Graph-based tracing of filamentary structured objects with applications in neuronal and retinal images

De, Jaydeep

2016

De, J. (2016). Graph-based tracing of filamentary structured objects with applications in neuronal and retinal images. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/65993>

<https://doi.org/10.32657/10356/65993>

GRAPH-BASED TRACING OF FILAMENTARY STRUCTURED OBJECTS WITH APPLICATIONS IN NEURONAL AND RETINAL IMAGES

A PhD Thesis
Submitted to the School of Computer Engineering
of Nanyang Technological University

by

Jaydeep De

for fulfilment of the requirement
for the Degree of Doctor of Philosophy

February 2016

Declaration

I, Jaydeep De, declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given in the bibliography.

Acknowledgment

I feel very fortunate that I have the opportunity to study as a graduate student in NTU for the past four years. My sincere gratitude goes to plenty of peoples who played very important roles on the completion of this thesis as well as my life.

I would like to express my special appreciation and thanks to my supervisor Dr. Li Cheng, you have been a tremendous mentor for me. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. Your advice on both research as well as on my career have been priceless. I would also like to thank my NTU supervisor Assoc. Professor Dr. Lin Feng for his guidance about research work and thesis writing. I also want to thank you for handling the administrative matters from NTU side in a smooth and easy manner. I would also like to thank Dr. Manoranjan Dash and Dr. Vladimir Kuznetsov for accepting me as a PhD student. I would especially like to thank the administrative staff of BII Ms. Betty Kee, Christine and Norani and IT staff Mr. Robin, Mr. Ananthan for your constant support for the research activities in BII.

A special thanks to my family. Words cannot express how grateful I am to my mother, father, mother-in law, father-in-law, sister, sister-in-law, uncle and aunty for all of the sacrifices that youve made on my behalf. Your prayer for me was what sustained me thus far. I would also like to thank all of my friends who supported me in writing, and incited me to strive towards my goal. This last word of acknowledgment I have saved for my dear wife Monalisa, who has been with me all these years and has made them the best years of my life.

Abstract

Filamentary structured objects are everywhere in nature and man-made architectures. These are found in the network of neurons, blood vessels, roads and others, which can scale from a range of few microns (in case of neurons) to thousands of kilometers (in case of roads). Image-based analysis of these filamentary structured objects are of great interest as they offer important information about the structure and connectivity of such networks which can help us in applications such as drug screening for neurological disorders or computer-aided diagnosis of diabetic retinopathy.

However, the task is challenging due to the bottleneck of filament *crossover* issue, which essentially hinders the use of existing systems in large-scale applications. The aim of the thesis is to build a generic graph-based framework for tracing the images of filamentary structured objects. The framework consists of two steps: preprocessing step consisting of segmentation, skeleton extraction, digraph representation and tracing step using graph-based methods. The main focus of this thesis is the second step.

After the preprocessing step, the skeleton is represented as a directed graph structure where skeleton segments are represented as nodes in the graph and there is an edge between two nodes if the corresponding skeletons are touching each other. Now, the goal is to cluster the graph into disjoint trees where the number of clusters depends on the number of tree structures in the filament. This is achieved by assuming that at least one node is labeled in each cluster of the graph. Given the structure of the graph and one labeled node per cluster, the problem becomes that of assigning class labels to the rest of the unlabeled nodes.

The problem is formulated as a label propagation problem on a weighted digraph using the Matrix Forest Theorem. The normalized conductance of a rooted spanning converging forest of a digraph is used as a similarity score between the labeled and

unlabeled nodes. The label propagation problem can also be formulated as a random walk on an Absorbing Markov Chain (AMC). After converting the original graph into an AMC, the Fundamental Matrix is computed, which is the expected number of visits from one node to another before absorption. For tracing problem, this is another kind of similarity score between the labeled and unlabeled nodes. For both approaches, the class label for an unlabeled node is the same as that of a labeled node with largest similarity. If the weight matrix is formulated as parametric functions then the parameters can be determined by maximizing the sum of the logarithm of expected number of visits from unlabeled nodes to the labeled nodes. For the scenario where the class labels for only very few nodes are known, the parameter estimation problem is solved by Expectation Maximization (EM) algorithm. For comparison purpose, the tracing problem is also formulated as a Maximum A Posteriori (MAP) inference problem in Undirected Graphical Models.

Empirical analysis is conducted for all the approaches using in-house dataset of 2D neuronal images¹ and publicly available benchmark dataset of retinal images which proves the superiority of my approach compared to current state of art methods as well as widely used commercial software.

¹Downloadable at <http://web.bii.a-star.edu.sg/~jaydeepd/tracing.htm>.

List of Abbreviations

HTS	High Throughput Screening
GIS	Geometric Information System
MST	Minimal Spanning Tree
GMM	Gaussian Mixture Models
AMC	Absorbing Markov Chain
EM	Expectation Maximization
MFTD	Matrix Forest Theorem on a Digraph
PARW	Partially Absorbing Random Walks
PAMC	Parametric Absorbing Markov Chain
PAMC_Co	Parametric Absorbing Markov Chain with Complete data
PAMC_In	Parametric Absorbing Markov Chain with Incomplete data
PAMC_Ra	Parametric Absorbing Markov Chain with Random weights
MF	Mean Field Message Passing
LBP	Loopy Belief Propagation
TRW	Tree-reweighted Message Passing
LLGC	Learning with Local and Global Consistency method
SGL	Symmetrized Graph Laplacian
ZFL	Zero-mode Free Laplacian
DIADEM	Digital Reconstruction of Axonal and Dendritic Morphology challenge
DRIVE	Retinal fundus image dataset
STARE	Retinal fundus image dataset
NeuB1	Neuron dataset batch 1 from BII
NeuB2	Neuron dataset batch 2 from BII

List of Notations

V	set of nodes of digraph
n	cardinality of a graph
$\mathcal{E}$	set of edges of a digraph
i, j, k	general symbols to denote indices
$W = [w_{ij}]$	weight matrix of size $n \times n$ of a digraph with elements w_{ij}
$G = (V, \mathcal{E}, W)$	a digraph of nodes V , edges E and weight matrix W
V_U	set of nodes of undirected graph
E_U	set of edges of a undirected graph
$G_U = (V_U, E_U)$	a undirected graph of nodes V_U , edges E_U
v_i	i^{th} node of a graph
d_i^+	out degree of a node v_i
D	Diagonal out-degree matrix a graph
P	Transition probability matrix of a graph
$L = D - W$	Laplacian matrix of a graph
V_u	set of unlabeled nodes
V_l	set of labeled nodes
K	number of class labels
$C = \{1, 2, \dots, K\}$	set of K class labels
$Y = [y_{ik}]$	a label matrix of size $n \times K$ with elements y_{ik}
y_i^*	predicted label for i^{th} unlabeled node
α, β, τ	parameter symbols
θ_{ij}	angle between two segments i and j
θ_c	a threshold angle
d_i^-	In-degree of a node i
$\tilde{P}$	Transition probability matrix of the augmented graph
$\tilde{Q}$	Transition probability matrix of the absorbing graph
$E = [e_{ij}]$	Fundamental Matrix of the absorbing graph with elements
ϕ_{ij}^l	e_{ij} l^{th} feature between pair of nodes i and j
ϕ_{ij}	feature vector between a pair of nodes i and j of length m
θ	a parameter vector of length m
$W(\theta)$	parametric weight matrix
$E(\theta)$	parametric Fundamental Matrix
$\ell(\theta), \ell'(\theta)$	Likelihood functions
$Q_i(k)$	a probability distribution of i^{th} node over k class labels

Publication

- (i) Jaydeep De, Xiaowei Zhang, Li Cheng, Feng Lin and others. “A Graph-theoretical Approach for Tracing Filamentary Structures in Neuronal and Retinal Image”, In *IEEE Tran. of Medical Imaging*, 35(1), 257-272, 2015.
- (ii) Jaydeep De, Huiqi Li, and Li Cheng. “Tracing retinal vessel trees by transductive inference”, In *BMC Bioinformatics*, 15,1-20, 2014.
- (iii) Li Cheng, Jaydeep De, Xiaowei Zhang, Feng Lin, Huiqi Li. “Tracing Retinal Blood Vessels by Matrix-Forest Theorem of Directed Graphs”, In *Medical Image Computing and Computer-Assisted Intervention*, Boston, MA, USA, 2014.
- (iv) Jaydeep De, Tengfei Ma, Huiqi Li, M. Dash, and Li Cheng. “Automated Tracing of Retinal Blood Vessels Using Graphical Models”, In *Scandinavian Conference on Image Analysis*, Espoo, Finland, 2013.
- (v) Jaydeep De, Xiaowei Zhang, Li Cheng and Feng Lin. “Transduction on directed graphs via absorbing random walks”, Submitted In *IEEE Tran. of Pattern Analysis and Machine Intelligence*.

Contents

Contents	i
List of Figures	v
List of Tables	xi
1 Introduction	1
1.1 Scientific Significance of Tracing Filamentary Structured Objects .	1
1.2 Literature Review of Tracing Filamentary Structures	4
1.2.1 Literature Review of Neuron Tracing	4
1.2.2 Literature Review of Retinal Blood Vessel Tracing	8
1.3 What is a <i>Crossover</i> ?	9
1.4 A Graph-based Approach for Tracing Filamentary Structured Ob- jects	10
1.5 Contributions	12
1.6 Thesis Organization	15
2 Pre-processing for the Tracing System	16

2.1	Segmentation Using Penalized Likelihood Estimate with Graph Laplacian	17
2.2	Segmentation Result $\rightarrow$ Skeleton Map	22
2.3	Constructing the Digraph	24
3	Tracing with Matrix Forest Theorem on a Directed Graph	26
3.1	Introduction	27
3.2	Problem Set-up	28
3.3	Matrix Forest Theorem on A Digraph	29
3.4	Inference Using Matrix Forest Theorem on A Digraph	31
3.5	Computational complexity	31
3.6	PAC-Bayesian Label Propagation Bound	32
3.7	Finding the Weight Matrix $W = [w_{ij}]$	35
3.8	Summary	37
3.9	Appendix	37
4	Tracing with Absorbing Markov Chain on a Directed Graph	39
4.1	Introduction	40
4.2	Basics	42
4.3	Problem Set-up	42
4.4	Absorbing Markov Chain and its Fundamental Matrix	44
4.5	Inference on Absorbing Markov Chain	47
4.6	Connections to Existing Methods	52
4.7	Computational Complexity	53
4.8	Theoretical Error Bound Analysis	54
4.9	Summary	57

4.10	Appendix	58
5	Parameter Estimation for Absorbing Markov Chain	73
5.1	Introduction	74
5.2	Problem Set-up	75
5.3	Expectation Maximization based Parameter Estimation for Incomplete Data Scenario	76
5.3.1	E-step	76
5.3.2	M-step	77
5.4	Parameter Estimation for Complete Data Scenario	80
5.5	Feature vectors ϕ_{ij}	82
5.6	Summary	84
6	Applications of the Tracing System	86
6.1	Dataset Description	87
6.1.1	Neuronal Datasets	87
6.1.2	Retinal Datasets	88
6.1.3	Tracing Ground-truth Generation and Storage	88
6.2	Evaluation Criteria	90
6.3	Comparison methods	90
6.3.1	Undirected Graphical Models	91
6.3.2	Other Methods and Softwares for Tracing	96
6.4	Quantitative Results and Discussion	98
6.4.1	Tracing Results for MFTD	98
6.4.2	Tracing with Undirected Graphical Models	101
6.4.3	Tracing Results with Parametric Absorbing Markov Chain	103

CONTENTS

6.4.4	Comparison with other methods	104
6.5	Qualitative Results and Discussion	107
6.6	Additional Applications on Large Scale Datasets	119
7	Conclusion and Future Work	132
7.1	Thesis Summary	132
7.2	Recommendations for Future Work	133
	References	134

List of Figures

1.1	Different examples of filamentary structures. (A) Drosophila olfactory axonal projection image with its reconstruction (in red) from DIADEM dataset [1]. (B) Rat brightfield micrograph image and its reconstruction (in color) from EPFL [2]. (C) In-vitro mouse embrionic stem-cell image and its reconstruction (in color) from BII. (D) X-ray angiogram of left coronary artery its reconstruction (in color) [3]. (E) Retinal fundus image its reconstruction (in color) from DRIVE dataset [4]. (F) Aerial images of road networks and reconstruction (in red) [2].	2
1.2	<i>Crossover</i> situation in Neuronal and Retinal images and the expected tracing results.	10
1.3	The graph based approach for tracing filamentary structured objects.	12
2.1	Overview of our two-step tracing method.	17

2.2	Comparing segmentation results of GMM likelihood estimate vs. our penalized likelihood estimate. (a) A cropped input neuronal image. Middle column: Heatmaps of filament probability images of the GMM likelihood estimate (b) vs. the proposed penalized likelihood estimate (d). Right column: Segmentation results of the GMM likelihood estimate (c) vs. the proposed penalized likelihood estimate (e).	20
2.4	Running example of digraph preparation process in the tracing step. (a) Original image. (b) Segmentation result. (c) Skeleton map, with (d) being a fragment inside the green box. (e) Undirected graph G_U . (f) Digraph G	23
2.3	Example of filament (in red) and junction (in green).	23
2.5	Synthetic example of rule 3.	25
3.1	Plot for the energy function $f(\theta_{ij})$ and the corresponding weight w_{ij} , both as functions of θ_{ij} , the angle between the i -th and j -th filaments.	37

4.1	(a) An illustrative example of a weighted digraph-based transduction setting: Two different class labels are to be propagated from the labeled nodes (the green and the red nodes, each for one class) to the rest nodes following the graph structure. Here only a subset of the graph nodes and edges are displayed. Quantities such as W and P can be computed accordingly. (b) As some nodes might have zero in-degree (i.e. source nodes), a new node is further added with directed edges to every nodes including itself, which gives $\tilde{P}$. According to the $\mathbf{c}$ vector, the edges toward those previous source nodes are weighted by 1, and the rest are $1 - \alpha$. (c) Its transpose, $\tilde{Q}$, corresponds to the same graph but with edge directions being reversed. This facilitates the evaluation of affinity scores flowing from unlabeled nodes (e.g. leaf nodes) to the labeled nodes (e.g. source nodes).	44
5.1	Figure for functions $f^1(\theta_{ij}; \kappa, \theta_c)$ and $f_2(x; \kappa, c)$	83
6.1	Figure for <i>Message Passing</i> algorithm.	93
6.2	Figure for energy function $\psi_{ij}(x_i, x_j)$	95
6.3	Effect of varying the internal parameters α or β of our four MFTD tracing variants. From top to bottom, each row presents one of the four datasets, namely NeuB1, NeuB2, DRIVE, and STARE; From left to right, each column is for one of the four MFTD tracing variants. In each panel, α or β are presented in x-axis, while y-axis shows the DIADEM score performance.	99
6.4	Variation of performance with the parameters k and θ_c	101

LIST OF FIGURES

6.5	The tracing results for various methods from an image in NeuB1. The red transparent circle in gold standard tracing denotes the <i>crossover</i> situation.	111
6.6	The tracing results for various methods from an image in NeuB1. The red transparent circle in gold standard tracing denotes the <i>crossover</i> situation.	112
6.7	The tracing results for various methods from an image in NeuB1. The red transparent circle in gold standard tracing denotes the <i>crossover</i> situation.	113
6.8	The tracing results for various methods from an image in NeuB2. The red transparent circle in gold standard tracing denotes the <i>crossover</i> situation.	114
6.9	The tracing results for various methods from an image in NeuB2. The red transparent circle in gold standard tracing denotes the <i>crossover</i> situation.	115
6.10	The tracing results for various methods from an image in NeuB2. The red transparent circle in gold standard tracing denotes the <i>crossover</i> situation.	116
6.11	The tracing results for various methods from an image in DRIVE dataset. The red transparent circle in gold standard tracing de- notes the <i>crossover</i> situation.	117
6.12	The tracing results for various methods from an image in DRIVE dataset. The red transparent circle in gold standard tracing de- notes the <i>crossover</i> situation.	117

6.13	The tracing results for various methods from an image in STARE dataset. The red transparent circle in gold standard tracing denotes the <i>crossover</i> situation.	118
6.14	The tracing results for various methods from an image in STARE dataset. The red transparent circle in gold standard tracing denotes the <i>crossover</i> situation.	118
6.15	Comparisons of accuracy & efficiency on the citation problem (i.e. datasets of US Patent, CoRA and CiteseerX). Plots of the first column display the micro-averaged accuracy (AC) on US Patent, CoRA and CiteseerX, respectively; Plots of the second column show the corresponding CPU time spent. In all plots, the horizontal axis denotes the label ratio (percentage of labeled nodes) varying from 10% to 90% with 10% increment. See text for details.	121
6.16	Comparisons of F1-Score on the multi-label problem of Google+ and Twitter datasets. In both plots, the horizontal axis denotes the label ratio (percentage of labeled nodes) varying from 10% to 90% with 10% increment. See text for details.	124
6.17	Empirical time-complexity of batch (4.3) vs. online (7,8) updates of the fundamental matrix, E . (A) and (B) show the CPU-time (log-seconds) of batch update vs. online update for changing one row using (4.7), and for inserting / deleting a new node with (4.8), respectively. See text for details.	126

6.18	Averaged absolute difference between online and batch updates. The red curve (with a square mark) and the blue curve (with a star mark) shows the average differences for changing one row using (4.7), and for inseting/deleting a new node with (4.8), respectively.	126
6.19	Comparison with state-of-the-art methods based on undirected graphs.	128
6.20	Robustness of our system vs. changing α values between .01 and .99. (A) For CoRA and CiteseerX, the performance of our system is rather stable (with around .001 variation) when α is within .01 and 0.9, and start to decrease slightly (around .01 variation) when α value goes beyond .9. (B) The performance remains constant regardless of α values in vessel tracing problems.	129
6.21	Robustness of our system with respect to varying label ratio. The error bar of each labeling ratio (r) displays 5 – 95 percentile of accuracy when α values are systematically sampled between .01 and .99 with an .01 increment. The narrow deviations from median as shown in the error bar (usually less than 2%) clearly suggest that our system is rather stable against changes of α values.	129
6.22	Empirical discriminative ability of our approach. Intra- and inter-class accumulated affinities are displayed over different datasets. Intuitively the larger the gap between intra- and inter-class, the better its performance would be. See text for details.	130

List of Tables

2.1	Comparison of segmentation performance w.r.t. F1 score. See text for details.	22
5.1	Set of features using local absolute difference of intensity $ \mathcal{I}_i - \mathcal{I}_j $ and local absolute difference of filament diameter $ d_i - d_j $ between two attached segments.	84
6.1	Description of SWC file	89
6.2	Description of structure identifier	89
6.3	DIADEM scores for MFTD _a , MFTD _b , MFTD _c and MFTD _d on the neuronal and the retinal datasets.	100
6.4	One of the four Netmets scores, geometric false positive rate, on the neuronal and the retinal datasets.	100
6.5	One of the four Netmets scores, geometric false negative rate, on the neuronal and retinal datasets.	100
6.6	One of the four Netmets scores, connectivity false positive rate, on the neuronal and the retinal datasets.	100
6.7	One of the four Netmets scores, connectivity false negativity rate, on the neuronal and the retinal datasets.	100

LIST OF TABLES

6.8	DIADEM scores of three inference methods of graphical model on the neuronal and the retinal datasets.	102
6.9	One of the four Netmets scores, geometric false positive rate, on the neuronal and the retinal datasets.	102
6.10	One of the four Netmets scores, geometric false negative rate, on the neuronal and retinal datasets.	102
6.11	One of the four Netmets scores, connectivity false positive rate, on the neuronal and the retinal datasets.	102
6.12	One of the four Netmets scores, connectivity false negativity rate, on the neuronal and the retinal datasets.	102
6.13	Performance of Absorbing Markov Chain with singles features and with multiples features with random weights or learned parameters.	103
6.14	Brief description of the methods compared.	104
6.15	DIADEM scores of comparison methods on the neuronal and the retinal datasets. The first four columns are variants of our tracing system, while the next six columns are reported by replacing the tracing component of our system with different methods. The last two columns are results of two existing neuronal tracing systems. .	106
6.16	One of the four Netmets scores, geometric false positive rate, on the neuronal and the retinal datasets.	106
6.17	One of the four Netmets scores, geometric false negative rate, on the neuronal and retinal datasets.	106
6.18	One of the four Netmets scores, connectivity false positive rate, on the neuronal and the retinal datasets.	106

LIST OF TABLES

6.19 One of the four Netmets scores, connectivity false negativity rate, on the neuronal and the retinal datasets.	106
---	-----

Chapter 1

Introduction

1.1 Scientific Significance of Tracing Filamentary Structured Objects

Filamentary structured objects are ubiquitous in nature and man made structures and can be found in network of neurons, blood vessels, roads and others. They can scale from a range of few microns (in case of neurons) to thousands of kilometers (in case of roads). Image based analysis of these filamentary structured objects are of great interest as they provide important information about the structure and connectivity of such networks.

For example, the cognitive functions of our brain depends on structure and the connectivity of the neurons [5, 6]. Loss of connectivity of neurons in our brain due to disease like Parkinson and Alzheimers or due to accidental damage can cause the loss or degradation of higher level cognitive abilities like movement, memory etc [7, 8, 9, 10]. To identify the drugs to restore or improve these cog-

Figure 1.1: Different examples of filamentary structures. (A) *Drosophila* olfactory axonal projection image with its reconstruction (in red) from DIADEM dataset [1]. (B) Rat brightfield micrograph image and its reconstruction (in color) from EPFL [2]. (C) In-vitro mouse embryonic stem-cell image and its reconstruction (in color) from BII. (D) X-ray angiogram of left coronary artery its reconstruction (in color) [3]. (E) Retinal fundus image its reconstruction (in color) from DRIVE dataset [4]. (F) Aerial images of road networks and reconstruction (in red) [2].

nitive functions, it is important to conduct high throughput screens (HTS) of microscopic neuronal images, which naturally requests for automated neuronal tracers [11].

A very similar situation has also been observed in tracing retinal blood vessel trees from fundus images. The topological and geometrical properties of retinal vessel trees provide valuable clinical information in diagnosing diseases such as proliferative diabetic retinopathy, glaucoma, and hypertensive retinopathy. Take as an example the disease of proliferative diabetic retinopathy, a leading cause

of blindness in the working-age population of most developed countries. It is the result of progressive damage to the network of tiny blood vessels that supply blood to the retina, and is specifically characterized by the formation of newly formed vessels in the retina [12]. It has also been observed that the change of length-to-diameter ratio of an artery tree is related to diseases of hypertension and diabetes [13, 14]. This thus depends on the description of blood vessel tree structure in clinical diagnosis, and as a result, calls for proper tracing of the vessel trees from fundus images.

Kong et al. [15] has proposed a method for quantifying the regional myocardial performance using the 3D motion of coronary artery from X-ray angiograms. After that several research efforts have been made for complete 3D reconstruction of coronary arteries with temporal information [16, 17, 18]. These temporal 3D reconstructions of coronary arteries provide valuable information about vessel length, volume and flow of blood to coronary muscles at different phases of cardiac cycles [19, 20]. These are some of the important factors for cardiovascular disease diagnosis and therapy planning. Therefore automatic tracing of 3D X-ray angiograms is of prime importance for quick diagnosis and cure of coronary artery disease.

In geology and remote sensing field, automatic detection and tracking of road networks from high resolution satellite images has several applications like automatic correction and updating of Geometric Information System (GIS) [21, 22] and detection of change in temporal road network images [23]. These are the core driving engine behind the popular map-software like Google Maps, Apple Maps or Bing Maps. Therefore automatic tracing of road network plays the central role behind these successful commercial softwares.

Figure 1.1 shows various examples of filamentary structures and their reconstruction, where subfigure (A), (B) and (C) show some examples of different kinds of neuronal structures, subfigure (D) shows an example of coronary artery angiogram, subfigure (E) shows an example of retinal fundus image and subfigure (F) shows an example of road network.

1.2 Literature Review of Tracing Filamentary Structures

In this section I will discuss the related literature of two applications of tracing system which are the main focus of this thesis. The first application is neuron tracing and the second application is retinal blood vessel tracing.

1.2.1 Literature Review of Neuron Tracing

Computer based analysis of neuronal morphology is dated back to 1965 by Glaser et al. [24] where computer based interactive microscope is used by human operator to store the critical point co-ordinates. In the next two decades several efforts have been made to reduce the human intervention but they are limited by the lack of computational power [25]. It has been recognized as a very difficult problem [26] as it requires the computer to mimic some very complex functions performed by our visual system. In the past two decades, there has been a great development in computational power and computer vision algorithms which triggered many research in this area. This research areas in neuronal reconstruction can broadly categorized into following five groups.

-
- Sequential tracing algorithms start from few seed points and sequentially trace the neuron filaments by fitting a local model for filamentary structures. Aylward et al. [27] has proposed a local model based on eigenvector of Hessian matrix whereas Al-Kofahi et al. [28] has proposed a local model based on edge detectors. Both the methods suffer from discontinuity in the filamentary structure and requires significant manual post-processing for curve completion.
 - Skeletonization method proposed by He et al. [29] starts with 3D deconvolution of images using microscope point spread function followed by 3D skeletonization proposed by Cohen et al. [30]. Isolated skeleton fragments are connected by different curve completion schemes proposed by the authors. In a more recent approach Yuan et al. [31] has proposed a more robust gray-scale image based skeletonization approach. Nonetheless these methods are very expensive in time and space due to the deconvolution operation.
 - Minimal path based tracing approaches trace the path between two user defined points based on Dijkstra's shortest path. E Meijering et al. [32] has proposed a ridge detector based cost function for connecting the pixels in eight-neighborhood whereas H. Peng et al. [33] has proposed geodesic metric function based cost function. Choen et al. [30, 34] has proposed a Fast Marching method based tracing approach. Though minimal path based tracing approaches are good for quick editing and semiautomatic tracing but they are not suitable for complete automation of tracing.
 - The Minimal Spanning Tree (MST) based tracing methods starts with de-

tecting few anchor points and then connecting those points by Minimal Spanning Tree [31]. González et al. [35] has proposed a K-Minimum Spanning Tree based tracing approach where K number of points are used for building the spanning trees out of N anchor nodes. The approximately optimal spanning tree is selected by minimizing a global objective function using ant colony optimization algorithm.

- Active contour or “snake” based tracing methods are used by Schmitt et al. [36] where a snake pixel or snaxel is defined by 4D points of pixel locations and radius. The external force on the snake is defined by the combination of offset and central medialness of a snaxel. Stepanyants et al. [37] has proposed a snake model for optimizing the trace after initial coarse estimation of filaments. These snake models are not good for complete automation of tracing methods as they require the user to provide the branching points and end points.

Lack of computational tools for complete automation of large scale 3D neurons has inspired the DIADEM challenge competition [38], which attracted a series of researchers from computer science to come and solve this problem. In the next paragraph I will briefly discuss about the five finalist algorithms of DIADEM challenge.

The joint winner of the challenge B. Roysham’s team has proposed a 3D open-curve active contour (snake) model for tracing. The algorithm starts with image denoising and continuity enhancement using curvelet transformation and scalar voting [39]. Then a Gradient Vector Field based deforming force is used to formulate the energy function of the open-curve snake. The branching points

of the filament is detected by using the points where the snakes collide with each other. This algorithm fails to perform when there is a *crossover* of neurons due to lower resolution of imaging or physical touching of neurites. The authors have suggested the use of Adaboost detector [40] for handling the *crossover* situation explicitly. P. Chotani et al [41] has proposed a optimization based method for tracing. After initial preprocessing steps of filtering, thresholding and skeleton extraction, skeletons are merged into tree structure by solving a two optimization problem. The first optimization is based on intensity integrated along the trace and the elastic energy of the trace whereas the second one is based on the intensity and straightness of the trace. Though this method provides a completely automatic solution to the problem but it has been mentioned that the *crossover* problem is not solved in this work. T. Zhao et al. [42] has proposed a 3D tube model and minimum spanning tree based approach for tracing neuron filaments. this method explicitly takes care of the *crossover* and it is locally resolved using pairing of branches by Hungarian algorithm. González et al. [2] has proposed a K-minimum spanning based approach for tracing filamentary structure. In the follow up research work E. Türetken et al. [43, 44, 45, 46] has proposed a quadratic mixed integer programming formulation for solving the K-MST problem. But none of their reserach work has addressed the problem of *crossover*. E. Bas et al. [47] has proposed a recursive principle curve based tracing approach. This method has also left the problem of *crossover* for their future work.

