RELATIONAL DATA CLUSTERING ALGORITHMS WITH BIOMEDICAL APPLICATIONS

A Dissertation

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

MOHAMMED A. KHALILIA

Mihail Popescu, Dissertation Supervisor

MAY 2014

The undersigned, appointed by the dean of the Graduate School, have examined the dissertation entitled

RELATIONAL DATA CLUSTERING ALGORITHMS WITH BIOMEDICAL APPLICATIONS

presented by Mohammed A. Khalilia,

candidate for the degree of Doctor of Philosophy

and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Mihail Popescu

Dr. James M. Keller

Dr. Marjorie Skubic

Dr. Glen Cameron

DEDICATION

I dedicate this dissertation to my family and my closest friends. This work would not have been possible without the support and encouragement of my loving family, my dad, Abdul-Sattar, my mom, Amal, my sisters Mais, Samah and my brother Faisal.

I cannot also forget the support of my closest friends who helped me throughout this process. Thank you for listening and for the advice. I will always appreciate all they have done, especially Keila Pena-Hernandez for introducing me to my advisor, Dr. Mihail Popescu, with whom I have enjoyed doing research with.

ACKNOWLEDGMENTS

I would like to start by thanking my committee members, James Keller, Marjorie Skubic and Glen Cameron, for their support and advice.

In addition, I would like to thank Dr. James Keller (Red Sam) and my advisor, Dr. Mihail Popescu for giving me the honor to work with James Bezdek (Mad Sam). It is probably a once in a lifetime opportunity to work with James Bezdek, James Keller and Mihail Popescu together on clustering problems. It has been a great honor to coauthor a paper with these three researchers. Working with James Bezdek provided a new set of skills in math and relational clustering algorithms. The research collaboration with James Bezdek also taught me how to appreciate data, regardless of its size, better understand relational clustering algorithms and how to write more creatively.

Last but not least, I would like to thank my very supportive advisor, Dr. Mihail Popescu. As we all know, the road that leads to a PhD is full of obstacles. Therefore, without the support and patience of my advisor, I wouldn't have succeeded. I appreciate all of his effort in the last few years and the countless number of hours spent reading, revising, encouraging and advising. Thank you for giving me the opportunity to work on diverse and exciting projects, for giving me the chance to travel and meet the brightest leaders in the field. I am especially grateful for meeting those key researchers with whom I later collaborated.

My experiences and memories at the University of Missouri are forever engraved. These memories will be held as my treasure. I sincerely thank you all for your service and determination in helping make my academic dream come true.

Table of Contents

ACKNOWLEDGMENTS ii
LIST OF ILLUSTRATIONS vi
LIST OF TABLES ix
LIST OF ABBREVIATIONS xi
ABSTRACT xiii

1 Introduction 1
1.1. The Problem 2
1.2. Contributions 5
1.2.1. Pattern Recognition in Relational Data 5
1.2.2. Predicting Disease Risks from Unbalanced Data 6
1.2.3. Quantifying the Amount of Care Coordination from Nursing Notes Using NLP and Ontologies 7
1.3. Outline 7
1.4. List of Relevant Publications 8

2 Relational Fuzzy Self-Organizing Map 10
2.1. Introduction 10
2.2. Self-Organizing Map 15
2.3. Relational Self-Organizing Map 17
2.4. Fuzzy Self-Organizing Map 18
2.5. Relational Fuzzy Self-Organizing Map 22
2.6. Neurons Sharing Information 26
2.7. Evaluation Criteria 28
2.7.1. Quantization Error 28
2.7.2. Topographic Error 28
2.7.3. Visualization Based Approach Based on U-matrix 29
2.8. SOM Summarization 30
2.9. Experimental Results 31
2.9.1. Hepta Dataset 33
2.9.2. Well-Separated Three Gaussians (WS3G) 37
2.9.3. Overlapping Three Gaussians (O3G) 38
2.9.4. Parallel Lines 39
2.9.5. Congressional Voting Record (CVR) 40
2.9.6. Gene Ontology Dataset (GPD194) 40
2.9.7. Activity of Daily Livings (ADL) 42
2.10. Conclusion 46

3 Topology Presevation in Fuzzy Self-Organizing Maps 48
3.1. Introduction 48
3.2. Relational Fuzzy Self-Organizing Maps 50
3.3. Topology Preservation in SOM 51

3.4. Topology Preservation in RFSOM 53
3.5. Experimental Results 60
3.5.1. Fuzzy Topographic Error on O3G 60
3.5.2. Fuzzy Topographic Error and Map Dimensions 63
3.6. Conclusion 65

4 Improvements to the Relational Fuzzy *c*-Means Clustering Algorithm 67
4.1. Introduction 67
4.2. Euclidean Distance Matrices (EDM) and the iRFCM Algorithm 72
4.3. Experimental Results 80
4.3.1. Example 1. Mutation Data 80
4.3.2. Example 2. GDP194 Data 85
4.3.3. Example 3. Iris Data 89
4.4. Conclusion and Discussion 90

5 Predicting Disease Risks from Highly Imbalanced Data using Random Forest 94
1.1. Background 95
5.2. Data 98
5.3. Methods 101
5.3.1. Data Pre-processing 101
5.3.2. Learning from Imbalanced Data 102
5.3.3. Repeated Random Sub-Sampling 103
5.3.4. Random Forest 105
5.3.5. Splitting Criterion 107
5.3.6. Variable Importance 107
5.3.7. Classification with Repeated Random Sub-Sampling 109
5.3.8. Model Evaluation 109
5.4. Results 110
5.4.1. Result set I: Comparison of RF, bagging, boosting and SVM 111
5.4.2. Result set II: Sampling vs. non-sampling 116
5.5. Conclusions and Future Work 119

6 Improving Disease Prediction Using ICD-9 Ontological Features 121
6.1. Introduction 121
6.2. ICD-9 Medical Taxonomy and Similarity Measure 123
6.3. Study Methodology 125
6.3.1. The HCUP2005 dataset 125
6.4. ICD-9 based Ontological Features 126
6.4.1. Example 1 127
6.5. Classifiers used 128
6.6. Experiments 129
6.7. Results 130
6.8. Conclusion 132

7 Quantifying Care Coordination Dose using Natural Language Processing and Domain Specific Ontology 134
7.1. Background 135
7.2. Methods 137
7.2.1. Setting and Sample 137
7.2.2. Dataset Description 137
7.2.3. Care Coordination Ontology 139
7.2.4. Mining Nurses Narratives 143
7.2.5. Problem Profiles 145
7.2.6. Care Coordination Dose 148
7.3. Results 150
7.4. Discussion and Conclusion 155

8 Perspectives and Open Problems 158
8.1. Improved Non-Euclidean Relational Fuzzy c-Means (iNERF) 160
8.2. Cluster Analysis for Big Relational Data 162
8.2.1. iNERFCM for Big Data 164
8.2.2. RFSOM for Big Data 164

APPENDIXES 167
BIBLIOGRAPHY 173
CODE LISTING 183
VITA 184

LIST OF ILLUSTRATIONS

Figure Page

Fig. 2.1. Synthetic datasets 32

Fig. 2.2. Topographic map for the Hepta dataset 34

Fig. 2.3. FSOM (dashed line) and RFSOM (continuous line) behavior on Hepta dataset over one run of 10 iterations 34

Fig. 2.4. RFSOM HL-matrix for a given stimulus at various iterations/times 36

Fig. 2.5. Neuron coefficients for map location in Fig. 2.4d 37

Fig. 2.6. Topographic maps for the WS3G dataset 38

Fig. 2.7. Topographic maps for the O3G dataset 39

Fig. 2.8. Topographic maps for the Parallel Lines dataset. **Error! Bookmark not defined.**

Fig. 2.9. Topographic maps for the Congressional Voting Record dataset 40

Fig. 2.10. Topographic map of the GPD194 dataset generated by RFSOM 42

Fig. 2.11. Distance matrix produced by DTW 44

Fig. 2.12. Topographic map generated by RFSOM using the relational data produced using DTW 45

Fig. 3.1. Topology preservation for some random point o_k 55

Fig. 3.2. Topology preservation in FSOM/RFSOM using image segmentation 57

Fig. 3.3. A topology preserving input pattern k 59

Fig. 3.4. Topology preservation in FSOM/RFSOM process 60

Fig. 3.5. Topology preservation from the O3G 61

Fig. 3.6. The membership values for a random point 62

Fig. 3.7. Topology preservation vs. map size 64

Fig. 3.8. Fuzzy topographic error vs. map size 64

Fig. 4.1. Possible solutions RFCM can utilize when input D is non-Euclidean 71

Fig. 4.2. Example of a minimum spanning tree 76

Fig. 4.3. Mutation phylogeny tree (Fig. 2 in [64]) 81

Fig. 4.4. iVAT image of the *squared* Mutation data D_{mut} 82

Fig. 4.5. The max, mean and standard deviation of the elements in $\widetilde{D}_{mut}$ for the 5 input matrices compared to D_{mut} 85

Fig. 4.6. The induced dissimilarity images $D(U)$ produced from clusterings of the GDP194 dataset using iRFCM with different choices of Δ and $c = 3$ 88

Fig. 4.7. The induced dissimilarity images $D(U)$ produced from clusterings of the Iris dataset using iRFCM with different choices of Δ and $c = 3$ 90

Fig. 5.1. Disease codes and categories hierarchical relationship 99

Fig. 5.2. Distribution of patient across age, race and sex 100

Fig. 5.3. Flow diagram of random forest and sub-sampling approach 105

Fig. 5.4. RF behaviour when the number of trees (ntree) varies 111

Fig. 5.5. ROC curve for diabetes mellitus 113

Fig. 5.6. ROC curve for hypertension 113

Fig. 5.7. ROC curve for breast cancer 114

Fig. 5.8. ROC curve for breast cancer (sampling vs. non-sampling) 117

Fig. 5.9. ROC curve for other circulatory diseases (sampling vs. non-sampling) 118

Fig. 5.10. ROC curve for peripheral atherosclerosis (sampling vs. non-sampling) 118

Fig. 6.1. Partial view of the ICD-9 hierarchy 123

Fig. 6.2. The crisp (red) and fuzzy ontological (blue) features for patient P from example 1..... 128

Fig. 6.3. ROC curves for diabetes prediction obtained with random forest (blue) and SVM (red). 130

Fig. 6.4. ROC curves for ateriosclerosis prediction obtained using random forest (blue) and SVM (red) 131

Fig. 6.5. ROC curves for hypertension prediction obtained using random forest (blue) and SVM (red) 132

Fig. 7.1. Top level concepts of care coordination activities and focus 143

Fig. 7.2. A flow diagram of the activity-focus extraction process..... 150

Fig. 7.3. Number of problem profiles representing each activity 151

Fig. 7.4. Top 20 most frequent activity-focus pairs in Case Management and the percentage of occurrence in every group 152

Fig. 7.5. Number of unique activity-focus pairs by Omaha problem in Case Management 153

Fig. 8.1. The world of data [112] 159

Fig. 8.2. Running time of the MST and SU as n increases..... 161

Fig. 8.3. Population X_∞, and samples X_{VL}, X_L and X_S [113] 164

LIST OF TABLES

Table Page

Table 2.1. Notations 16

Table 2.2. Datasets 32

Table 2.3. Algorithm Parameters 33

Table 2.4. WS3G Dataset Properties 37

Table 2.5. O3G Dataset Properties 38

Table 2.6. Charactertistics of the GDP194 Dataset 41

Table 2.7. RFSOM Results 45

Table 3.1. Behaviour of te_c and te_f when varying σ_0 62

Table 4.1. Transformations of $D \rightarrow \widetilde{D}$ 74

Table 4.2. Hardened 4-partitions found by RFCM(D_{mut}) and iRFCM($\widetilde{D}_{mut}$) 83

Table 4.3. Characteristics of the GDP194 dataset 86

Table 5.1. The 10 most prevalent diseases categories 100

Table 5.2. Some of the most imbalanced diseases categories 100

Table 5.3. Sample Dataset, the shaded column represents the category to predict 102

Table 5.4. RF,SVM, bagging and boosting performance in terms of AUC on eight disease categories 112

Table 5.5. Statistical comparison of RF and boosting ROC curves, the lower the value the more significant the difference is 114

Table 5.6. Top four most importance variable for the eight disease categories 116

Table 5.7. RF, SVM, bagging and boosting performance without sub-sampling in terms of AUC on eight disease categories 117

Table 6.1. The prevalence of three diseases in the HCUP2005 dataset 126

Table 6.2. AROC results for diabetes prediction 130

Table 6.3. AROC results for artheriosclerosis prediction 131

Table 6.4. AROC results for hypertension prediction 131

Table 7.1. Characteristics of the dataset by Omaha category and group (AIP, $n = 217$; HHC, $n = 691$) 139

Table 7.2. A sample patient dataset 139

Table 7.3. Summary of the ontology 142

Table 7.4. Care coordination dose by problem and category in AIP and HHC groups .. 155

LIST OF ABBREVIATIONS

AIP	Aging in Place
ANN	Artificial Neural Network
AROC	Area under the Receiver Operator Characteristic curve
CDC	Center for Disease Control
CMS	Center for Medicare Services
EDM	Euclidean Data Matrix
EF	Exponential Fit
FCM	Fuzzy c-Means
FSOM	Fuzzy Self-Organizing Map
HCUP	Healthcare Cost and Utilization Project
HHC	Home Health Care
ICD-9	International Classification of Disease, Version 9
iNERF	Improved Non-Euclidean Relational Fuzzy c-Means
iRFCM	Improved Relational Fuzzy c-Means
iVAT	Improved Visual Assessment of Tendency
LF	Log Fit
MDS	Minimum Data set
ML	Machine Learning
MST	Minimum Spanning Tree
NERFCM	Non-Euclidean Relational Fuzzy c-Means
NHANES	National Health and Nutrition Examination Survey
NIS	National Inpatient Sample
NLP	Natural Language Processing
PCM	Possibilistic c-Means
PF	Power Fit
PNN	Probabilistic Neural Network

RF	Random Forest
RFCM	Relational Fuzzy *c*-Means
RFSOM	Relational Fuzzy Self-Organizing Map
ROC	Receiver Operating Characteristic
RSOM	Relational Self-Organizing Map
SOM	Self-Organizing Map
SU	Subdominant Ultrametric
SVM	Support Vector Machine
VAT	Visual Assessment of Tendency

ABSTRACT

In real world problems some datasets can only be represented in a relational data matrix because either the underlying objects are unknown or objects cannot be represented as feature vectors. Unlike object-based datasets where we have a well-defined set of features, relational data describes the relations among objects, which can be dissimilarity or a similarity.