1.2.2 Literature Review of Retinal Blood Vessel Tracing

Existing efforts in retinal vessel analysis can be roughly categorized into two groups, namely, segmentation-based and tracking-based. The segmentation-based methods often use pixel classification [48, 49, 50, 51, 52, 53, 54, 55] to produce a binary segmentation, where a pixel is classified into vessel or non-vessel. Ricci et al. [56] work with orthogonal line operators and support vector machine to perform pixelwise segmentation. Mendonca et al. [48] use four directional differential operators to detect the vessel centerlines, which are then engaged for morphologically reconstructing the vessels. An unsupervised curvature based vessel segmentation method is proposed by Garg et al. [49]. Meanwhile, a deformable contour model is adopted by Espona et al. [50], by incorporating the snake method with domain specific knowledge such as the topological properties of blood vessels. Soares et al. [52] adopt eighteen dimensional Gabor response feature to train two Gaussian Mixture Models (GMM), which are further employed to produce a binary probability map for a test image. The tracking-based methods [57, 58, 59, 60, 61, 41, 62, 43, 44, 45, 46], on the other hand, usually start with a seed and track the intended vessel based on local intensity or texture information. The authors of [58] divide the image into non-overlapping grid and considered each grid separately for seed finding, which are followed by the tracking procedure to uncover the vessel network structure. In a series of research efforts [43, 44, 45, 46], the authors extract *tubularity* measure of all image pixels and connect those pixels having high tubularity measure values. Then, an optimal set of trees over these tubular pixels are selected by minimizing a global objective function with prior geometric constraints, such as orientation and width

of the vessel structure. In a recent work [63], the tracking problem is formulated as a constrained optimization problem based on vessel orientation and topology, and a candidate enumeration algorithm is devised for the proposed optimization problem, which prunes the search space by maintaining a lower bound on the objective function.

It has been observed that segmentation-based methods tend to produce many disconnected and isolated segments [56], which are less favorable towards retaining the important topological properties of vessel networks [13]. Vessel tracking methods, on the other hand, often preserve the connectivity structure of vessel segments. Nonetheless they encounter great difficulties with the occurrence of *crossover* [61] at the junction points. Current methods often fail to trace properly, as it is nontrivial to predict whether the vessel segments touching at a junction point belong to one tree, or two and more trees, and for the later case, to which tree each segment belongs.

1.3 What is a *Crossover*?

Crossover is a situation where two filamentary structures overlap with each other and appear to be connected due to the image modality though they are physically separated in space. Figure 1.3 shows the examples of crossovers in Neuronal and Retinal images. All the previous methods in this field talk about the *Crossover* problem but fail to solve the situation. In this thesis we have proposed a graph based representation of the problem which proves to be better in solving this problem, which will be discussed later in this thesis.

Figure 1.2: *Crossover* situation in Neuronal and Retinal images and the expected tracing results.

1.4 A Graph-based Approach for Tracing Filamentary Structured Objects

In this thesis, a graph based approach for tracing filamentary structured objects are proposed. It starts with a preprocessing step where the filamentary structures are detected using simple thresholding or supervised methods. Filament skeletons are extracted and the skeleton is represented as a directed graph (digraph) structure. After the digraph representation the goal is to cluster the graph into disjoint trees where the number of clusters depend on the number of tree structure in the

filament. This is achieved by assuming that at least one node is labeled in each cluster of the graph. Knowing the graph structure and one labeled node per class, the problem can be presented by well studied algebraic graph theoretical methods like Matrix Forest Theorem, random walk based methods like Absorbing Markov Chain or probabilistic methods like Undirected Graphical Models. As all these methods fit into our situation and they are all based on graphical representation of a problem, therefore I call them graph-based methods.

Figure 1.3 is a schematic representation of the full tracing system. Figure 1.3-(A) is a small example of the skeleton structure whereas subfigure (B) is a graph representation of the same skeleton. Skeleton segments are represented as nodes in the graph and there is an edge between two nodes if the corresponding skeleton segments are touching each other. Edge directions are found based on some prior knowledge which will be defined in later chapters. Two skeleton segments and their corresponding nodes, marked in red and blue, are two labelled nodes belonging to two different class labels. Subfigure (C) is an animated representation of the output of graph based methods, where each node with different proportion of red and blue denotes the chance of that node belonging to the red or blue class. Subfigure (D) is the disjoint tree of two different class labels after maximum chance label assignment on each node from subfigure (C). Subfigure (E) is the corresponding skeleton structure representation where the skeletons are colored with red or blue class labels.

Figure 1.3: The graph based approach for tracing filamentary structured objects.

1.5 Contributions

The contribution of this thesis is five fold.

The first contribution is the formulation of tracing problem in terms of graph based label assignment problem. In contrast to other methods [64] where the branching points are represented as nodes in the graph and filament segments are represented as edges, my method represents the filament segments as nodes and nodes are connected if the filament segments are connected through the junction points (branching or *crossover* points). My representation is particularly useful for resolving the *crossover* situation encountered in many tracing applications which is the main drawback for DIADEM challenge finalist algorithms. This contribution is discussed in Chapter-2 and it is partly published in SCIA-2013 and BMC Bioinformatics, 2014.

The second contribution is formulating the tracing problem as a label propagation problem on a weighted digraph using Matrix Forest Theorem. According to the Matrix Forest Theorem, the normalized conductance of a rooted spanning

converging forest of a digraph is defined by $(I + \alpha L)^{-1}$, where α is a parameter and L is a graph laplacian matrix of the digraph. For tracing problem, this is used as a similarity score between the labeled and unlabeled nodes. If an unlabeled node is more similar to a particular labeled node than others, then it can be assigned the same class label as that labeled node. This contribution is discussed in Chapter-3 and partly published in BMC Bioinformatics, 2014, MICCAI-2014 and IEEE TMI, 2015.

The third contribution is formulating the tracing problem as a label propagation problem on a weighted digraph using random walk on an Absorbing Markov Chain (AMC). After converting the original graph into an AMC, its Fundamental Matrix is computed, which is the expected number of visits from one node to another before absorption. For tracing problem, this is another kind of similarity score between the labeled and unlabeled nodes. Class labels for the unlabeled nodes can be assigned using the similar approach as the previous one. In addition, an optimal one-step increment/decrement update (aka online update) is introduced, which is handy in scenarios where local changes are made to graphs over time, or prediction are made on out-of-sample instances. The proposed algorithm also bears interesting connections to existing works including undirected graph Laplacian, diffusion graph kernels, spanning forest of graphs, graph-based proximity measure, among others. Few interesting properties of the Fundamental Matrix are also studied. This contribution is discussed in Chapter-4 and in review of IEEE TPAMI during the submission of thesis (2016).

The fourth contribution is the parameter estimation for the Absorbing Markov Chain based tracing problem. If the weight matrix is formulated as parametric functions then the parameters can be determined by maximizing the sum of

the logarithm of expected number of visits from unlabeled nodes to the labeled nodes. For the scenario, where the class labels for a very few nodes are known, the parameter estimation problem is solved by Expectation Maximization (EM) algorithm. For a different scenario, where there is a separate training set for which the class labels for all the nodes are known, the parameters are estimated using maximum-likelihood formulation. This contribution is discussed in Chapter-5 and submitted in MICCAI-2016.

The fifth contribution of this thesis is the thorough empirical evaluation of the methods discussed in this thesis on two applications namely neuron tracing and retinal blood vessel tracing. For neuron tracing, I have used our in-house neuronal microscopic image dataset dedicated to the task of tracing with crossovers. The dataset, together with the ground-truth annotations, the evaluation metric implementation as well as subsequent comparisons with baseline methods, have been made publicly available ¹. This is the first benchmark of such kind to my knowledge ², and I wish it can provide an option for researchers in the field to compare performance on the same ground. For retinal blood vessel tracing application, publicly available dataset DRIVE [4] and STARE [65] is used. The tracing annotation for these two datasets is prepared by the experts in this field and made available online. This contribution is discussed in Chapter-6 and partly published in all the publications and submissions mentioned above.

¹Downloadable at <http://web.bii.a-star.edu.sg/~jaydeepd/tracing.htm>.

²The DIADEM challenge [1] focuses on 3D neuronal tracing instead of the 2D tracing as considered in this thesis.

1.6 Thesis Organization

The thesis is organized in eight more chapters. In Chapter-2, the pre-processing steps for tracing are explained. In particular this chapter focuses on the representation of the skeleton structure into a digraph. In Chapter-3 Matrix Forest Theorem on Digraph based formulation is explained. In Chapter-4, I have described the solution of the tracing problem using Absorbing Markov Chain on a directed graph. In Chapter-5, the parameter estimation for Absorbing Markov Chain is explained. Chapter-6 describes the two application of the tracing problem that is neuron tracing and retinal blood vessel tracing. Experimental evaluation of the tracing system is done in this chapter. In Chapter-7, I have discussed about the conclusion and the future direction of the work.

Chapter 2

Pre-processing for the Tracing System

In this chapter, I will briefly discuss the pre-processing steps of our tracing system. The input raw image is converted to binary segmented image using penalized likelihood estimate using graph laplacian. The segmentation result is converted to its digraph representation by two intermediate stages: First, the segmented image undergoes a thinning stage to a skeleton map; Second, a digraph is constructed using the skeleton map. Details are as follows:

The first step of our pipeline is segmentation where we convert the color retinal fundus image into a binary image where the true pixels are vessel pixels and false pixels are non-vessel ones. Here the goal of the segmentation step here is to extract the vessel skeleton while maintaining the structural connections, as well as the corresponding pointwise radii along the skeleton (A point radius is to measure the thickness of a skeleton point in the orthogonal direction), based

Figure 2.1: Overview of our two-step tracing method.

on which the retinal vessels can be faithfully reconstructed. This differs notably from the usual aim of most existing segmentation work, where the emphasis is to achieve a high classification accuracy. As the number of vessel pixels are much fewer comparing to the number of background pixels, often a high accuracy is achieved by missing many vessel pixels – a situation we try to avoid. In fact, our goal can be better described as *segmentation with a high recall*. It is critical for us to retain the vessel pixels that keep the local vein and artery branches from being broken or entirely missing. To achieve this, we have resort to a cascade of two segmentation modules for producing our final segments. The first one in the cascade is a supervised segmenter as being described next, and the second one is an unsupervised segmenter that is specialized at recovering parallel thin branches, which often tend to be merged into one thick branch by the first module.

2.1 Segmentation Using Penalized Likelihood Estimate with Graph Laplacian

Denote $\{1, 2, \dots, K_s\}$ a set of class labels, and $\mathbf{x} = \{x_1, x_2, \dots, x_{n_s}\}$ a set of n_s data points. Further, consider a set of K_s class-priors $\{p(k)\}_{k=1}^{K_s}$, and a set of class-conditional density functions $\{p(x_i|k)\}_{k=1}^{K_s}$. It is standard to formulate the

Gaussian mixture model (GMM) as the following form of likelihood functions (e.g. [66])

$$p(k|x_i) = \frac{p(x_i|k)p(k)}{\sum_{k=1}^{K_s} p(x_i|k)p(k)}.$$

An interesting observation is that the above equation can be re-cast as an equivalent matrix representation

$$T = M^{-1}Q,$$

where $T = [t_{ik}]$ and $Q = [q_{ik}]$ are $n_s \times K_s$ matrices, with $t_{ik} = p(k|x_i)$ and $q_{ik} = p(x_i|k)p(k)$, and $M = \text{diag}(\sum_k p(x_i|k)p(k))$ a $n_s \times n_s$ diagonal matrix.

Consider now a graph representation of the set of n_s data points $\mathbf{x}$, with n_s vertices and a $n_s \times n_s$ symmetric weight matrix $W_s = [w_{s,ij}]$ to characterize the set of edges. Let $D_s = \text{diag}(\sum_j w_{s,ij})$ be the diagonal degree matrix. The graph Laplacian $L_s = D_s - W_s$ acts as a smoothing operator on the function space on this graph. This inspires us to propose a graph Laplacian regularized GMM by solving the following augmented objective function:

$$T^* = \arg \min_{\hat{T}} \left(\|\hat{T} - M^{-1}Q\|_M^2 + \gamma \text{tr}(\hat{T}^T L_s \hat{T}) \right),$$

where $\gamma > 0$ is a tuning parameter, $\text{tr}(\cdot)$ is the matrix trace operator, and $\|\hat{T} - M^{-1}Q\|_M^2 := \text{tr}((\hat{T} - M^{-1}Q)^T M (\hat{T} - M^{-1}Q))$.

Since the above objective function is a quadratic function of $\hat{T}$, there exists a unique matrix-valued solution, $T^* = [t_{ik}^*]$, which can be obtained by solving the system of linear equations

$$(M + \gamma L_s)T^* = Q,$$

where $(M + \gamma L_s)$ is a positive definite matrix. It is interesting to note that the

solution to this penalized form remains a valid probability function, which can be verified straightforwardly by observing

$$\begin{aligned} T^* \mathbf{e}_{K_s} &= (I + \gamma M^{-1} L_s)^{-1} M^{-1} Q \mathbf{e}_{K_s} \\ &= (I + \gamma M^{-1} L_s)^{-1} \mathbf{e}_{n_s} = \mathbf{e}_{n_s}, \end{aligned}$$

where $\mathbf{e}_{K_s}$ and $\mathbf{e}_{n_s}$ are column vectors of K_s and n_s elements with all elements being 1, respectively, and the last equality holds since $(I + \gamma M^{-1} L_s) \mathbf{e}_{n_s} = \mathbf{e}_{n_s}$.

Finally, the class labels are obtained by row-wisely identifying the element with the maximum probability value,

$$y_i = \arg \max_k t_{ik}^*.$$

To illustrate the effect of the proposed penalized segmenter, Fig. 2.2 presents an exemplar neuronal image: Comparing to the existing GMM likelihood based segmenter as shown in the top row, it is clear that our segmenter delivers a much smoother and less noisy segmentation result, as displayed in the bottom row. Note similar results are also obtained for retinal images. Implementation detail and quantitative evaluation of the segmentation step are put off to the later section on empirical evaluations.

The Segmentation Step: Implementation Details and Evaluations During segmentation step, the set of data points, $\mathbf{x} = \{x_1, x_2, \dots, x_n\}$, contains features of all the pixels in training images. The number of class categories is naturally set to $K_s = 2$, as the focus here is to segment out the filamentary structures as foreground, with the rest pixels being background. More specifically,

Figure 2.2: Comparing segmentation results of GMM likelihood estimate vs. our penalized likelihood estimate. (a) A cropped input neuronal image. Middle column: Heatmaps of filament probability images of the GMM likelihood estimate (b) vs. the proposed penalized likelihood estimate (d). Right column: Segmentation results of the GMM likelihood estimate (c) vs. the proposed penalized likelihood estimate (e).

each data point $x_i \in \mathbb{R}^5$ containing five features: The first feature is the normalized image intensity, and the next four are normalized maximum Gabor filter responses among 18 directions in $[0, \pi]$ for 4 different scales. Our GMM likelihood model follows that of [66]. Throughout our experiments, a mixture of 15 Gaussian components is employed which is empirically to lead to good performance. The set of points (i.e. pixels) n_s is set to 10^6 , obtained by randomly selecting pixels from training images. The Laplacian is obtained from an 8-neighbor grid graph following the image structure, where each image pixel corresponds to a node, and the weight matrix is constructed by $w_{s,ij} = w_{s,ji} = \exp(-\beta_s \|x_i - x_j\|^2)$ for each (i, j) -th element with $\beta = 5000$. Furthermore, the segmentation related

parameter is set as $\gamma = 1$. Performance evaluation is reported using the average F-1 score over the test images. Following is the definition of F_1 score.

$$Precision = \frac{\#TP}{\#(TP + FP)}$$

$$Recall = \frac{\#TP}{\#(TP + FN)}$$

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

where, $\#TP$ is the number of True Positives, $\#FN$ is the number of False Negatives and $\#FP$ is the number of False Positives. Table 2.1 provides an comparison of our segmentation results to the GMM-based method (referred to as Soares) [66] as well as FRVD [67], OOF [68], Frangi [69], and NN [70], five state-of-the-art methods for retinal blood vessel segmentation. The method of Soares [66] is based on Gabor wavelet and GMM models, and is the closest to our segmentation module. FRVD [67] utilizes a wavelet-based thresholding and edge refinement scheme, meanwhile, OOF or optimally oriented flux method is widely known for its ability to enhance and detect irregular filamentary structures and usually outperforms the traditional Hessian based methods. Frangi [69] is a local Hessian based method, and NN [70] uses neural network together with gray-level and moment invariants-based features. The results of the five comparison methods are reported by running their original source codes with parameters tuned for best performance. Empirically our segmentation module outperforms the comparison methods on both datasets of neurons and retinal vessels. This we believe could be due to the introduction of graph Laplacian based regularization which helps noticeably in providing smoother and cleaner segmentation results,

as is visually illustrated in Fig. 2.2.

	Ours	Soares [66]	FRVD [67]	OOF [68]	Frangi [69]	NN [70]
NeuB1	0.76	0.74	0.69	0.51	0.59	-
NeuB2	0.80	0.77	0.65	0.59	0.73	-
DRIVE	0.78	0.77	0.71	0.73	0.56	0.77
STARE	0.77	0.76	0.64	0.69	0.57	0.75

Table 2.1: Comparison of segmentation performance w.r.t. F1 score. See text for details.

2.2 Segmentation Result $\rightarrow$ Skeleton Map

Applying the segmentation step on an input image provides us the filamentary foreground segmentation. This is then followed by medial axis transform and postprocessing to produce a skeleton map of 1-pixel thickness. The obtained skeleton pixels can be divided into the following categories:

- *Root pixels* – For neuronal tracing, the root pixels are the DAPI-tagged nuclei pixels; For retinal vessel tracing, the root pixels are the pixels touching the boundary of the optical disk.
- *Body pixels* – Pixels with two neighbours.
- *Junction pixels* – Pixels with three or more neighbours.
- *Terminal pixels* – Pixels with only one neighbour and are not the root pixels.

Furthermore, a *filament* in our context refers to a set of connected body/root/terminal pixels, and a *junction* is the set of connected junction pixels, as illustrated in Fig. 2.3 in red and green, respectively. In this thesis, a filament is also interchangeably referred to as a filamentary segment or simply a segment.

Figure 2.4: Running example of digraph preparation process in the tracing step. (a) Original image. (b) Segmentation result. (c) Skeleton map, with (d) being a fragment inside the green box. (e) Undirected graph G_U . (f) Digraph G .

Figure 2.3: Example of filament (in red) and junction (in green).

2.3 Constructing the Digraph

Now I start to construct the digraph $G = (V, \mathcal{E}, W)$, with V , $\mathcal{E}$, W being the node set, edge set, and weight matrix, respectively. Each node in the graph corresponds to a *filament*; There is an edge between two nodes if these two filaments are directly connected in the skeleton map via a *junction*; A *Root node* is a filament containing root pixels, while a *Terminal node* is a filament containing terminal pixels. This gives rise to essentially an undirect graph $G_U = (V_U, E_U)$, from which our digraph representation G can be obtained by the following intuitive rules:

1. A root node possesses only outgoing edges, while a terminal node contains only incoming edges.
2. If a k -clique ($k \geq 3$) of G_U has more than $k - 2$ terminal nodes, then the edge among these terminal nodes will be removed. For example, if a 4-clique has three terminal nodes, then the edge between those three nodes will be removed.
3. In a 3-clique of nodes i, j and k , if the nodes j and k has outgoing connections to rest of the graph other than i , then (i, j) and (i, k) edge will be $i \rightarrow j$ and $i \rightarrow k$ respectively. The edge (j, k) will be removed.
4. All remaining undirected edges are converted to a pair of directed edges with opposite directions.

Fig. 2.4 illustrates an example of executing this digraph construction stage. After filamentary foregrounds (b) are obtained from input image (a), its skeleton map is obtained (c), and for a zoomed-in crop of the filamentary foregrounds, its undirected graph G_U and finally the digraph representation G is constructed,

where the root nodes are denoted as R1 and R2, respectively. The clique of nodes $\{1, 2, 3, 4\}$ in G_U of (e) is converted to its corresponding directed subgraph in (f) marked within a dotted boundary, obtained by applying rule 2. A synthetic example of rule 3 is shown in Fig. 2.5. Note the introduction of the intermediate undirected graph G_U here is to make the process intuitive and easy to understand. In practice, the digraph representation G as in Fig. 2.4(f) can be directly obtained using the skeleton map as in Fig. 2.4(d).

Figure 2.5: Synthetic example of rule 3.

Chapter 3

Tracing with Matrix Forest

Theorem on a Directed Graph

In this chapter the tracing problem is formulated as a label propagation problem on a weighted digraph using Matrix Forest Theorem. According to the Matrix Forest Theorem, the normalized conductance of a rooted spanning converging forest of a digraph is defined by $(I + \alpha L)^{-1}$, where α is a parameter and L is a graph laplacian matrix of the digraph. For tracing problem, this is used as a similarity score between the labeled and unlabeled nodes. If an unlabeled node is more similar to a particular labeled node than others, then it can be assigned the same class label as that labeled node. Four different variants of this method and an interesting connection between this method and random walk based label propagation method is also established.

3.1 Introduction

Very few label propagation methods are specifically dedicated to digraphs, including notably [71, 72]. The most related work is that of [71], which generalizes their earlier framework of regularized risk minimization on undirected graphs [73] to digraphs by discriminatively normalizing in-links and out-links, as well as adopting the symmetrized graph Laplacian of [74]. One major difficulty in learning with digraphs lies in the asymmetric nature of the weight matrix introduced by these directed edges or links. This is often regarded as cumbersome when aligning with the key concepts developed for undirected graphs that are symmetric by nature, such as graph Laplacians. It thus leads to the widely adopted symmetrization trick in e.g. the construction of graph Laplacians [74], or co-link similarity matrices [75]. Unfortunately important information conveyed by edge directions are lost. In contrast, my approach directly works with asymmetric matrices, which is the key in preserving edge directions. In addition, the construction in both methods [71, 72] rely on *strongly connected* digraphs, that is, there always exists a directed path from any node to any other node of the graph. Since in practice the digraphs are usually not necessarily strongly connected, a teleporting random walk trick (e.g [76]) is adopted by inserting bi-directional edges between *all* node pairs with a *tiny* weight. The resulting method thus works with non-sparse matrices, instead my algorithm works with a much broader set of digraphs called *weakly connected* digraphs, i.e. for an arbitrary partitioning of the digraph node set into two subsets, there is at least an edge connecting them regardless of the edge direction. This enable my algorithm to well-preserve the sparse graph structures and edge directional information of the input, which I believe is an important advan-

tage over existing label propagation methods on graphs (being either digraphs or undirected graphs). This hypothesis has indeed been empirically validated through extensive experiments in Chapter-7 where my label propagation method consistently outperforms existing label propagation methods when operating on the same ground.

3.2 Problem Set-up

Given this digraph G , denote the out-degree of each node v_i as $d_i^+ = \sum_{j=1}^n w_{ij}$. Further, define the out-degree matrix as $D = \text{diag}(d_1^+, \dots, d_n^+)$, or in matrix forms, $D = \text{diag}(W\mathbf{e}_n)$. Denote the digraph Laplacian $L = D - W$. A row-stochastic transition probability matrix $P = [p_{ij}]$ can be constructed as $p_{ij} = w_{ij}/d_i^+$, or equivalently as $P = D^{-1}W$. Note that undirected graphs can be regarded as special digraphs characterized algebraically by their symmetric weight matrix W , i.e. the symmetric pair w_{ij} & w_{ji} correspond to bi-directional edges with equal weights. The focus here on a transductive inference scenario where labels from the set of few labeled nodes V_l are to be propagated to the rest unlabeled nodes V_u , with $V = V_l \cup V_u$. These labels are of multiclass nature, where each class corresponds to an individual neuron or vessel tree. To simplify the notations I assume V_l contains the first l nodes, $V_l = \{v_1, \dots, v_l\}$. To accommodate label information, define a label matrix $Y = [y_{ik}]$ of size $n \times K$ (assuming there are K class labels available), with each entry y_{ik} containing 1 if node i belongs to V_l and is labeled with class k , and 0 otherwise. Also define the length n ground-truth label vector $\mathbf{y}$ that contains two disjoint parts $\mathbf{y}_l$ and $\mathbf{y}_u$: $\mathbf{y}_l$ is the input label vector of length l over the set of labeled nodes, with each entry

y_i for the input class assignment of node $v_i \in V_l$; $\mathbf{y}_u$ is the hold-out ground-truth label for the unlabeled nodes, i.e. a vector of length $n - l$. Similarly, define the initial label vector $\hat{\mathbf{y}}$ containing also two parts, $\hat{\mathbf{y}}_l := \mathbf{y}_l$ and $\hat{\mathbf{y}}_u = \mathbf{0}$, where $\mathbf{0}$ is an all-zero vector of length $n - l$. Define the prediction vector $\mathbf{y}^*$ with also two parts $\mathbf{y}_l^* := \mathbf{y}_l$, as well as $\mathbf{y}_u^*$ of length $n - l$, containing the prediction results, where each y_i^* denotes the predicted class assignment for a node $v_i \in V_u$.

3.3 Matrix Forest Theorem on A Digraph

The proposed label propagation algorithm (shown in Algorithm 1 and referred to as MFTD) is derived based on matrix-forest theorem [77] of algebraic digraph theory [78] as follows. The forest matrix is defined as

$$S_1 = (I + \alpha L)^{-1}, \quad (3.1)$$

a normalized forest matrix where each (i, j) -th entry denotes the number of spanning trees rooted at node i that also include the j -th node, as in Theorem 4 of [77]. In a way it can be viewed as a generalization of the celebrated matrix-tree theorem (e.g. [79]) from undirected graphs to digraphs. Let $w_{\max}$ denote the entry in W containing the strongest signal, i.e. $w_{\max} = \max_{i,j} |w_{ij}|$. By varying the preprocessing schemes of normalizing W , I have two variants: MFTD_a starts with a preprocessing effort to normalize W , $W \leftarrow \frac{W}{w_{\max}}$; MFTD_b considers a different normalization of W as $W \leftarrow D^{-1}W$ instead.

Proposition 1. Under normalization scheme of $W \leftarrow D^{-1}W$, the forest matrix becomes $S_1 = (1 - \tau)(I - \tau P)^{-1}$, with $\tau = \frac{\alpha}{1+\alpha}$.

Algorithm 1 Label Propagation by Matrix-Forest Theorem of Digraphs (MFTD)

Input: A digraph $G = (V, E, W)$, label information Y , $\mathbf{y}_l$, and $\alpha \in (0, 1)$.

Output: $\mathbf{y}_u^*$

1. Compute the out-degree matrix D .
 2. Compute the affinity matrix by $A = SY$ and (3.1) (or (3.2)).
 3. Predict $\mathbf{y}_u^*$: Compute the i -th entry by (4.5), for any unlabeled node $v_i \in V_u$.
-

Proof. By normalization we know $P = W$. Apply the fact $L = D - W = I - P$, we have $I + \alpha L = (1 + \alpha)(I - \frac{\alpha}{1+\alpha}P) = \frac{1}{1-\tau}(I - \tau P)$. $\square$

Further, let $\tilde{L} := \lim_{\alpha \rightarrow \infty} (I + \alpha L)^{-1}$, which is a matrix of normalized rooted spanning forests. Both S_1 and $\tilde{L}$ has few interesting properties [80]: Each entry of both matrices is non-negative, and both matrices are row-stochastic; $\tilde{L}$ resides in the null space of digraph Laplacian L , as $L\tilde{L} = \tilde{L}L = 0$; $\text{rank}(L) = n - \text{rank}(\tilde{L})$; $L + \beta\tilde{L}$ is non-singular for any $\beta > 0$, and is the “complementary perturbation of L ” [81]. Indeed, this brings forward the second forest matrix,

$$S_2 = (L + \beta\tilde{L})^{-1}, \quad (3.2)$$

which is also termed the matrix of dense forest in [77]. When the same preprocessing schemes of normalizing W are applied to the forest matrix S_2 , two additional variants are similarly obtained, and are denoted as MFTD_c & MFTD_d, respectively.

3.4 Inference Using Matrix Forest Theorem on A Digraph

One can interpret the (i, j) -th entry s_{ij} of the forest matrix S (being either $S := S_1$ or $S := S_2$) as quantifying the accessibility of a particle from a node v_i to visit node v_j along the digraph structure. This provides a notion of affinity from state i to j . Now define the affinity matrix as

$$A = SY, \tag{3.3}$$

a matrix of size $n \times K$, with each entry a_{ik} being associated with an affinity score of state i belonging to class k . To infer $\mathbf{y}_u^*$ of the unlabeled states V_u , my algorithm predicts each entry's class assignment by identifying a label with the largest affinity score,

$$\mathbf{y}_i^* = \arg \max_k a_{ik}, \quad \forall v_i \in V_u. \tag{3.4}$$

Now I have all the ingredients ready to present the digraph-theoretical label propagation algorithm, MFTD, as described in Algorithm 2. With a slight abuse of notation, MFTD is also used to refer to the entire tracing system.