We will address three main issued related to relational data clustering and analysis: (i) adapting the Kohonen Self-Organizing Maps (SOM) to relational data and incorporating a fuzzy membership function, which results in a new algorithm called Relational Fuzzy SOM (RFSOM); (ii) proposing a new technique to measure the topology preservation in RFSOM; and (iii) extending the well-known Relational Fuzzy c-Means (RFCM) to handle non-Euclidean relational datasets.

We found that (i) by incorporating fuzzy membership into FSOM/RFSOM a better and less noisy visualization is produced, (ii) for a given stimulus, adjacent neurons will have similar membership, but as the distance between the neuron increases, so does the difference in the membership, based on which we measure the topology preservation and (iii) Euclideanizing a relational matrix D using the subdominant ultrametric transformation leads to best clustering performance, while the β-spread one does the worst. We demonstrate our clustering algorithms on various biomedical datasets, such as the patient activity of daily living and gene ontology datasets.

We also investigate the biomedical problem of predicting future patient diagnoses based on current diseases using the data provided by the Healthcare Cost and Utilization

Project (HCUP). First, we will discuss the problem of patient disease classification using random forest (RF) followed by improvement to the prediction model using ontological features. The ontological features are computed using an ICD9 ontological similarity approach. We found that the classification accuracy using ontological feature surpasses the accuracy using the crisp features.

Finally, we focus on quantifying health care coordination dose using Natural Language Processing (NLP) and nursing Electronic Medical Record (EMR) notes. Care coordination, which includes transitional care services, is seen as a way to improve healthcare, resulting in improved health, and reduced costs. The main innovation of our approach is employing a novel domain specific ontology to guide the NLP process. Using the extracted activities from 139,173 notes we evaluate the amount of care coordination received by every patient in our dataset. We concluded that "Communicate" and "Manage" activities are widely used in care coordination. That confirmed the expert hypothesis that nurse care coordinators spent most of their time communicating about their patients and managing problems. Overall, nurses performed care coordination in both Again in Place (AIP) and Home Health Care (HHC), but the aggregated dose is larger in AIP.

CHAPTER 1

INTRODUCTION

Pattern recognition is a subject that has been studied for many years, yet it never ceases to evolve, grow and adapt to fit new scientific challenges and solve real world problems. In fact, we need pattern recognition more than ever to build intelligent systems able to analyze the huge amount of data that continues to accumulate in computers around the world due, in part, to the pervasiveness of technology in our daily lives. Larger and larger amount of data is being collected almost in every domain. For example, the prevalence of smartphones and ubiquity of the various embedded sensors facilitates the collection of granular information about users, all without human intervention. Wearable sensor-based systems are used in health monitoring and prognosis. Such systems can have various sensors such as pulse oximeter that is used to measure the amount of oxygen carried in the blood and phonocardiograph, which records the heart sound [1]. Other devices such as the iWatch that also be used for health monitoring [2]. Or non-wearable (environmental) sensors used in smart homes to collect a wide array of data for monitoring elderly living alone [3]. Two things are important to point out. First, while the current technology does an excellent job in collecting and storing data, it lacks the ability to understand and convert such data into knowledge that can help us make informed decisions. Second, pattern recognition is no longer applicable to only few domains such as biology and medicine, in fact, it is being utilized in every domain that is data driven, such as ecommerce, defense, advertising, finance, etc.

Clustering is a key component of pattern recognition and it is widely used in numerous applications and fields. Given a set of unlabeled objects, clustering algorithms

attempts to categorize those objects into natural groups, where objects within one group exhibit similar properties. What makes clustering even more important is the fact that most data that exist is in unlabeled form and trying to label a subset of this data is not sufficient and not scalable due to the amount data being generated constantly. Although many clustering algorithms have been proposed in the literature, clustering remains an open field for research. In this work, we will focus on relational clustering and relational topographic maps. In addition, we will investigate applications in biomedical and nursing informatics that utilizes classifications and NLP.

1.1. The Problem

Objects $O = \{o_1, o_2, \dots, o_n\}$ can be described in two ways. They can be represented by numerical features, $X = \{x_1, x_2, \dots, x_n\}$, where x_i is the numerical feature vector representing object o_i and every dimension in x_i is a feature value (object data). Objects can also be described by their relation with each other (relational data). Relational data may be represented as a square $n \times n$ matrix $R = \{[r_{ij}] | 1 \leq i, j \leq n\}$ where r_{ij} represents the relation between o_i and o_j. The relation can be either a similarity or dissimilarity (distance) between the two objects. A dissimilarity relation D satisfies the following conditions:

$$d_{ii} = 0 \qquad \forall\, i = 1, \dots, n, \tag{1.1a}$$

$$d_{ij} \geq 0 \qquad \text{for } i = 1, \dots n \text{ and } j = 1, \dots, n, \tag{1.1b}$$

$$d_{ij} = d_{ji} \quad \text{for } i = 1, \dots n \text{ and } j = 1, \dots, n. \tag{1.1c}$$

One can see the relational data as a more general form for representing objects. In fact, we can convert any object data to relational data by computing the distance between

objects o_i and o_j as $d_{ij} = \|x_i - x_j\|_2^2, \forall\, 1 \leq i, j \leq n$. If we do not have access to the feature vectors we can compute the distance among objects using dynamic time warping [4] to measure distance between two time series, or semantic distance [5] to measure distance between two concepts, etc. Thus, relational clustering algorithms can group objects based on their relations and, from this point of view, they can be seen as more general than object data clustering algorithms.

Numerous relational and non-relational algorithms have been proposed in the literature. Best known non-relational clustering algorithms are k-means [6], k-medoids [7], Fuzzy c-Means (FCM) [8] and possibilistic c-Means (PCM) [9]. Best known relational clustering algorithms are hierarchical clustering [10], Relational Fuzzy c-Means (RFCM) [11] and the Non-Euclidean Relational c-Means (NERCM) [12]. Description of more clustering algorithms can be found in [13]. There is another class of clustering algorithms, namely, algorithms that are aimed at data visualization and exploration such as the Self-Organizing Map (SOM) or Kohonen Network.

All clustering algorithms have one objective, that is, given n objects represented in X (object-based clustering) or R (relational clustering) the goal is to group them into c clusters, where $1 \leq c \leq n$. If $c = 1$ or $c = n$ we assume that the data does not exhibit any clusters and objects are either all grouped together in one cluster or every object is its own cluster, respectively. We can represent the result of any clustering algorithm using a $c \times n$ partition matrix $U = \{[u_{ik}] | \, 1 \leq i \leq c, 1 \leq k \leq n\}$, where each element u_{ik} measures the degree of belongingness of o_k in cluster i. There are three types of partitions: crisp, fuzzy and possibilistic. The crisp clustering approach gives the output of the cluster analyses as matrices from the set of the hard c-partitions, which is defined as

$$M_{hcn} = \left\{ U \in \mathbb{R}^{c \times n} \middle| u_{ik} \in \{0,1\}, \sum_{k=1}^{n} u_{ik} > 0, \sum_{i=1}^{c} u_{ik} = 1, \forall\, 1 \le i \le c \text{ and } 1 \le k \le n \right\} \quad (1.2)$$

The fuzzy c-partition is more general than the crisp c-partition. Unlike crisp partitions where an object is assigned equivocally to one cluster, in fuzzy partitions an object can belong to multiple clusters with varying degree of membership. This concept is important as an object may exhibit characteristics of multiple clusters. A fuzzy partition is defined as

$$M_{fcn} = \left\{ U \in \mathbb{R}^{c \times n} \middle| u_{ik} \in [0,1], \sum_{k=1}^{n} u_{ik} > 0, \sum_{i=1}^{c} u_{ik} = 1, \forall\, 1 \le i \le c \text{ and } 1 \le k \le n \right\} \quad (1.3)$$

A possibilistic partition is a generalization of the fuzzy partitions, where the columns in the partition matrix do not necessarily sum to 1. Possibilistic clustering is also effective in finding coincidental or overlapping clusters. A set of possibilistic c-partitions is defined as

$$M_{pcn} = \left\{ U \in \mathbb{R}^{c \times n} \middle| u_{ik} \in [0,1], \sum_{k=1}^{n} u_{ik} > 0, \forall\, 1 \le i \le c \text{ and } 1 \le k \le n \right\} \quad (1.4)$$

In possibilistic partitions, u_{ik} is referred to as the typicality of object o_k in cluster i and one can see that M_{hcn} is a subset of M_{fcn}which is a subset of M_{pcn} ($M_{hcn} \subset M_{fcn} \subset M_{pcn}$).

Overall, cluster analysis seeks to answers three questions: 1) does X or R have substructure at any value of c? 2) If substructures exist, how can find them? 3) Once clusters are found, how can we validate them? [14] In this work we focus on the second question for the case of relational topographic map and relational clustering.

1.2. Contributions

We start with a discussion about relational topographic maps, visualization of objects represented in a square relational data and relational clustering. Later in the discussion I will address the topic of disease risks classification of patients based on their diagnoses and will end it with an application to NLP in nursing informatics.

1.2.1. Pattern Recognition in Relational Data

Chapter 2 presents a new algorithm, Relational Fuzzy Self-Organizing Map (RFSOM), which is a generalization of the crisp Relational SOM (RSOM) and a variant of RFCM with topological constraint. The notion of Best-Matching Unit (BMU) in RFSOM is replaced by a membership function, that is, every neuron is a BMU of an input object with a certain degree of membership. We tested the algorithm on twelve different datasets that assess different aspects of the performance of the algorithm. The results obtained show that the fuzzy membership smoothes the map and results in better and less noisy visualization.

One of the important properties of SOM is its topology preservation of the input data. The topographic error is one of the techniques proposed to measure how well the continuity of the map is preserved. However, this topographic error is only applicable to the crisp SOM algorithms and cannot be adapted to the fuzzy SOM (FSOM) since FSOM/RFSOM does not assign a unique winning neuron to the input patterns. In chapter 3, we propose a new technique to measure the topology preservation of the FSOM algorithms. The new measure relies on the distribution of the membership values on the map. A low topographic error is achieved when neighboring neurons share similar membership values to a given input pattern.

But what happens if the input relational data matrix is not Euclidean? That is the discussion of chapter 4, which applies to any algorithms that is based on RFCM, such as the RFSOM. When D is not Euclidean, RFCM can fail to execute if it encounters negative relational distances. To overcome this problem we can Euclideanize the relation D prior to clustering. There are different ways to Euclideanize D such as the β-spread transformation, where some constant is added to the off-diagonal elements of D. There are at least four alternatives to the β-spread method. In chapter 4 we compare five methods for Euclideanizing D to $\widetilde{D}$. The quality of $\widetilde{D}$ for our purpose is judged by the ability of RFCM to discover the apparent cluster structure of the objects underlying the data matrix D. We conclude that the subdominant ultrametric transformation gives the best results, producing much better partitions of $\widetilde{D}$ than the other four methods.

1.2.2. **Predicting Disease Risks from Unbalanced Data**

We present a method using random forest (RF) for predicting disease risk of individuals based on their medical history. Medicare data is used, which is publicly available through Healthcare Cost and Utilization Project (HCUP). The data set is highly unbalanced. Therefore, in order to overcome the class imbalance problem, we used an ensemble learning method that consists in repeated random sub-sampling. This technique divides the training data into multiple sub-samples, allowing each sub-sample to be fully balanced. The performance of support vector machine (SVM) is compared to RF in predicting the risk of eight chronic diseases. In combining repeated random sub-sampling with RF, one can overcome the class imbalance problem and achieve good results.

The feature values generated from the HCUP dataset are binary meaning that the patient either has the diagnoses or not. However, from our point of view ICD9 represents

an ontology, i.e. a controlled vocabulary overlaid with a "is-a" term hierarchy. The controlled vocabulary allows for detection of synonymy when two diagnoses are compared. The hierarchy (tree) structure allows for assessing the semantic similarity between diagnoses. Therefore, we can substitute the binary features with fuzzy membership values. The fuzzy membership features were computed using an ICD-9 ontological similarity approach. The prediction results obtained on three diseases (diabetes, atherosclerosis and hypertension) using two classifiers, RF and SVM, show a significant improvement in the area under the Receiver Operating Characteristic (ROC) curve (AROC) compared to the results obtained using the binary features.

1.2.3. Quantifying the Amount of Care Coordination from Nursing Notes Using NLP and Ontologies

We employ NLP aided with a domain specific ontology to guide the extraction of care coordination activities and the focus (object) upon which the specific activity was performed. Using the extracted nursing activities from about 139,000 notes, we evaluate the amount of care coordination received by every patient. We compared two groups of patients: Aging in Place (AIP) who received enhanced care coordination and Home Healthcare (HHC) who received traditional care. We found that patients in AIP received higher care coordination doses than the patients in HHC.

1.3. Outline

The first section encompasses three chapters focused on pattern recognition in relational data. We start by introducing the RFSOM algorithm in Chapter 2, followed by a discussion on how to measure the map continuity and topology preservation in RFSOM in chapter 3. Since RFSOM is based in RFCM, we make a transition into chapter 4 which

addresses the possible failure that can occur in RFCM based algorithms if the dissimilarity matrix D is not Euclidean.

In the second section, which includes chapter 5 and 6, we will discuss the classification of disease risk from the unbalanced HCUP dataset. In chapter 6 we will compare the results obtained using crisp and ontological feature values.

Chapter 7 presents an application of NLP and ontologies in the nursing domain. We quantify the amount of care coordination based on nursing notes by employing NLP aided with a domain specific ontology to guide the extraction of care coordination activities and the focus (object) upon which the specific activity was performed.

Chapter 8 talks about open pattern recognition problems and future work. I will talk about the challenges that RFCM/NERFCM/RFSOM have and needs to be overcome in order to scale those algorithms for large datasets.