3.5 Computational complexity

As the forest matrix S (being either $S := S_1$ or $S := S_2$) is invertible, denote its inverse as $E := S^{-1}$. Usually it is computationally more efficient to evaluate E than S , since $E = I + \alpha L$ or $E = L + \beta \tilde{L}$. Let us investigate the complexity

of Algorithm 2, which is clearly dominated by the cost of computing the affinity matrix A in (4.4). In fact, it can be accomplished by instead solving the linear system of

$$EA = Y,$$

thus eliminating the need to explicitly compute any matrix inverse. For a general dense matrix E , the computational time is $O(n^3 + n^2K)$. This is about the same complexity of [71], one of the main competing methods. Fortunately, E (as well as S) is usually a sparse matrix in my context, which can be exploited to reduce the computational time. There are many efficient solvers for large sparse linear systems, including both direct [82, 83] and iterative methods [84, 85]. In my implementation, I have adopted the direct solver UMFPACK [82] which exists as a built-in routine (for LU, backslash, and forward slash functions) in MATLAB. The specific complexity depends on the size (n), the number of non-zero entries and the sparsity pattern of E , which remains a challenging task to provide an explicit complexity measure dedicated to my context.

3.6 PAC-Bayesian Label Propagation Bound

A PAC-Bayesian error bound is provided here for our MFTD algorithm (both (3.1) and (3.2)). This bound is an adaptation of the PAC-Bayesian bound for general transductive learning developed in [86], which improves the PAC-Bayesian bound of [87].

In the transductive setting, a learning method is given the full sample $V = V_l \cup V_u$ and labels of the labelled data (i.e., y_i for $v_i \in V_l$), and learns a classifier $h : V \rightarrow \mathcal{Y} := \{1, 2, \dots, K\}$ that correctly classifies the unlabelled data

in V_u . The hypothesis space of classifiers determined by the learning method is denoted as $\mathcal{H}$. For any classifier $h \in \mathcal{H}$, we define the *test error* as $\mathcal{L}_{l,n}(h) := \frac{1}{n-l} \sum_{i=l+1}^n \ell(h(v_i), y_i)$ with respect to the 0/1 loss function ℓ satisfying $\ell(h(v_i), y_i) = 1$ if $h(v_i) \neq y_i$ and $\ell(h(v_i), y_i) = 0$ otherwise. Similarly, we define the *empirical error* of h as $\hat{\mathcal{L}}_{l,n}(h) := \frac{1}{l} \sum_{i=1}^l \ell(h_i, y_i)$, the *full sample error* of h as $\mathcal{L}_n(h) := \frac{1}{n} \sum_{i=1}^n \ell(h_i, y_i)$. Therefore, we have

$$\mathcal{L}_n(h) = \frac{1}{n} \left(l \hat{\mathcal{L}}_{l,n}(h) + (n-l) \mathcal{L}_{l,n}(h) \right). \quad (3.5)$$

In the error bound presented in the next theorem, we show that the test error can be upper bounded by the empirical error plus some complexity term.

Theorem 2. For any deterministic classifier h determined by our Algorithm 2 and any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over random draws of V_l from V , the following bound holds

$$\begin{aligned} \mathcal{L}_{l,n}(h) &\leq \hat{\mathcal{L}}_{l,n}(h) \\ &\quad + \sqrt{\frac{1}{2(1 - \frac{l}{n})} \left(\ln \frac{n}{l} + \frac{1}{l} \ln \frac{C(l, n)}{\delta} + \ln(Ke) \right)}, \end{aligned} \quad (3.6)$$

where $C(l, n) = (\sqrt{2} \ln l + 8) \sqrt{l(1 - \frac{l}{n})}$.

Proof. Our proof is based on the PAC-Bayesian bound for *Gibbs classifier* $h_{\mathbf{q}}$ presented in Corollary 7 of [86]. For a given distribution $\mathbf{q}$ on $\mathcal{H}$ and a sample v , $h_{\mathbf{q}}$ randomly draws a classifier h according distribution $\mathbf{q}$ and classifies v as $h(v)$. The test error of $h_{\mathbf{q}}$ is defined as $\mathcal{L}_{l,n}(h_{\mathbf{q}}) := \frac{1}{n-l} \sum_{i=l+1}^n \mathbb{E}_{h \sim \mathbf{q}} \ell(h(v_i), y_i)$. The empirical error $\hat{\mathcal{L}}_{l,n}(h_{\mathbf{q}})$ and full sample error $\mathcal{L}_n(h_{\mathbf{q}})$ are similarly defined.

An adaptation of Corollary 7 of [86] leads to the following result: For any *prior distribution* $\mathbf{p}$ on $\mathcal{H}$ and any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over random draws of V_l from V , we have

$$\mathcal{L}_n(h_{\mathbf{q}}) \leq \hat{\mathcal{L}}_{l,n}(h_{\mathbf{q}}) + \sqrt{\frac{1 - \frac{l}{n}}{2l} \left(KL(\mathbf{q}||\mathbf{p}) + \ln \frac{C(l, n)}{\delta} \right)}, \quad (3.7)$$

where $KL(\mathbf{q}||\mathbf{p})$ is the Kullback-Leibler divergence between two distributions and $C(l, n)$ is defined as in (4.14).

For any deterministic classifier $h \in \mathcal{H}$, we can derive an error bound for h by choosing the posterior distribution $\mathbf{q}$ as the one which assigns probability 1 to h . In this way, all errors of $h_{\mathbf{q}}$ reduce to the corresponding errors of h and the KL-divergence term $KL(\mathbf{q}||\mathbf{p})$ reduces to $\ln(1/\mathbf{p}(h))$, which implies that

$$\mathcal{L}_n(h) \leq \hat{\mathcal{L}}_{l,n}(h) + \sqrt{\frac{1 - \frac{l}{n}}{2l} \left(\ln \frac{1}{\mathbf{p}(h)} + \ln \frac{C(l, n)}{\delta} \right)}. \quad (3.8)$$

Now, we propose a technique for selecting a prior distribution $\mathbf{p}$ over $\mathcal{H}$ based on the full sample V . It is noted that Algorithm 2 is deterministic and generates only one classifier h once V_l and the corresponding labels are given. Since there are $\binom{n}{l}$ ways to draw V_l from V and there are K possible labels for each sample in V_l , it implies that $\mathcal{H}$ has $\binom{n}{l} K^l$ hypotheses. That is, $|\mathcal{H}| = \binom{n}{l} K^l$. We then define the prior distribution $\mathbf{p}$ to be an uniform distribution over $\mathcal{H}$, or equivalently, $\mathbf{p}(h) = \frac{1}{\binom{n}{l} K^l}$ for any $h \in \mathcal{H}$. Substituting $\mathbf{p}(h)$ into (3.8) and using inequality $\binom{n}{l} \leq (en/l)^l$ leads to

$$\mathcal{L}_n(h) \leq \hat{\mathcal{L}}_{l,n}(h) + \sqrt{\frac{1 - \frac{l}{n}}{2} \left(\ln \frac{n}{l} + \frac{1}{l} \ln \frac{C(l, n)}{\delta} + \ln(Ke) \right)}. \quad (3.9)$$

In addition, we have for any constant c that $\mathcal{L}_n(h) \leq \hat{\mathcal{L}}_{l,n}(h) + c$ is equivalent to $\mathcal{L}_{l,n}(h) \leq \hat{\mathcal{L}}_{l,n}(h) + \frac{1}{1-\frac{l}{n}}c$, which together with inequality (3.9) leads to the desired result. $\square$

Notice that when $\beta_l \leq l/n \leq \beta_u$ for any constant β_l, β_u satisfying $0 < \beta_l \leq \beta_u < 1$ (e.g., $l/n = \beta_0$ the number of labelled sample is a constant proportion of the number of full sample), the complexity term (i.e., the second term on the right-hand side of (4.14)) converges to $\sqrt{\frac{\ln(\frac{Ke}{\beta_l})}{2(1-\beta_u)}}$ as $n \rightarrow \infty$, which means that in such a case the test error $\mathcal{L}_{l,n}(h)$ will not exceed the empirical error $\hat{\mathcal{L}}_{l,n}(h)$ by a constant value.

3.7 Finding the Weight Matrix $W = [w_{ij}]$

The weight matrix $W = [w_{ij}]$ is computed as follows: w_{ij} is set to 0 if the i -th and j -th filaments involved are not contacting (i.e. immediately connected via a junction). Due to the edge sparsity nature of my digraph representation, majority of the matrix elements are therefore 0. A weight matrix element may attain a positive values only if the two filaments are contacting, which can be obtained based on the angle θ_{ij} between these two filaments as

$$w_{ij} = \exp\left(-f(\theta_{ij})\right).$$

Note θ_{ij} is in the range of $[0, \pi]$, and the induced energy function takes the form

$$f(\theta_{ij}) = \begin{cases} -\frac{\sin(\theta_{ij})}{k} & \text{if } \theta_{ij} \in [0, \theta_c) \\ -\frac{\sin(\theta_c)}{k} & \text{if } \theta_{ij} \in [\theta_c, \arccos(\frac{-\sin \theta_c}{k^2}) \\ k \cos(\theta_{ij}) & \text{if } \theta_{ij} \in [\arccos(\frac{-\sin \theta_c}{k^2}), \pi]. \end{cases}$$

It is worth noting that each element in the weight matrix is designed to capture the following intuition: For two consecutive filaments i and j belonging to the same vessel tree or neuron, they should not bend too much. In other words, the angle between these two filaments, θ_{ij} , tends to be close to π . In the context of this thesis it amounts to endow its corresponding w_{ij} a higher weight value, and the other way around if there is an abrupt bend (i.e. a tiny weight value if θ_{ij} is an acute angle). The meaning of the two tuning constants, θ_c and k , are illustrated in Fig. 6.3.1 for both the energy function f and the weight w_{ij} : θ_c marks the turning point in the function f , while k brings in the weight difference between highest and the lowest values. Throughout empirical experiments, θ_c and k are fixed to $\pi/7$ and 5, respectively.

Figure 3.1: Plot for the energy function $f(\theta_{ij})$ and the corresponding weight w_{ij} , both as functions of θ_{ij} , the angle between the i -th and j -th filaments.

3.8 Summary

In this chapter an interesting graph theoretical method called Matrix Forest Theorem is exploited for tracing applications on a digraph. We have also proposed four different variants of this method and one interesting connection between Matrix Forest Theorem and Random Walk based method is also established. In Chapter-7 through empirical evaluation as well as optimal parameter selection of all these methods are investigated.

3.9 Appendix

Proof of Proposition 1

Following [88], let $L := D - W$, and denote the forest matrix $\mathcal{Q} := (I + \tau L)^{-1}$. Suppose $\mathcal{F}$ is the set of all spanning forests of digraph G and $\mathcal{F}^{ij}$ the set of spanning forests containing edge $i \leftarrow j$. It has been shown in [88, 89] that each

(i, j) -th entry of $\mathcal{Q} = [q_{ij}]$ concerns the normalized counts of spanning out-forests, as $q_{ij} = \frac{\epsilon(\mathcal{F}^{ji})}{\epsilon(\mathcal{F})}$, with $\epsilon(\cdot)$ denoting the count of specified forests.

Further, denote $P^T = I - \mu L$, $\mu > 0$, and let $q = \left(1 + \frac{\tau}{\mu}\right)^{-1}$, it can be easily shown that

$$\mathcal{Q} = q \left(I - \frac{\tau}{\mu + \tau} P^T \right)^{-1}.$$

In other words, $\mathcal{Q}$ is a scaled version of $(I - \alpha P^T)^{-1}$, with $\alpha := \frac{\tau}{\mu + \tau}$.

Chapter 4

Tracing with Absorbing Markov Chain on a Directed Graph

In this chapter the tracing problem is formulated as a label propagation problem on a weighted digraph using random walk on an Absorbing Markov Chain (AMC). After converting our original graph into an AMC, we can compute its Fundamental Matrix, which gives us the expected number of visits from one node to another before absorption. For tracing problem, this is another kind of similarity score between the labeled and unlabeled nodes. Class labels for the unlabeled nodes can be assigned using the similar approach as the previous one. In addition, an optimal one-step increment/decrement update (aka online update) is introduced, which is handy in scenarios where local changes are made to graphs over time, or prediction are made on out-of-sample instances. The proposed algorithm also bears interesting connections to existing works including undirected graph Laplacian, diffusion graph kernels, spanning forest of graphs, graph-based proximity measure, among others. Few interesting properties of the Fundamental

Matrix are also studied in this chapter.

4.1 Introduction

We focus on the following graph transduction problem: Given a directed graph with certain nodes labeled, make predictions on the rest unlabeled nodes by propagating the class labels following the underlining directed graph structure. Different from undirected graphs that delineate symmetric weight between adjacent nodes, in directed graphs, edge or link directions are well preserved in its weight matrix. This information is particularly useful in many real life applications that can be naturally characterized using directed graphs, including automated delineation of filamentary structures in images and videos [90, 91, 92, 93, 94], image classification [72] and clustering [95], network & link analysis in hyperlinks of webpages as well as citations of papers [96, 97, 75, 98], among others. Moreover, the fast pace of big data production and storage, together with the scarcity of annotated labels, also create the need for algorithms capable of scaling up to make predictions on large-scale directed graphs with few known labels.

Since being introduced by Vapnik in the nineties, many research efforts have already been devoted to graph-based transduction or transductive learning, as can be found in the comprehensive reviews of [99, 100]. Most of the existing literature work with undirected graphs. For example, the harmonic functions on Gaussian fields [101], the local and global consistency of [73] the quadratic criterion [102], the commute time kernel [98], the partially absorbing random walks [103], and the greedy max-cut [104]. Among these methods, a few has been specifically dedicated to directed graphs, including in particular [71, 105, 72]. One major

difficulty in learning with directed graphs lies in the asymmetric nature of the weight matrix introduced by these directed edges or links. This is often regarded as cumbersome when aligning with the key concepts developed for undirected graphs that are symmetric by nature, such as graph Laplacians. It thus leads to the widely adopted symmetrization trick in e.g. the construction of graph Laplacians [71], or co-link similarity matrices [75], or covariance kernel [105]. Unfortunately important information conveyed by edge directions are still lost. There have been a few methods for large-scale transduction such as [106, 107], which are however not ready to be used for directed graphs.

We propose a random walk on absorbing Markov chains approach to the problem of transductive learning on directed graphs, where the edge directions – the key aspect of directed graphs, are well preserved. Our algorithm is intuitive to understand, easy to implement, and by working with absorbing Markov chains [108], the sparse nature of the graph structure is also retained, which would be important in the context of predictions on large-scale directed graphs.

The most related work is that of [71], which generalizes their earlier framework of regularized risk minimization on undirected graphs [73] to directed graphs by discriminatively normalizing in-links and out-links, as well as adopting the directed graph Laplacian of [74]. The method of [71] involves symmetric matrices due to the construction of digraph Laplacian [74], where the symmetrization trick is utilized. In contrast, we directly work with asymmetric matrices, which is the key in preserving edge directions. In addition, the construction in [71] relies on an irreducible Markov chain, which however only apply to *strongly connected* directed graphs, that is, there is a directed path from any node to any other node of the graph. Since in practice the directed graphs are usually not necessar-

ily strongly connected, a teleporting random walk trick (e.g [76]) is adopted by inserting bi-directional edges between all node pairs with an equal weight. The resulting method thus works only with non-sparse matrices, instead our algorithm is able to preserve sparse graph structures and edge directional information of the input. Besides, our algorithm is able to efficiently work with large-scale graphs that might be too big to be considered by [71]. More recently, the family of partially absorbing random walks (PARW) is proposed in [103], which can be formulated as a special case of the absorbing Markov chains considered in our approach. Note absorbing random walks have been considered by [109] for an entirely different problem in natural language processing.

4.2 Basics

Definition 4.2.1. An *Absorbing Markov Chain* is a Markov chain in which every state can reach an absorbing state. An *Absorbing State* is a state that, once entered, cannot be left.

4.3 Problem Set-up

Let $G = (V, \mathcal{E}, W)$ denote a directed graph or digraph, where $V = \{v_1, v_2, \dots, v_n\}$ is the set of nodes, E is the set of directed edges each connecting two adjacent nodes, and $W = [w_{ij}] \in \mathbb{R}^{n \times n}$ is the asymmetric non-negative matrix with $w_{ij} \geq 0$ denoting the weight associated with the directed edge from v_i to v_j . The in-degree of each node v_j is computed as $d_j^- = \sum_{i=1}^n w_{ij}$, and define in-degree matrix $D = \text{diag}(d_1^-, \dots, d_n^-)$. A column stochastic transition probability matrix,

$P = [p_{ij}]$, can be constructed as $p_{ij} = \frac{w_{ij}}{d_j}$, or equivalently as $P = WD^{-1}$. An important remark is that random walks on an undirected graph can be regarded as a special case in our context, since in its weight matrix W , the symmetric pair w_{ij} and w_{ji} correspond to bi-directional edges with the same weights – which can be characterized by a reversible Markov chain. In fact, as e.g. illustrated in Figure 4.1(a), *loops*, *self-loops*, and *bi-directional* edges (i.e. two edges between adjacent nodes), as well as mixed graphs (of directed and undirected edges) are all within the scope of our digraph definition. In this chapter, we focus on a transductive inference scenario where labels from the set of few labeled nodes V_l are to be propagated to the rest nodes, namely the set of unlabeled nodes V_u , with $V = V_l \cup V_u$. The labels here could be binary or multiclass. To simplify the notation we assume V_l contains the first l nodes, $V_l = \{v_1, \dots, v_l\}$. To accommodate label information, define a label matrix Y of size $n \times K$ (assuming there are K class labels available), with each entry Y_{ik} containing 1 provided the node i belongs to V_l and is labeled with class k , and 0 otherwise. Besides, define the ground-truth label vector $\mathbf{y}$, a vector of length n including two disjoint parts $\mathbf{y}_l$ and $\mathbf{y}_u$: $\mathbf{y}_l$ is the input label vector of length l over the set of labeled nodes, with each entry y_i for the input class assignment of node $v_i \in V_l$; $\mathbf{y}_u$ is the hold-out ground-truth label for the unlabeled nodes, i.e. a vector of length $n-l$. Similarly, define the initial label vector $\hat{\mathbf{y}}$ containing also two parts, $\hat{\mathbf{y}}_l := \mathbf{y}_l$ and $\hat{\mathbf{y}}_u = \mathbf{0}$, where $\mathbf{0}$ is an all zero vector of length $n-l$. Define the prediction vector $\mathbf{y}^*$ with also two parts $\mathbf{y}_l^* := \mathbf{y}_l$, as well as $\mathbf{y}_u^*$ of length $n-l$, containing the prediction results, where each y_i^* denotes the predicted class assignment for a node $v_i \in V_u$.

Figure 4.1: (a) An illustrative example of a weighted digraph-based transduction setting: Two different class labels are to be propagated from the labeled nodes (the green and the red nodes, each for one class) to the rest nodes following the graph structure. Here only a subset of the graph nodes and edges are displayed. Quantities such as W and P can be computed accordingly. (b) As some nodes might have zero in-degree (i.e. source nodes), a new node is further added with directed edges to every nodes including itself, which gives $\tilde{P}$. According to the $\mathbf{c}$ vector, the edges toward those previous source nodes are weighted by 1, and the rest are $1 - \alpha$. (c) Its transpose, $\tilde{Q}$, corresponds to the same graph but with edge directions being reversed. This facilitates the evaluation of affinity scores flowing from unlabeled nodes (e.g. leaf nodes) to the labeled nodes (e.g. source nodes).

4.4 Absorbing Markov Chain and its Fundamental Matrix

There however exists an issue: the graph often contains source nodes (i.e. nodes with $d_j^- = 0$ ¹), for which the corresponding columns in P are zero vectors – then P is not stochastic anymore. Inspired by the PageRank method used in Google search engine [76], we consider an augmented graph to ameliorate this situation: One extra node v_{n+1} is introduced that is further connected to each of the source nodes V_l with an equal weight of 1 and connected to the rest nodes with an equal weight of $(1 - \alpha)$ where $\alpha \in (0, 1)$. A self-connecting

¹Usually source nodes are within V_l , i.e. they are labeled nodes.

edge $e = (v_{n+1}, v_{n+1})$ with a transition probability 1 is further imposed on this new node v_{n+1} , to save itself from being another source node. This leads to a digraph as displayed in Figure 4.1(b), which also corresponds to an augmented column-stochastic transition probability matrix

$$\tilde{P} = \left(\begin{array}{c|c} \alpha P & 0 \\ \hline \mathbf{c} & 1 \end{array} \right) \in \mathbb{R}^{(n+1) \times (n+1)},$$

where $\mathbf{c} \in \mathbb{R}^{1 \times n}$ is a vector with the elements corresponding to source nodes being 1 and the rest elements being $1 - \alpha$. The reason of introducing α is as follows: Each transient node has to secure a positive value in $\mathbf{c}$ to ensure the final absorption into v_{n+1} for a valid absorbing Markov chain. This naturally introduces α to down-weight P to αP .

Let us pause for a moment and recall that in our context, labels of the source nodes (if there are any) are usually known and are to be propagated to the rest nodes including the leaf nodes. Edges of the input graph however flows from source to leaf nodes, as exemplified in Figure 4.1(b) for $\tilde{P}$. It is more informative to *reverse* all the edge directions, to compute instead the affinity of each unlabeled node toward the labeled nodes (where many are source nodes). As presented in Figure 4.1(c), algebraically this corresponds to the matrix *transpose*, $\tilde{Q} := \tilde{P}^T$. Surprisingly, now this row-stochastic transition probability matrix

$$\tilde{Q} = \left(\begin{array}{c|c} \alpha P^T & \mathbf{c}^T \\ \hline 0 & 1 \end{array} \right) = [\tilde{q}_{ij}] \quad (4.1)$$

defines an *absorbing Markov chain* on the augmented digraph, $\tilde{G}$. Considering

random walks in the Markov chain theory [108], each node v_i of the digraph is equivalent to a Markov chain state s_i , and collectively, the set of nodes naturally identifies a set of states $S = \{s_1, s_2, \dots, s_{n+1}\}$ in the Markov chain induced by the graph $\tilde{G}$ of $n + 1$ nodes. In what follows node v and state s are sometimes used interchangeably. In addition, s_{n+1} is the only absorbing state, while $S_t = \{s_1, s_2, \dots, s_n\}$ denotes the set of transient or non-absorbing states connecting to s_{n+1} by at least one path. Here each entry $\tilde{q}_{ij}$ with $s_i, s_j \in S_t$ denotes the transition probability from state s_i to state s_j . It is also known that within a finite number of steps, a particle in state s_i moving randomly by $\tilde{Q}$ will be absorbed into s_{n+1} with probability 1.

Let us inspect further the upper left block of the matrix $\tilde{Q}$ in (4.1), denote $Q = \alpha P^T = [q_{ij}]$, and I an identity matrix, both of size $n \times n$. From Markov chain theory [108] we know every element of $Q^t = \underbrace{Q \dots Q}_t$ denotes the probability of a particle starting from s_i to visit s_j in t steps. The expected number of visits from s_i to s_j ($s_i \rightarrow s_j$) in t steps is $e_t(s_i \rightarrow s_j) = \sum_{k=0}^t q_{ij}^{(k)}$, or in its matrix form

$$E_t = I + Q + Q^2 + \dots + Q^t. \quad (4.2)$$

Proposition 3. The fundamental matrix of the absorbing Markov chain $\tilde{Q}$ is

$$E = (I - \alpha P^T)^{-1} = [e_{ij}]. \quad (4.3)$$

The detailed proof is described in Appendix 4.10. In what follows, we present a transductive learning algorithm based on the fundamental matrix of the above

absorbing Markov chain $\tilde{Q}$.

4.5 Inference on Absorbing Markov Chain

An important fact [108] about the fundamental matrix E of our Markov chain $\tilde{Q}$ is that its (i, j) -th entry e_{ij} provides the *expected number of times* a particle from a transient state s_i visits the transient state s_j . This provides a notion of affinity from state i to j . The intuition is, if a state j is close to the initial state i in terms of graph structure, it will be visited by the particle more often than if it is far away from initial state (We visit our close relatives more often than our distant ones). Now define the affinity matrix as

$$A = EY = (I - \alpha P^T)^{-1}Y. \quad (4.4)$$

It is a matrix of size $n \times K$, with each entry a_{ik} being associated with an affinity score of state i belonging to class k . In other words, it is the accumulated expected number of visits from state v_i to those states in V_l that are labeled with class k . Here $\alpha \in (0, 1)$ acts as a parameter which controls how long the random walker stays among the transient states before it gets absorbed. If α is closer to 1 then the random walker stays for a longer period of time before getting absorbed, and vice versa. Empirically it is observed that our approach is insensitive to varying α values to anything between .01 and .99. We set $\alpha = 0.1$ throughout the experiments. To infer $\mathbf{y}_u^*$ of the unlabeled states V_u , our algorithm predicts each entry's class assignment by identifying a label with the largest affinity score,

namely

$$y_i^* = \arg \max_k a_{ik}, \quad \forall v_i \in V_u. \quad (4.5)$$

Multi-label Inference With slight modification our approach is also able to work with multi-label classification. That is, starting with a few nodes of the input graph being labeled, to predict multiple target labels for each of the rest nodes. Instead of Y , we consider a matrix $\tilde{Y}$ of size $n \times \tilde{K}$ for the input label matrix. Here, each column of $\tilde{Y}$ is for one of the $\tilde{K}$ labels, and each entry contains $\tilde{Y}_{ik} = 1$ if the i -th instance is positive for label k , -1 if it is a negative instance, or 0 if it is unlabeled. Instead of $\mathbf{y}^*$, define $\tilde{Y}^*$ the prediction matrix of size $n \times \tilde{K}$. To infer the row vectors $\tilde{Y}_u^*$ of the unlabeled states V_u from the incurred affinity matrix $A = [a_{ik}]$, we replace (4.5) with the following one: $\forall v_i \in V_u, k \in \{1, \dots, K\}$, $\tilde{y}_{ik}^* = 1$ if $a_{ik} > 0$, and $\tilde{y}_{ik}^* = -1$ otherwise. In other words, a particular entry is assigned positive, if its accumulative expected number of visits to positively labeled instances is more than that to those negative ones, and vice versa. The same procedure can be carried on over all labels.

Algorithm 2 Transduction by Random Walks on Directed Graphs

Input: A digraph $G = (V, \mathcal{E}, W)$, label information Y , $\mathbf{y}_l$, and $\alpha \in (0, 1)$.
Output: $\mathbf{y}_u^*$
 Compute the in-degree matrix D .
 Compute the transition probability matrix $P = WD^{-1}$.
 Compute the affinity matrix A by (4.4).
 Produce prediction $\mathbf{y}_u^*$. The i -th entry is computed by (4.5), for an unlabeled node $v_i \in V_u$.

One-step Increment/Decrement Update (aka Online Update) In a dynamic graph setting, over the time its graph weights or even structure might

subject to changes, being either inserting or deleting edges or nodes of the graph, or merely adjustments of the edge weights. Note the node insertion case corresponds to the out-of-sample instance scenario. These operations can all be accomplished by one-step increment/decrement edge update. In what follow we present a simple $O(n)$ procedure to perform such update in our context. Consider G (or G') being a digraph (its updated digraph) with transient submatrix $Q = \alpha P^T$ (Q'), and fundamental matrix E (E'), respectively. Our aim is to efficiently update E in the following three cases: (1) Delete an edge or decrease an edge weight, $\Delta q_{ij} < 0$; (2) Add an edge or increase an edge weight, $\Delta q_{ij} > 0$, and $\Delta q_{ij} e_{ji} \neq 1$; (3) Add a new node with its edges. Furthermore, the matrix Q has the property that the summation of each row equals either α or 0. In this case, a change of weight entry q_{ij} will lead to changes in the entire i -th row of Q . Fortunately, as described in Proposition 4 below, the cases described above can all be addressed, once we establish a mean to update E' by Proposition 4(i) focusing on the change of only a single entry q_{ij} between Q and Q' :

Proposition 4. (i) Suppose for an arbitrary entry q_{ij} , the amount of change, Δq_{ij} , satisfies $|\Delta q_{ij}| \leq q_{ij}$ if $\Delta q_{ij} < 0$, and $\Delta q_{ij} e_{ji} \neq 1$ otherwise. The incurred amount of change in E is

$$\Delta E := E' - E = \frac{\Delta q_{ij}}{1 - \Delta q_{ij} e_{ji}} E_{:i} E_{j:} \quad (4.6)$$

where $E_{:i}$ and $E_{j:}$ denote the i -th column and j -th row of E , respectively.

(ii) Consider the changes in the entire i -th row of Q , and assume the amount of change in each entry satisfies the condition described above. To update matrix E , we can either apply (4.6) n times with each time dealing with one entry change,

or equivalently apply the following result:

$$E' = E + \frac{E_{:i}(\Delta Q_{i:} E)}{1 - \Delta Q_{i:} E_{:i}}, \quad (4.7)$$

where $\Delta Q_{i:}$ denotes the amount of change in the i -th row of Q .

(iii) Suppose a new node is added to the graph such that the matrix Q becomes $Q' = \begin{bmatrix} Q & u \\ v^T & q \end{bmatrix}$, then the new fundamental matrix E' is given by

$$E' = \begin{bmatrix} E + \gamma(Eu)(v^T E) & \gamma(Eu) \\ \gamma(v^T E) & \gamma \end{bmatrix}, \quad (4.8)$$

where $\gamma = \frac{1}{(1-q)-v^T E u}$.

The proof is detailed in Appendix 4.10. Notice the imposed condition of $\Delta q_{ij} e_{ji} \neq 1$ in (i) for adding an edge is to guarantee that E is well-defined. Empirically online updates are shown to produce the same results as the batch update counterpart (i.e. our normal algorithm), with negligible entry-wise difference (on the order of 10^{-10}) but with an order of magnitude speedup.

Properties of E We present here several interesting properties regarding the fundamental matrix, E , a central element in our approach. To our best knowledge, many of them are not shown before for the fundamental matrix of an absorbing Markov chain.

- **Nonnegativity.** For a digraph, elements of its fundamental matrix satisfies

$$e_{ij} \geq 0, 1 \leq i, j \leq n.$$

-
- **Edge reversal property.** By simply reversing all the edge directions of a digraph with a fundamental matrix E , the corresponding new fundamental matrix is equal to E^T .
 - **Connectivity and Transitivity.** (i) For any edge $(i, j) \in \{1, \dots, n\}$ in a digraph, $e_{ij} > 0$ iff there is at least one path from v_i to v_j ; (ii) For any $i, j, k \in \{1, \dots, n\}$, if $e_{ij} > 0$ and $e_{jk} > 0$, then $e_{ik} > 0$. As boundary condition we assume there is one path of length 0 from any node to itself.

Moreover, if $\max_{i,j} q_{ij} < \frac{1}{n}$, the following properties are also hold true:

- **Diagonal dominance.** For $i, j \in \{1, \dots, n\}$ of a digraph with $i \neq j$, $e_{ii} > \max\{e_{ij}, e_{ji}\}$.
- **Triangular inequality.** For $i, j, k \in \{1, \dots, n\}$ of a digraph with $j \neq i$ and $k \neq i$, $e_{ii} \geq \max\{e_{ij} + e_{ik} - e_{jk}, e_{ji} + e_{ki} - e_{kj}\}$.
- **Transit inequality.** For distinct indices $i, j, k \in \{1, \dots, n\}$ of a digraph, if there exists a path from v_i to v_k and each path from v_i to v_t includes v_k , then $e_{ik} > e_{it}$.
- **Monotonicity.** Suppose the entry q_{kt} concerning the edge from v_k to v_t is increased by $\Delta q_{kt} > 0$. Then
 - (i) $\Delta e_{kt} > 0$, and for any $i, j \in \{1, \dots, n\}$ such that $i \neq k$ or $j \neq t$ we have $\Delta e_{kt} > \Delta e_{ij}$;
 - (ii) for any $i \in \{1, \dots, n\}$, if there is a path from v_i to v_k , then $\Delta e_{it} > \Delta e_{ik}$.

Detailed proofs are presented in Appendix 4.10, which are adapted based on [110] tackling proximity measure in a more general setting.

4.6 Connections to Existing Methods

Graph Kernels Algebraically our algorithm is similar to several graph kernels, including the Von Neumann Kernel $K_{VND} = \sum_{k=0}^{\infty} \alpha^k W^k = (I - \alpha W)^{-1}$ [111], and the regularized commute time kernel $K_{RCT} = (D - \alpha W)^{-1}$ [73, 98]. These kernel functions are however constructed specifically from undirected graphs (i.e. within the cone of symmetric positive definite matrices) and based on considerably different motivations and derivations.

PageRank and Digraph Laplacian Our approach is also related to PageRank [76], which resolves the issue of source nodes by teleporting random walks that introduce bi-directional edges to all node pairs with equal weights, i.e. $\bar{P} = (1 - \eta)P + \frac{\eta}{n}\mathbf{e}\mathbf{e}^T$ with $\mathbf{e}$ a $n \times 1$ vector of all ones, and η a tiny positive real. A very similar idea is also used by the closely related work [71] based on digraph Laplacian [74]. They are very different from our approach. First, both operate on graphs with *irreducible* Markov chains rather than the *absorbing* Markov chains considered in our context. By definition irreducibility requires each node can be reached from any other node, i.e. a strongly connected graph – algebraically this often gives rise to a dense matrix, as shown in the teleporting operation. Second, as side-effects of introducing the teleporting operation, the input graph structure is not well preserved, and weak edge signals also tend to be washed away. In contrast our approach is able to retain the input graph structure as well as weak signals. This is more pronounced when $\tilde{Q}$ is sparse and of large size: The matrices in both methods are dense and thus demand significant amount of computation and storage efforts, while our approach is able to maintain the induced sparse matrix structure and is much more efficient in term of computation and storage.

Graph Laplacian in Undirected Graphs Our algorithm also works with *undirected* graphs as a special case (i.e. equivalent to bi-directional edges with equal weights). An interesting observation is that here our algorithm can be shown as a scaled variant of the graph Laplacian based method in [73], which has been specifically developed for undirected graphs. This is discussed in details in Appendix 4.10.

Partially Absorbing Random Walks (PARW) [103] It can be shown that the absorbing Markov chains considered in our context is quite general: The random walks of [103] correspond to a very special kind of such absorbing Markov chains where the submatrix of W concerning transient nodes forms a symmetric non-negative matrix. In other words, the transient nodes are inter-connected with undirected edges, while the edges from transient to absorbing nodes are still directed. Details are relegated to Appendix 4.10.

Spanning Forest of Digraphs [89] The celebrated Matrix-Tree theorem has been extended to general digraphs [88], where the quantity $\mathcal{Q} := (I + \tau L)^{-1}$ with $L := D - W$ is shown to be the normalized counts of spanning out-forests. It turns out $\mathcal{Q}$ is a scaled version of $(I - \alpha P^T)^{-1}$, the central piece of our approach. Details are relegated to Appendix 4.10.

4.7 Computational Complexity

The complexity of Algorithm 2 is dominated by the cost of computing the affinity matrix A in (4.4), which can be accomplished by solving the following linear

system

$$(I - \alpha P^T)A = Y.$$

For a general dense matrix P , the computational time is $O(n^3 + n^2K)$. This is e.g. about the same complexity of [71], one of our main competing methods. Fortunately, P is usually a sparse matrix in our context, which can be exploited to reduce the computational time. There are many efficient solvers for large sparse linear systems, including both direct [82, 83] and iterative methods [84, 85]. In our implementation, we adopt the direct solver UMFPACK [82] which exists as a built-in routine (for LU, backslash, and forward slash functions) in MATLAB. The specific complexity depends on the size (n), the number of non-zero entries and the sparsity pattern of P , which remains a challenging task to provide a tighter complexity measure dedicated to our context. Nevertheless, our approach is practically much more efficient comparing to state-of-the-art methods including [71], as is empirically verified in experiments.

4.8 Theoretical Error Bound Analysis

Error Bound Based on Transductive Rademacher Complexity A data-dependent generalization error bound is provided for the proposed algorithm, where we focus on the binary-class case for the sake of simplicity. The bound provided by our analysis is built on top of the work of [112] on transductive Rademacher complexity.

We start by reformulating our algorithm (4.4) as an equivalent representation

$$\mathbf{h} = E\hat{\mathbf{y}} = (I - \alpha P^T)^{-1}\hat{\mathbf{y}}, \quad (4.9)$$

where $\hat{\mathbf{y}}$ is the initial label vector with partial labels $\hat{y}_i \in \{\pm 1\}$ for $v_i \in \mathcal{V}_l$, and $\hat{y}_i = 0$ otherwise. The obtained $\mathbf{h}$ is the “soft” label vector with h_i being the “soft” label for node v_i , which will be assigned with class label $\text{sign}(h_i)$ when making predictions ¹. We denote by $\mathcal{H}_{out}$ the set of feasible soft label vectors generated by our algorithm (4.9). Since there are l labeled nodes, it follows that

$$\mathcal{H}_{out} \subseteq \mathcal{H} := \left\{ \mathbf{h} \mid \mathbf{h} = (I - \alpha P^T)^{-1} \hat{\mathbf{y}}, \|\hat{\mathbf{y}}\|_2 \leq \sqrt{l} \right\}, \quad (4.10)$$

which naturally admits a *vanilla unlabeled-labelled representation* proposed in [112]. We proceed with the definition of transductive Rademacher complexity.

Definition 5. [112] Let $\mathcal{F} \subseteq \mathbb{R}^n$ and $p \in [0, 1/2]$. The transductive Rademacher complexity of $\mathcal{F}$ with parameter p is defined as

$$R_{l,n}(\mathcal{F}, p) := \left(\frac{1}{l} + \frac{1}{n-l} \right) \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\mathbf{f} \in \mathcal{F}} \boldsymbol{\sigma}^T \mathbf{f} \right], \quad (4.11)$$

¹We should remark that predictions made in this way are exactly the same as the predictions made by Algorithm 2 in the binary case. Let $\mathcal{J}_1$ and $\mathcal{J}_2$ denote the index sets of labeled data from classes 1 and 2, respectively, it follows that $Y_{i1} = 1$ if $i \in \mathcal{J}_1$, $Y_{i2} = 1$ if $i \in \mathcal{J}_2$ and $Y_{ij} = 0$ otherwise; $\hat{y}_i = 1$ if $i \in \mathcal{J}_1$, $\hat{y}_i = -1$ if $i \in \mathcal{J}_2$ and $\hat{y}_i = 0$ otherwise. Then, from equations (4.4) and (4.9) we have

$$A = \left[\sum_{i \in \mathcal{J}_1} E_{:i} \quad \sum_{i \in \mathcal{J}_2} E_{:i} \right] \text{ and } \mathbf{h} = \sum_{i \in \mathcal{J}_1} E_{:i} - \sum_{i \in \mathcal{J}_2} E_{:i},$$

which implies that for $1 \leq i \leq n$

$$\text{sign}(h_i) = \arg \max_{k \in \{1,2\}} a_{ik}.$$

where $\boldsymbol{\sigma} = (\sigma_1, \dots, \sigma_n)^T$ is a vector of i.i.d. random variables such that

$$\sigma_i := \begin{cases} 1, & \text{with probability } p, \\ -1, & \text{with probability } p, \\ 0, & \text{with probability } 1 - 2p. \end{cases} \quad (4.12)$$

Different from inductive Rademacher complexity [113], the transductive complexity does not depend on any underlying distribution. Besides, for any label vector $\mathbf{h}$, define the *test error* as $\mathcal{L}_{l,n}(\mathbf{h}) := \frac{1}{n-l} \sum_{i=l+1}^n \ell(\mathbf{h}_i, y_i)$ with respect to its 0/1 loss function ℓ satisfying $\ell(\mathbf{h}_i, y_i) = 1$ if $\mathbf{h}_i \neq y_i$ and $\ell(\mathbf{h}_i, y_i) = 0$ otherwise, and define the *empirical error* of $\mathbf{h}$ as $\hat{\mathcal{L}}_{l,n}(\mathbf{h}) := \frac{1}{l} \sum_{i=1}^l \ell(\mathbf{h}_i, y_i)$. Based on the aforementioned transductive Rademacher complexity, in what follows we present our risk bound and relegate the proof to Appendix 4.10.

Theorem 6. Let $\mathcal{H}_{out}$ be the set of feasible soft label vectors generated by applying (4.9) to all possible sample set $\{(v_i, y_i)\}_{i=1}^n$. Let $c_0 := \sqrt{32 \ln(4e)/3}$, $q := 1/l + 1/(n-l)$ and $s := \frac{n}{(n-1/2)(1-1/(2 \max(l, n-l)))}$. For any $\delta \in (0, 1)$, with probability $1 - \delta$ over random draws of sample $\{(v_i, y_i)\}_{i=1}^n$, for all $\mathbf{h} \in \mathcal{H}_{out}$,

$$\mathcal{L}_{l,n}(\mathbf{h}) \leq \hat{\mathcal{L}}_{l,n}(\mathbf{h}) + \sqrt{\frac{2l}{n(n-l)} \|(I - \alpha P^T)^{-1}\|_F^2} + c_0 q \sqrt{\min(l, n-l)} + \sqrt{\frac{sq}{2} \ln \frac{1}{\delta}}, \quad (4.13)$$

It is easy to see that when $l \rightarrow \infty$, and $(n-l) \rightarrow \infty$, $s \rightarrow 1$. Then the convergence rate is determined by the slack terms $c_0 q \sqrt{\min(l, n-l)} + \sqrt{\frac{sq}{2} \ln \frac{1}{\delta}}$, which is in the order of $O\left(\frac{1}{\sqrt{\min(l, n-l)}}\right)$. So far we provide an transductive Rademacher bound for the binary scenario. In addition, a transductive bound based on PAC-Bayes is also provided in what follows for general multiclass setting where binary

classification can be regarded as a special case. It is known that the first bound is tighter but more focused on binary classification, while the PAC-Bayes bound is more general.

PAC-Bayesian Transduction Bound In this section we present another error bound for our algorithm. This bound is a simple application of the PAC-Bayesian bound for transductive learning developed in [87] and can be applied to both binary-class and multi-class cases. The proof is given in Appendix 4.10.

Theorem 7. Let $\hat{\mathcal{L}}_{l,n}(\mathbf{h})$ and $\mathcal{L}_{l,n}(\mathbf{h})$ be the empirical error and test error, respectively, defined the same as in the previous section with respect to the 0/1 loss function ℓ . Then, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over random draws of $\mathcal{V}_l$ from $\mathcal{V}$, the following bound holds for any $\mathbf{h} \in \mathcal{H}$

$$\mathcal{L}_{l,n}(\mathbf{h}) \leq \hat{\mathcal{L}}_{l,n}(\mathbf{h}) + \sqrt{\left(\frac{2\hat{\mathcal{L}}_{l,n}(\mathbf{h})n}{n-l}\right) \frac{\ln \frac{1}{\delta} + 7\ln(n+1)}{l-1}} + \frac{2\left(\ln \frac{1}{\delta} + 7\ln(n+1)\right)}{l-1}. \quad (4.14)$$

Notice that when $\hat{\mathcal{L}}_{l,n}(\mathbf{h}) = 0$ (i.e. the “realizable case”) the bound converges to 0 as $l \rightarrow \infty$ and the number of unlabeled instances is a t -th ordered polynomial of l (i.e. $n - l = O(l^t)$) for any t . This is consistent with what we have previously for the data-dependant bound.

4.9 Summary

A novel random walk approach is proposed on digraphs that is able to preserve edge directions and is shown to perform competitively against the state-of-the-art

methods in Chapter-7. In the next chapter we will extend this approach to learn the parameters for finding the weight matrix.

4.10 Appendix

Proof of Proposition 3

Proof. We know [108] that the fundamental matrix of $\tilde{Q}$ is $E = \sum_{t=0}^{\infty} Q^t$, and are left to show that $(I - \alpha P^T)^{-1}$ exists, and $E = (I - \alpha P^T)^{-1}$. $(I - \alpha P^T)^{-1}$ exists, since its spectral radius $\rho(P)$ defined as the absolute value of its largest eigenvalue is always 1, and $\alpha < \rho(P)^{-1}$ since $\alpha \in (0, 1)$. we also have $Q^\infty = (\alpha P^T)^\infty = 0$ and the series $I + Q + Q^2 + \dots$ will converge to $(I - \alpha P^T)^{-1}$. $\square$

Proof of Proposition 4

Proof. (i) We focus only on the change in q_{ij} . The difference between Q and Q' is given by

$$Q' - Q = \Delta q_{ij} \varepsilon_i \varepsilon_j^T,$$

where ε_i (ε_j) is a column vector with the i -th (j -th) entry being 1 and all other entries being 0. By the Sherman-Morrison-Woodbury formula [114], we have

$$\begin{aligned} E' &= (I - Q')^{-1} = (I - Q - \Delta q_{ij} \varepsilon_i \varepsilon_j^T)^{-1} \\ &= (I - Q)^{-1} + \frac{\Delta q_{ij}}{1 - \Delta q_{ij} \varepsilon_j^T (I - Q)^{-1} \varepsilon_i} (I - Q)^{-1} \varepsilon_i \varepsilon_j^T (I - Q)^{-1} \\ &= E + \frac{\Delta q_{ij}}{1 - \Delta q_{ij} e_{ji}} E_{:i} E_{j:}, \end{aligned}$$

which completes the proof of part (i).

(ii) The update of E is obtained by noting that

$$Q' - Q = \varepsilon_i \Delta Q_i.$$

and applying the Sherman-Morrison-Woodbury formula as in part (i).

(iii) By the definition of E , we have

$$\begin{aligned} E' &= (I - Q')^{-1} = \begin{bmatrix} I - Q & -u \\ -v^T & 1 - q \end{bmatrix}^{-1} \\ &= \begin{bmatrix} (I - Q)^{-1} + \gamma(I - Q)^{-1}uv^T(I - Q)^{-1} & \gamma(I - Q)^{-1}u \\ \gamma v^T(I - Q)^{-1} & \gamma \end{bmatrix} \\ &= \begin{bmatrix} E + \gamma(Eu)(v^T E) & \gamma(Eu) \\ \gamma(v^T E) & \gamma \end{bmatrix}, \end{aligned}$$

where $\gamma = \frac{1}{(1-q)-v^T(I-Q)^{-1}u} = \frac{1}{(1-q)-v^T E u}$.

□

Proofs for Properties of E

The first two properties, namely nonnegativity and edge reversal property follow directly from the definition of E . We thus omit their proofs and focus on the proofs of the remaining properties.

Proof of Connectivity and Transitivity

Proof. It is trivial when $i = j$, so we concentrate on the case that $i \neq j$.

“ $\Rightarrow$ ” If $e_{ij} > 0$, from Proposition 3, there exist some $t \geq 1$ such that $(Q^t)_{ij} > 0$, and

$$(Q^t)_{ij} = \sum_{i_1, i_2, \dots, i_{t-1}} q_{ii_1} q_{i_1 i_2} \cdots q_{i_{t-1} j} > 0.$$

Since each term in the summation is nonnegative, there exists at least one term $q_{ii_1} q_{i_1 i_2} \cdots q_{i_{t-1} j} > 0$, which implies that $v_i \rightarrow v_{i_1} \rightarrow \cdots \rightarrow v_{i_{t-1}} \rightarrow v_j$ is a path from v_i to v_j .

“ $\Rightarrow$ ” If there exists a path from v_i to v_j , without loss of generality, we represent this path as $v_i \rightarrow v_{i_1} \rightarrow \cdots \rightarrow v_{i_{t-1}} \rightarrow v_j$, then $q_{ii_1} q_{i_1 i_2} \cdots q_{i_{t-1} j} > 0$, which implies that $(Q^t)_{ij} > 0$. Hence, $e_{ij} > 0$.

□

Proof of Diagonal Dominance

Proof. By reversal property, we only need to prove $e_{ii} > e_{ij}$. Without loss of generality, we assume $i = 1$. Consider the digraph $G^{(1)}$ constructed by removing all edges directed to v_1 from the completed digraph ¹ with the weight of each edge being $q_{max} := \max_{i,j} q_{ij}$, we have

$$q_{ij}^{(1)} = \begin{cases} q_{max}, & \text{if } i \neq j \text{ and } j > 1, \\ 0, & \text{otherwise,} \end{cases}$$

¹By complete graph, we refer to the graph with edges in two directions between any two nodes.

which leads to

$$e_{11}^{(1)} = 1, e_{1j}^{(1)} = \frac{q_{max}}{1 - q_{max}(n - 2)} \text{ if } j > 1, e_{i1}^{(1)} = 0 \text{ if } i > 1.$$

Since $q_{max} < \frac{1}{n}$, it follows that $e_{11}^{(1)} > e_{1j}^{(1)}$. For $t > 1$, reduce the weight $q_{kt}^{(1)}$ by the amount of $\Delta q_{kt}^{(1)}$ such that $q_{kt}^{(1)} - \Delta q_{kt}^{(1)} = q_{kt}$, resulting a new graph $G^{(2)}$. From Proposition 4, we get

$$E^{(2)} - E^{(1)} = -\frac{\Delta q_{kt}^{(1)}}{1 + \Delta q_{kt}^{(1)} e_{tk}^{(1)}} E_{:k}^{(1)} E_{t:}^{(1)},$$

which implies that $e_{11}^{(2)} = 1, e_{1j}^{(2)} \leq e_{1j}^{(1)} < e_{11}^{(1)} = e_{11}^{(2)}$ if $j > 1$ and $e_{i1}^{(2)} = 0$ if $i > 1$. Repeat this process, we can eventually get a new graph $G^{(3)}$ such that $e_{11}^{(3)} = 1, e_{1j}^{(3)} < e_{11}^{(3)}$ if $j > 1, e_{i1}^{(3)} = 0$ if $i > 1$, and the only difference between $G^{(3)}$ and G is the lack of edges directed to v_1 in $G^{(3)}$. Now, add an edge with weight q_{s1} ($s > 1$) from v_s to v_1 to $G^{(3)}$ and denote the resulting graph as $G^{(4)}$, then

$$E^{(4)} - E^{(3)} = -\frac{q_{s1}}{1 - q_{s1} e_{1s}^{(3)}} e_{:s}^{(3)} e_{1:}^{(3)},$$

which implies that

$$e_{11}^{(4)} = \frac{e_{11}^{(3)}}{1 - q_{s1} e_{1s}^{(3)}}, \quad e_{1j}^{(4)} = \frac{E_{1j}^{(3)}}{1 - q_{s1} E_{1s}^{(3)}}.$$

Thus,

$$e_{11}^{(4)} - e_{1j}^{(4)} = \frac{e_{11}^{(3)} - e_{1j}^{(3)}}{1 - q_{s1} e_{1s}^{(3)}} > 0, \text{ if } j > 1.$$

Repeat this process, we can eventually get the original graph G and $e_{11} > e_{1j}$ if $j > 1$, which completes the proof. $\square$

Proof of the Triangular Inequality

Proof. Similar to the proof of diagonal dominance, we only need to prove $e_{ii} \geq e_{ij} + e_{ik} - e_{jk}$, and, without loss of generality, assume $i = 1$. Let $G^{(1)}$ be the digraph constructed by removing all edges directed to v_1 from the completed digraph with the weight of each edge being $\frac{1}{n}$. Then,

$$e_{11}^{(1)} = 1, \quad e_{1j}^{(1)} = \frac{1}{2} \text{ if } j > 1, \quad e_{i1}^{(1)} = 0 \text{ if } i > 1,$$

which lead to $e_{11}^{(1)} - (e_{1j}^{(1)} + e_{1k}^{(1)}) = 0$, and

$$e_{11}^{(1)} - (e_{1j}^{(1)} + e_{1k}^{(1)} - e_{jk}^{(1)}) = e_{jk}^{(1)} \geq 0.$$

For $t > 1$, reduce the weight $q_{kt}^{(1)}$ by the amount of $\Delta q_{kt}^{(1)}$ such that $\frac{1}{n} - \Delta q_{kt}^{(1)} = q_{kt}$, resulting a new graph $G^{(2)}$. From Proposition 4, we get

$$\begin{aligned} e_{11}^{(2)} &= e_{11}^{(1)}, \\ e_{1j}^{(2)} &= e_{1j}^{(1)} - \frac{\Delta q_{kt}^{(1)} e_{1k}^{(1)}}{1 + \Delta q_{kt}^{(1)} e_{tk}^{(1)}} e_{tj}^{(1)} \leq e_{1j}^{(1)} \text{ if } j > 1, \\ e_{i1}^{(2)} &= 0 \text{ if } i > 1, \end{aligned}$$

which imply $e_{11}^{(2)} - (e_{1j}^{(2)} + e_{1k}^{(2)}) \geq e_{11}^{(1)} - (e_{1j}^{(1)} + e_{1k}^{(1)}) = 0$, and

$$e_{11}^{(2)} - (e_{1j}^{(2)} + e_{1k}^{(2)} - e_{jk}^{(2)}) \geq e_{jk}^{(2)} \geq 0.$$

Repeat this process, we can eventually get a new graph $G^{(3)}$ such that $e_{11}^{(3)} = 1$, $e_{i1}^{(3)} = 0$ if $i > 1$, $e_{11}^{(3)} - (e_{1j}^{(3)} + e_{1k}^{(3)}) \geq 0$, and the only difference between $G^{(3)}$ and

G is the lack of edges directed to v_1 in $G^{(3)}$. Now, add an edge with weight q_{s1} ($s > 1$) from v_s to v_1 to $G^{(3)}$ and denote the resulting graph as $G^{(4)}$, then

$$e_{11}^{(4)} = \frac{e_{11}^{(3)}}{1 - q_{s1}e_{1s}^{(3)}}, \quad e_{1j}^{(4)} = \frac{e_{1j}^{(3)}}{1 - q_{s1}e_{1s}^{(3)}} \text{ if } j > 1.$$

Thus,

$$e_{11}^{(4)} - (e_{1j}^{(4)} + e_{1k}^{(4)}) = \frac{e_{11}^{(3)} - (e_{1j}^{(3)} + e_{1k}^{(3)})}{1 - q_{s1}e_{1s}^{(3)}} \geq 0.$$

Repeat this process, we can eventually get the original graph G and

$$\begin{aligned} e_{11} - (e_{1j} + e_{1k} - e_{jk}) &= e_{jk} + \left((e_{11} - (e_{1j} + e_{1k})) \right) \\ &\geq e_{jk} \geq 0, \end{aligned}$$

which completes the proof. □

Remark 8. The diagonal dominance and triangular inequality can be applied to construct a metric on node set V . Indeed, define a function $\mathcal{M} : V \times V \rightarrow \mathbb{R}$ as

$$\mathcal{M}(v_i, v_j) := e_{ii} + e_{jj} - e_{ij} - e_{ji},$$

then we can prove that $\mathcal{M}$ is a metric on V (i.e., it satisfies the axioms of a metric).

Proof of Transit Inequality

Proof. We prove this result by contradiction. Suppose there exist digraphs that violate the transit inequality. Let $G^{(1)}$ be the digraph with the minimum number

of edges among the digraphs that violate the transit inequality. It follows that $G^{(1)}$ has a path from v_i to v_k and every path from v_i to v_t ($t \neq k$) includes v_k , but $e_{ik}^{(1)} \leq e_{it}^{(1)}$.

We first show that there exist at least one path from v_i to v_k that has more than one edges. Otherwise, every path from v_i to v_k has only one edge. Considering the ik -entry of the matrices on both sides of the equality $(I - Q^{(1)})E^{(1)} = I$, we have

$$e_{ik}^{(1)} = (Q^{(1)}E^{(1)})_{ik} = \sum_{j=1}^n q_{ij}^{(1)} e_{jk}^{(1)} = q_{ik}^{(1)} e_{kk}^{(1)},$$

since $q_{ij}^{(1)} e_{jk}^{(1)} = 0$ if $j \neq k$. (Otherwise, there is one path from v_i to v_k having more than one edge.) Similarly, we have $e_{it}^{(1)} = q_{ik}^{(1)} e_{kt}^{(1)}$. Thus, by diagonal dominance,

$$e_{ik}^{(1)} - e_{it}^{(1)} = q_{ik}^{(1)} (e_{kk}^{(1)} - e_{kt}^{(1)}) > 0,$$

which is a contradiction to the construction of $G^{(1)}$.