1.4. List of Relevant Publications

The research described in this dissertation is based on materials from the following publications:

1. Mohammed A. Khalilia, James Bezdek, Mihail Popescu, James Keller (2014). "Improved Relational Fuzzy *c*-Means". Pattern Recognition (Under Review)
2. Mohammed A. Khalilia; Lori L. Popejoy, PhD, APRN, GNS-BC; Mihail Popescu, PhD; Colleen Galambos, PhD, MSW; Vanessa Lyons; Marilyn Rantz, PhD, RN, FAAN; Lanis Hicks, PhD; Frank Stetzer, PhD (2014). Quantifying Care Coordination Dose using Natural Language Processing and Domain Specific Ontology. Journal of the American Medical Informatics Association (Under Review)
3. M. Khalilia and M. Popescu, "Fuzzy Relational Self-Organizing Maps," International journal of uncertainty fuzziness and knowledge-based system, 2013 (Under Review)

4. Khalilia, M. and Popescu, M. "Topology Preservation in Fuzzy Self-Organizing Maps". Adv. Trends Soft Computing. (2014)
5. M. Khalilia and M. Popescu, "Fuzzy relational self-organizing maps," in 2012 IEEE International Conference on Fuzzy Systems, 2012, pp. 1–6.
6. M. Popescu and M. Khalilia, "Improving Disease Prediction Using ICD-9 Ontological Features," in IEEE International Conference On Fuzzy Systems, 2011, pp. 1805-1809.
7. M. Khalilia, S. Chakraborty, and M. Popescu, "Predicting disease risks from highly imbalanced data using random forest.," BMC medical informatics and decision making, vol. 11, no. 1, p. 51, Jan. 2011 (Highly accessed)

CHAPTER 2

RELATIONAL FUZZY SELF-ORGANIZING MAP

The notion of Best-Matching Unit (BMU) in the proposed Relational Fuzzy Self-Organizing (RFSOM) algorithm is replaced by a membership function where every neuron has a certain degree of matching to an input object. By employing a monotonically increasing fuzzifier and a monotonically decreasing neighborhood kernel, RFSOM initially assigns winning neurons. However, as time progresses adjacent neurons begin communicating and sharing information about the stimulus received. The amount of information being shared at a given time is governed by the fuzzifier and the number of neurons sharing information is controlled by the neighborhood kernel. In this chpater we show that RFSOM is the relational dual of Fuzzy SOM (FSOM). We will compare both FSOM and RFSOM on synthetic and real datasets. Then we will assess the performance of RFSOM on two real relational datasets, Gene Ontology and a patient data consisting of Activity of Daily Living score trajectories.

2.1. Introduction

Data visualization and exploration tools help us understand a domain and inform decision making. For instance, such tools can assist in confirming certain assumptions we make about the data or explore other datasets that exist in high dimensions. Many algorithms have been developed for visualization and dimensionality reduction of high dimensional data. The class of such algorithms includes, but is not limited to, Principle Component Analysis (PCA) [15], Multi-dimensional Scaling (MDS) [16], Isomap [17], Locally Linear Embedding (LLE)[18] and Laplacian Eigenmaps[19]. Some of these

techniques, such as Laplacian Eigenmaps, construct a weighted graph of n nodes, where every node represents a point in a higher dimensional space[19]. Weighted edges connect adjacent nodes. The weights are defined using what is called a heat kernel, which assigns weights based on the distance between two points in the high dimensional space. Similar to the heat kernel, Kohonen's Self-Organizing Maps (SOM) [20] employs a neighborhood kernel that assigns weights for neurons based on their proximity in lower dimensions. Also, contrary to LLE where the number of nodes is equal to n, SOM maps n points to a predefined c neurons.

SOM is an unsupervised learning technique aimed at data exploration, clustering and visualization. It projects an s-dimensional input space into a low dimensional, usually two dimensional, lattice or grid of neurons. SOM is a powerful algorithm and has been used in many applications such as the system used to analyze the Sydney 2000 Olympic results using Viscovery SOMine software [21] or the SOM embedded in an Android device used for fall detection [22]. Another application is omeSOM software and the biological SOM which are used in biological sciences for data visualization [23] and the WEBSOM that is used for information retrieval and document clustering [24]. These are only a handful of numerous possible applications and different implementations of SOM that are tailored to address various problems.

Other variations of SOM are the Fuzzy SOM (FSOM) algorithms. The general idea of FSOM is to integrate fuzzy set theory into neural networks to give SOM the capabilities of handling uncertainty in the data. For instance, a FSOM algorithm for object data was proposed in [25] which is, in some sense, a regularization of the Fuzzy c-Means (FCM) algorithm. The FSOM in [25] is based on a cost function that is derived by

introducing two modifications to the generalized FCM. First, the code vectors are distributed on a regular, low dimensional grid as in SOM, and a penalty term is added to guarantee a smooth distribution of the codebook vector values on the grid to help preserve the topological structure of the data. In [26] fuzzy object SOM based on fuzzy inputs and fuzzy weights for market segmentation of credit cards was proposed. FCM is applied for fuzzy clustering to identify the ambiguous sampled data located near the border between the clusters. In [27], a fuzzy SOM was developed by replacing the neurons of the original SOM with fuzzy rules, which are composed of fuzzy sets. The output of each rule is a singleton. For that reason, the algorithm maps the *s*-dimensional input space to a one dimensional output space. In [28], a hybrid SOM is proposed to predict overlapping clusters of high dimensional data and to detect the uncertainty that comes from the overlapping data. This approach is based on rough set theory to generate soft clustering. In [29], the same authors proposed a variation to [28] in which a two-level stage SA-Rough SOM (Simulated Annealing Rough Self-Organizing Map) was proposed.

Another fuzzy online Kohonen clustering networks was proposed in [30]. The authors address major problems of SOM, such as the termination criteria, convergence and the SOM dependency on the sequence of the input data. To address these problems, FCM model is integrated into the learning rate allowing the neuron weights update function to be inversely proportional to their distance from the kth data point, o_k. As the fuzzifier gets smaller, updating the weights reverts back to Hard c-Means (winner take-all). As it gets larger, the weights are updated with lower individual learning rates. A fuzzy SOM algorithm was proposed in [31] where the FCM membership function was

used to compute the degree of belongingness between a neuron and an object. However, the author abandoned the neighborhood function since it increases the computational complexity of SOM.

SOM was also extended to handle relational data. As previously mentioned, objects can be described by feature vectors or by pair-wise relations. Object data $X = \{x_1, \dots, x_n\} \subset R^s$ consists of s-tuples of numerical feature vectors, x_k, that describe the objects o_k. On the other hand, relational data is presented as an $n \times n$ matrix R. Every element in R, r_{jk}, measures the relationship (similarity or dissimilarity) between objects o_j and o_k. For example, a dissimilarity relation satisfies the following conditions:

$$r_{jj} = 0 \qquad \forall\, j = 1, \dots, n, \tag{2.1a}$$

$$r_{jk} \geq 0 \qquad \text{for } k = 1, \dots n \text{ and } j = 1, \dots, n, \tag{2.1b}$$

$$r_{jk} = r_{kj} \quad \text{for } k = 1, \dots n \text{ and } j = 1, \dots, n. \tag{2.1c}$$

Several extensions of SOM were proposed to handle relational data [32]–[34]. In [32], a Self-Organizing Map for dissimilarity data was proposed. The authors described each neuron by a codebook that represents a subset of input vectors. The codebooks are then updated using a cost function that resembles the c-means objective function and accounts for both the relational data and the neighborhood topology. In [33], the authors extended Neural Gas (NG) and SOM to relational data based on the relational dual of the c-means clustering algorithm derived in [12], [35]. Every neuron is represented as a coefficient vector $\alpha_i \in R^s$ (this algorithm will be discussed in detail in Section 2.3).

An Ontological Self-Organizing Map (OSOM) was proposed to visualize and summarize datasets composed of words [34]. Ontological based similarity measures, such

as the generalized outer product and ordered weighted average, in addition to the relational clustering distance measure, were integrated into SOM. Unlike the RSOM, where the coefficient vectors $\alpha_i \in R^s$, in OSOM, a prototype is a weight vector of fuzzy membership representation of all the terms in the dataset. Hence, the dimensions of the prototype vector can be less than s. For instance, the authors used Gene Ontology dataset of 194 gene products to test their algorithm. The dataset contains 64 terms; therefore, the length of the weight vector is 64 rather than 194 as would be the case in RSOM.

Regardless of the SOM algorithm being used, be it fuzzy, relational or ontological, the goal is to produce a low dimensional map, usually two-dimensional, to visualize the data. One type of maps that is widely used is the Unified Distance Matrix (U-matrix) which visualizes the topology of the data. A U-matrix contains valleys representing the clusters separated by mountain ranges that act as boundaries between the valleys. A good SOM produces a nontrivial U-matrix [36]. U-matrix is "nontrivial" when its watershed order or the number of distinct catchment basins is greater than one, but much less than n. This chpater demonstrates the ability of RFSOM to produce a "nontrivial" U-matrix that can preserve the data topology. The reader is referred to [36] for more details about the concept of "nontrivial" U-matrix and its significance.

In this chpater, we present the theoretical framework of RFSOM, and extend and elaborate on the RFSOM algorithm concept we proposed in [37]. Here, we present a new Fuzzy Self-Organizing Maps (FSOM), which employs a monotonically increasing fuzzifier. Based on FSOM, we then derive its fuzzy relational dual. In both FSOM and RFSOM the notion of the Best Matching Unit (BMU) no longer exists. Instead, by using a monotonically increasing fuzzifier and a monotonically decreasing neighborhood size

we give neurons the ability to share and communicate information about the stimulus. To our knowledge, this is the first attempt to apply fuzzification to the relational SOM. We demonstrate the benefits of RFSOM with extensive evaluations and comparisons to the FSOM using multiple synthetic and real datasets.

The rest of the chapter is organized as follows: Section 2.2 provides a brief overview of the Online and Batch SOM (BSOM) algorithm. Section 2.3 briefly introduces the RSOM algorithm. Section 2.4 presents the first contribution, which is the Fuzzy SOM (FSOM). Section 2.5 presents the second contribution, where we derive the relational dual of FSOM algorithm and sets a few theorems that link RFSOM to RSOM. Section 2.6 justifies the use of a monotonically increasing fuzzifier and the ability of neurons to share information. Section 2.7 addresses the criteria used to evaluate the proposed algorithm. Section 2.8 briefly explains the technique used for SOM summarization. Section 2.9 presents results obtained on synthetic and real dataset which demonstrates the effectiveness of RFSOM. Finally, Section 2.10 concludes with analysis, remarks and future work.

2.2. Self-Organizing Map

SOM is an unsupervised learning technique that has been widely used in data visualization, exploration and clustering. SOM performs dimensionality reduction from a high-dimensional data space, $\mathbb{R}^s$, to a lower dimensional lattice or a map, usually two-dimensional. This feature allows us to visualize the cluster tendency of high-dimensional data. SOM forms a network structure that can be two-dimensional square, hexagonal grid or toroidal. Every node or neuron in the structure is connected to its neighbor using a neighborhood kernel, which gives SOM the topology preserving characteristic.

The online SOM algorithm starts by drawing a random data point, x_k, from the input data which causes the weight vectors, m_i, to move closer towards x_k according to a neighborhood function, h, and a learning rate, η (notations are summarized in Table 2.1). The learning rate $\eta(t)$ determines the amount of influence x_k has on every neuron at iteration t, while $h(t)$ determines the amount of influence based on the proximity of the neuron to the BMU (proximity is defined as the distance between neuron's i coordinate a_i and neuron's j coordinate a_j in 2D). BMU, denoted by w_k, is the closest neuron to the input x_k and therefore is influenced the most. In other words, SOM assigns a full membership for o_k to the winning neuron i ($u_{ik} = 1$). However, if the entire dataset is available, one can use batch SOM (BSOM). BSOM can be significantly faster and does not require the specification of the learning parameter η [38]. Algorithm 2.1 outlines the BSOM procedure.

Table 2.1. Notations

Symbol	Description
c	Number of neurons in the lattice
i	Refers to the ith neuron, $1 \leq i \leq c$
a_i	Neuron's i position in 2D space
x_k	Feature vector representing o_k, where $x_k \subset \mathbb{R}^s$
m_i	The weight vector of the ith neuron in the non-relational SOM
h_{ip}	Neighborhood function between neuron i and p
η	Learning rate
w_k	Refers to the position or index of the winning neuron of o_k
u_{ik}	The membership grade of o_k in neuron i
n	Number of objects in the dataset
o_k	The kth object in the dataset, $1 \leq k \leq n$
α_i	The coefficient vector of the ith neuron in the relational SOM
r_{jk}	The distance between o_j and o_k
d_{ik}	The distance between m_i and o_k
σ_0	Initial neighborhood size or radius
σ_f	Final neighborhood size or radius
q_0	Initial fuzzifier
q_f	Final fuzzifier
$q = q(t)$	Fuzzifier value at time t
t_{max}	Number of training epochs

N_i	Set of immediate neighboring neurons to i

Algorithm 2.1. Batch Self-Organizing Map (BSOM)

1 **Input:** Data $O = \{o_1, \dots, o_n\}$, where o_k is represented by a feature vector x_k, *map size*, *c neurons*, initial radius σ_0, final radius σ_f

2 **Output:** U, V_R

3 **Initialize:** random weight vectors $m \subset \mathbb{R}^s$

4 **while** $t \leq t_{max}$

5 $$d_{ik} = \|m_i - x_k\|^2;\ \forall\ 1 \leq i \leq c \text{ and } 1 \leq k \leq n \tag{2.2}$$

6 $$w_k = \arg\min_i d_{ik};\ \forall\ 1 \leq k \leq n. \tag{2.3}$$

7 $$h_{i,w_k}(t) = \exp\left(\frac{-\|a_i - a_{w_k}\|^2}{2\sigma^2(t)}\right);\ \forall\ 1 \leq i \leq c \text{ and } 1 \leq k \leq n. \tag{2.4}$$

8 $$m_i(t+1) = \sum_{k=1}^{n} h_{i,w_k} x_k \Big/ \sum_{k=1}^{n} h_{i,w_k};\ \forall\ 1 \leq i \leq c. \tag{2.5}$$