Let (v_i, v_j) be the first edge of some path from v_i to v_k such that $j \neq k$, and $G^{(2)}$ be the digraph obtained by removing the (v_i, v_j) edge from $G^{(1)}$. Then,

$$\begin{aligned} \Delta e_{ik} = e_{ik}^{(2)} - e_{ik}^{(1)} &= -\frac{q_{ij}^{(1)} e_{ii}^{(1)}}{1 + q_{ij}^{(1)} e_{ji}^{(1)}} e_{jk}^{(1)}, \\ \Delta e_{it} = e_{it}^{(2)} - e_{it}^{(1)} &= -\frac{q_{ij}^{(1)} e_{ii}^{(1)}}{1 + q_{ij}^{(1)} e_{ji}^{(1)}} e_{jt}^{(1)}. \end{aligned}$$

If there is no path from v_i to v_k , then $e_{ik}^{(2)} = 0$, $e_{it}^{(2)} = 0$, and

$$\Delta e_{ik} - \Delta e_{it} = e_{it}^{(1)} - e_{ik}^{(1)} \geq 0.$$

If there is some path from v_i to v_k , then $e_{ik}^{(2)} > e_{it}^{(2)}$ (Otherwise, $G^{(2)}$ also violates the transit inequality but has less edges than $G^{(1)}$, which is a contradiction.), which leads to

$$\Delta e_{ik} - \Delta e_{it} = (e_{ik}^{(2)} - e_{it}^{(2)}) + (e_{it}^{(1)} - e_{ik}^{(1)}) > 0.$$

In both cases, we have $\Delta e_{ik} - \Delta e_{it} \geq 0$, which together with

$$\Delta e_{ik} - \Delta e_{it} = \frac{q_{ij}^{(1)} e_{ii}^{(1)}}{1 + q_{ij}^{(1)} e_{ji}^{(1)}} (e_{jt}^{(1)} - e_{jk}^{(1)}),$$

implies $e_{jt}^{(1)} \geq e_{jk}^{(1)}$. Thus,

$$e_{jt}^{(2)} = \frac{e_{jt}^{(1)}}{1 + q_{ij}^{(1)} e_{ji}^{(1)}} \geq \frac{e_{jk}^{(1)}}{1 + q_{ij}^{(1)} e_{ji}^{(1)}} = e_{jk}^{(2)}.$$

Since $G^{(2)}$ has a path from v_j to v_k , every path from v_j to v_t includes v_k and $e_{jk}^{(2)} \leq e_{jt}^{(2)}$, it follows that $G^{(2)}$ also violates the transit inequality but has less edges than $G^{(1)}$, which is a contradiction to the construction of $G^{(1)}$. $\square$

Proof of Monotonicity

Proof. (i) From Proposition 4, we have

$$\Delta e_{kt} = \frac{\Delta q_{kt}}{1 - \Delta q_{kt} e_{tk}} e_{kk} e_{tt}, \quad \Delta e_{ij} = \frac{\Delta q_{kt}}{1 - \Delta q_{kt} e_{tk}} e_{ik} e_{tj}.$$

Thus, $\Delta e_{kt} > 0$ and

$$\begin{aligned}\Delta e_{kt} - \Delta e_{ij} &= \frac{\Delta q_{kt}}{1 - \Delta q_{kt} e_{tk}} (e_{kk} e_{tt} - e_{ik} e_{tj}) \\ &= \frac{\Delta q_{kt}}{1 - \Delta q_{kt} e_{tk}} \left((e_{kk} - e_{ik}) e_{tt} + e_{ik} (e_{tt} - e_{tj}) \right) \\ &> 0,\end{aligned}$$

if $i \neq k$ or $j \neq t$, due to the diagonal dominance.

(ii) Similar to case (i), we have

$$\Delta e_{it} = \frac{\Delta q_{kt} e_{ik}}{1 - \Delta q_{kt} e_{tk}} e_{tt}, \quad \Delta e_{ik} = \frac{\Delta q_{kt} e_{kk}}{1 - \Delta q_{kt} e_{tk}} e_{tk},$$

which imply

$$\Delta e_{it} - \Delta e_{ik} = \frac{\Delta q_{kt} e_{ik}}{1 - \Delta q_{kt} e_{tk}} (e_{tt} - e_{tk}) > 0.$$

□

Connections to Graph Laplacian in undirected Graphs

Here we shown that when operating as random walks on undirected graphs, our algorithm is equivalent to a scaled variant of the graph Laplacian method of [73]. For an undirected graph G , denote $S = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$, and define $P = W D^{-1}$, where $W = [w_{ij}]$ is a symmetric matrix and $D = \text{diag}(d_1, \dots, d_n)$, with $d_i = \sum_j w_{ij}$. Now consider applying our algorithm (i.e. (4.4) and (4.5)) on undirected

graph G . Since

$$A = (I - \alpha P^T)^{-1}Y = D^{-\frac{1}{2}}(I - \alpha S)^{-1}D^{\frac{1}{2}}Y,$$

we have

$$D^{\frac{1}{2}}A = (I - \alpha S)^{-1}\left(D^{\frac{1}{2}}Y\right).$$

Notice that since the goal is to choose the best element from the current row i as in (4.5), the result will not change by multiplying an additional constant $d_i^{\frac{1}{2}}$ to all elements in the row. Define $\hat{A} := D^{\frac{1}{2}}A$, and let $\hat{Y} := D^{\frac{1}{2}}Y$ we now have

$$\hat{A} = (I - \alpha S)^{-1}\hat{Y},$$

which recovers the update formula of [73], with the only difference that instead of Y , $\hat{Y}$ is used here as a row-wise scaled variant.

Connections to Partially Absorbing Random Walks (PARW) [103]

The random walks considered in [103] deal with a *special* form of *absorbing Markov chains* where the submatrix of its weight matrix concerning transient nodes forms a symmetric non-negative matrix with diagonal entries taking zero

values. More formally, denote this submatrix as

$$W_P = \begin{pmatrix} 0 & w_{12} & \cdots & w_{1n} \\ w_{21} & 0 & \cdots & w_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ w_{n1} & \cdots & w_{nn-1} & 0 \end{pmatrix},$$

which is a $n \times n$ symmetric non-negative matrix with zero diagonal values. Let $\Lambda_P = \text{diag}(\lambda_1, \dots, \lambda_n)$ with $\lambda_i > 0 \ \forall i$, and define $\lambda_{\mathbf{P}} = \text{vec}(\Lambda_P)$, where the operator $\text{vec}(\cdot)$ extracts the diagonal elements of the input matrix to produce a column vector. The weight matrix of the PARW family proposed in [103] can be regarded as an extended matrix of W_P by introducing an additional absorbing node, as

$$\tilde{W}_P = \left(\begin{array}{c|c} W_P & \lambda_{\mathbf{P}} \\ \hline 0 & 1 \end{array} \right),$$

with 0 here referring to a $1 \times n$ vector of zero values. Define the degree matrix $D_P = \text{diag}(d_1, \dots, d_n)$ with each element computed as the sum of the corresponding row of W_P , $d_i = \sum_j w_{ij}$. denote the (sub-) graph Laplacian $L_P = D_P - W_P$, and let $P_W = (\Lambda_P + D_P)^{-1}W_P$, and $P_\lambda = (\Lambda_P + D_P)^{-1}\Lambda_P$. At this point, we are ready to obtain the *probability transition matrix*:

$$\tilde{P}_P = \left(\begin{array}{c|c} P_W & \text{vec}(P_\lambda) \\ \hline 0 & 1 \end{array} \right),$$

which is exactly the same form as of $\tilde{Q}$ defined earlier in Equation (4.1) of our approach. Note that the random walks considered in [103] is a *special* form of (4.1) with W_P confined to being a symmetric matrix with zero diagonal entries. Following Proposition 3, its fundamental matrix becomes

$$E_P = \sum_{t=0}^{\infty} (P_W)^t = (I - (\Lambda_P + D_P)^{-1} W_P)^{-1} = (\Lambda_P + L_P)^{-1} (\Lambda_P + D_P).$$

It is interesting to observe that the absorption probability matrix, A_P , proposed and discussed in [103] can be related to E_P as

$$A_P = \sum_{t=0}^{\infty} (P_W)^t P_{\Lambda} = E_P [(\Lambda_P + D_P)^{-1} \Lambda_P].$$

Interestingly, it corresponds to a special form of the absorbing probabilities of the Markov chain [?, see e.g. Theorem 3.3.7 of]KemSne:SpringerBook76.

Connections to Spanning Forest of Digraphs

Following [88], let $L := D - W$, and denote the forest matrix $\mathcal{Q} := (I + \tau L)^{-1}$. Suppose $\mathcal{F}$ is the set of all spanning forests of digraph G and $\mathcal{F}^{ij}$ the set of spanning forests containing edge $i \leftarrow j$. It has been shown in [88, 89] that each (i, j) -th entry of $\mathcal{Q} = [q_{ij}]$ concerns the normalized counts of spanning out-forests, as $q_{ij} = \frac{\epsilon(\mathcal{F}^{ji})}{\epsilon(\mathcal{F})}$, with $\epsilon(\cdot)$ denoting the count of specified forests.

Further, denote $P^T = I - \mu L$, $\mu > 0$, and let $q = (1 + \frac{\tau}{\mu})^{-1}$, it can be easily

shown that

$$\mathcal{Q} = q \left(I - \frac{\tau}{\mu + \tau} P^T \right)^{-1}.$$

In other words, $\mathcal{Q}$ is a scaled version of $(I - \alpha P^T)^{-1}$, with $\alpha := \frac{\tau}{\mu + \tau}$.

Proof of Theorem 6

We start by stating the risk bound developed in [112].

Theorem 9. [112] For a transductive learning algorithm, let $\mathcal{F}$ be the set of all soft labels of the algorithm obtained by applying it to all possible sample $\{(v_i, y_i)\}_{i=1}^n$. Let c_0 , q and s be defined the same way as in Theorem 6. For any $\delta \in (0, 1)$, with probability $1 - \delta$ over random draws of sample $\{(v_i, y_i)\}_{i=1}^n$, for all $\mathbf{f} \in \mathcal{F}$,

$$\mathcal{L}_{l,n}(\mathbf{f}) \leq \hat{\mathcal{L}}_{l,n}(\mathbf{f}) + R_{l,n}(\mathcal{F}, p_0) + c_0 q \sqrt{\min(l, n - l)} + \sqrt{\frac{sq}{2} \ln \frac{1}{\delta}}, \quad (4.15)$$

where $p_0 = \frac{l(n-l)}{n^2} < \frac{1}{2}$.

This risk bound is quite general and can be applied to estimate the generalization error for a suite of transductive learning algorithms if one can find the transductive Rademacher complexity $R_{l,n}(\mathcal{F}, p_0)$ of the hypothesis set $\mathcal{F}$. Fortunately, in what follows we show that for the hypothesis set $\mathcal{H}_{out}$, the transductive Rademacher complexity $R_{l,n}(\mathcal{H}_{out}, p_0)$ can be found efficiently, which consequentially leads to the generalization error bound for our algorithm. From Definition

5, we notice that

$$R_{l,n}(\mathcal{H}_{out}, p_0) \leq R_{l,n}(\mathcal{H}, p_0), \text{ if } \mathcal{H}_{out} \subseteq \mathcal{H}. \quad (4.16)$$

In other words, by comparing Theorem 9 and Theorem 6, it is sufficient to prove

$$R_{l,n}(\mathcal{H}, p_0) \leq \sqrt{\frac{2l}{n(n-l)} \|(I - \alpha P^T)^{-1}\|_F^2}. \quad (4.17)$$

Note $E = (I - \alpha P^T)^{-1}$, and for convenience denote $E_{i,:}$ as the i th row of E . From the Definition 5, we have that

$$\begin{aligned} R_{l,n}(\mathcal{H}, p_0) &= q \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\mathbf{h} \in \mathcal{H}} \boldsymbol{\sigma}^T \mathbf{h} \right] \\ &= q \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\|\hat{\mathbf{y}}\| \leq \sqrt{l}} \sum_{i=1}^n \sigma_i \langle E_{i,:}, \hat{\mathbf{y}} \rangle \right] \\ &= q \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\|\hat{\mathbf{y}}\| \leq \sqrt{l}} \left\langle \sum_{i=1}^n \sigma_i E_{i,:}, \hat{\mathbf{y}} \right\rangle \right] \\ &= q \sqrt{l} \mathbb{E}_{\boldsymbol{\sigma}} \left[\left\| \sum_{i=1}^n \sigma_i E_{i,:} \right\|_2 \right] \\ &= q \sqrt{l} \mathbb{E}_{\boldsymbol{\sigma}} \left[\sqrt{\sum_{i,j=1}^n \sigma_i \sigma_j \langle E_{i,:}, E_{j,:} \rangle} \right] \\ &\leq q \sqrt{l} \sqrt{\sum_{i,j=1}^n \mathbb{E}_{\boldsymbol{\sigma}} [\sigma_i \sigma_j] \langle E_{i,:}, E_{j,:} \rangle} \\ &= q \sqrt{l} \sqrt{\sum_{i=1}^l 2p_0 \langle E_{i,:}, E_{i,:} \rangle} \\ &= \sqrt{\frac{2l}{n(n-l)} \|E\|_F^2}, \end{aligned}$$

where we have used

$$\sup_{\|\hat{\mathbf{y}}\| \leq \sqrt{l}} \left\langle \sum_{i=1}^n \sigma_i E_{i,:}, \hat{\mathbf{y}} \right\rangle = \sqrt{l} \left\| \sum_{i=1}^n \sigma_i E_{i,:} \right\|_2$$

in the forth equality, the Jensen inequality in the first inequality, as well as $q\sqrt{Lp_0} = \sqrt{\frac{l}{n(n-l)}}$ in the last equality. Substituting the above bound of $R_{l,n}(\mathcal{H}, p_0)$ into (4.15) and (4.16), the risk bound in (4.13) is obtained.

Proof of Theorem 7

The proof of Theorem 7 follows immediately from the transductive PAC-Bayesian error bound below [87, adapted from Corollary 23 of]:

Theorem 10. Let the conditions of Theorem 7 hold. Then, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over random draws of $\mathcal{V}_l$ from $\mathcal{V}$ the following bound holds for any $\mathbf{h} \in \mathcal{H}$

$$\begin{aligned} \mathcal{L}_{l,n}(\mathbf{h}) &\leq \hat{\mathcal{L}}_{l,n}(\mathbf{h}) + \sqrt{\left(\frac{2\hat{\mathcal{L}}_{l,n}(\mathbf{h})n}{n-l} \right) \frac{\ln \frac{1}{P(\mathbf{h})} + \ln \frac{l}{\delta} + 7\ln(n+1)}{l-1}} \\ &\quad + \frac{2 \left(\ln \frac{1}{P(\mathbf{h})} + \ln \frac{l}{\delta} + 7\ln(n+1) \right)}{l-1}. \end{aligned} \tag{4.18}$$

Since our algorithm is a deterministic algorithm and the selection of the prior distribution $P(\mathbf{h})$ can be made prior to observing the data, we can simply select $P(\mathbf{h}) = 1$ for our algorithm, which leads to the error bound (4.14) in Theorem 7.

Chapter 5

Parameter Estimation for Absorbing Markov Chain

In this chapter, the parameter estimation for the Absorbing Markov Chain based tracing problem is proposed. If the weight matrix is formulated as parametric functions then the parameters can be determined by maximizing the sum of the logarithm of expected number of visits from unlabeled nodes to the labeled nodes. For the scenario where the class labels for only very few nodes are known, the parameter estimation problem is solved by Expectation Maximization (EM) algorithm. For a different scenario where we have separate training set for which know the class label for all the nodes, the parameters are estimated using maximum-likelihood formulation.

5.1 Introduction

Accuracy of the graph-based label propagation algorithms depend on the weight matrix of the input graph, which captures the manifold of the input data [115, 116, 117, 118, 119]. Prior research work on learning the weight matrix of the input data for label propagation methods can be categorized into two classes: similarity based and local linear embedding (LLE) [120] based approaches. The similarity based approaches tries to tune the parameters of the similarity functions like Gaussian kernel, whereas LLE based approaches tries to capture the true manifold of the data by minimizing some local reconstruction error [118, 121, 122, 123]. The similarity based approaches are more difficult to tune but are robust to noise [119] whereas LLE based approaches has less number of parameters to tune but they work less accurately in noisy data [124]. In a more recent work [119], Adaptive Edge Weighting (AEW) approach is proposed which is a similarity based approach but also captures the data manifold. This approach tries to find the width parameter of the Gaussian kernel based similarity function by minimizing a local reconstruction error. Nonetheless all these approaches assume an undirected graph structure with a symmetric weight matrix, whereas in many practical applications it is more meaningful to work on the directed graph. In our approach we have formulated the likelihood function as the sum of the log-likelihood of expected number of visits from an unlabeled node to the labeled nodes. Expectation maximization based solution is provided for maximizing this likelihood function.

5.2 Problem Set-up

Let's say $\mathcal{G} = \{G_1, G_2, \dots, G_b\}$ is the set of directed graphs extracted from b training set images $\mathcal{I} = \{I_1, I_2, \dots, I_b\}$. Let's say $\{n_1, n_2, \dots, n_b\}$ are the number of nodes, $\{W_1, W_2, \dots, W_b\}$ are the weight matrices, $\{P_1, P_2, \dots, P_b\}$ are the transition probability matrices, $\{E_1, E_2, \dots, E_b\}$ are the fundamental matrices for each of those graphs. We construct an ensemble graph of $N = \sum_{f=1}^b n_f$ nodes with weight matrix $W = \text{diag}(W_1, W_2, \dots, W_b)$. We can compute the corresponding $E = \text{diag}(E_1, E_2, \dots, E_b)$, $P = \text{diag}(P_1, P_2, \dots, P_b)$ by computing them separately for each graph and then putting them in a block-diagonal form.

It can be proved (See Appendix) that $E' = [e'_{ij}] = (1 - \alpha)E$ is row stochastic for non-source nodes. For source-nodes it will row sum to zero as the source nodes have outgoing connections with the absorbing nodes only. We define the probability of visiting j^{th} node starting from i^{th} node as $p(j|i) = e'_{ij}$.

Let's say V_u, V_l are the set of unlabeled and labeled nodes respectively and $C = \{1, 2, \dots, L\}$ is the set of the class labels. $Y = [y_{ik}]$ is a $N \times L$ dimensional label matrix, where y_{ik} is 1 if i^{th} node belongs to class k , otherwise it is 0. The set C and the label matrix Y is obtained by collecting the labeled nodes from each graphs and then correspondingly mapping them into the ensemble graph.

If $\phi_{ij} = (\phi_{ij}^1, \phi_{ij}^2, \dots, \phi_{ij}^m)^T$ denotes $m \times 1$ dimensional feature vector for a pair of nodes i and j and $\theta = (\theta_1, \theta_2, \dots, \theta_m)^T$ is weight parameters for these features then we define the following θ -parameterized weight for a pair of nodes i and j

$$w_{ij}(\theta) = \exp(\theta^T \phi_{ij}) \quad (5.1)$$

So the weight matrix can be expressed in matrix form as $W(\theta) = [w_{ij}(\theta)]_{N \times N}$.

Similarly the fundamental matrix can also be expressed in matrix form as $E'(\theta) = [e'_{ij}(\theta)]_{N \times N} = [p_\theta(j|i)]_{N \times N}$.

The parameters θ should be such that it increases the expected number of visits from the unlabeled nodes i to the labeled nodes j , therefore I propose the following likelihood function to maximize.

$$\theta := \arg_\theta \max \ell(\theta) = \arg_\theta \max \sum_{j \in V_l} \sum_{i \in V_u} \log p_\theta(j|i) \quad (5.2)$$

5.3 Expectation Maximization based Parameter Estimation for Incomplete Data Scenario

For the tracing scenario, only few nodes are labeled therefore it is an incomplete data scenario where the missing variable is the class label k for the node i . For incomplete data scenario, it is impossible to find the θ to maximize $\ell(\theta)$, therefore expectation-maximization (EM) based iterative update formulation is proposed for finding the optimal θ . In the E-step (expectation step), the lower bound for $\ell(\theta)$ is repeatedly reconstructed whereas in M-step (maximization step) that lower bound is maximized.

5.3.1 E-step

If k is a random variable denoting the class label for the i^{th} node and $Q_i(k)$ is the probability that the i^{th} node belongs to the class k , then our likelihood function can be expressed as following

$$\begin{aligned}
\ell(\theta) &= \sum_{j \in V_l} \sum_{i \in V_u} \log \sum_{k \in C} p_\theta(j, k|i) \\
&= \sum_{j \in V_l} \sum_{i \in V_u} \log \sum_{k \in C} Q_i(k) \frac{p_\theta(j, k|i)}{Q_i(k)} \tag{5.3}
\end{aligned}$$

$$\geq \sum_{j \in V_l} \sum_{i \in V_u} \sum_{k \in C} Q_i(k) \log \frac{p_\theta(j, k|i)}{Q_i(k)} = L(\theta) \tag{5.4}$$

The above inequality in eq. 5.4 is from the Jensen's inequality. The equality hold for the eq. 5.4 if $\frac{p_\theta(j, k|i)}{Q_i(k)} = c$, or equivalently $Q_i(k) \propto p_\theta(j, k|i)$. As we know the $Q_i(k)$ is a distribution, so $\sum_k Q_i(k) = 1$. So the above equation will be

$$\begin{aligned}
Q_i(k) &= \frac{p_\theta(j, k|i)}{\sum_{j \in V_l} p_\theta(j, k|i)} \\
&= \frac{p_\theta(j|i)p(k|j, i)}{\sum_{j \in V_l} p_\theta(j|i)p(k|j, i)} \tag{5.5}
\end{aligned}$$

$$= \frac{e'_{ij}p(k|j)}{\sum_{j \in V_l} e'_{ij}p(k|j)} \tag{5.6}$$

where $p(k|j, i) = p(k|j)$ in eq. 5.6 is because of the fact that i and k are independent of each other. $p(k|j) = 1$ if the j^{th} labeled node belongs to class label k , otherwise it is 0.

5.3.2 M-step

In M-step, the lower bound of $\ell(\theta)$ is maximized

$$\theta := \arg_\theta \max L(\theta) \tag{5.7}$$

where

$$\begin{aligned}
L(\theta) &= \sum_{j \in V_l} \sum_{i \in V_u} \sum_{k \in C} Q_i(k) \log \frac{p_\theta(j, k|i)}{Q_i(k)} \\
&= \sum_{j \in V_l} \sum_{i \in V_u} \sum_{k \in C} Q_i(k) \log \frac{p_\theta(j|i)p(k|j, i)}{Q_i(k)} \\
&= \sum_{j \in V_l} \sum_{i \in V_u} \sum_{k \in C} Q_i(k) \log \frac{e'_{ij}p(k|j)}{Q_i(k)} \\
&= \sum_{j \in V_l} \sum_{i \in V_u} \sum_{k \in C} \left(Q_i(k) \log(e'_{ij}p(k|j)) - Q_i(k) \log(Q_i(k)) \right) \quad (5.8)
\end{aligned}$$

For finding the local maxima of $L(\theta)$ we need to find the local gradient of $L(\theta)$ and then do gradient based line search. Through $\log(0)$ is mathematically undefined, for implementation purpose $\log(0)$ is defined as 0 here.

$$\begin{aligned}
\frac{\partial L(\theta)}{\partial \theta_l} &= \sum_{j \in V_l} \sum_{i \in V_u} \sum_{k \in C} Q_i(k) \frac{\partial}{\partial \theta_l} \log(e'_{ij}p(k|j)) \\
&= \sum_{j \in V_l} \sum_{i \in V_u} \sum_{k \in C} Q_i(k) \frac{1}{e'_{ij}} \frac{\partial e'_{ij}}{\partial \theta_l} p(k|j) \quad (5.9)
\end{aligned}$$

Let's say $F(\theta) = (E'(\theta))^{-1}$, then $F(\theta)E'(\theta) = I$. Taking partial derivatives from both the sides and rearranging the terms we get

$$\frac{\partial E'}{\partial \theta_l} = -E'(\theta) \frac{\partial F(\theta)}{\partial \theta_l} E'(\theta) \quad (5.10)$$

where,

$$\frac{\partial F(\theta)}{\partial \theta_l} = -\alpha \frac{\partial (D(\theta)^{-1}W(\theta)^T)}{\partial \theta_l} \quad (5.11)$$

If we define $D(\theta)^{-1}W(\theta)^T = \hat{W}(\theta) = [\hat{w}_{ij}]$, then

$$\frac{\partial F(\theta)}{\partial \theta_l} = -\alpha \frac{\partial \hat{W}(\theta)}{\partial \theta_l} = -\alpha \left[\frac{\partial \hat{w}_{ij}}{\partial \theta_l} \right] \quad (5.12)$$

Putting it back to eq. 5.10 we get,

$$\frac{\partial E'}{\partial \theta_l} = \alpha E'(\theta) \left[\frac{\partial \hat{w}_{ij}}{\partial \theta_l} \right] E'(\theta) \quad (5.13)$$

Let's say $f(\theta) = \exp(\theta^T \phi_{ij}) = w_{ij}$ and $g(\theta) = \sum_m \exp(\theta^T \phi_{im}) = \sum_m \gamma_m(\theta)$, where $\gamma_m(\theta) = \exp(\theta^T \phi_{im}) = w_{im}$. Here m denotes a neighbor of the i^{th} node.

Now we define the partial derivatives of $f(\theta)$ and $g(\theta)$ as following,

$$\frac{\partial f(\theta)}{\partial \theta_l} = \phi_{ij}^l w_{ij} \text{ and } \frac{\partial g(\theta)}{\partial \theta_l} = \sum_m \phi_{im}^l w_{im} \quad (5.14)$$

Then, the partial derivative of $\hat{w}_{ij}$ is as following,

$$\begin{aligned} \frac{\partial \hat{w}_{ij}}{\partial \theta_l} &= \frac{\partial}{\partial \theta_l} \left(\frac{\exp(\theta^T \phi_{ij})}{(\sum_m \exp(\theta^T \phi_{im}))} \right) \\ &= \frac{\partial}{\partial \theta_l} \left(\frac{f(\theta)}{g(\theta)} \right) \\ &= \frac{1}{g(\theta)^2} \left(g(\theta) \frac{\partial f(\theta)}{\partial \theta_l} - f(\theta) \frac{\partial g(\theta)}{\partial \theta_l} \right) \\ &= \frac{1}{(\sum_m w_{im})^2} \left[\phi_{ij}^l w_{ij} \sum_m w_{im} - w_{ij} \sum_m \phi_{im}^l w_{im} \right] \end{aligned} \quad (5.15)$$

Putting eq. 5.15 back in eq. 5.13 we get the partial derivative $\nabla L = \left[\frac{\partial L(\theta)}{\partial \theta_l} \right]$.

Then we apply the following gradient accent step to update θ

$$\theta^{t+1} = \theta^t + \lambda(t) \frac{\nabla L}{\|\nabla L\|} \quad (5.16)$$

where $\lambda(t) = \frac{\eta}{\frac{t}{\tau} + 1}$ is a adaptive learning parameter, where η and τ are constants.

Algorithm 3 Algorithm for Parameter Estimation on Absorbing Markov Chain for Incomplete Data

Input: $G = (V, E), \phi_{ij}, V_u, V_l$
Output: $\theta = (\theta_1, \theta_2, \dots, \theta_n)^T$
Init: $\theta^0 = \text{rand}(0,1)$
 $\Delta = 10^6, \Delta' = 10^6$
 $\eta = 1, \tau = 10$
 $E'(\theta^0), Q^0, L(\theta^0, Q^0)$
while $\Delta > 10^{-6}$ **do**
 /* E Step */
 while $\Delta' > 10^{-6}$ **do**
 /* M Step */
 Gradient Ascent : $\theta^{t+1} = \theta^t + \lambda(t) \frac{\nabla L}{\|\nabla L\|}$
 $\Delta' = L(\theta^{t+1}, Q^n) - L(\theta^t, Q^n)$
 end while
 Estimate Q^{n+1}
 $\Delta = L(\theta, Q^{n+1}) - L(\theta, Q^n)$
end while

5.4 Parameter Estimation for Complete Data Scenario

For complete data scenario, we have a training data set where we know the labels for all the nodes. Therefore we can maximize the expected number of visits from i^{th} node to the j^{th} node belonging to the same class k . So likelihood function can be formulated as following

$$\ell'(\theta) = \sum_{k \in C} \sum_{i, j \in V} \log p_\theta(j|i, k) \quad (5.17)$$

Now we have the following maximization problem

$$\theta := \arg_\theta \max \ell'(\theta) \quad (5.18)$$

where

$$\begin{aligned} \ell'(\theta) &= \sum_{k \in C} \sum_{i, j \in V} \log p_\theta(j|i, k) \\ &= \sum_{k \in C} \sum_{i, j \in V} \log \left[\frac{p(k|j, i)p_\theta(j|i)}{p(i, k)} \right] \end{aligned} \quad (5.19)$$

$$\propto \sum_{k \in C} \sum_{i, j \in V} \log \left[e'_{ij} p(k|j, i) \right] \quad (5.20)$$

where $p(k|j, i)$ from eq. 5.19 and 5.20 are 1 if j^{th} and i^{th} node belong to the same class k , otherwise they are 0. Here we assume that i and k are independent of each other, therefore $p(i, k) = p(i)p(k)$ are same for all i and k . Through $\log(0)$ is mathematically undefined, for implementation purpose $\log(0)$ is defined as 0 here.