9 $$\sigma(t+1) = \sigma_0 \cdot \left(\sigma_f/\sigma_0\right)^{t/t_{max}}. \tag{2.6}$$

10 $t \leftarrow t + 1$

11 **end while**

2.3. Relational Self-Organizing Map

SOM is a very effective technique when the objects in the dataset are represented by feature vectors. However, when objects are described in relational form, one needs to use the Relational Self-Organizing Map (RSOM) [33]. In RSOM, it is not necessary to know the vectorial representation of the input data to compute the cluster prototypes or weight vectors. Instead, a weight vector, m_i, is expressed as a linear combination of the input data points $m_i = \sum_{k=1}^{n} \alpha_{ik} x_k$ where $\sum_{k=1}^{n} \alpha_{ik} = 1$. The dissimilarity between a weight vector m_i and object o_k is computed based on the coefficients α and the

dissimilarity matrix R (2.8). The goal in RSOM is to minimize the following objective function [33]

$$L(U;R) = \sum_{i=1}^{c} \frac{1}{2\sum_{k=1}^{n} h_{i,w_k}} \sum_{k=1}^{n} \sum_{k'=1}^{n} h_{i,w_k} h_{i,w_{k'}} r_{kk'}. \tag{2.7}$$

RSOM algorithm is outlined below:

Algorithm 2.2. Relational Self-Organizing Map (RSOM)

Input: $n \times n$ relational data matrix R, *map size*, c, σ_0, σ_f
Output: U, V_R
Initialize: random weight vectors $m \subset \mathbb{R}^s$

while $t \leq t_{max}$

$$d_{ik} = \|m_i - x_k\|^2 = (R \cdot \alpha_i)_k - \frac{(\alpha_i^t \cdot R \cdot \alpha_i)}{2}; \quad \forall\, 1 \leq i \leq c \text{ and } 1 \leq k \leq n \tag{2.8}$$

Assign the winning neuron using (2.3)
Compute the neighborhood function using (2.4)
Update the coefficients vector values

$$\alpha_{ik}(t+1) = \frac{h_{i,w_k}}{\sum_{k=1}^{n} h_{i,w_k}}; \ \forall\, 1 \leq i \leq c \text{ and } 1 \leq k \leq n. \tag{2.9}$$

Update the neighborhood radius using (2.6)
$t \leftarrow t + 1$
end while

2.4. Fuzzy Self-Organizing Map

Numerous fuzzy SOM algorithms were proposed. This section presents a new Fuzzy SOM (FSOM) algorithm which will serve as a foundation for the next section, the Fuzzy Relational SOM.

Contrary to the "winner takes-all" paradigm, as in BSOM and RSOM, FSOM gives neurons the ability to share a stimulus with their neighboring neurons. The amount of sharing is controlled by the fuzzifier and the number of neurons involved in sharing is governed by the neighborhood kernel. More discussion about the fuzzifier and information sharing will be presented in Section 2.6. The goal of FSOM is to find a fuzzy partition $U \in M_{fcn}$, where

$$M_{fcn} = \left\{ U \in \mathbb{R}^{c\times n} \middle| \begin{array}{c} u_{ik} \in [0,1], \sum_{k=1}^{n} u_{ik} > 0, \\ \sum_{i=1}^{c} u_{ik} = 1, \forall\, 1 \le i \le c \text{ and } 1 \le k \le n \end{array} \right\} \tag{2.10}$$

and codebook vectors $m = \{m_1, \dots, m_c\}$, $m_i \in \mathbb{R}^s$, that minimizes the objective function

$$J_q(U, m; X) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^q \sum_{j=1}^{c} h_{ij}^q \, d_{jk}. \tag{2.11}$$

The objective function (2.11) is similar to the probabilistic SOM proposed in [39]. And in contrast to fuzzy clustering methods where the fuzzifier, q, remains constant with time, in RFSOM, q changes at every iteration. For reasons we will discuss in Section 2.6, q uses a monotonically increasing function (2.15) where q varies within a range $[q_0\ q_f]$, for example, $q_0 = 1$ and $q_f = 2$. The fuzzifier q in (2.11) and other equations we will encounter refers to the fuzzifier value at time t, $q(t)$. Therefore, the notations q and $q(t)$ are equivalent and we will use q for convenience.

Theorem 1. *Let $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^s$ be the set of feature vectors, $m = \{m_1, \dots, m_c\}$ be the c neurons codebook vectors, $\|x_k - m_j\|^2$ is the distance between feature vector x_k and codebook vector m_j ($k = 1, \dots, n; j = 1, \dots, c$), and $U \in M_{fcn}$. Then the set (U, m) might be a minimizer of J_q only if $u_{ik} = 1/\sum_{j=1}^{c}\left(\frac{\sum_{a=1}^{c} h_{ia}^q d_{ak}^2}{\sum_{b=1}^{c} h_{jb}^q d_{bk}^2}\right)^{1/q-1}$ and $m_j = \frac{\sum_{i=1}^{c}\sum_{k=1}^{n} h_{ij}^q u_{ik}^q x_k}{\sum_{i=1}^{c}\sum_{k=1}^{n} h_{ij}^q u_{ik}^q}$.*

Proof. To derive the necessary conditions and membership update equations, we set Lagrange optimization problem to minimize (2.11) under the constraint $\sum_{i=1}^{c} u_{ik} = 1, \forall k$ as follows:

$$L(m, X, U, \Lambda) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{q} \sum_{j=1}^{c} h_{ij} d_{jk}^{2} + \sum_{k=1}^{n} \lambda_k \left(1 - \sum_{i=1}^{c} u_{ik}\right)$$

Take the partial derivative with respect to u_{ik} and set it equal to 0.

$$\frac{\partial}{\partial u_{ik}} L(m, X, U, \Lambda) = q\, u_{ik}^{q-1} \sum_{j=1}^{c} h_{ij} d_{jk}^{2} + \lambda_k = 0$$

Solving for λ_k

$$\lambda_k = \left(\sum_{i=1}^{c} \left(q \sum_{j=1}^{c} h_{ij} d_{jk}^{2} \right)^{1-q} \right)^{1-q}$$

Solving for u_{ik}

$$u_{ik} = \left(\frac{\lambda_k}{q \sum_{j=1}^{c} h_{ij} d_{jk}^{2}} \right)^{\frac{1}{q-1}}$$

By substitute λ_k in u_{ik} we find that U might be a minimum of J_q only if

$$u_{ik} = 1 \Big/ \left[\sum_{j=1}^{c} \left(\frac{\sum_{a=1}^{c} h_{ia}^{q} d_{ak}^{2}}{\sum_{b=1}^{c} h_{jb}^{q} d_{bk}^{2}} \right)^{\frac{1}{q-1}} \right]. \tag{2.12}$$

To derive the necessary condition for the weight update equation we rewrite (2.11) as

$$J_{FSOM}(m, X, U) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{q} \sum_{j=1}^{c} h_{ij} \left(x_k - m_j\right)^{t} \left(x_k - m_j\right)$$

Differentiating (2.11) with respect to m_j and setting it to 0

$$\frac{\partial}{\partial m_j} J_{FSOM}(m, X, U) = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{q} \sum_{j=1}^{c} h_{ij} \left(-2\left(x_k - m_j\right)\right) = 0$$

Solving for m_j leads to codebook update equation

$$m_j = \sum_{i=1}^{c}\sum_{k=1}^{n} h_{ij}^{q} u_{ik}^{q} x_k \Big/ \sum_{i=1}^{c}\sum_{k=1}^{n} h_{ij}^{q} u_{ik}^{q}. \tag{2.13}$$

As we can see, o_k is assigned a partial or fuzzy membership in multiple neurons. Consequently, the notion of "best-matching unit" is no longer applicable to FSOM and it is replaced by the membership degree. In fact, one can think of every neuron i as the winning neuron (BMU) of object o_k with degree u_{ik}. For this reason, none of the neurons in the proposed algorithm is empty, meaning that for every neuron $\sum_{k=1}^{n} u_{ik} > 0$.

By having $U \in M_{fcn}$, FSOM leads to a smoother topology. The smoother topology is mainly due to eliminating empty neurons and the fact that adjacent neurons share similar memberships to object o_k. The importance and practicality of a smoother topology becomes more apparent when they are incorporated into the U-matrix. For instance, a less distorted and smoother U-matrix may help in increasing the accuracy of the U-matrix segmentation which is one of the methods used for the clustering of SOM [40], [41]. The FSOM is summarized in Algorithm 2.3.

Algorithm 2.3. Fuzzy Self-Organizing Map (FSOM)

Input: Data $O = \{o_1, \dots, o_n\}$, where o_k is represented by a feature vector x_k, *map size*, c, initial fuzzifier q_0, final fuzzifier q_f, σ_0, σ_f

Output: U, V_R

Initialize: random weight vectors $m \subset \mathbb{R}^s$

while $t \leq t_{max}$

  Calculate the distance between neuron i and o_k using (2.2)

  Compute the membership values, u_{ik} using (2.12)

$$h_{ij} = \exp\left(\frac{-\|a_i - a_j\|^2}{2\sigma^2(t)}\right). \tag{2.14}$$

  Update weight vectors using (2.13)

  Update the neighborhood radius using (2.6)

$$q(t+1) = q_0 \cdot \left(q_f/q_0\right)^{t/t_{max}}. \tag{2.15}$$

  $t \leftarrow t + 1$

end while

2.5. Relational Fuzzy Self-Organizing Map

In this section, we will derive RFSOM theoretical framework, which is the major contribution of this chapter. We will derive the membership and coefficient update equations and the fuzzy relational dual of the FSOM. As mentioned earlier, in relational clustering it is not necessary to know the vectorial representation of the input data to compute the cluster prototypes or weight vectors. Instead, the codebook vector m_i is expressed as linear combination of the input data points $m_i = \sum_{k=1}^{n} \alpha_{ik} x_k, \sum_{k=1}^{n} \alpha_{ik} = 1$, where α_{ik} is computed as

$$\alpha_{ik} = \delta_{ik} \Big/ \sum_{k\prime=1}^{n} \delta_{ik\prime} \tag{2.16}$$

and δ_{ik} is defined as

$$\delta_{ik} = \sum_{j=1}^{c} h_{ij}^{q} \cdot u_{jk}^{q}. \tag{2.17}$$

Using equations (2.11)-(2.13), (2.16) and (2.17), we can re-formulate the fuzzy optimization scheme in terms of the relational data, R, which we will do next.

Definition. Let $U \in M_{fcn}$ be a fuzzy partition of R, the RFSOM functional K_q is defined as

$$min\left\{K_q(U;R) = \sum_{i=1}^{c} \frac{1}{2\sum_{k=1}^{n}\delta_{ik}} \sum_{k=1}^{n}\sum_{k'=1}^{n} \delta_{ik} \cdot \delta_{ik'} \cdot r_{kk'}\right\}. \quad (2.18)$$

The objective function (2.18) attempts to minimize the sum of pairwise distances among objects belonging to the same neuron. To better understand the RFSOM objective function, we will decompose it into its basic elements:

$h_{ij}^q \cdot u_{jk}^q =$ grade of membership of o_k contributed to neuron i by j, weighted by the neighborhood function;

$\delta_{ik} = \sum_{j=1}^{c} h_{ij}^q \cdot u_{jk}^q =$ sum of memberships of o_k contributed to i from the surrounding neurons;

$\sum_{k=1}^{n}\sum_{k'=1}^{n} \delta_{ik} \cdot \delta_{ik'} \cdot r_{kk'} =$ within neuron i sum of pairwise distances among objects weighted by the contributed memberships of o_k and $o_{k'}$ from neighboring neurons.

Notice that the neurons in FSOM and RFSOM become less competitive than the "winner takes-all" SOM. And while o_k has a membership in neuron i, o_k enjoys an additional membership in i, contributed by its neighbors.

Theorem 2. *Let $q > 1$, $\|\cdot\|$ be a norm induced by inner product on $\mathbb{R}^s$, $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^s$ be a set of feature vectors, and $R = [r_{jk}] = \left[\|x_j - x_k\|^2\right]$ be the set of corresponding relational distance data. Then $U \in M_{fcn}$ is a minimizer of K_q if and only if (U, m) is a minimizer of J_q in $U \in M_{fcn}$.*

Proof. By substituting (2.16) in (2.8) which in turn is substituted in (2.11) we can derive the RFSOM objective function K_q as follows:

$$K_q(U,m;R) = \sum_{i=1}^{c}\sum_{k=1}^{n}\sum_{j=1}^{c} u_{ik}^q h_{ij}^q (R\cdot\alpha_i)_k - \frac{(\alpha_i^t\cdot R\cdot\alpha_i)}{2}$$

$$= \sum_{i=1}^{c}\sum_{k=1}^{n}\sum_{j=1}^{c} u_{ik}^q h_{ij}^q \left(R\cdot\frac{\sum_{j'=1}^{c} h_{ij'}^q u_{j'}^q}{\sum_{j'=1}^{c}\sum_{k'=1}^{n} u_{ij'}^q h_{j'k'}^q}\right)_k$$

$$-\frac{1}{2}\sum_{i=1}^{c}\sum_{k=1}^{n}\sum_{j=1}^{c}\left(\frac{\sum_{j'=1}^{c} h_{ij'}u_{j'}^q}{\sum_{j'=1}^{c}\sum_{k'=1}^{n} u_{j'k'}^q h_{ij'}}\cdot R\cdot\frac{\sum_{j'=1}^{c} h_{ij'}u_{j'}^q}{\sum_{j'=1}^{c}\sum_{k'=1}^{n} u_{j'k'}^q h_{ij'}}\right)$$

$$= \sum_{i=1}^{c}\frac{(\sum_{j=1}^{c} u_j^q h_{ij}^q)R\left(\sum_{j'=1}^{c} u_{j'}^q h_{ij'}^q\right)}{\sum_{j'=1}^{c}\sum_{k'=1}^{n} u_{ij'}^q h_{j'k'}^q} - \frac{1}{2}\sum_{i=1}^{c}\frac{(\sum_{j=1}^{c} u_j^q h_{ij}^q)R\left(\sum_{j'=1}^{c} u_{j'}^q h_{ij'}^q\right)}{\sum_{j'=1}^{c}\sum_{k'=1}^{n} u_{ij'}^q h_{j'k'}^q}$$

$$= \sum_{i=1}^{c}\frac{(\sum_{j=1}^{c} u_j^q h_{ij}^q)R\left(\sum_{j'=1}^{c} u_{j'}^q h_{ij'}^q\right)}{2\sum_{j'=1}^{c}\sum_{k'=1}^{n} u_{ij'}^q h_{j'k'}^q}$$

$$= \sum_{i=1}^{c}\frac{\sum_{k=1}^{n}\sum_{k\prime=1}^{n}(\sum_{j=1}^{c} u_j^q h_{ij}^q)(\sum_{j\prime=1}^{c} u_{j\prime k\prime}^q h_{ij\prime}^q) r_{kk\prime}}{2\sum_{j\prime=1}^{c}\sum_{k\prime=1}^{n} u_{ij\prime}^q h_{j\prime k\prime}^q}$$

By setting $\delta_{ik} = \sum_{j=1}^{c} u_{jk}^q h_{ij}^q$ we can re-write K_q as:

$$min\left\{K_q(U;R) = \sum_{i=1}^{c}\frac{1}{2\sum_{k=1}^{n}\delta_{ik}}\sum_{k=1}^{n}\sum_{k'=1}^{n}\delta_{ik}\cdot\delta_{ik\prime}\cdot r_{kk\prime}\right\}.$$

From the above proof, we can see that $K_q(U;R) = J_q(U,m;X)$. By Theorem 1, we proved that $m = \{m_1,\dots,m_c\} \subset \mathbb{R}^s$ uniquely minimizes $J_q(U,m;X)$ for a fixed $U \in M_{fcn}$. It follows that U is a minimizer of $K_q(U;R)$ if and only if U is a minimizer of $J_q(U,m;X)$.

Theorem 2 establishes the close connection between the functionals K_q and J_q and in theory, we should be able to find the same partitions of objects using K_q in the relational scheme that is found using the object based FSOM. Also, we can show the close connection between the functionals K_q and L which states that K_q is a generalization of L (2.7).

Corollary 1: If $\{q_0 = q_f\} \overset{+}{\rightarrow} 1$, the rate at which neurons share information decreases and the sharing stops when $q_0 = q_f = 1$. At that point, the RFSOM reduces to the "winner takes-all" scheme as in the RSOM algorithm.

Proof: When $q_0 = q_f$, the fuzzifier q remains constant throughout the iterations, which means $q = q_0 = q_f$. Also, the limit property of (2.12) states the following:

$$\lim_{q\to 1} u_{ik} = \begin{Bmatrix} 1, & \sum_{a=1}^{c} h_{ia}^{q} d_{ak}^{2} < \sum_{b=1}^{c} h_{jb}^{q} d_{bk}^{2} \ \forall i \neq j \\ 0, & otherwise \end{Bmatrix};$$

$$\forall 1 \leq i \leq c; 1 \leq k \leq n.$$

Therefore, at $q_0 = q_f = 1$, δ_{ik} (2.17) converges to the neighborhood function,

$$\lim_{q\to 1} \delta_{ik} = \lim_{q\to 1} \sum_{j=1}^{c} h_{ij}^{q} \cdot u_{jk}^{q} = h_{i,w_k}.$$

Combining those results, we conclude that when $q_0 = q_f = 1$, $K_1(U;R) = L(U;R)$.

Corollary 1 states that RSOM is a special case of the RFSOM and it can be reduced to the "winner-take-all" scheme by using a constant fuzzifier and setting $q = 1$.

Theorem 3. *When σ is large, δ_{ik} is influenced by the memberships of o_k in the neighboring neurons to i. However, as neighborhood size shrinks with time and approaches its final value and as $\sigma_f \to 0$, δ_{ik} converges to u_{ik}^{q}.*

Proof: The limit property of (2.14) states the following:

$$\lim_{\sigma\to 0} h_{ij} = \begin{Bmatrix} 1, & \forall i = j \\ 0, & otherwise \end{Bmatrix} \forall 1 \leq i, j \leq c.$$

Therefore, we conclude that δ_{ik} converges to u_{ik}^{q} as $\sigma \to 0$

$$\lim_{\sigma\to 0} \delta_{ik} = \lim_{\sigma\to 0} \sum_{j=1}^{c} h_{ij}^{q} \cdot u_{jk}^{q} = u_{ik}^{q}.$$

Note that these theorems also apply to the FSOM discussed earlier in Section 2.4. The complete RFSOM algorithm is outlined below:

Algorithm 2.4. Relational Fuzzy Self-Organizing Map (RFSOM)

Input: $n \times n$ relational data matrix R, map size, c, σ_0, σ_f

Output: U, V_R

Initialize: random weight vectors $m \subset \mathbb{R}^s$

while $t \leq t_{max}$

$$d_{ik} = \|m_i - x_k\|^2 = (R \cdot \alpha_i)_k - \frac{(\alpha_i^t \cdot R \cdot \alpha_i)}{2}; \quad \forall\, 1 \leq i \leq c \text{ and } 1 \leq k \leq n$$

$$u_{ik} = 1 \Big/ \left[\sum_{j=1}^{c} \left(\frac{\sum_{a=1}^{c} h_{ia}^q d_{ak}^2}{\sum_{b=1}^{c} h_{jb}^q d_{bk}^2} \right)^{\frac{1}{q-1}} \right]; \ \forall\, i, k.$$

for $i = 1$ *to* c **do**

$$h_{ij} = \exp\left(\frac{-\|a_i - a_j\|^2}{2\sigma^2(t)} \right); \forall\, j.$$

$$\alpha_{ik} = \delta_{ik} \Big/ \sum_{k\prime=1}^{n} \delta_{ik\prime}, \delta_{ik} = \sum_{j=1}^{c} h_{ij}^q \cdot u_{jk}^q \,; \forall\, k.$$

end for

$$\sigma(t+1) = \sigma_0 \cdot \left(\sigma_f / \sigma_0\right)^{t/t_{max}}.$$

$$q(t+1) = q_0 \cdot \left(q_f / q_0\right)^{t/t_{max}}.$$

$t \leftarrow t + 1$

end while

2.6. Neurons Sharing Information

The remaining question is why does RFSOM employ a monotonically increasing fuzzifier? Why not simply use a constant fuzzifier similar to some clustering algorithms such as Fuzzy *c*-Means? (Note that this Section references RFSOM only, but the same argument applies to the FSOM).

There are two reasons for employing this kind of fuzzifier. First, RFSOM has more degrees of freedom, map size, number of neurons initial and final radius. Introducing a new parameter, q, can cause undesirable interactions with some of the existing parameters. Indeed, both q and σ are related and one has to exercise extra care

when setting them. Therefore, it is important to understand the behavior of the neighborhood function in relation to the fuzzifier. Recall that SOM starts with a larger neighborhood radius σ_0 which decreases with time until it reaches σ_f and as $\sigma_0 \to \infty$ the neighborhood function $h_{ij} \to 1$ (2.14) causing the membership function (2.12) $u_{ik} \to 1/c$ (assuming a constant fuzzifier, i.e. $q = 2$). The experiments have shown that $\sigma_0 = 2$ is large enough to cause this problem. Thus, to alleviate this issue and prevent $u_{ik} \to 1/c$, RFSOM has to balance q and σ by employing a monotonically increasing fuzzifier and a monotonically decreasing radius. RFSOM will start as "winner take-all" with large radius and small fuzzifier ($q = 1$) and the membership values become fuzzifier as q increases with time.

Second, a monotonically increasing fuzzifier allows the neurons to share information about the sensed stimulus. As stimulus o_k is sensed at $t = 1$, a winning neuron i is assigned to that stimulus. However, neuron i loses the unity membership at $t > 1$ as it starts sharing and communicating information with its neighbors about o_k, that is, when crisp memberships start becoming softer. The harmony between the monotonically increasing fuzzifier and the monotonically decreasing neighborhood kernel governs and limits the sharing of information. The fuzzifier restricts the amount of information being shared with other neurons; as the fuzzifier increases, the membership values become more distributed and the amount of information communicated increases. On the other hand, the neighborhood kernel limits the intensity and the number of neurons sharing information. For instance, a Gaussian neighborhood kernel limits the sharing of information among distant neurons, so by the time RFSOM converges, only a small number of adjacent neurons should have a high firing strength for some stimulus.

Hence, as time progresses more information is communicated among a smaller subset of neurons. In the experimental results, we will show an example demonstrating information sharing.

2.7. Evaluation Criteria

In this section we present the evaluation criteria used to assess RFSOM performance. For the evaluation we use: quantization error, topographic error and visualization based approach (U-matrix).

2.7.1. Quantization Error

Quantization error (*qe*), a widely used SOM evaluation measure [42], is defined as the average distance between the objects in the dataset and their corresponding winning neurons[43]. A good map is expected to have a small *qe*.

$$qe = \frac{1}{n}\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{q}\, d_{ik}. \tag{2.19}$$

Equation (2.19) weights the distance d_{ik} with the fuzzy membership u_{ik}. When $q_0 = q_f = 1$, (2.19) reduces to the original *qe* formulation[43].

2.7.2. Topographic Error

The quantization error is a good overall measure, but it does not reflect the topological preservation of the map. Different approaches for measuring the topology preservation were proposed [43]. In this chpater we use the topographic error (*te*) to measure the topology preservation. *te* measures the proportion of the input vector for which the first and second BMU are not adjacent neurons.

$$te = \frac{1}{n}\sum_{k=1}^{n} adj(o_k). \tag{2.20}$$

$$adj(o_k) = \begin{cases} 0, if\ the\ two\ closest\ neurons\ to\ o_k\ are\ adjacent \\ 1, if\ the\ two\ closest\ neurons\ to\ o_k\ are\ not\ adjacent \end{cases}.$$

To evaluate RFSOM, however, we do not use the two closest neurons, but rather the two neurons with the highest membership of o_k. Similar to *qe*, a good map is expected to yield a small *te*.

2.7.3. **Visualization Based Approach Based on U-matrix**

Although the U-matrix is not a quantitative evaluation technique, nonetheless, it provides a quick overall qualitative assessment of how the algorithm performed. A U-matrix is calculated in the weight vector space ($\mathbb{R}^s$) or coefficient space in relational algorithms and displayed in the lattice space, which is usually two-dimensional. A U-height of a neuron i, $uh(m_i)$, is defined as the sum of distances from m_i to the neighboring neurons of i, N_i [44]. For instance, in a rectangular grid, N_i refers to the four immediate neighbors and the U-Height for neuron i is computed as:

$$uh(m_i) = \sum_{m_j \in N_i} \|m_i - m_j\|^2. \tag{2.21}$$

However, in relational SOM, such as RFSOM and RSOM, $\|m_i - m_j\|^2$ is calculated in terms of α and R as (see proof in Appendix I)

$$\|m_i - m_j\|^2 = \alpha_j^t R\alpha_i - \frac{1}{2}\alpha_j^t R\alpha_j - \frac{1}{2}\alpha_i^t R\alpha_i. \tag{2.22}$$

A U-matrix is generated when the U-height of every neuron is calculated at that neuron's coordinates. The matrix can be displayed in 2D as a planar or in 3D to visualize

the topology. When the U-matrix is visualized in 3D, the "mountain ranges" point to cluster boundaries and "valleys" refer to cluster centers.

2.8. SOM Summarization

Several techniques exist to summarize the results of SOM algorithms. One can either cluster the neurons and then identify which objects belong to which cluster[45], or use image processing techniques to segment the U-matrix to find its distinct regions [46], [47]. In this chapter, we chose the latter technique and used the MATLAB implementation of the Watershed algorithm [48]. Once the Watershed algorithm is applied on the U-matrix, it returns multiple regions. Every region represents a catchment basin encompassing a group of similar neurons. While it is not always the case that we will identify the exact number of regions since Watershed algorithm can over-segment the U-matrix[49], for the purposes of SOM summarization, over-segmentation is not an issue.

Once the regions are identified, we can uncover the similar neurons grouped in every region. For every neuron, we find the most representative object (the object with the highest weight or membership value to that neuron). Once every neuron has a representative object, a majority voting among neurons in that region is performed to vote for an overall representative label for that region. The label is then placed in the centroid of the region in question. Therefore, to label the region it is assumed that every object has an assigned label. For instance, an object in Hepta dataset is assigned a label $l \in \{1, 2, 3, 4, 5, 6, 7\}$, while patients in the ADL dataset are labeled with their own and unique ADL trajectory. As we will see later in the results section, every region is

assigned an identifier as $R\#:l$, where # corresponds to the region index or number and l is the region label.

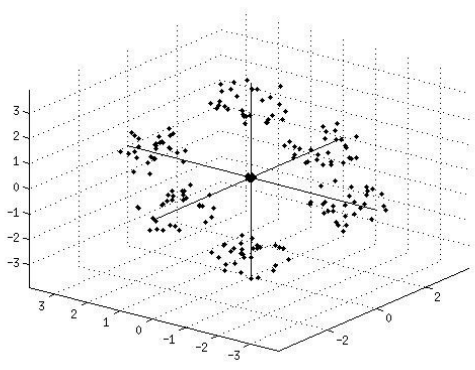
2.9. Experimental Results

To evaluate the proposed algorithm, RFSOM is compared to the FSOM using synthetic and real datasets (Fig. 2.1), which are summarized in Table 2.2. For object-based datasets, the dissimilarities among objects, r_{ij}, are calculated using the Euclidean distance. The well-separated three Gaussians (WS3G) and the overlapping three Gaussians (O3G) datasets are two-dimensional and they differ by the inter-cluster variance. Lines dataset contains three parallel lines. The congressional voting record dataset is published by the University of California-Irvine (UCI) Machine Learning Repository [50]. Hepta dataset is part of the Fundamental Clustering Problem Suite (FCPS) [41], and it is used to demonstrate the behavior of the FSOM and RFSOM.