For finding the local maxima of $\ell'(\theta)$ we need to find the local gradient of $\ell'(\theta)$ and then do gradient based line search.

$$\frac{\partial \ell'(\theta)}{\partial \theta_l} = \sum_{k \in C} \sum_{i,j \in V} \frac{\partial}{\partial \theta_l} \log [e'_{ij} p(k|j, i)] \quad (5.21)$$

$$= \sum_{k \in C} \sum_{i,j \in V} \frac{1}{e'_{ij}} \frac{\partial e'_{ij}}{\partial \theta_l} p(k|j, i) \quad (5.22)$$

Using eq. 5.13 from the previous section we get $\frac{\partial e'_{ij}}{\partial \theta_l}$ and using the same gradient update step as in eq. 5.16 we can obtain the local maxima of $\ell'(\theta)$.

5.5 Feature vectors ϕ_{ij}

Before discussing about the features let's define few important terms used to define the features.

Local angle θ_{ij} : The local angle $\theta_{ij} \in (0, \pi)$ between two filament segments can be defined as the local absolute difference between two filament directions.

Local absolute difference of intensity $|\mathcal{I}_i - \mathcal{I}_j|$: It is the local average intensity difference for 10 pixels.

Local absolute difference of filament diameter $|d_i - d_j|$: It is the local average difference of diameters for 10 pixels.

The features ϕ_{ij} are based on the local angle θ_{ij} , local absolute difference of intensity $|\mathcal{I}_i - \mathcal{I}_j|$ and local absolute difference of filament diameter $|d_i - d_j|$

between two attached segments. Let's define a function $f_1(\theta_{ij}; k, \theta_c)$ as following

$$f_1(\theta_{ij}; \kappa, \theta_c) = \begin{cases} \frac{\sin \theta_{ij}}{\kappa} & \text{if } \theta_{ij} \in [0, \theta_c) \\ \frac{\sin \theta_c}{\kappa} & \text{if } \theta_{ij} \in [\theta_c, \arccos(\frac{-\sin \theta_c}{\kappa^2}) \\ -k \cos \theta_{ij} & \text{if } \theta_{ij} \in [\arccos(\frac{-\sin \theta_c}{\kappa^2}), \pi] \end{cases}$$

where, θ_{ij} is the local angle between two filaments and κ and θ_c are two parameters. The first feature is defined as $\phi_{ij}^1 = f^1(\theta_{ij}; 5, 50^\circ)$ and the second feature is defined as $\phi_{ij}^2 = f^1(\theta_{ij}; 5, 100^\circ)$.

Figure 5.1: Figure for functions $f^1(\theta_{ij}; \kappa, \theta_c)$ and $f_2(x; \kappa, c)$.

The function $f^1(\theta_{ij}; \kappa, \theta_c)$ capture the following intuition: For two consecutive filaments i and j belonging to the same vessel tree or neuron, they should not bend too much. In other words, the angle between these two filaments, θ_{ij} , tends to be close to π . In our context it amounts to endow its corresponding w_{ij} a higher weight value, and the other way around if there is an abrupt bend (i.e. a tiny weight value if θ_{ij} is an acute angle). The meaning of the two tuning constants, θ_c and κ , are illustrated in Fig. 5.1(A) for both the energy function f and the weight w_{ij} : θ_c marks the turning point in the function f , while κ brings

in the weight difference between highest and the lowest values.

Let's define sigmoid function $f_2(x; \kappa, c)$ centered at c

$$f_2(x; \kappa, c) = \frac{\kappa}{1 + \exp(-x + c)}$$

where x is either $\frac{1}{|\mathcal{I}_i - \mathcal{I}_j|}$ or $\frac{1}{|d_i - d_j|}$ and κ is the asymptotic limit of the sigmoid. We define six more features as shown in Table 5.1.

ϕ_{ij}^3	ϕ_{ij}^4	ϕ_{ij}^5	ϕ_{ij}^6	ϕ_{ij}^7	ϕ_{ij}^8
$f^2(\frac{1}{ \mathcal{I}_i - \mathcal{I}_j }; 5, 3)$	$f^2(\frac{1}{ \mathcal{I}_i - \mathcal{I}_j }; 5, 5)$	$f^2(\frac{1}{ \mathcal{I}_i - \mathcal{I}_j }; 5, 7)$	$f^2(\frac{1}{ d_i - d_j }; 5, 3)$	$f^2(\frac{1}{ d_i - d_j }; 5, 5)$	$f^2(\frac{1}{ d_i - d_j }; 5, 7)$

Table 5.1: Set of features using local absolute difference of intensity $|\mathcal{I}_i - \mathcal{I}_j|$ and local absolute difference of filament diameter $|d_i - d_j|$ between two attached segments.

The intuition of this function is to assign higher weight to a pair of segments which has similar local intensities or diameter and lower weight to a pair which has higher difference in local intensity or diameter. The sigmoid function here works as a soft-thresholding function centred at c . κ is the maximum value of the sigmoid which also has a same value of 5 like the angular features to keep all the features in a same range. We have fixed this value of κ to five for all experiments. In the experimental section we will show the variation of performance with different κ for all the datasets.

5.6 Summary

In summary, I have proposed a novel approach for finding the weight matrix of label propagation method proposed in Chapter-4. Our method is more general compared to other approaches which assumes a undirected graph structure. The algorithm for finding the weight matrix for incomplete and complete data scenario

is also provided. Through empirical evaluation of our approach will be provided in Chapter-7.

Appendix

Proof : E' is row-stochastic for all non-source nodes or
 $E'\mathbf{1} = \mathbf{1}$

Since $E = \sum_{t=0}^{\alpha} (\alpha P^T)^t$, $P^T \mathbf{1} = \mathbf{1}$ and $P^T(P^T \mathbf{1}) = \mathbf{1}$

Therefore,

$$\begin{aligned} E\mathbf{1} &= \sum_{t=0}^{\alpha} (\alpha P^T)^t \mathbf{1} \\ &= \sum_{t=0}^{\alpha} \alpha^t \\ &= \frac{1}{1 - \alpha} \end{aligned}$$

So,

$$(1 - \alpha)E\mathbf{1} = \mathbf{1}$$

$$E'\mathbf{1} = \mathbf{1}$$

Chapter 6

Applications of the Tracing System

In this chapter, two applications of the tracing system will be discussed. The first application is tracing neurons from 2D fluorescence microscopic images of in-vitro mouse embryonic stem cells. Due to the lack of publicly available dataset, we have used the in-house dataset for this application. The dataset preparation and tracing gold standard generation tools and protocols will be discussed in the next section. The second application is tracing retinal blood vessels from 2D retinal fundus images. For this application, we have used the publicly available DRIVE [4] and STARE [90] dataset.

In the next section I will describe the dataset preparation and gold standard generation protocols followed by the pre-processing steps required for each applications.

6.1 Dataset Description

The primary focus of this thesis is tracing filamentary structures in neuronal and retinal images. Two datasets have been collected and annotated for neuronal images and for retinal images two publicly available datasets namely DRIVE and STARE has been used.

6.1.1 Neuronal Datasets

Mouse embryonic stem cells are used to collect images of varying morphologies with an aim that it would be sufficiently representative. The data acquisition follows the following protocol: First the cells are grown in-vitro into neurospheres. Single cells are then collected from these neurospheres and seeded onto PLL/laminin (with $10\mu g/ml$ concentration, from Invitrogen Inc. [125]) coated 13 mm coverslips at 1×10^4 cells/coverslip, and cultured in differentiation medium for 7 days. Then the cells are stained with anti- βIII tubulin (with 1:200 concentration, from Sigma-Aldrich Inc. [126]) with secondary antibodies Alexa Fluor (AF)-488 goat anti-mouse, while the cell nuclei are counter-stained with DAPI. Furthermore, two microscopic imaging systems are used: The first is an Olympus Fluoview FV1000 laser scanning confocal microscope with photomultiplier tube detector (PMT), while the second is an Olympus lx-83 live-cell inverted with Coolsnap HQ2 camera. Both are with a 12-bit A/D converter, and with a magnification of 20x. Image resolution is 0.994 micrometer per pixel. In practice, two batches of neurons are cultured, with *batch 1* of 112 images acquired with the first microscope, and *batch 2* of 98 images obtained using the second microscope. We subsequently refer to these two batches as NeuB1 and NeuB2, respectively. It

is worth noting that images of the second batch are more challenging, with more noticeable background noises and more blurred foregrounds. It nevertheless contributes to our dataset with added variability, which we feel could be very helpful to examine the reliability of the tracing systems.

6.1.2 Retinal Datasets

DRIVE [4]: This dataset is collected and made available publicly by Imaging Science Institute, Netherlands. 40 images are captured with Canon CR5 3CCD camera with a 45° field of view (FOV) and saved as 8-bit compressed JPEG image with 768×584 pixels resolution. Ground truth for segmentation were generated by two specialists. 20 images are used for training whereas 20 images are used for testing.

STARE [90]: This Dataset is collected and made available by National Institute of Health, USA. 20 images are captured with TopCon TRV-50 camera with a 35° field of view (FOV) and saved as 24-bit compressed JPEG image with 605×700 pixels resolution. Ground truth for segmentation were generated by two specialists. 10 images are used for training whereas 10 images are used for testing.

6.1.3 Tracing Ground-truth Generation and Storage

Neuromantic [127] tool has been used to generate the tracing ground truth for both neuronal and retinal datasets. It saves the filament morphology as a tree structure starting from a root point. The morphology is saved as SWC test file where each line of this file denotes a node in the reconstruction morphology. The description of the columns of a SWC file as following.

1	2	3	4	5	6	7
Sample number	Structure Identifier	x position	y position	z position	radius	parent sample

Table 6.1: Description of SWC file

Structure Identifier	Description
0	unidentified
1	soma
2	axon
3	basal dendrite
4	apical dendrite
5	fork point
6	end point
7,8,9	custom

Table 6.2: Description of structure identifier

The first column of each line is the sample number followed by the structure identifier in the next column. The description of structure identifier is shown in Table 6.1. The third, fourth and fifth columns are the x,y and z coordinates respectively whereas the sixth and seventh columns are radius and parent sample number. The parent sample number for a root point is -1 .

For neuronal datasets NeuB1 and NeuB2, each image has multiple neurons therefore each image contains multiple root nodes whereas in retinal datasets each image has one root node.

6.2 Evaluation Criteria

Performance evaluation is carried out using both the widely-used DIADEM score [1] and the relatively recent Netmets score [128]. DIADEM score compares the similarity between a ground truth filamentary object and the corresponding tracing result, and returns a score within $[0, 1]$, where 0 indicates a completely mismatch and 1 means a perfect match. Note the x-y threshold of the DIADEM score is fixed to 30 pixels throughout our experiments.

The Netmets score, on the other hand, provides matching score based on geometric and connectivity related information, which is represented as both false positive and false negative rates. Unlike in DIADEM score, in Netmets lower value is better. In particular, in terms of connectivity false positive and false negative rates, Netmets score penalizes the extra or missing junctions *twice* by considering extra or missing junctions and related filaments separately, which suggests the error rates would often be much higher than that of e.g. the DIADEM score where only *one* round of extra or missing junctions are considered.

The DIADEM score has been the primary choice for performance evaluation of the tracing systems during empirical simulations, while the Netmets score is also used in most of the occasions to provide a complementary source of information.

6.3 Comparison methods

In this section I will talk about some other comparison methods for tracing. The main contribution of this thesis is tracing using graph based methods, therefore we compare our performance with other graph based methods. First we will

compare with two undirected graph methods : Undirected Graphical Models [129] and Learning with Local and Global Consistency method (LLGC) [73]. We will also compare with other directed graph methods : Symetrized Graph Laplacian (SGL) [130] and Zero-mode Free Laplacian (ZFL) [72].

6.3.1 Undirected Graphical Models

In this chapter, the graph based label assignment problem is formulated as a Maximum A Posteriori (MAP) inference problem [129] in Undirected Graphical Models. Local potential function between a pair of node is defined based on the relative orientation of a pair of segments. The overall conditional joint distribution function of all the nodes is formulated based on well-known Hammersley-Clifford theorem. The marginal probability of each node is computed using variational mean field, sum-product and tree-reweighted message passing algorithm, which gives us the maximum probable label assignment for each node.

Problem Set-up

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote the undirected graph where $\mathcal{V} = \{v_1, v_2, \dots, v_n\}$ is the set of nodes and $\mathcal{E}$ is the set of undirected edges between two adjacent nodes, $\mathcal{X} = \{X_1, X_2, \dots, X_k\}$ denotes the k observed random variables and $\mathcal{Y} = \{Y_1, Y_2, \dots, Y_{n-k}\}$ denotes $n - k$ unobserved random variables. If each of the nodes in $\mathcal{G}$ represents a random variable from the set $\{\mathcal{X} \cup \mathcal{Y}\}$ then this is a *Undirected Graphical* or *Markov Random Field* representation of the joint probability distribution $p(\mathcal{X}, \mathcal{Y})$ of the random variable set $\{\mathcal{X} \cup \mathcal{Y}\}$. Let $\mathcal{C} = \{1, 2, \dots, L\}$ denote the L number of possible labels which each random variables can take, $\mathbf{x} = \{x_1, x_2, \dots, x_{n-k}\}$ denote the label assignment set for random variable set $\mathcal{X}$ and $\mathbf{y} = \{y_1, y_2, \dots, y_n\}$

denote the label assignment set for random variable set $\mathcal{Y}$.

The maximum probable label assignment for the unobserved nodes can be formulated as following *Maximum A Posteriori (MAP)* inference problem

$$\mathbf{y}^* = \arg \max_{\mathbf{y}} p(\mathcal{Y} = \mathbf{y} | \mathcal{X} = \mathbf{x}), \quad (6.1)$$

where $\mathbf{y}^*$ is the optimal assignment for the unobserved nodes and $p(\mathcal{Y} = \mathbf{y} | \mathcal{X} = \mathbf{x})$ is the conditional probability of $\mathcal{Y} = \mathbf{y}$ given $\mathcal{X} = \mathbf{x}$. As we know that $p(\mathcal{Y} = \mathbf{y} | \mathcal{X} = \mathbf{x}) = \frac{p(\mathcal{Y}=\mathbf{y}, \mathcal{X}=\mathbf{x})}{p(\mathcal{X}=\mathbf{x})}$ and $p(\mathcal{X} = \mathbf{x}) = 1$, therefore

$$\mathbf{y}^* = \arg \max_{\mathbf{y}} p(\mathcal{Y} = \mathbf{y}, \mathcal{X} = \mathbf{x}) \quad (6.2)$$

For undirected graphical models, following the celebrated Hammersley-Clifford theorem ([131]), the probability function possesses a factorized form, as

$$p(\mathcal{Y} = \mathbf{y}, \mathcal{X} = \mathbf{x}) = \frac{1}{Z} \prod_{(i,j) \in E} \psi_{ij}(y_i, y_j), \quad (6.3)$$

where $\psi_{ij}(y_i, y_j)$ is the local potential function defined over a pair of nodes connected by an edge, Z is the normalization constant called *Partition Function*. Consider $\psi_{ij}(y_i, y_j) = \exp(-E_{ij}(y_i, y_j))$, where $E_{ij}(y_i, y_j)$ refers to the local *Pairwise Energy Function*. By combining (6.2) (6.3), and by the fact that log is a monotone function, the problem can be reformulated as

$$\mathbf{y}^* = \arg \min_{\mathbf{y}} \sum_{(i,j) \in E} E_{ij}(y_i, y_j). \quad (6.4)$$

Figure 6.1: Figure for *Message Passing* algorithm.

MAP Inference on Graphical Models

Map inference on undirected graphical models with cycles is generally a NP-hard problem and approximate solution for this problem can be found either by solving a LP relaxed integer programming problem [132, 133, 134] or by using approximate dynamic programming approach using message passing on max-product algebra [135, 136, 137]. In this thesis three different message passing approaches has been considered while the other approaches for solving the problem are left for the future scopes of this work. In the subsequent sections, I will first discuss the general approach for message passing algorithms and then I will continue with the three variants of message passing algorithms.

General Framework of Message Passing Algorithms

The max-marginals are used to compute the MAP inference of a joint distribution and for a tree structured distribution max-marginals give the exact MAP inference [136, 137, 135]. For a general graph with cycles pseudo max-marginals are defined in [134] and is used for approximate MAP inference. The message

passing framework is used to compute the max-marginals in an iterative manner with a complexity of $O(n)$. In general it starts with random initiation of messages over the edges in both directions. These messages are updated using some local messages coming from the neighbouring nodes and then it is repeated until some convergence criteria is met. Different message update equations give us different message passing algorithms. In this work we have used variational mean field, sum-product/max-product and tree-reweighted message passing algorithms. In the results and discussion chapter we have discussed the advantage of one algorithm over others for our tracing applications.

In Fig 6.3, a general framework of a message passing algorithm on a graph with cycles has been shown. The message from t to s node has been shown as μ_{ts} . After convergence the max-marginal probability of j^{th} variable taking x_j assignment can be computed as the product of all the incoming messages as below.

$$p(x_j) \propto \prod_{i \in N(j)} \mu_{ij}(x_j) \quad (6.5)$$

Variational Mean Field Message Passing

Variational mean field starts with a variational optimization formulation of our MAP inference problem and with a very coarse assumption that every random variable is independent of each other gives a very simple update equation as following.

$$\mu_s(x_s) \leftarrow \frac{1}{1 + \exp(\sum_{t \in N(s)} E_{st}(x_s, x_t) \mu_t(x_t))} \quad \forall x_t, x_s \in \mathcal{C} \quad (6.6)$$

Sum-product Message Passing

Figure 6.2: Figure for energy function $\psi_{ij}(x_i, x_j)$

It's a generalization of celebrated Viterbi algorithm [138] which exploits dynamic programming and the distributive property of multiplication over summation [135]. The update equation is as following.

$$\mu_{ts}(x_s) \leftarrow \sum_{x'_t \in \mathcal{C}} [\psi_{st}(x_s, x'_t) \prod_{i \in N(t) \setminus s} \mu_{it}(x_t)] \quad (6.7)$$

Tree-reweighted Message Passing

In their work [134] Wainwright et. al has proposed a new way of message passing via agreement on spanning trees of a graph. If $\mathcal{T}(\mathcal{G})$ is the set of all spanning trees of a graph $\mathcal{G}$, ρ_{ij} is the probability that an edge (i, j) belongs to a randomly selected spanning tree from $\mathcal{T}(\mathcal{G})$ and κ is a normalization constant then message update equation is as following.

$$\mu_{ts}(x_s) \leftarrow \kappa \sum_{x'_t \in \mathcal{C}} \left\{ \exp \left[-\frac{E_{st}(x_s, x'_t)}{\rho_{st}} \right] \frac{\prod_{i \in N(t) \setminus s} [\mu_{it}(x_t)]^{\rho_{it}}}{[\mu_{st}(x_t)]^{(1-\rho_{st})}} \right\} \quad (6.8)$$

Pairwise Potential Function $\psi_{ij}(x_i, x_j)$

The pairwise potential function $\psi_{ij}(x_i, x_j)$ can be defined as following

$$\psi_{ij}(x_i, x_j) = \begin{cases} \exp(-f(\theta_{ij})) & \text{if } x_i = x_j \\ \exp(-1/(f(\theta_{ij}) + \epsilon)) & \text{if } x_i \neq x_j \end{cases} \quad (6.9)$$

where

$$f(\theta_{ij}) = \begin{cases} -\frac{\sin(\theta_{ij})}{k} & \text{if } \theta_{ij} \in [0, \theta_c) \\ -\frac{\sin(\theta_c)}{k} & \text{if } \theta_{ij} \in [\theta_c, \arccos(\frac{-\sin \theta_c}{k^2}) \\ k \cos(\theta_{ij}) & \text{if } \theta_{ij} \in [\arccos(\frac{-\sin \theta_c}{k^2}), \pi), \end{cases} \quad (6.10)$$

where, θ_{ij} is the local angle between two filaments and k and θ_c are two parameters. The function $f(\theta_{ij})$ capture the following intuition: For two consecutive filaments i and j belonging to the same vessel tree or neuron, they should not bend too much. In other words, the angle between these two filaments, θ , tends to be close to π . In our context it amounts to endow its corresponding $\psi_{ij}(x_i, x_j)$ a higher value when $x_i = x_j$ and lower value when $x_i \neq x_j$, and it is the other way around if there is an abrupt bend between two filaments (i.e. a tiny weight value if θ_{ij} is an acute angle). The meaning of the two tuning constants, θ_c and k , are illustrated in Fig. 6.3.1 for both f and the potential function ψ_{ij} : θ_c marks the turning point in the function f , while k brings in the weight difference between highest and the lowest values. The selection of these two parameters θ_c and k are discussed in the experimental section.

6.3.2 Other Methods and Softwares for Tracing

These are few other comparison methods on directed and undirected graph.

Learning with Local and Global Consistency method (LLGC) This method [73] is also an undirected graph method where the similarity matrix is proposed as following

$$S = (I - \alpha D^{-1/2} W D^{1/2})^{-1}$$

Symetrized Graph Laplacian (SGL) This method [130] is based on the symmetrized graph laplacian by Dr. Fan Chung [74] where the similarity matrix is defined as following

$$S = (I - \alpha \Theta)^{-1}$$

where Θ is the symmetrized graph laplacian [74].

Zero-mode Free Laplacian (ZFL) This method [72] is also a directed graph method where the similarity matrix is defined as Zero-mode Free Laplacian .

$$L^+ = D - W + \frac{W_{++}}{n^2} \mathbf{e} \mathbf{e}^T$$

where $W_{++} = \sum_{ij} W_{ij}$ and $\mathbf{e}$ is a all-one vector of size n .

Neurocyto Neurocyto [139] is a freely available tool for neuron tracing which works with 2D neuronal images in fully automatic fashion. The method is based on level-set method and mumford shah model.

Metamorph NX (MMNX) This is a widely used commercial software from Molecular Devices [140] which claims to solve the crossover issue. The exact details of the implementation is unknown but it requires 14 parameters to tune to obtain optimal results.

HCA Vision (HCA) The neurite tracing module of HCA Vision [141] can trace 2D neurons. We have used this method for qualitative comparison only.

6.4 Quantitative Results and Discussion

Here in this section I will discuss about the quantitative results of the methods discussed in this thesis.

6.4.1 Tracing Results for MFTD

This is the brief of different MFTD methods discussed in this thesis.

- MFTDa : $S_1 = (I + \alpha L)^{-1}$ with $W \leftarrow \frac{W}{W_{\max}}$.
- MFTDb : $S_1 = (I + \alpha L)^{-1}$ with $W \leftarrow D^{-1}W$.
- MFTDc : $S_2 = (L + \alpha \tilde{L})^{-1}$ with $W \leftarrow \frac{W}{W_{\max}}$.
- MFTDd : $S_2 = (L + \alpha \tilde{L})^{-1}$ with $W \leftarrow D^{-1}W$.

Quantitative tracing results for MFTD_a, MFTD_b, MFTD_c and MFTD_d are shown in tables 6.3 to 6.7. Both from DIADEM score and Netmets score we can see that MFTD_a works better than others, whereas MFTD_b is the second best among four methods. The better performance of MFTD_a over MFTD_b is due to the row-wise normalization of MFTD_b which tends to bias the solution toward local minimums. For all the datasets, MFTD_c and MFTD_d fails to perform well.

Figure 6.3 shows the variation of performance with the internal parameters α and β in terms of DIADEM score. For MFTD_a, any value of α greater than 1 gives good performance whereas the performance reduces drastically if α is close to 0.

For MFTD_b, $\alpha = 10$ usually gives best performance but $\alpha = 1$ is also quite close to the optimum performance. For other values of α MFTD_b fails to perform well. MFTD_c gives optimum performance for β values close to 0 and the performance reduces when β values are greater than 1. For MFTD_d, $\beta = 10^{-1}, 10^0, 10^1$ usually gives better performance.

Figure 6.3: Effect of varying the internal parameters α or β of our four MFTD tracing variants. From top to bottom, each row presents one of the four datasets, namely NeuB1, NeuB2, DRIVE, and STARE; From left to right, each column is for one of the four MFTD tracing variants. In each panel, α or β are presented in x-axis, while y-axis shows the DIADDEM score performance.

	MFTDa	MFTDb	MFTDc	MFTDd
NeuB1	0.52	0.51	0.51	0.41
NeuB2	0.26	0.24	0.25	0.19
DRIVE	0.79	0.76	0.75	0.63
STARE	0.42	0.41	0.42	0.26

Table 6.3: DIADEM scores for $MFTD_a$, $MFTD_b$, $MFTD_c$ and $MFTD_d$ on the neuronal and the retinal datasets.

	MFTDa	MFTDb	MFTDc	MFTDd
NeuB1	0.21	0.22	0.21	0.25
NeuB2	0.38	0.41	0.39	0.45
DRIVE	0.09	0.11	0.12	0.15
STARE	0.26	0.28	0.28	0.31

Table 6.4: One of the four Netmets scores, geometric false positive rate, on the neuronal and the retinal datasets.

	MFTDa	MFTDb	MFTDc	MFTDd
NeuB1	0.20	0.22	0.21	0.26
NeuB2	0.36	0.38	0.37	0.43
DRIVE	0.08	0.10	0.10	0.13
STARE	0.25	0.27	0.26	0.30

Table 6.5: One of the four Netmets scores, geometric false negative rate, on the neuronal and retinal datasets.

	MFTDa	MFTDb	MFTDc	MFTDd
NeuB1	0.77	0.78	0.78	0.81
NeuB2	0.81	0.85	0.83	0.91
DRIVE	0.62	0.64	0.63	0.69
STARE	0.79	0.80	0.80	0.83

Table 6.6: One of the four Netmets scores, connectivity false positive rate, on the neuronal and the retinal datasets.

	MFTDa	MFTDb	MFTDc	MFTDd
NeuB1	0.79	0.80	0.80	0.85
NeuB2	0.83	0.85	0.86	0.91
DRIVE	0.65	0.67	0.66	0.69
STARE	0.79	0.80	0.81	0.88

Table 6.7: One of the four Netmets scores, connectivity false negativity rate, on the neuronal and the retinal datasets.

6.4.2 Tracing with Undirected Graphical Models

Quantitative tracing results for three inference methods of graphical model namely - Variational Mean Field (MF), Tree Re-weighted Belief Propagation (TRW) and Max-product loopy belief propagation (LBP) is shown in Table 6.8. Both from DIADEM score and Netmets score we can see that TRW works better than others whereas LBP is the second best among these three. MF performs worst among the three because of the coarse assumption that each nodes are independent of each other.

Figure 6.4 shows the variation of performance with the parameters k and θ_c in terms of DIADEM score. For all the datasets, it is observed that $k = 5$ and $\theta_c = 90^\circ$ gives optimum performance.

Figure 6.4: Variation of performance with the parameters k and θ_c

	MF	TRW	LBP
NeuB1	0.38	0.46	0.42
NeuB2	0.15	0.23	0.19
DRIVE	0.58	0.68	0.63
STARE	0.29	0.35	0.32

Table 6.8: DIADEM scores of three inference methods of graphical model on the neuronal and the retinal datasets.

	MF	TRW	LBP
NeuB1	0.30	0.23	0.26
NeuB2	0.51	0.47	0.48
DRIVE	0.20	0.13	0.18
STARE	0.37	0.31	0.34

Table 6.9: One of the four Netmets scores, geometric false positive rate, on the neuronal and the retinal datasets.

	MF	TRW	LBP
NeuB1	0.29	0.25	0.27
NeuB2	0.47	0.43	0.45
DRIVE	0.12	0.18	0.15
STARE	0.31	0.33	0.31

Table 6.10: One of the four Netmets scores, geometric false negative rate, on the neuronal and retinal datasets.

	MF	TRW	LBP
NeuB1	0.85	0.81	0.82
NeuB2	0.91	0.88	0.90
DRIVE	0.71	0.75	0.73
STARE	0.81	0.88	0.83

Table 6.11: One of the four Netmets scores, connectivity false positive rate, on the neuronal and the retinal datasets.

	MF	TRW	LBP
NeuB1	0.88	0.83	0.86
NeuB2	0.94	0.92	0.93
DRIVE	0.71	0.69	0.68
STARE	0.89	0.85	0.86

Table 6.12: One of the four Netmets scores, connectivity false negativity rate, on the neuronal and the retinal datasets.

6.4.3 Tracing Results with Parametric Absorbing Markov Chain

As we know that the Absorbing Markov Chain (AMC) is a special case of Parametric Absorbing Markov Chain (PAMC), we present the results together and compare their performance in Table 6.13. For AMC, I have used single angular feature ϕ_{ij}^1 , single diameter feature ϕ_{ij}^4 and single image intensity feature ϕ_{ij}^7 separately and present their results in the first three row of Table 6.13. For PAMC, random weights (denoted as PAMC_Ra), learned weights with incomplete data (denoted as PAMC_In) and learned weights with complete data (denoted as PAMC_Co) are presented in 4th, 5th and 6th row of Table 6.13 respectively.

	NeuB1	NeuB2	DRIVE	STARE
$AMC(\phi_{ij}^1)$	0.5113	0.2434	0.7683	0.4173
$AMC(\phi_{ij}^4)$	0.3468	0.2215	0.6965	0.3267
$AMC(\phi_{ij}^7)$	0.3261	0.1847	0.6517	0.3173
$PAMC_Ra$	0.3942	0.2136	0.6236	0.3373
$PAMC_In$	0.5646	0.2891	0.8078	0.4920
$PAMC_Co$	0.6155	0.3160	0.8364	0.5323

Table 6.13: Performance of Absorbing Markov Chain with singles features and with multiples features with random weights or learned parameters.

It is observed that AMC with angular feature (ϕ_{ij}^1) works better than other features. PAMC_Ra doesn't work well but it is better than AMC with diameter or intensity feature. PAMC_Co is the best method among all whereas PAMC_In is the second best method. The better performance of PAMC_Co compared to PAMC_In owes to the fact that PAMC_Co has completely labeled data for training whereas PAMC_In has only one labeled data point per-class for training.

Independent softwares	MMNX, Neurocyto
Methods using same pre-processing steps	PAMC_Co, PAMC_In, MFTD_a, AMC, SGL, ZFL, LLGC, TRW
Directed graph methods	PAMC_Co, PAMC_In, MFTD_a, AMC, SGL, ZFL
Undirected graph methods	LLGC, TRW
Methods using angular feature only	MFTD_a, AMC, SGL, ZFL, LLGC, TRW
Methods using all the features	PAMC_Co, PAMC_In

Table 6.14: Brief description of the methods compared.

6.4.4 Comparison with other methods

In this section, I will discuss about the comparative performance of the methods discussed in this thesis with some other label propagation methods and some commercial as well as open source tracing softwares. The best methods among the variants discussed in this thesis will be compared with other label propagation methods on undirected graph like learning with local and global consistency method (LLGC) [73] and on directed graph like zero mode free laplacian method (ZFL) [72] and symmetrized graph laplacian method(SGL) [71]. We will also compare our algorithms with a widely used commercial software Metamorph NX [140] (MMNX) and a opensource tool NeuroCyto [139]. For SGL, ZFL and LLGC methods, segmentation and preprocessing steps are same as our methods, whereas MMNX and Neurocyto works independently. It is also important to notice that SGL, ZFL and LLGC are primarily unsupervised learning methods without training therefore we have only used angular feature ϕ_{ij}^1 for experiments. Out of all, following methods used angular feature only : MFTD_a, AMC, TRW, SGL, ZFL and LLGC and following methods used all the features : PAMC_Co and PAMC_In. Overall PAMC_Co, PAMC_In, MFTD_a, AMC, SGL and ZFL are

directed graph methods whereas LLGC and TRW are undirected graph methods. For all the datasets, PAMC_Co performs best among all followed by PAMC_In. MFTD_a is the best method among those using only one feature. Among all the methods using undirected graph, LLGC is usually as par or better than the directed graph methods SGL or ZFL. MFTD_a or AMC is always better than both the undirected graph methods. The commercial software MMNX only works with neuronal datasets NeuB1 and NeuB2 and usually is slightly worse than undirected graph based TRW. The opensource tool Neurocyto performs worst among all.

	PAMC_Co	PAMC_In	MFTDa	AMC	SGL	ZFL	LLGC	TRW	MMNX	NeuroCyto
NeuB1	0.62	0.56	0.52	0.51	0.45	0.47	0.50	0.46	0.43	0.36
NeuB2	0.32	0.29	0.26	0.24	0.21	0.22	0.21	0.23	0.22	0.14
DRIVE	0.84	0.81	0.79	0.76	0.68	0.73	0.71	0.68	-	-
STARE	0.53	0.49	0.42	0.41	0.35	0.39	0.38	0.35	-	-

Table 6.15: DIADEM scores of comparison methods on the neuronal and the retinal datasets. The first four columns are variants of our tracing system, while the next six columns are reported by replacing the tracing component of our system with different methods. The last two columns are results of two existing neuronal tracing systems.

	PAMC_Co	PAMC_In	MFTDa	AMC	SGL	ZFL	LLGC	TRW	MMNX	NeuroCyto
NeuB1	0.10	0.14	0.21	0.22	0.24	0.23	0.22	0.23	0.25	0.33
NeuB2	0.33	0.36	0.38	0.41	0.49	0.43	0.46	0.47	0.43	0.52
DRIVE	0.02	0.04	0.09	0.11	0.14	0.13	0.12	0.13	-	-
STARE	0.15	0.21	0.26	0.28	0.31	0.29	0.27	0.31	-	-

Table 6.16: One of the four Netmets scores, geometric false positive rate, on the neuronal and the retinal datasets.

	PAMC_Co	PAMC_In	MFTDa	AMC	SGL	ZFL	LLGC	TRW	MMNX	NeuroCyto
NeuB1	0.11	0.14	0.20	0.22	0.25	0.23	0.23	0.25	0.26	0.32
NeuB2	0.32	0.35	0.36	0.38	0.42	0.40	0.43	0.43	0.42	0.51
DRIVE	0.04	0.05	0.08	0.10	0.13	0.12	0.12	0.15	-	-
STARE	0.18	0.21	0.25	0.27	0.29	0.28	0.35	0.33	-	-

Table 6.17: One of the four Netmets scores, geometric false negative rate, on the neuronal and retinal datasets.

	PAMC_Co	PAMC_In	MFTDa	AMC	SGL	ZFL	LLGC	TRW	MMNX	NeuroCyto
NeuB1	0.58	0.63	0.77	0.7	0.81	0.79	0.80	0.81	0.82	0.89
NeuB2	0.75	0.79	0.81	0.85	0.89	0.87	0.88	0.88	0.86	0.93
DRIVE	0.49	0.53	0.62	0.64	0.68	0.67	0.76	0.75	-	-
STARE	0.62	0.68	0.79	0.80	0.83	0.82	0.81	0.83	-	-

Table 6.18: One of the four Netmets scores, connectivity false positive rate, on the neuronal and the retinal datasets.

	PAMC_Co	PAMC_In	MFTDa	AMC	SGL	ZFL	LLGC	TRW	MMNX	NeuroCyto
NeuB1	0.62	0.67	0.79	0.80	0.84	0.83	0.81	0.83	0.85	0.88
NeuB2	0.79	0.81	0.83	0.85	0.93	0.91	0.89	0.92	0.91	0.96
DRIVE	0.53	0.59	0.65	0.67	0.69	0.68	0.67	0.69	-	-
STARE	0.65	0.69	0.79	0.80	0.86	0.83	0.81	0.85	-	-

Table 6.19: One of the four Netmets scores, connectivity false negativity rate, on the neuronal and the retinal datasets.

6.5 Qualitative Results and Discussion

In this section, I will visually compare our tracing results with some widely used commercial softwares like Metamorph NX (MMNX) and HCA Vision (HCA). In Figure 6.5, the top row first image is an example image from the NeuB1 neuronal image dataset whereas the second image is the gold standard for tracing. In this image, the neurons are shown in different colors, whereas the red transparent circle shows the location of the *crossover* and touchings. The number beside the circle shows the index of the *crossover* or touching. Clearly we can see that *crossover* 1 is only resolved by PAMC_Co, whereas *crossover* 2 is resolved by PAMC_Co, PAMC_In, AMC and MFTDa. *Crossover* 3 is resolved by PAMC_Co, PAMC_In, AMC and MFTDa whereas *crossover* 4 is resolved by PAMC_Co, PAMC_In, PAMC_Ra, AMC and MFTDa. In brief none of the *crossovers* are resolved by MMNX and HCA whereas all the *crossovers* are resolved by PAMC_Co.

In the second example from NeuB1 dataset, shown in Figure 6.6, all the *crossovers* are resolved by PAMC_Co whereas for PAMC_In all except 2 is resolved. None of the *crossovers* are resolved by PAMC_Ra whereas for AMC, *crossovers* 1, 4 and 5 are resolved. For this example MFTDa works as good as PAMC_In whereas for TRW *crossovers* 1 and 5 are resolved successfully. For HCA, none of the *crossovers* are resolved successfully but for MMNX all the *crossovers* are resolved successfully.

In a more difficult example shown in Figure 6.7, *crossovers* 1 and 2 are resolved by PAMC_Co whereas for PAMC_In only one *crossovers* is resolved successfully. For AMC, two *crossovers* are resolved successfully whereas for MFTDa

only one crossover is resolved successfully. For TRW, HCA and MMNX none of the crossovers are resolved successfully. Bad tracing result for this image owes to the fact that is image much more complicated and it requires more sophisticated segmentation and skeleton extraction methods for more cleaner skeletons. Wrong skeletons are converted to digraph with wrong topological connectivity, therefore the tracing methods cannot give proper results.

Figure 6.8 shows an example from NeuB2. For this image, PAMC_Co has resolved *crossovers* 1 and 3, whereas PAMC_In has resolved *crossover* 3 only. For AMC none of the crossovers are resolved correctly whereas for MFTDa and TRW *crossover* 1 is resolved correctly. Following the same trend as before none of the crossovers are resolved correctly for HCA where as for MMNX *crossover* 1 is resolved correctly.

Another example from NeuB2 is shown in Figure 6.9. For this image all the *crossovers* and touchings are resolved correctly by PAMC_Co whereas for PAMC_In, *crossovers* 1, 3 and 4 are resolved correctly. Following the same trend as before none of the *crossovers* or touchings are resolved correctly by PAMC_Ra. For AMC two *crossovers* are resolved correctly whereas for MFTDa three *crossovers* are resolved correctly. For TRW, three *crossovers* are resolved correctly whereas following the same trend as before none of the crossovers are resolved correctly by HCA and MMNX.

One more example from NeuB2 is shown in Figure 6.10. For this image all the *crossovers* and touchings are resolved correctly by PAMC_Co and PAMC_In. One *crossovers* is resolved correctly by PAMC_Ra whereas for AMC two *crossovers* are resolved correctly. For MFTDa all three *crossovers* are resolved correctly. For TRW, two *crossovers* are resolved correctly whereas following the same trend

as before none of the crossovers are resolved correctly by HCA and MMNX.

In the example from DRIVE dataset, shown in Figure 6.11, seven out of twelve *crossovers* are resolved by PAMC_Co whereas for PAMC_In four out of twelve are resolved. Just one *crossover* is resolved by PAMC_Ra whereas for AMC, four *crossovers* are resolved. Both for MFTDa and TRW four *crossovers* are resolved correctly. None of the methods could solve the *crossovers* 7, 8, 9, 10 and 11 because the root nodes couldn't be detected correctly after the preprocessing steps. Correct detection of root nodes is very crucial to our method therefore if they are detected incorrectly it effects the tracing result.

In another example from DRIVE dataset, shown in Figure 6.12, ten out of fifteen *crossovers* are resolved by PAMC_Co whereas for PAMC_In nine out of fifteen are resolved. None of the *crossover* are resolved by PAMC_Ra whereas for AMC, two *crossovers* are resolved. For MFTDa four *crossovers* are resolved correctly whereas for TRW two *crossovers* are resolved. None of the methods could solve the *crossovers* 3, 4, 13, 14 and 15 because of the problem in root node detection.

In the example from STARE dataset, shown in Figure 6.13, seven out of eight *crossovers* are resolved by PAMC_Co whereas for PAMC_In four out of eight are resolved. Just one *crossover* is resolved by PAMC_Ra whereas for AMC, four *crossovers* are resolved. For MFTDa, four crossovers are resolved and for TRW none of the *crossovers* are resolved correctly. None of the methods could resolve the *crossover* 1 because of segmentation error.

In another example from STARE dataset, shown in Figure 6.14, three of seven *crossovers* are resolved by PAMC_Co and PAMC_In. None of the *crossovers* are resolved by PAMC_Ra, whereas one *crossover* is resolved by by AMC, MFTDa

and TRW. None of the methods could solve the *crossovers* 5 and 7 because of segmentation error whereas 1 and 2 couldn't be resolved because of the problem in root node detection.

Figure 6.5: The tracing results for various methods from an image in NeuB1. The red transparent circle in gold standard tracing denotes the *crossover* situation.

Figure 6.6: The tracing results for various methods from an image in NeuB1. The red transparent circle in gold standard tracing denotes the *crossover* situation.

Figure 6.7: The tracing results for various methods from an image in NeuB1. The red transparent circle in gold standard tracing denotes the *crossover* situation.

Figure 6.8: The tracing results for various methods from an image in NeuB2. The red transparent circle in gold standard tracing denotes the *crossover* situation.

Figure 6.9: The tracing results for various methods from an image in NeuB2. The red transparent circle in gold standard tracing denotes the *crossover* situation.

Figure 6.10: The tracing results for various methods from an image in NeuB2. The red transparent circle in gold standard tracing denotes the *crossover* situation.

Figure 6.11: The tracing results for various methods from an image in DRIVE dataset. The red transparent circle in gold standard tracing denotes the *crossover* situation.

Figure 6.12: The tracing results for various methods from an image in DRIVE dataset. The red transparent circle in gold standard tracing denotes the *crossover* situation.

Figure 6.13: The tracing results for various methods from an image in STARE dataset. The red transparent circle in gold standard tracing denotes the *crossover* situation.

Figure 6.14: The tracing results for various methods from an image in STARE dataset. The red transparent circle in gold standard tracing denotes the *crossover* situation.

6.6 Additional Applications on Large Scale Datasets

In this section, I will discuss about some other applications of label propagation methods like the citation problem and the social network problem [142]. For these datasets, the graph size is quite large which are in the range of few millions of nodes. These applications will show the usability of our approach on different fields as well as demonstrate the robustness of our algorithm. For the citation problem three datasets are employed: US Patent [143], CoRA [97] and CiteSeer [97]. For social network we consider the Google+ and the Twitter datasets of [142], where the goal is to identify the social circles of individual users. Our approach is also compared with eight state-of-the-art methods that directly work with directed graphs:

- Network-only Bayes Classifier (NBC) [144].
- Network-only Link Based classifier (NLB) [145].
- Class Distribution Relational Neighbor classifier (CDRN) [97].
- Weighted Vote Relational Neighbor classifier (WVRN) [97].
- Digraph variant of the Commute Time Kernel classifier (CTKd) [98].
- Symmetrized Graph Laplacian (SGL) [71].
- Zero-mode Free Laplacian (ZFL) [75].
- Sum Over Path covariance kernel (SOP) [105].

Out of these methods, four (NBC, NLB, CDRN, and WVRN) are implemented by NetKit [97] in Java, SOP [105] is obtained from the authors, while the rest

(CTKd, SGL, ZFL, and Ours) are implemented by ourselves in MATLAB. Note the original Commute Time Kernel classifier (or CTKu) only works with undirected graphs. To work with digraphs, we instead replace its original undirected graph Laplacian with the symmetrized digraph Laplacian of [74]. As a result, this variant is referred to as CTKd in this thesis. To ensure fair evaluations, the internal parameters of the comparison methods are either set to as is from the authors' original source code, or as suggested by their respective authors: For example, according to [71], the regularization parameter is set to 0.1 and the jumping factor used in teleporting random walk is set to 0.01. In particular, for the four methods implemented by NetKit, the uniform local-classifier is used as the "local" model, and for collective inference relaxation labeling has been used for NBC, CDRN, and WVRN, while iterative classification is used for NLB. This setting is reported [97] to deliver the best performance.

In term of evaluation metric, the micro-averaged accuracy (AC) [146] is adopted in most of the experiments as the accuracy measure, which is the sum of all true positive counts divided by the total number of instances. For the social network applications where there is a need to evaluate the partial correctness of predicted labels in the multi-label setting, a modified version of F_1 score [147] is used:

$$F_1 = \frac{1}{nk} \sum_{i,k} \frac{2|\tilde{y}_{ik}^* \cap z_{ik}|}{|\tilde{y}_{ik}^*| + |z_{ik}|} \quad (6.11)$$

where $\tilde{y}_{ik}^*$ and z_{ik} are predicted and true labels for the k -th label of the i -th instance.

Figure 6.15: Comparisons of accuracy & efficiency on the citation problem (i.e. datasets of US Patent, CoRA and CiteseerX). Plots of the first column display the micro-averaged accuracy (AC) on US Patent, CoRA and CiteseerX, respectively; Plots of the second column show the corresponding CPU time spent. In all plots, the horizontal axis denotes the label ratio (percentage of labeled nodes) varying from 10% to 90% with 10% increment. See text for details.

Citation Problem Paper citations naturally form a digraph and the aim here is to predict a prescribed topic for each of the unlabeled papers at hand, provided a few are labeled a priori. We first examine our approach on a *large-scale* dataset, US Patent [143], meanwhile we also conduct evaluations on the other two standard datasets: CoRA [97] and CiteseerX [148]. The US Patent dataset consists of 13 million directed edges connecting 2.7 million nodes that can be categorized into 418 distinct topics. The CoRA dataset contains a citation digraph of 2,708 nodes and 5,429 directed links (edges) on computer science research papers spanning 7 topics. CiteseerX is another citation dataset of 3,312 papers and 4,732 citations (directed edges) from 6 categories. Here the adjacency matrices are adopted as their corresponding weight matrices.

To evaluate the system performance against varying size of labeled nodes in digraphs, the following strategy is adopted: For each of the K classes, certain percentage (i.e. label ratio, also denoted as r) of instances (i.e. nodes) in this class is uniformly selected as labeled nodes – this gives one empirical data sample. This procedure is repeated 50 times to produce an averaged performance estimate. We then vary the label ratio r from 10% to 90% with 10% increment, and compare the averaged performance (AC) of competing methods as presented in Figure 6.15 first column. Note that during these experiments, when the labeled nodes are selected, the nodes with “zero-knowledge” components [97] will be temporarily removed from consideration, as are the graph nodes that have no directed path connecting to any node in $\mathcal{V}_l$. Besides, Figure 6.15 second column displays the average CPU time for each of the competing methods: For US Patent dataset, the timing is of a single run, while for CoRA and CiteseerX, the timing is instead recorded over 1,000 runs to ensure all timing values are above zero in the log-

second scale.

Overall our approach consistently outperforms the state-of-the-arts while consuming the least amount of computational resource. For US Patent dataset, NLB, CTKd, SGL, ZFL, and SOP fail to work as these methods require dense matrices of 2.7 million by 2.7 million entries. So only our results are compared with only NBC, WVRN and CDRN on US Patent dataset. In term of accuracy, our method performs consistently the best and with a very significant gap (10%-20%) comparing to the others across different labeling ratios. The consistently superior performance of our approach still carries on for CoRA and CiteseerX datasets: ZFL becomes the second best method, which is followed by WVRN and others, while CTKd and SOP often produce the least favorable results. Note the performance of SGL, a closely related method of ours, is almost at the lower end of the middle regime of performers. We attribute this to the fact that both CoRA and CiteseerX are not very dense digraphs and their edge weights are quite asymmetric, which seems to be difficult for SGL, as source information is not well kept after utilizing teleporting Markov chain as well as the symmetrized graph Laplacian [74]. Our results are also align with existing evaluations¹, although the results are not directly comparable, due to the randomized nature when sampling instances for each of the label ratios. In term of CPU time as in Figure 6.15 second column, our method consumes significantly less time compare to the other methods, and is with a significant gap from the second best, WVRN. On the flip side, ZFL, and SOP are the most computational intensive of all, which is closely followed by SGL. In particular, our method is shown (Figure 6.15 (B))

¹E.g. Fig.6 of [97] on CoRA where the best performer delivers around 0.8-0.9 by varying the label ratios.

Figure 6.16: Comparisons of F1-Score on the multi-label problem of Google+ and Twitter datasets. In both plots, the horizontal axis denotes the label ratio (percentage of labeled nodes) varying from 10% to 90% with 10% increment. See text for details.

to be at least over 100 times more efficient when working with the large-scale US patent dataset, where it takes merely around 26 seconds for our approach to make predictions on this million-node dataset using a standard desktop. Note that comparing to ZFL, our method delivers consistently better results on CoRA and CiteseerX, and runs much more efficiently. In particular our method also works well with large-scale datasets such as the US Patent, which is however not possible to use ZFL.

Social Network Application It is a non-trivial task to identify social circles in social networks. Usually such a problem involves many different labels (circles) and is of large size. In particular, we consider the problem of identifying 327 social circles in the Google+ dataset, and 3,127 social circles in the Twitter dataset. both datasets are from [142]. The Google+ dataset consists of a graph of 1.4 million nodes and 30 million directed edges belonging to 133 users. As only part of the nodes are with their ground-truth labels, those nodes with no label information are trimmed away – we are thus left with 19,327 nodes and 3,294,465 directed

edges. Similarly, the Twitter dataset has 81,306 nodes and 2.4 million directed edges from 1,000 users. After removing nodes with no label information, we obtain a digraph with 19,270 nodes and 490,667 directed edges. For experimental evaluation, the labelling ratios are varied from 10% to 90% with 10% increment, and the F_1 score in (6.11) are computed. NBC, NLB, CDRN and WVRN are only able to work with single-label classification problem. In the meantime, the teleporting random walks introduced in SGL tends to wash away weak signals, which seems to significantly deteriorate the performance over all label ratios. As a result, our approach are compared with three remaining methods: CTKd, ZFL and SOP, as presented in Figure 6.16. Our approach clearly outperforms the other three state-of-the-arts by a very large margin in both datasets. For the Google+ dataset, ours produces a series of increasing F1-scores of 0.7–0.95 with the increment of label ratios r , where ZFL and SOP are the best runner-ups with combined best performance of merely 0.25–0.65 during the same range of r . CTKd seems to perform least well. These phenomenons are similarly observed for the Twitter dataset. The drop of performance in the comparison methods seems to be attributed to the combined influences of large label size and large data size (In the large-sized US patent dataset we also observe a rather significant margin between our method and the best runner-up). The superior performance of our approach, on the other hand, suggests that our approach is particularly reliable when dealing with large-sized data with many labels.

Empirical Time-complexity of Batch vs. Online Updates Here we focus on the dynamic graph scenario where a small fraction of the digraph structure might change over time, being either a single edge weight, or inserting/deleting

Figure 6.17: Empirical time-complexity of batch (4.3) vs. online (7,8) updates of the fundamental matrix, E . (A) and (B) show the CPU-time (log-seconds) of batch update vs. online update for changing one row using (4.7), and for inserting / deleting a new node with (4.8), respectively. See text for details.

Figure 6.18: Averaged absolute difference between online and batch updates. The red curve (with a square mark) and the blue curve (with a star mark) shows the average differences for changing one row using (4.7), and for inserting/deleting a new node with (4.8), respectively.

a single node. In our context, this boils down to efficient computation of the fundamental matrix E . Our approach is capable of addressing these changes in E , as presented in (4.7) and (4.8) for online updates, as well as in (4.3) for batch update. Ideally, the online updates are expected to be carried out more efficiently and the results should be the same as of batch update. To show this, we design the following synesthetic experiments: The weight matrix of a sparse digraph of size n is randomly generated with its E matrix computed. This is followed by either changing a single edge weight, or inserting/deleting a single node from the digraph, which subsequently gives E' . Its respectively online update is then computed by (4.7) or (4.8), vs. the batch update of (4.3). The above process is repeated 20 times, for each of the following four different digraph sizes, namely $n \in \{10, 100, 1000, 10,000\}$, and the median running time is displayed in Figure 6.17(A) and (B). Note that this comparison is not entirely fair as the implementation set-up is less favorable for the online update: To compute (4.3) for batch update, the Matlab implementation of UMFPACK direct solver is highly optimized and runs on multi-cores, while our implementation of the online updates, namely (4.7) and (4.8) are in Matlab script as is (without any optimization). Nevertheless, as presented in Figure 6.17 the online updates runs always an order of magnitude faster. Besides, empirically the numerical difference value between batch and online updates are numerically negligible. As displayed in Figure 6.18, on average the absolute difference value is always below 10^{-5} in the above mentioned experiments. In addition, this numerical error decreases dramatically with the increase of digraph sizes.

Figure 6.19: Comparison with state-of-the-art methods based on undirected graphs.

Comparison with Undirected-graph based Methods So far we have compared our approach to a number of methods that can directly work with digraphs. One may still wonder how conventional undirected-graph based methods would perform in our context. For this purpose we implement in matlab two state-of-the-art such methods, namely the original Commute Time Kernel classifier (CTKu) [98], and the Learning with Local and Global Consistency (LLGC) method in [73], both operate on undirected graphs. The comparison is performed on CoRA and CiteseerX datasets with results presented in Figure 6.19 when the label ratios are changed from 0.1 to 0.9. Our approach maintains a clear performance gap of about 10% overall for both datasets. LLGC and CTKu alternate in both datasets as being the runner-up, with a further margin of around 10% over the worst performing one of the three methods.

The Effect of α We also provide empirical analysis to study the effect of varying α value to the performance of the proposed system: As presented in

Figure 6.20: Robustness of our system vs. changing α values between .01 and .99. (A) For CoRA and CiteseerX, the performance of our system is rather stable (with around .001 variation) when α is within .01 and 0.9, and start to decrease slightly (around .01 variation) when α value goes beyond .9. (B) The performance remains constant regardless of α values in vessel tracing problems.

Figure 6.21: Robustness of our system with respect to varying label ratio. The error bar of each labeling ratio (r) displays 5 – 95 percentile of accuracy when α values are systematically sampled between .01 and .99 with an .01 increment. The narrow deviations from median as shown in the error bar (usually less than 2%) clearly suggest that our system is rather stable against changes of α values.

Figure 6.22: Empirical discriminative ability of our approach. Intra- and inter-class accumulated affinities are displayed over different datasets. Intuitively the larger the gap between intra- and inter-class, the better its performance would be. See text for details.

Figures 6.20, the empirical performance accross a wide range of applications is relatively stable against changing α values, especially during the range of .09 to .25. This observation is further confirmed in Figure 6.21 with varying label ratios, where different α values usually result in less than 2% variations in its performance. The insensitive pattern of α is experienced throughout empirical experiments. This motivate us to simply fix α to certain value (0.1) during the rest experiments. We note in the passing that performance degradation is to be expected when α taking extreme values being too close to either 0 or 1, since which renders E to be too close to either the identity matrix or an ill-conditioned matrix, respectively.

Analyzing Discriminative Ability of Our Approach Further, we compare the intra- and inter-class accumulated affinity scores over different datasets, which offers an empirical explanation for the discriminative ability of our approach. The

results displayed in Figure 6.22 are obtained as follows: For each non-zero (i, j) -th entry in E there is a directed path connected both nodes. Now group all entries in E into two sets: Those with both nodes belonging to the same class (i.e. intra-class), and the rest that each from a different classes (i.e. inter-class). Then accumulate the scores within each set and normalize – which produces the final scores. The intra-class score is expected to outnumber the inter-class one, and the larger the gap (or ratio) between the two suggests a better discriminative ability on the particular dataset. As revealed in Figure 6.22, the ratios are all very large across various datasets used in this chapter, which indeed suggests that our algorithm is expected to deliver good performance regardless of any particular set of input labels.

Chapter 7

Conclusion and Future Work

7.1 Thesis Summary

In this thesis, a general framework for tracing filamentary structure has been proposed. Thorough empirical evaluation has been done on two applications: neuronal image tracing and retinal image tracing. Quantitative comparisons among the proposed methods, other state of art methods and commercial softwares are done using widely used DIADEM score as well as Netmets score. For qualitative comparison, visual examples of tracing results from different methods and softwares are shown in figures with a focus on the ability of solving the *crossover* issue. It is observed that both qualitatively and quantitatively Parametric Absorbing Markov Chain with Complete data (PAMC-Co) works best among all followed by Parametric Absorbing Markov Chain with Incomplete data (PAMC-In). The third and the fourth best method are MFTDa and AMC respectively. Among the comparison methods on directed graphs ZFL works best whereas among the methods on undirected graphs LLGC is the best method. Among the softwares Metamorph NX (MMNX) works better than others. There are primarily two source of errors for our methods: missing filament due to missing segmentation

and incorrect tracing due to wrong topological connectivity during skeleton extraction and graph conversion. I believe that a better segmentation and skeleton extraction might be able to solve many errors.

7.2 Recommendations for Future Work

For future direction, I would like to recommend to try with better segmentation and skeleton extraction algorithm to obtain a graph with more accurate topological connectivity.

Another direction of work would be working with Undirected Graphical Models. Learning with Graphical Models is not done in thesis which can be considered as future direction of research. Graphical Models with higher order graphs can also be exploited for better tracing performance.

Another direction of research would be using the methods discussed in this thesis with 3D data with crossovers as well as other application on 2D data like aerial images of road networks. Working with 2d+t or 3D+t data would be another direction of future work for my thesis.

Finally the methods we have learned can be used to simulate filamentary structures which might give us additional insight into the growth of filamentary structures in particular for clinical applications.