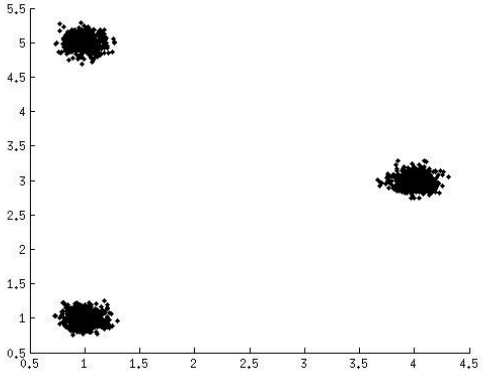
In addition to the synthetic datasets that allow us to compare RFSOM to FSOM, we test RFSOM using two real datasets, the Gene Ontology (GPD194) and the Activity of Daily Living (ADL). GPD194 is a pure relational dataset containing pairwise distances among Gene terms measured using fuzzy distance measure [34], and ADL is a relational dataset containing pairwise distances among 3,963 patients measured using Dynamic Time Warping (DTW) [4]. Notice that RFSOM expects an Euclidean relational matrix R and non-Euclidean matrix needs to be converted into an Euclidean one using techniques such as the β-Spread Tranformation [12]. However, this tranformation is necessary and important to perform if equation (2.8) results in negative values [12]. This situation was not encounter for GPD194 and ADL datasets and therefore no tranformation was necessary.

Table 2.2. Datasets

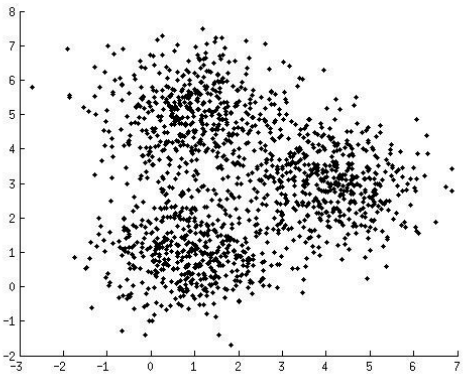
	Name	Size	Dimensions	No. of Classes
A	Hepta	212	3	7
B	Well Separated Gaussians (WS3G)	1,500	2	3
C	Overlapping Gaussians (O3G)	1,500	2	3
D	Parallel Lines	300	2	3
F	Congressional Voting Record (CVR)	435	16	2
G	Gene Ontology (GPD194)	194	-	-
H	Activity of Daily Living (ADL) Patients	3,963	-	-



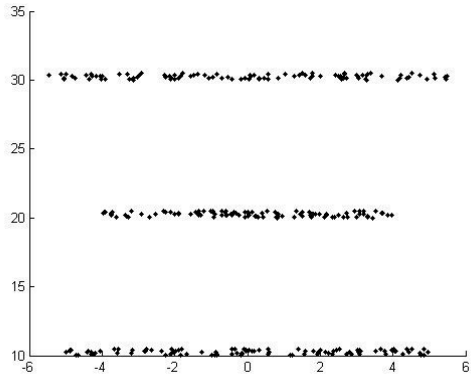
(a) Hepta



(b) WS3G



(c) O3G



(d) Parallel Lines

Fig. 2.1. Synthetic datasets

The average and standard deviation of the qe, te and objective function values over 10 runs were calculated. For most datasets we used the parameters listed in Table 2.3. Notice that σ_0 is not fixed across all datasets, rather it is dataset specific. σ_0 is the

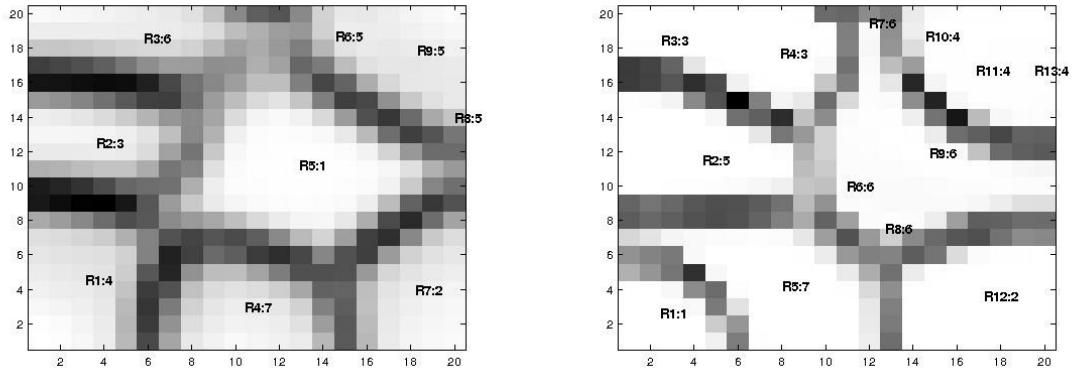
only parameter that was varied and it was determined by trial and error. Overall, we found $\sigma_0 \in [1, 3]$ to give reasonable results for all datasets presented in this chapter.

Table 2.3. Algorithm Parameters

Parameter	Value
q_0	1.01
q_f	2
σ_0	1-3
σ_f	0.5
C	400
Map size	20×20
Neighborhood function	Gaussian
Training length	10 epochs

2.9.1. **Hepta Dataset**

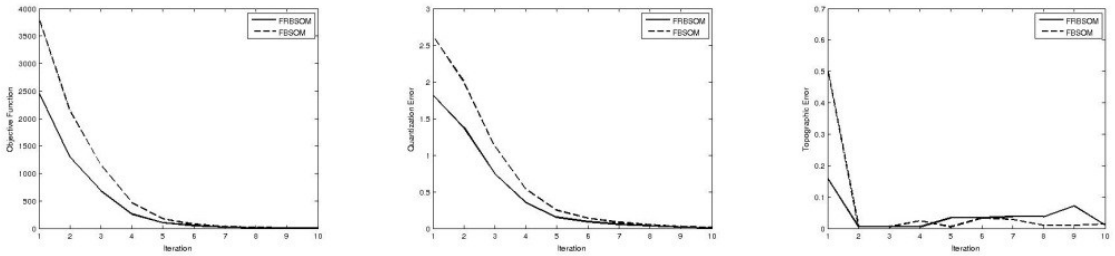
The Hepta dataset contains 212 data points divided into the seven classes of 30 points each and two additional points in the center group [41]. The center group of points is about twice as dense as any of the six groups (Fig. 2.1a, Table 2.2.A, which refers to row A in Table 2.2). The goal of the Hepta dataset is to validate if the clustering algorithm can find the clusters with varying densities. The maps generated by FSOM and RFSOM are shown in Fig. 2.2a-b, respectively. Every map is summarized by displaying the most representative labels for the catchment basins. The average qe, te and objective function values are presented in Table 2.7.A.



(a) FSOM (b) RFSOM

Fig. 2.2. Topographic map for the Hepta dataset

The behavior of the objective function, quantization error and topographic error is shown in Fig. 2.3a-c for one specific run. RFSOM starts with lower values (continuous line in Fig. 2.3), but as time progresses both algorithms behave similarly. In fact, these properties were observed in every dataset throughout the experiments (see Table 2.7). Thus, figures describing the objective function, *qe* and *te,* are only shown for this dataset.



(a) Objective function (b) quantization error (c) topographic error

Fig. 2.3. FSOM (dashed line) and RFSOM (continuous line) behavior on Hepta dataset over one run of 10 iterations

As mentioned before, for a given stimulus all neurons are winners to some degree. Neurons with high firing strength to stimulus will have the highest membership while neurons that have a weaker response will have a lower membership. The first few iterations of RFSOM will assign crisp memberships since q is very small. In other words, RFSOM's first few iterations resemble the RSOM behavior, meaning we start with a

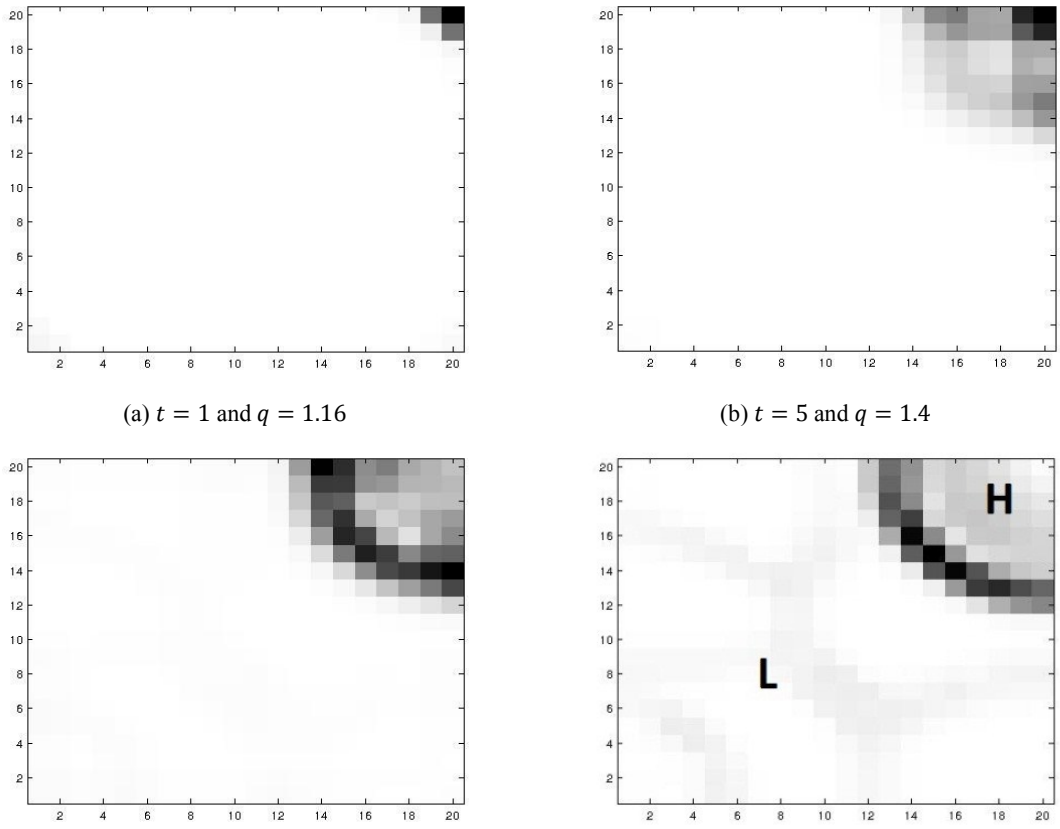
winning neuron. As time progresses and the fuzzifier value increases the memberships become fuzzier.

To illustrate this phenomenon for a given stimulus, we need to have a good understanding of the membership distribution among neurons. To do that, we will construct what we call the HL-matrix. One HL-matrix is constructed for one stimulus and has the same dimensions of the U-matrix. Given a stimulus o_k, the value of the ith neuron in the HL-matrix is computed as follows:

$$HL(i) = \sum_{j \in N_i} |u_{ik} - u_{jk}| \tag{2.23}$$

This calculation resembles the U-matrix, except we are now looking at the membership level. Of course, the goal is to minimize the values of the HL-matrix. Smaller values mean that adjacent neurons share similar memberships with the stimulus. The first iteration of RFSOM will assign crisp memberships similar to RSOM which result in an HL-matrix as shown in Fig. 2.4a. Fig. 2.4a demonstrates the "winner take-all" at $t = 2$: one neuron (top right corner) has a membership close to 1 and every other neuron has membership close to 0 (slight information sharing since $q = 1.16$). At $t = 5$ (Fig. 2.4b), q increases to 1.4 and the winning neuron starts sharing information about the stimulus with its neighbors and the membership value of that stimulus gets divided among those adjacent neurons. At $t = 8$ (Fig. 2.4c), where $q = 1.7$ a boundary begins to form. This separates the region in which neurons have high firing strength to the stimulus (see the region in the upper right corner of Fig. 2.4c) from the remaining neurons that have a weaker response. These two regions and the boundary become more defined in the last iteration at $t = 10$, where $q = 2$. We clearly see the L region (region with low active neurons) and the H region (region with high active neurons) separated by the boundary.

Hence, communication among neurons intensifies with time and the maximum sharing of information occurs in the last iterations (Fig. 2.4d). The similarities between Fig. 2.4d and the U-matrix in Fig. 2.2b are evident. Fig. 2.4d shows the catchment basin containing the set of neurons that responded to that stimulus, which corresponds to the catchment basin labeled R10, R11, and R13 in Fig. 2.2b.

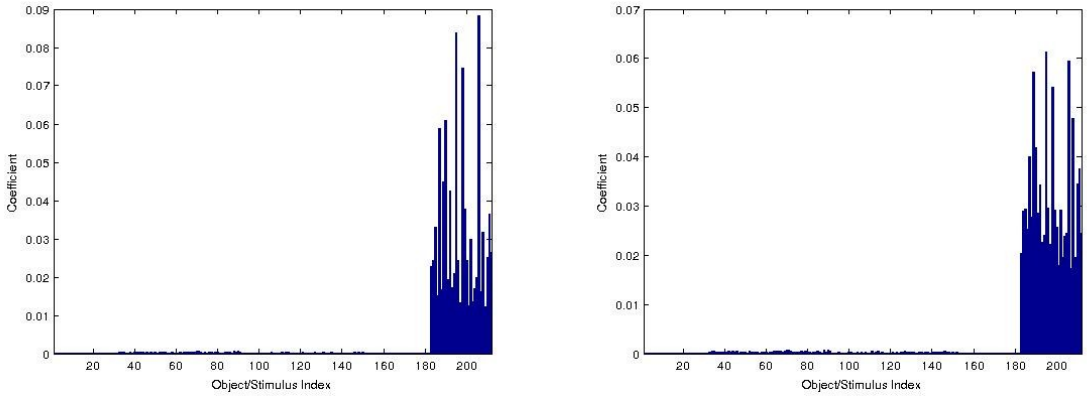


(a) $t = 1$ and $q = 1.16$ (b) $t = 5$ and $q = 1.4$

(c) $t = 8$ and $q = 1.7$ (d) $t = 10$ and $q = 2$

Fig. 2.4. RFSOM HL-matrix for a given stimulus at various iterations/times

We can verify that two neighboring neurons represent and sense similar stimuli by inspecting their coefficient vectors. For instance, let us select two neurons from the upper right corner of Fig. 2.4d, more specifically, neuron (18, 18) and (19, 16) whose coefficient vectors are shown in Fig. 2.5a-b, respectively. It is clear both neurons represent the same stimuli, but with varying weights.