References

- [1] T. A. Gillette, K. M. Brown, and G. A. Ascoli. The diadem metric: comparing multiple reconstructions of the same neuron. *Neuroinformatics*, 9(2-3):233–245, 2011. [v](#), [2](#), [14](#), [90](#)
- [2] E. Turetken, F. Benmansour, and P.Fua. Automated reconstruction of tree structures using path classifiers and mixed integer programming. In *CVPR*, 2012. [v](#), [2](#), [7](#)
- [3] Guy Shechter, Guy Shechter Abc, Frederic Devernay, Elliot R. McVeigh, Eve Coste-Maniere, and Elliot R. Mcveigh Ca. Temporal tracking of 3d coronary arteries in projection angiograms. 2002. [v](#), [2](#)
- [4] M. Abramoff, M. Niemeijer, M. Viergever, and B. Ginneken. Ridge based vessel segmentation in color images of the retina. *IEEE Trans. Med. Imag.*, 23(4):501–509, 2004. [v](#), [2](#), [14](#), [86](#), [88](#)
- [5] M. F. Bear, B. W. Connors, and M. A. Paradiso. *Neuroscience: Exploring the Brain*. Williams & Wilkins, 1996. [1](#)
- [6] Eric R. Kandel, James H. Schwarz, and Thomas M. Jessel. *Principles of Neural Science*. McGraw-Hill, 4th ed. edition, 2000. [1](#)

REFERENCES

- [7] J.W. Moore, A.E. Stuart, and J.G. Nicholls. *From Neuron to Brain 4th Ed.* Sinauer Associates Incorporated, 2001. [1](#)
- [8] JG Nicholls, AR Martin, BG Wallace, and PA Fuchs. *From neuron to brain.* Sinauer Associates, 4th ed edition, 2001. [1](#)
- [9] D Purves, GJ Augustine, D Fitzpatrick, WC Hall, AS LaMantia, JO McNamara, and LE White. *Neuroscience.* Sinauer Associates, 4th ed edition, 2007. [1](#)
- [10] J. DeFelipe. From the connectome to the synaptome: an epic love story. *Science*, 330(6008):1198–1201, 2010. [1](#)
- [11] Hanchuan Peng. Bioimage informatics: a new area of engineering biology. *Bioinformatics*, 24(17):1827–1836, 2008. [2](#)
- [12] K Viswanath and D. McGavin. Diabetic retinopathy: Clinical findings and management. *Community Eye Health*, 16(46):21–24, 2003. [3](#)
- [13] M. Martinez-Perez, A. Hughes and A. Stanton, S. Thom, N. Chapman, A. Bharath, and K. Parker. Retinal vascular tree morphology: a semi-automatic quantification. *IEEE Trans Biomed Eng.*, 49(8):912–917, 2002. [3](#), [9](#)
- [14] L. King, A. Stanton, P. Sever, S. Thom, and A. Hughes. Arteriolar length-diameter (l:d) ratio: a geometric parameter of the retinal vasculature diagnostic of hypertension. *J Hum Hypertens*, 10(6):417–424, 1996. [3](#)

REFERENCES

- [15] Assessment of regional myocardial performance from biplane coronary cineangiograms. *The American Journal of Cardiology*, 27(5):529 – 537, 1971. [3](#)
- [16] A.A. Young, P.J. Hunter, and B.H. Smaill. Estimation of epicardial strain using the motions of coronary bifurcations in biplane cineangiography. *Biomedical Engineering, IEEE Transactions on*, 39(5):526–531, May 1992. [3](#)
- [17] G. Coppini, M. Demi, P. Marraccini, and A. L’Abbate. 3-d heart motion from x-ray angiography. In *Computers in Cardiology 1995*, pages 71–74, Sep 1995. [3](#)
- [18] J. Puentes, C. Roux, M. Garreau, and J.L. Coatrieux. Dynamic feature extraction of coronary artery motion using dsa image sequences. *Medical Imaging, IEEE Transactions on*, 17(6):857–871, Dec 1998. [3](#)
- [19] N. Guggenheim, P.A. Dorsaz, P.A. Doriot, C. Suilen, F. Chappuis, and W. Rutishauser. 3d determination of the intravascular volume and flow of coronary arteries. *International Journal of Bio-Medical Computing*, 35(1):13 – 23, 1994. [3](#)
- [20] A. Wahle, E. Wellnhofer, I. Mugaragu, H.U. Saner, H. Oswald, and E. Fleck. Assessment of diffuse coronary artery disease by quantitative analysis of coronary morphology based upon 3-d reconstruction from biplane angiograms. *Medical Imaging, IEEE Transactions on*, 14(2):230–241, Jun 1995. [3](#)

REFERENCES

- [21] Jiuxiang Hu, Anshuman Razdan, John C. Femiani, Ming Cui, and Peter Wonka. P.: Road network extraction and intersection detection from aerial images by tracking road footprints. *IEEE Transactions on Geoscience and Remote Sensing*, pages 4144–4157, 2000. [3](#)
- [22] J.B. Mena. State of the art on automatic road extraction for {GIS} update: a novel classification. *Pattern Recognition Letters*, 24(16):3037 – 3058, 2003. [3](#)
- [23] M. Xia and B. Liu. Image registration by "super-curves". *Image Processing, IEEE Transactions on*, 13(5):720–732, May 2004. [3](#)
- [24] E.M. Glaser and H. Van Der Loos. A semi-automatic computer-microscope for the analysis of neuronal morphology. *Biomedical Engineering, IEEE Transactions on*, BME-12(1):22–31, Jan 1965. [4](#)
- [25] J.J. Capowski and M.J. Sedivec. Accurate computer reconstruction and graphics display of complex neurons utilizing state-of-the-art interactive techniques. *Computers and Biomedical Research*, 14(6):518 – 532, 1981. [4](#)
- [26] Thomas S. Ford-Holevinski, Todd A. Dahlberg, and Bernard W. Agranoff. A microcomputer-based image analyzer for quantitating neurite outgrowth. *Brain Research*, 368(2):339 – 346, 1986. [4](#)
- [27] S.R. Aylward and E. Bullitt. Initialization, noise, singularities, and scale in height ridge traversal for tubular object centerline extraction. *Medical Imaging, IEEE Transactions on*, 21(2):61–75, Feb 2002. [5](#)

REFERENCES

- [28] K. Al-Kofahi, S. Lasek, D. Szarowski, C. Pace, G. Nagy, J. Turner, and B. Roysam. Rapid automated three-dimensional tracing of neurons from confocal image stacks. *Trans. Info. Tech. Biomed.*, 6(2):171–187, 2002. [5](#)
- [29] Wenyun He, Thomas A. Hamilton, Andrew R. Cohen, Timothy J. Holmes, Christopher Pace, Donald H. Szarowski, James N. Turner, and Badrinath Roysam. Automated three-dimensional tracing of neurons in confocal and brightfield images. *Microsc. Microanal.*, 9:296–310, 2003. [5](#)
- [30] LaurentD. Cohen and Ron Kimmel. Global minimum for active contour models: A minimal path approach. *International Journal of Computer Vision*, 24(1):57–78, 1997. [5](#)
- [31] Xiaosong Yuan, JoshuaT. Trachtenberg, SteveM. Potter, and Badrinath Roysam. Mdl constrained 3-d grayscale skeletonization algorithm for automated extraction of dendrites and spines from fluorescence confocal images. *Neuroinformatics*, 7(4):213–232, 2009. [5](#), [6](#)
- [32] Erik H. W. Meijering, Mathews Jacob, J.-C. Floyd Sarria, and Michael Unser. A novel approach to neurite tracing in fluorescence microscopy images. In *SIP*, pages 491–495. IASTED/ACTA Press, 2003. [5](#)
- [33] Hanchuan Peng, Zongcai Ruan, Deniz Atasoy, and Scott Sternson. Automatic reconstruction of 3d neuron structures using a graph-augmented deformable model. *Bioinformatics*, 26(12):i38–i46, 2010. [5](#)
- [34] Fethallah Benmansour and LaurentD. Cohen. Tubular structure segmentation based on minimal path method and anisotropic enhancement. *International Journal of Computer Vision*, 92(2):192–210, 2011. [5](#)

REFERENCES

- [35] G. Gonzalez, E. Turetken, F. Fleuret, and P. Fua. Delineating trees in noisy 2d images and 3d image stacks. In *CVPR*, 2010. 6
- [36] S Schmitt, JF Evers, C Duch, M Scholz, and K Obermayer. New methods for the computer-assisted 3-d reconstruction of neurons from confocal image stacks. *NeuroImage*, 23:1283–1298, 2004. 6
- [37] Z. Vasilkoski and A. Stepanyants. Detection of the optimal neuron traces in confocal microscopy images. *Journal of Neuroscience Methods*, 178:197–204, 2009. 6
- [38] G. A. Ascoli and K. Svoboda. Diadem challenge. available at <http://www.diademchallenge.org/>. 6
- [39] Arunachalam Narayanaswamy, Yu Wang, and Badrinath Roysam. 3-d image pre-processing algorithms for improved automated tracing of neuronal arbors. *Neuroinformatics*, 9(2-3):219–231, 2011. 6
- [40] Robert E. Schapire. A brief introduction to boosting. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence - Volume 2*, pages 1401–1406, 1999. 7
- [41] P. Chothani, V. Mehta, and A. Stepanyants. Automated tracing of neurites from light microscopy stacks of images. *Neuroinformatics*, 9(2-3):263–278, 2011. 7, 8
- [42] Ting Zhao, Jun Xie, Fernando Amat, Nathan Clack, Parvez Ahammad, Hanchuan Peng, Fuhui Long, and Eugene Myers. Automated reconstruction

- of neuronal morphology based on local geometrical and global structural models. *Neuroinformatics*, 9(2-3):247–261, 2011. [7](#)
- [43] E. Turetken, F. Benmansour, and P. Fua. Automated reconstruction of tree structures using path classifiers and mixed integer programming. In *Conference on Computer Vision and Pattern Recognition*, June 2012. [7](#), [8](#)
- [44] E. Turetken, G. Gonzalez, C. Blum, and P. Fua. Automated reconstruction of dendritic and axonal trees by global optimization with geometric priors. *Neuroinformatics*, 9:279–302, May 2011. [7](#), [8](#)
- [45] E. Turetken, C. Blum, G. Gonzalez, and P. Fua. Reconstructing geometrically consistent tree structures from noisy images. In *International Conference on Medical Image Computing and Computer Assisted Intervention*, 2010. [7](#), [8](#)
- [46] G. Gonzalez, E. Turetken, F. Fleuret, and P. Fua. Delineating trees in noisy 2d images and 3d image stacks. In *Conference on Computer Vision and Pattern Recognition*, June 2010. [7](#), [8](#)
- [47] Erhan Bas and Deniz Erdogmus. Principal curves as skeletons of tubular objects. *Neuroinformatics*, 9(2-3):181–191, 2011. [7](#)
- [48] A. Mendonca and A. Campilho. Segmentation of retinal blood vessels by combining the detection of centerlines and morphological reconstruction. *IEEE Trans. Med. Imag.*, 25(9):1200–1213, 2006. [8](#)
- [49] Garg S, J. Sivaswamy, and S. Chandra. Unsupervised curvature-based retinal vessel segmentation. In *ISBI*, pages 1200–1213, 2007. [8](#)

REFERENCES

- [50] L. Espona, M. Carreira, M. Penedo, and M. Ortega. Retinal vessel tree segmentation using a deformable contour model. In *ICPR*, 2008. [8](#)
- [51] M. Martinez-Perez, A. Hughes, S. Thom, A. Bharath, and K. Parker. Segmentation of blood vessels from red-free and uorescein retinal images. *Med. Image Anal.*, 11(1):47–61, 2007. [8](#)
- [52] J. Soares, J. Leandro, R. Cesar, H. Jelinek, and M. Cree. Retinal vessel segmentation using the 2d gabor wavelet and supervised classification. *IEEE Trans. Med. Imag.*, 25(9):1214–1222, 2007. [8](#)
- [53] D. Marin, A. Aquino, M. Gegundez-Arias, and J. Brav. A new supervised method for blood vessel segmentation in retinal images by using gray-level and moment invariants-based features. *IEEE Trans. Med. Imag.*, 30(1):146–158, 2011. [8](#)
- [54] P. Bankhead, C. Scholfield, J. McGeown, and T. Curtis. Fast retinal vessel detection and measurement using wavelets and edge location refinement. *PLoS ONE*, 7(3):e32435, 2012. [8](#)
- [55] L. Wang and A. Bhalerao. Model based segmentation for retinal fundus images. In *Scandinavian conf. on Image analysis*, pages 422–429, 2003. [8](#)
- [56] E. Ricci and R. Perfetti. Retinal blood vessel segmentation using line operators and support vector classification. *IEEE Trans. Med. Imag.*, 26(10):1357–1365, 2007. [8](#), [9](#)

- [57] X. Xu, M. Niemeijer, Q. Song, M. Sonka, M. Garvin, J. Reinhardt, and M. Abramoff. Vessel boundary delineation on fundus images using graph-based approach. pages 1184–1191, 2011. [8](#)
- [58] A. Can, H. Shen, J. Turner, H. Tanenbaum, B. Roysam, and B. Roysam. Rapid automated tracing and feature extraction from retinal fundus images using direct exploratory algorithms. pages 125–138, 1999. [8](#)
- [59] E. Grisan, A. Pesce, A. Giani, M. Foracchia, and A. Ruggeri. A new tracking system for the robust extraction of retinal vessel structure. In *IEEE EMBS*, 2004. [8](#)
- [60] E. Bekkers, R. Duits, B. ter Haar Romeny, and T. Berendschot. A new retinal vessel tracking method based on orientation scores. *CoRR*, abs/1212.3530, 2012. [8](#)
- [61] B. Al-Diri, A. Hunter, and D. Steel. An active contour model for segmenting and measuring retinal vessels. *IEEE Trans Med. Imaging*, 28(9):1488–1497, 2009. [8](#), [9](#)
- [62] Y.A. Tolias and S.M. Panas. A fuzzy vessel tracking algorithm for retinal images based on fuzzy clustering. *Medical Imaging, IEEE Transactions on*, 17(2):263–273, 1998. [8](#)
- [63] Q. Lau, M. Lee, W. Hsu, and T. Wong. Simultaneously identifying all true vessels from segmented retinal images. *IEEE Transactions on Biomedical Engineering*, 2013. [9](#)

- [64] E. Turetken, G. Gonzalez, C. Blum, and P. Fua. Automated reconstruction of dendritic and axonal trees by global optimization with geometric priors. *Neuroinformatics*, 9:279–302, 2011. [12](#)
- [65] A.D. Hoover, V. Kouznetsova, and M. Goldbaum. Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response. *Medical Imaging, IEEE Transactions on*, 19(3):203–210, March 2000. [14](#)
- [66] J. Soares, J. Leandro, R. Cesar, H. Jelinek, and M. Cree. Retinal vessel segmentation using the 2d gabor wavelet and supervised classification. *IEEE Trans. Med. Imag.*, 25(9):1214–1222, 2007. [18](#), [20](#), [21](#), [22](#)
- [67] P. Bankhead, C. Scholfield, J. McGeown, and T. Curtis. Fast retinal vessel detection and measurement using wavelets and edge location refinement. *PLoS ONE*, 7(3):e32435, 2012. [21](#), [22](#)
- [68] M. Law and A. Chung. Three dimensional curvilinear structure detection using optimally oriented flux. In *ECCV*, 2008. [21](#), [22](#)
- [69] R. Frangi, W. Niessen, K. Vincken, and M. Viergever. Multiscale vessel enhancement filtering. In *MICCAI*, 1998. [21](#), [22](#)
- [70] D. Marin, A. Aquino, M. Gegundez-Arias, and J. Brav. A new supervised method for blood vessel segmentation in retinal images by using gray-level and moment invariants-based features. *IEEE Trans. Med. Imag.*, 30(1):146–158, 2011. [21](#), [22](#)

- [71] D. Zhou, J. Huang, and B. Schölkopf. Learning from labeled and unlabeled data on a directed graph. In *ICML*, 2005. [27](#), [32](#), [40](#), [41](#), [42](#), [52](#), [54](#), [104](#), [119](#), [120](#)
- [72] X. Cai, H. Wang, H. Huang, and C. Ding. Simultaneous image classification and annotation via biased random walk on tri-relational graph. In *ECCV*, 2012. [27](#), [40](#), [91](#), [97](#), [104](#)
- [73] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Scholkopf. Learning with local and global consistency. In *NIPS*, 2004. [27](#), [40](#), [41](#), [52](#), [53](#), [66](#), [67](#), [91](#), [97](#), [104](#), [128](#)
- [74] Fan Chung. Laplacians and the cheeger inequality for directed graphs. *Annals of Combinatorics*, 9(1):1–19, 2005. [27](#), [41](#), [52](#), [97](#), [120](#), [123](#)
- [75] H. Wang, C. Ding, and H. Huang. Directed graph learning via high-order co-linkage analysis. In *ECML*, 2010. [27](#), [40](#), [41](#), [119](#)
- [76] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. In *WWW*, 1998. [27](#), [42](#), [44](#), [52](#)
- [77] R. P. Agaev and P. Yu. Chebotarev. Spanning forests of a digraph and their applications. *Automation and Remote Control*, 62(3):443–466, 2001. [29](#), [30](#)
- [78] Frank Harary, Robert Norman, and Dorwin Cartwright. *Structural models : an introduction to the theory of directed graphs*. Wiley, 1965. [29](#)
- [79] R. Brualdi and H. Ryser. *Combinatorial Matrix Theory*. Cambridge Uni. Press, 1991. [29](#)

REFERENCES

- [80] P. Yu. Chebotarev and R. P. Agaev. Forest matrices around the laplacian matrix. *Linear Algebra and its Applications*, 356(1-3):253–247, 2002. [30](#)
- [81] C.D. Meyer and M.W. Stadelmaier. Singular m-matrices and inverse positivity. *Linear Algebra and its Applications*, 22:139–156, 1978. [30](#)
- [82] T. Davis. Algorithm 832: Umfpack v4.3—an unsymmetric-pattern multifrontal method. *ACM Trans. Math. Softw.*, 30:196–199, 2004. [32](#), [54](#)
- [83] J. Demmel, S. Eisenstat, J. Gilbert, X. Li, and J. Liu. A supernodal approach to sparse partial pivoting. *SIAM J. Matr. Anal. App.*, 20:720–755, 1999. [32](#), [54](#)
- [84] C. Paige and M. Saunders. Lsqr: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw.*, 8:43–71, 1982. [32](#), [54](#)
- [85] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2nd edition, 2003. [32](#), [54](#)
- [86] Luc Bégin, Pascal Germain, François Laviolette, and Jean-Francis Roy. PAC-Bayesian theory for transductive learning. In *AISTAT*, 2014. [32](#), [33](#), [34](#)
- [87] P. Derbeko, R. El-Yaniv, and R. Meir. Explicit learning curves for transduction and application to clustering and compression algorithms. *J. Artif. Intell. Res.*, 22:117–142, 2004. [32](#), [57](#), [72](#)
- [88] P. Chebotarev and E. Shamis. The matrix-forest theorem and measuring relations in small social groups. *Automat. Remote Control*, 58(9):1505–14, 1997. [37](#), [53](#), [69](#)

- [89] R. Agaev and P. Chebotarev. Spanning forests of a digraph and their applications. *Automat. Remote Control*, 62(3):443–466, 2001. [37](#), [53](#), [69](#)
- [90] A. Hoover, V. Kouznetsova, and M. Goldbaum. Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response. *IEEE Trans Med Imag*, 19(3):203–210, 2000. [40](#), [86](#), [88](#)
- [91] E. Meijering, M. Jacob, J. Sarria, P. Steiner, H. Hirling, and M. Unser. Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry A*, 58(2):167–76, 2004. [40](#)
- [92] J. Staal, M. Abramoff, M. Niemeijer, M. Viergever, and B. van Ginneken. Ridge based vessel segmentation in color images of the retina. *IEEE Trans. Med. Imag.*, 23(4):501–509, 2004. [40](#)
- [93] E. Turetken, C. Becker, P. Glowacki, F. Benmansour, and P. Fua. Detecting irregular curvilinear structures in gray scale and color imagery using multi-directional oriented flux. In *ICCV*, 2013. [40](#)
- [94] E. Turetken, F. Benmansour, B. Andres, H. Pfister, and P. Fua. Reconstructing loopy curvilinear structures using integer programming. In *CVPR*, 2013. [40](#)
- [95] M. Chen, M. Liu, J. Liu, and X. Tang. Isoperimetric cut on a directed graph. In *CVPR*, 2010. [40](#)
- [96] A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Inf. Retr.*, 3(2):127–163, 2000. [40](#)

- [97] S. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *JMLR*, 8:935–983, 2007. [40](#), [119](#), [120](#), [122](#), [123](#)
- [98] F. Fouss, K. Francoise, L. Yen, A. Pirotte, and M. Saerens. An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. *Neural Network*, 31:53–72, 2012. [40](#), [52](#), [119](#), [128](#)
- [99] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. *Semi-Supervised Learning*. MIT Press, 2006. [40](#)
- [100] X. Zhu and A. Goldberg. *Introduction to Semi-Supervised Learning*. Morgan & Claypool, 2009. [40](#)
- [101] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, 2003. [40](#)
- [102] Y. Bengio, O. Delalleau, and N. Le Roux. Label propagation and quadratic criterion. In O. Chapelle, B. Schölkopf, and A. Zien, editors, *Semi-Supervised Learning*, pages 193–216. MIT Press, 2006. [40](#)
- [103] X. Wu, Z. Li, A. So, J. Wright, and S. Chang. Learning with partially absorbing random walks. In *NIPS*, 2012. [40](#), [42](#), [53](#), [67](#), [68](#), [69](#)
- [104] J. Wang, T. Jebara, and S. Chang. Semi-supervised learning using greedy max-cut. *J. Mach. Learn. Res.*, 14(1):771–800, 2013. [40](#)
- [105] A. Mantrach, L. Yen, J. Callut, K. Françoisse, M. Shimbo, and M. Saerens. The sum-over-paths covariance kernel: A novel covariance measure between

- nodes of a directed graph. *IEEE Trans. PAMI*, 32(6):1112–1126, 2010. [40](#), [41](#), [119](#)
- [106] T. Gartner, Q. Le, S. Burton, A. Smola, and S. Vishwanathan. Large-scale multiclass transduction. In *NIPS*, 2005. [41](#)
- [107] A. Subramanya and J. Bilmes. Semi-supervised learning with measure propagation. *JMLR*, 12:3311–70, 2011. [41](#)
- [108] John G. Kemeny and J. Laurie Snell. *Finite Markov Chains*. Springer, 1976. [41](#), [46](#), [47](#), [58](#)
- [109] Xiaojin Zhu, Andrew Goldberg, Jurgen Van, and Gael David Andrzejewski. Improving diversity in ranking using absorbing random walks. In *HLT-NAACL*, 2007. [42](#)
- [110] P. Chebotarev and E. Shamis. On proximity measures for graph vertices. *Automat. and Remote Control*, 59:1443–1459, 1998. [51](#)
- [111] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002. [52](#)
- [112] R. El-Yaniv and D. Pechyony. Transductive Rademacher complexity and its applications. *J. Artif. Intell. Res.*, 35:193–234, 2009. [54](#), [55](#), [70](#)
- [113] P. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *JLMR*, 3:463–482, 2002. [56](#)
- [114] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996. [58](#)

REFERENCES

- [115] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, pages 912–919, 2003. [74](#)
- [116] Ashish Kapoor, Yuan (alan) Qi, Hyungil Ahn, and Rosalind W. Picard. Hyperparameter and kernel learning for graph based semi-supervised classification. In *Advances in NIPS*, pages 627–634, 2005. [74](#)
- [117] Xinhua Zhang and Wee Sun Lee. Hyperparameter learning for graph based semi-supervised learning algorithms. In Bernhard Schölkopf, John C. Platt, and Thomas Hoffman, editors, *NIPS*, pages 1585–1592. MIT Press, 2006. [74](#)
- [118] Fei Wang and Changshui Zhang. Label propagation through linear neighborhoods. In *ICML*, 2006. [74](#)
- [119] Masayuki Karasuyama and Hiroshi Mamitsuka. Manifold-based similarity adaptation for label propagation. In Christopher J. C. Burges, Lon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *NIPS*, pages 1547–1555, 2013. [74](#)
- [120] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000. [74](#)
- [121] Samuel I. Daitch, Jonathan A. Kelner, and Daniel A. Spielman. Fitting a graph to vector data. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML*, pages 201–208, 2009. [74](#)

REFERENCES

- [122] Hong Cheng, Zicheng Liu, and Jie Yang 0001. Sparsity induced similarity measure for label propagation. In *ICCV*, pages 317–324. IEEE, 2009. [74](#)
- [123] Wei Liu, Junfeng He, and Shih-Fu Chang. Large graph construction for scalable semi-supervised learning. In *ICML*, 2010. [74](#)
- [124] Jing Chen and Yang Liu. Locally linear embedding: a survey. *Artificial Intelligence Review*, 36(1):29–48, 2011. [74](#)
- [125] Invitrogen inc. <http://www.lifetechnologies.com/>, 2015. [87](#)
- [126] Sigma-Aldrich inc. <http://www.sigmaaldrich.com/>, 2015. [87](#)
- [127] D. Myatt, T. Hadlington, G. Ascoli, and S. Nasuto. Neuromantic - from semi-manual to semi automatic reconstruction of neuron morphology. *Frontiers in Neuroinformatics*, 6(4):1–14, 2012. [88](#)
- [128] David Mayerich, Chris Bjornsson, Jonathan Taylor, and Badrinath Roysam. Netmets: software for quantifying and visualizing errors in biological network segmentation. *BMC Bioinformatics*, 13(S8):1–19, 2012. [90](#)
- [129] M. Wainwright and M. Jordan. *Graphical models, exponential families, and variational inference*. Foundations and trends in machine learning. Now Publisher, Hanover, 2008. [91](#)
- [130] Dengyong Zhou, Jiayuan Huang, and Bernhard Scholkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems (NIPS) 19*, page 2006. MIT Press, 2006. [91](#), [97](#)

REFERENCES

- [131] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. [92](#)
- [132] Dimitris Bertsimas and John Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1st edition, 1997. [93](#)
- [133] László Lovász. Review of the book by alexander schrijver: Combinatorial optimization: Polyhedra and efficiency. *Oper. Res. Lett.*, 33(4):437–440, 2005. [93](#)
- [134] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Map estimation via agreement on trees: Message-passing and linear programming. *IEEE Trans. Inf. Theor.*, 51(11):3697–3717, 2005. [93](#), [95](#)
- [135] S.M. Aji and R.J. McEliece. The generalized distributive law. *Information Theory, IEEE Transactions on*, 46(2):325–343, Mar 2000. [93](#), [95](#)
- [136] Robert G. Cowell, Steffen L. Lauritzen, A. Philip David, and David J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer-Verlag New York, Inc., 1st edition, 1999. [93](#)
- [137] A.P. Dawid. Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, 2(1):25–36, 1992. [93](#)
- [138] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Inf. Theor.*, 13(2). [95](#)
- [139] W. Yu, H. Lee, s. Hariharan, W. Bu, and S. Ahmed. Quantitative neurite outgrowth measurement based on image segmentation with topological dependence. *Cytometry Part A*, 75A(4):289–297, 2009. [97](#), [104](#)

REFERENCES

- [140] Metamorph NX. <http://www.moleculardevices.com/systems/metamorph-research-imaging/metamorph-nx-microscopy-automation-image-analysis-software>, 2015. [97](#), [104](#)
- [141] Pascal Vallotton, Ryan Lagerstrom, Changming Sun, Michael Buckley, Dadong Wang, Melanie De Silva, Seong-Seng Tan, and Jenny M. Gunnensen. Automated analysis of neurite branching in cultured cortical neurons using hca-vision. *Cytometry Part A*, 71A(10):889–895, 2007. [98](#)
- [142] Julian J. McAuley and Jure Leskovec. Learning to discover social circles in ego networks. In *NIPS*, pages 548–556, 2012. [119](#), [124](#)
- [143] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: Densification laws, shrinking diameters and possible explanations. In *In KDD*, pages 177–187. ACM Press, 2005. [119](#), [122](#)
- [144] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *SIGMOD*, 1998. [119](#)
- [145] Q. Lu and L. Getoor. Link-based classification. In *ICML*, 2003. [119](#)
- [146] P. Sen and L. Getoor. link based classification. Technical report, CS department, Univ of Maryland, 2007. CS-TR-4858. [120](#)
- [147] Shantanu Godbole and Sunita Sarawagi. Discriminative methods for multi-labeled classification. In *PAKDD*, pages 22–30. Springer, 2004. [120](#)
- [148] C. Giles, K. Bollacker, and S. Lawrence. Citeseer: an automatic citation indexing system. In *Int. Conf. on Digital Libraries*, 1998. [122](#)