(a) (18, 18) (b) (19, 16)

Fig. 2.5. Neuron coefficients for map location in Fig. 2.4d

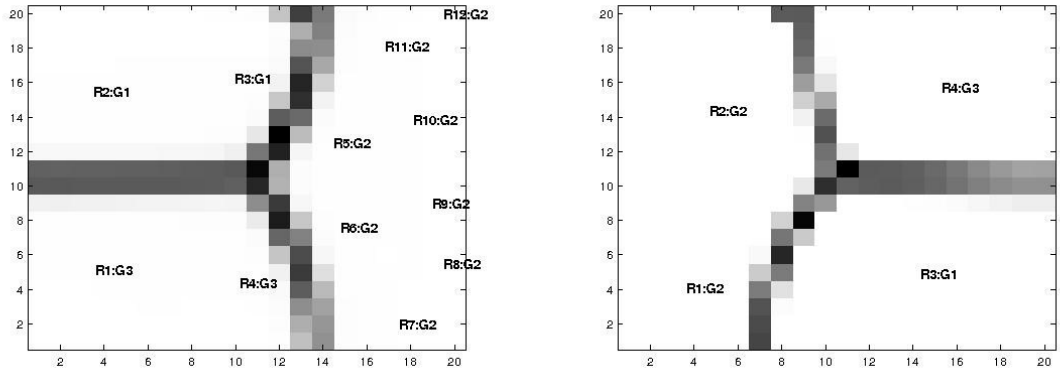
2.9.2. **Well-Separated Three Gaussians (WS3G)**

The WS3G dataset contains three classes that are well separated; this is a fairly easy dataset to cluster (Fig. 2.1b, Table 2.2.B). The main goal of this dataset is to demonstrate the behavior of the FSOM and RFSOM when the inter-cluster distance is large. Properties of the WS3G datasets are presented in Table 2.4.

Table 2.4. WS3G Dataset Properties

No. of Points	MEAN, M	Std. Dev., σ
500	(1,1)	0.1
500	(1,5)	0.1
500	(4,3)	0.1

The topographic maps for FSOM and RFSOM are shown in Fig. 2.6a-b, respectively. Both algorithms were able to successfully identify the three clusters. The average qe, te and objective function value are shown in Table 2.7.B. Overall, the average values calculated are very close in both algorithms.



(a) FSOM (b) RFSOM

Fig. 2.6. Topographic maps for the WS3G dataset

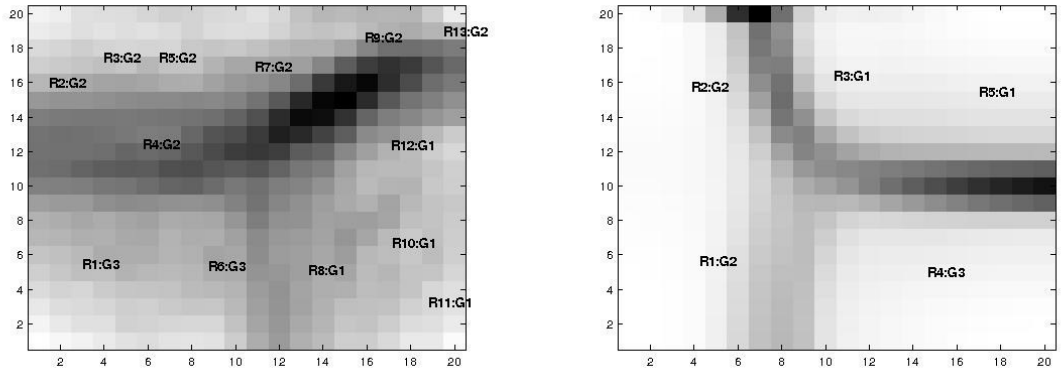
2.9.3. **Overlapping Three Gaussians (O3G)**

O3G is a similar dataset to WS3G; however, the clusters in O3G have larger variances which causes overlapping. Properties of the O3G datasets are presented in Table 2.5.

Table 2.5. O3G Dataset Properties

No. of Points	MEAN, M	Std. Dev., σ
500	(1,1)	1
500	(1,5)	1
500	(4,3)	1

The topographic maps produced by FSOM and RFSOM are shown in Fig. 2.7a-b, respectively. Since the three Gaussian clouds overlap, we expect more fuzziness in the maps. Both algorithms identified the three Gaussian clouds correctly as seen in Fig. 2.7. However, in FSOM boundaries between the clusters are fuzzier compared to RFSOM. Contrary to the WS3G dataset where we observed how close the errors in both algorithms are, on the O3G the qe, te and the objective function value are a little higher in FSOM than in RFSOM which may explain the difference between the topographic maps.

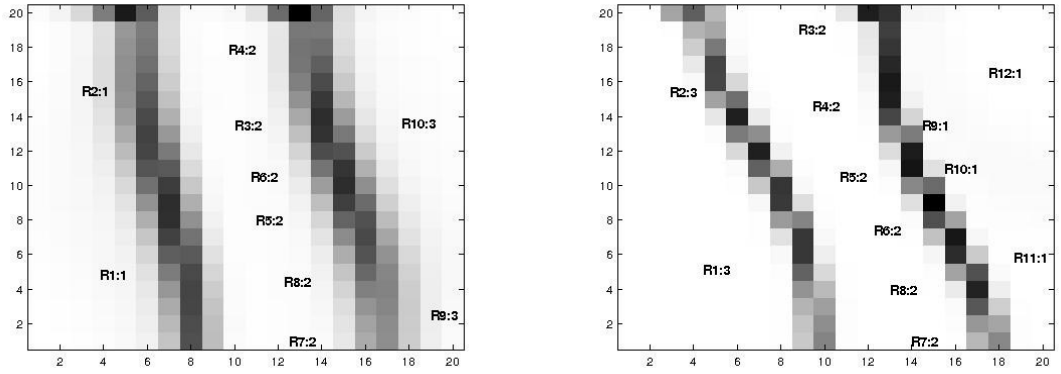


(a) FSOM (b) RFSOM

Fig. 2.7. Topographic maps for the O3G dataset

2.9.4. **Parallel Lines**

The lines dataset consists of three parallel lines of 100 points each (Fig. 2.1d, Table 2.2.D). The purpose of this dataset is to test whether or not FSOM and RFSOM would preserve the topology of the data. Indeed, both algorithms are capable of preserving the topology of the lines dataset (Fig. 2.8). The measured qe and objective function values are close and topographic errors are identical (Table 2.7.D).



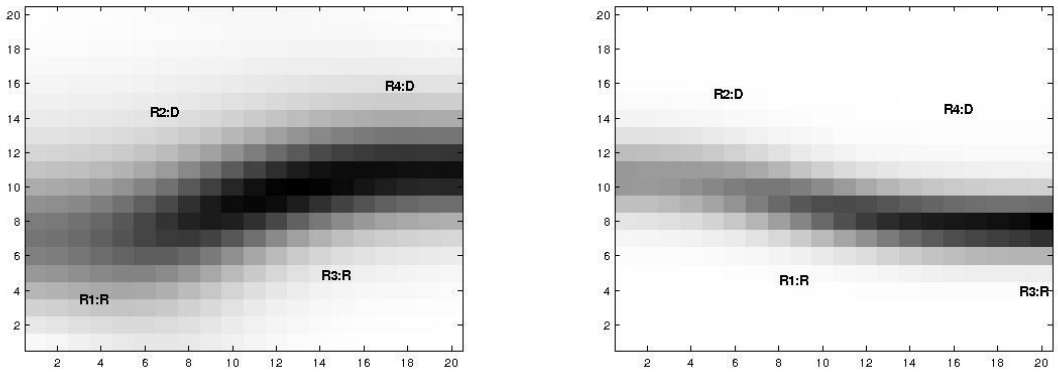
(a) FSOM (b) RFSOM

Fig. 2.8. Topographic maps for the Parallel Lines dataset

2.9.5. **Congressional Voting Record (CVR)**

The CVR dataset, obtained from UCI machine learning repository, contains 435 records. Each record corresponds to a congressman's vote on 16 issues. The class label for a record is either Democrat (D) or Republican (R).

The topographic maps generated by FSOM and RFSOM are shown in Fig. 2.9a-b, respectively. One can say that the maps are almost identical. Also, the errors produced from both maps are very close (Table 2.7.E). In fact, the average and the standard deviation of the topographic errors of both algorithms are identical (see topographic error column in Table 2.7.E).



(a) FSOM (b) RFSOM

Fig. 2.9. Topographic maps for the Congressional Voting Record dataset

2.9.6. **Gene Ontology Dataset (GPD194)**

The dataset discussed thus far originally existed in feature vectors and converted to relational data using a distance measure. This is not the case for the GPD194 dataset. It contains 194 sequences of human gene products and was obtained from ENSEMBL 2009 [51] and used by Havens *et. al.* to test the ontological SOM (OSOM) [34]. Table 2.6 describes the characteristics of the GPD194 dataset [52].

Table 2.6. Charactertistics of the GDP194 Dataset

ENSEMBL Family ID	F_i = Protein Family	Gene Symbols	No. of Genes	No. of Sequences
339	Myotubularin	MTMR1÷4, MTMR1÷4	7	21
73	Receptor Precursor	FGFR1÷4, RET, TEK, TIE1	7	87
42	Collagen Alpha Chain	COL1A2, COL21A2, COL24A2, COL27A2, COL2A1, COL3A1, COL4A1, COL4A2, COL4A3, COL4A6, COL5A3, COL9A1, COL9A2	13	86

The relational data GPD194 was produced using fuzzy measure similarity (FMS), which is based on Sugeno λ measure. Describing the FMS is outside the scope of this dissertation, and the reader is referred to [52] for more details about the GDP194 dataset and the FMS.

RFSOM was able to identify three main regions (Fig. 2.10), each representing one of the Protein families described in Table 2.6. The lower right corner of Fig. 2.10 corresponds to *myotubularin*, the lower left region represents *receptor precursor,* and the upper region corresponds to *collagen alpha chain*. Notice that the *collagen alpha chain* is further divided into three sub-regions: fibril forming collagens, type IV collagens, and fibril associated collagens with interrupted triple helices. Those regions are also observed and discussed in Popescu *et. al* [52].

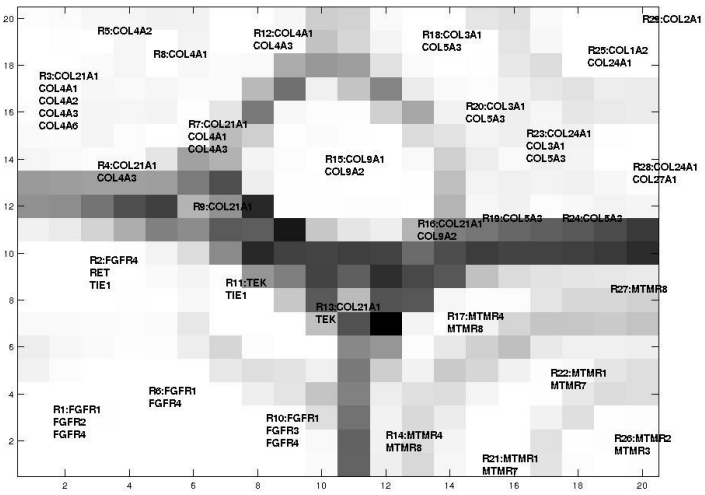


Fig. 2.10. Topographic map of the GPD194 dataset generated by RFSOM

2.9.7. **Activity of Daily Livings (ADL)**

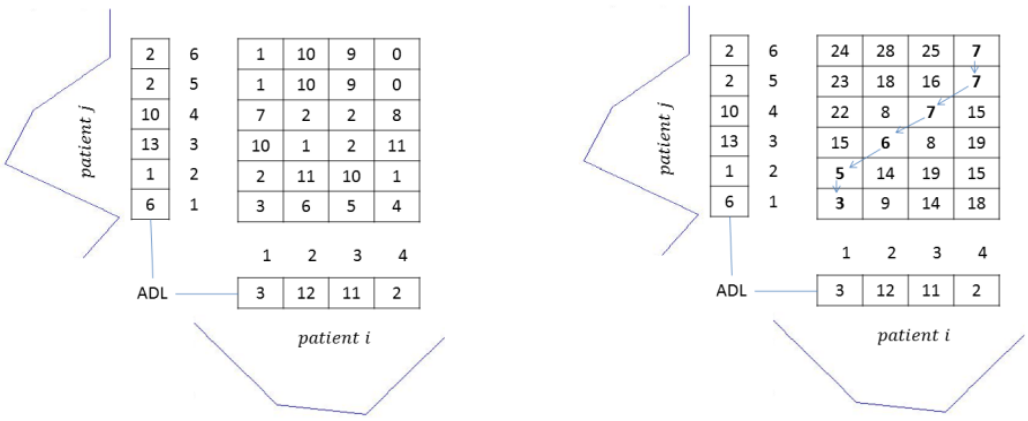
ADL dataset contains 3,963 patients originated from the 2006-2007 Minimum Data Set (MDS). Every patient has seven individual ADL item scores, each measuring the performance of a given activity. Those activities are: self-performance of bed mobility; transfer between surfaces; locomotion on the nursing unit; dressing; eating; toilet use; and personal hygiene. Patients' ADL scores are generally assessed every three months. For our analysis, we will not address the individual ADL scores, instead focusing on the seven-item ADL score, which is the sum of the individual ADL scores. The score varies from 0 to 28, where 0 indicates complete independence in all seven activities and 28 indicates complete dependence on others for all seven activities. For convenience we will refer to this score as ADL.

Not all patients were assessed every three months. Therefore, the length of the patients' trajectories varies depending on the number of times they were assessed. Some

patients may have four ADL scores while others may have 12. Hence, measuring the distance between patients is more difficult than computing the Euclidean distance. So, instead, we need to measure the distances among patients' trajectories represented as time series with uneven length and irregular time steps. To do that, we will use Dynamic Time Warping (DTW) distance measure [4].

DTW starts with two sequences (patients) $p_1 = p_{11}, p_{12}, \cdots, p_{1a}$ and $p_2 = p_{21}, p_{22}, \cdots, p_{2b}$ of lengths a and b, respectively. We construct an $a \times b$ matrix D where the element (i, j) corresponds to the distance between $p_{1\mathrm{i}}$ and $p_{2\mathrm{j}}$. For instance, $d(p_{1\mathrm{i}}, p_{2\mathrm{j}}) = |p_{1\mathrm{i}} - p_{2\mathrm{j}}|$. Therefore, each element in D corresponds to the alignment between points $p_{1\mathrm{i}}$, $p_{2\mathrm{j}}$ in the sequence p_1 and p_2, respectively. For example, in Fig. 2.11a patient i with four ADL scores is represented on the x-axis and patient j with six ADL scores is on the y-axis. The matrix in Fig. 2.12a shows the pairwise distances among the ADL scores of both patients. The goal of DTW is to minimize the cost of the warping path, which is done by computing a cumulative distance between points $p_{1\mathrm{i}}$, $p_{2\mathrm{j}}$ of the sequences (Fig. 2.11b), which is computed as follows [4]

$$\gamma(p_{1i}, p_{2j}) = d(p_{1i}, p_{2j}) + \min\left(\gamma(p_{1i-1}, p_{2j-1}), \gamma(p_{1i-1}, p_{2j}), \gamma(p_{1i}, p_{2j-1})\right) \tag{2.24}$$



(a) Pairwise distance between ADL scores belonging to the two patients

(b) Cumulative DTW distance matrix with the final distance $\gamma(p_4, q_6) = 7$

Fig. 2.11. Distance matrix produced by DTW

Based on the relational data produced using the DTW distance measure, RFSOM generated the topographic map as shown in Fig. 2.12. The map shows four distinct regions, each region representing a unique set of ADL trajectories. For instance, the upper left corner of Fig. 2.12 (R2 and R3) represents 471 patients whose ADL trajectory increased before it starting to decline. R5 contains 828 patients whose ADL trajectory decreased and it appears to stabilize. The lower left corner (R1 and R4) represents 1,328 patients whose trajectory consistently increases. R7 contains 780 who exhibit characteristics similar to patients in R1 and R4 except for an apparent improvement in their ADL score after the sudden increase. Lastly, the upper right corner (R6, R8 and R9) represents 556 patients whose ADL trajectory seems to be unstable and fluctuating.

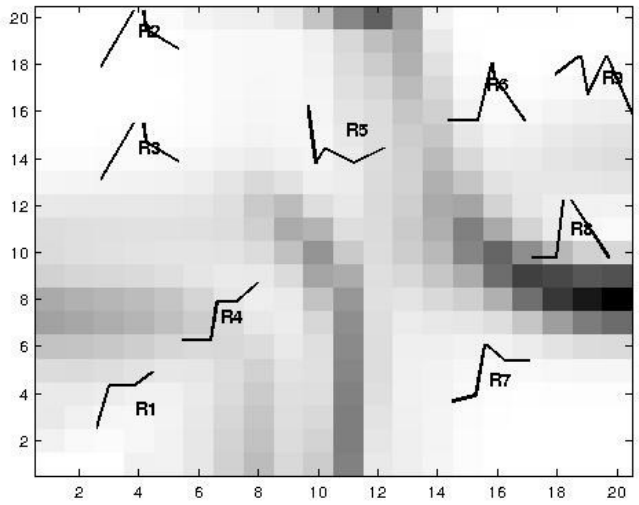


Fig. 2.12. Topographic map generated by RFSOM using the relational data produced using DTW

Table 2.7. RFSOM Results

	Dataset	QUANTIZATION ERROR		TOPOGRAPHIC ERROR		OBJECTIVE FUNCTION	
		FSOM	RFSOM	FSOM	RFSOM	FSOM	RFSOM
A	Hepta	0.0123 ($1x10^{-4}$)	0.0087 ($4x10^{-5}$)	0.0198 (0.0187)	0.0292 (0.0341)	2.9 (0.0275)	2.09 (0.0083)
B	Well Separated Gaussians (WS3G)	0.0018 ($7x10^{-6}$)	0.0013 ($3x10^{-6}$)	0.0023 (0.0015)	0.003 (0.0011)	3.18 (0.0145)	2.29 (0.0037)
C	Overlapping Gaussians (O3G)	0.0107 ($9x10^{-5}$)	0.0082 ($6x10^{-6}$)	0.0297 (0.0247)	0.0135 (0.0091)	18.02 (0.1433)	13.72 (0.0072)
D	Parallel Lines	0.0345 (0.0042)	0.0197 (0.0002)	0.0177 (0.0244)	0.0177 (0.0104)	11.54 (1.41)	6.51 (0.088)
E	Congressional Voting Record (CVR)	0.0102 ($1x10^{-5}$)	0.0071 ($2x10^{-6}$)	0.0002 (0.0007)	0.0002 (0.0007)	4.96 (0.0063)	3.44 (0.0004)
F	Gene Ontology (GDP194)	-	0.00004 ($2x10^{-5}$)	-	0.534 (0.316)	-	0.013 (0.0029)
G	Activity of Daily Living (ADL)	-	0.1204 (0.0003)	-	0.171 (0.172)	-	465.96 (7.774)

A summary of the FSOM and RFSOM performance, the average quantization and topographic errors and objective function value are estimated from 10 runs of the algorithms. The value in parentheses represents the standard deviation.

2.10. Conclusion

This chapter presented the RFSOM algorithm, which is a generalization of the RSOM[33]. RFSOM acts on relational data that describes the pairwise dissimilarities among objects. Unlike RSOM or the classical SOM, by incorporating fuzziness to the neurons, the notion of BMU no longer exists in the FSOM and RFSOM. Instead, every object is associated with a neuron with varying grade of membership.

RFSOM's initial iterations resemble the classical crisp SOM since the fuzzifier is small. In other words, RFSOM begins as a "winner takes-all" paradigm and as time progresses and the value of the fuzzifier increases, neighboring neurons begin to share and communicate information about the stimuli they sense. This is made possible by employing a monotonically increasing fuzzifier and monotonically decreasing neighborhood kernel. We demonstrated this concept using the HL-matrix, where neurons that exhibit strong responses to a given stimuli are separated by a boundary from the neurons that display a weaker response.

We employed five datasets (some from FCPS and the UCI repository) to test the performance of FSOM and RFSOM algorithms. Additionally, we tested the performance of the RFSOM on real relational data, Gene Ontology and Activity of Daily Living. On the datasets that we tested on both the FSOM and RFSOM, we saw similar topographic maps and very close error rates. In few cases, the error rates were identical.

Relational data clustering and visualization are two of the effective approaches to handle data that do not exist in object form. However, algorithms such as RFSOM are inherently complex, computationally expensive, and most importantly they lack, the ability to handle large relational data. For instance, the time complexity of RFSOM is

$O(c^2n^3t)$, where c is the number of neurons, n is the number of objects, and t is the number of iterations. The memory complexity is $O(n^2 + (c * n)^3)$, where n^2 is the memory complexity of the relational matrix R, $(c * n)^3$ is the complexity for storing the distance matrix (2.8), partition matrix (2.12) and the coefficient vectors (2.16). For example, if $n = 10{,}000$ and $c = 400$, the memory usage is estimated to be 2.9 GB. Additionally, RFSOM can contain from several hundreds to thousands of neurons, which in turn increases the complexity and memory usage of the algorithm. Although RSOM was proposed for large dissimilarity datasets[53], direct applicability of that technique to RFSOM is not trivial since RFSOM does not assign winning neurons to objects.

In the RFSOM evaluation, in section 2.7.2, we used the crisp topographic error to measure how well the RFSOM preserves the data topology. So, in order to use the crisp topographic error we have to find the two neurons with the highest membership value to that stimulus and pretend that those neurons are the first and second best-matching units. The flow in this approach is that it relies on only two neurons and does not take full advantage of the fuzzy membership values. The next chapter is dedicated to address this problem and proposed two new methods to overcome this problem.

CHAPTER 3

TOPOLOGY PRESERVATION IN FUZZY SELF-ORGANIZING MAPS

One of the important properties of SOM is its topology preservation of the input data. The topographic error is one of the techniques proposed to measure how well the continuity of the map is preserved. However, this existing topographic error is only applicable to the crisp SOM algorithms and cannot be adapted to the fuzzy SOM (FSOM) since FSOM does not assign a unique winning neuron to the input patterns. In this chpater, we propose a new technique to measure the topology preservation of the FSOM algorithms. The new measure relies on the distribution of the membership values on the map. A low topographic error is achieved when neighboring neurons share the same or similar membership values in a given input pattern.

3.1. Introduction

Self-Organizing Maps (SOM) is an unsupervised neural network algorithm. SOM tries to map the *s*-dimensional input patterns to a 2-dimensional lattice, preserve the topology of the data, and cluster the neurons that represent similar input patterns, which can be visualized using a 2D or 3D map such as the Unified Distance Matrix (U-Matrix) [44]. Several formulations and modifications were proposed to the classical SOM algorithm, such as the Self-Organizing Semantic Maps [20], Ontological SOM [34], Relational Topographic Maps [33], and WEBSOM [24]. Another class of SOMs is the fuzzy SOM algorithms. The general idea of FSOM is to integrate fuzzy set theory into neural networks to give SOM the capabilities of handling uncertainly in the data. FSOM can also be divided into two categories: object FSOM [25], [26], [29], [54], [55] where

input patterns are represented as feature vectors and the relational FSOM [37] which handles relational data.

Regardless of the type of SOM algorithm they all share one important feature that is topology preservation. Topology preservation means that neighboring data points in the input space are mapped to nearby neurons in the output space. Once a good mapping is established, SOM can represent the high dimensional input space in a 2-dimensional output map that preserves the topology of the input data. This in turn yields better visualization and reveals more information about the structure and the clusters presented in high dimensional input space. To ensure that SOM has established good mapping, we need to measure or quantify the goodness of SOM. Different measures are proposed to accomplish this goal, such as the quantization error and the topographic error. Those errors are widely used in SOM and while the quantization error was adapted for the object and relational FSOM [37], no formulation is yet proposed to measure the topological preservation or continuity of the map in the FSOM algorithms.

The topographic errors used in SOM are not directly applicable to FSOM due to the fact that FSOM does not assign a unique winning neuron for every object, instead every neuron is a winning a neuron of every object with a varying degree of membership. Therefore, in this work, we propose a technique to measure the topographic error in FSOM algorithms.

The reminder of the chapter is organized as follows: Section 3.2 gives an overview of the fuzzy relational SOM. Section 3.3 discusses some of the well-known methods to measure the goodness of SOM. Section 3.4 explains a new approach to

measure the topographic error in FSOM. Section 3.5 presents experimental results and we conclude this chpater with remarks and discussion in Section 3.6.

3.2. Relational Fuzzy Self-Organizing Maps

In this section we give a very brief overview of the fuzzy relational SOM algorithm (RFSOM) [37] on which the experimental results discussed in section 3.5 are based on. However, the same technique for evaluating the topology preservation can be used on object FSOM or any FSOM algorithm. For a complete analysis of RFSOM the reader is referred to [37].

Given n input objects $O = \{o_1, \dots, o_n\}$ described by feature vectors $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^s$or by a relational matrix $R = [r_{jk}] = \left[\|x_j - x_k\|^2\right]$[33], [37] SOM constructs a lattice or map of c number of neurons, that are connected using a neighborhood kernel, h, such the neighborhood between neuron i and j is given by

$$h_{ij} = \exp\left(\frac{-\|a_i - a_j\|^2}{2\sigma^2(t)}\right), \tag{3.1}$$

where a_i is the coordinate of the ith neuron in the output space (two dimensional space) and σ is a monotonically decreasing neighborhood size. Every neuron has a corresponding s-dimensional weight vector, $m = \{m_1, \dots, m_c\}$ or an n-dimensional coefficient vector in the relational algorithm. One of the goals of the classical crisp SOM algorithm is to assign every s-dimensional input signal, o_k, a winning or a best-matching unit (BMU), w_k, according to

$$w_k = \arg\min_i \|m_i - x_k\|^2 \quad \forall 1 \le i \le c \text{ and } 1 \le k \le n\,. \tag{3.2}$$

Effectively, SOM assigns a full membership of o_k in neuron w_k,

$$u_{ik} = \begin{cases} 1, \text{if } w_k = i \\ 0, \text{otherwise} \end{cases} \tag{3.3}$$

An alternative to this approach is to assign a fuzzy membership for all objects in every neuron as described in [37]. The RFSOM proposed in [37] produces fuzzy partitions $U \in M_{fcn}$ where

$$M_{fcn} = \left\{ U \in \mathbb{R}^{c \times n} \middle| \begin{array}{c} u_{ik} \in [0,1], \\ \sum_{k=1}^{n} u_{ik} > 0, \sum_{i=1}^{c} u_{ik} = 1, \\ \forall\ 1 \le i \le c \ and \ 1 \le k \le n \end{array} \right\}. \tag{3.4}$$

Introducing fuzzy memberships to SOM as in RFSOM adds another layer of complexity due to the fact that all neurons are winners of all objects to some degree. Thus, any error measurement made in RFSOM has to factor in all membership values of all input signals in all neurons. In [37] we showed that the quantization error in SOM can be easily adapted to the RFSOM, but this is not the case regarding the topographic error. In the next section we will briefly review two of the major SOM evaluation techniques followed by a new method to evaluate the topology preservation of RFSOM in section 3.4.

3.3. Topology Preservation in SOM

Several measures are proposed to measure the goodness of the map. Some measures, such as the quantization error, evaluate the fitness of SOM to the input data. This error calculates the average distance between the input patterns and their corresponding winning neurons. Optimal map is expected to produce a smaller error, which means the input patterns are close [43] to their winning neurons. Quantization error for SOM is shown in (3.5).

$$qe_c = \frac{1}{n}\sum_{k=1}^{n}\left\|x_k - m_{w_k}\right\| \tag{3.5}$$

Similarly, the FSOM quantization error is defined as [37]

$$qe_f = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{q}\left\|x_k - m_{w_i}\right\| \tag{3.6}$$

However, the crisp and fuzzy quantization errors in (3.5) and (3.6) may not accurately measure the topographic preservation of the map. Instead, one can quantify the relation between the codebook weight vectors and the associated neurons in the map as in the topographic product [43]. This gives a sense on how well the *s*-dimensional space is mapped to a 2-dimensional lattice [42]. A different approach is to use the topographic error.

The topographic error measures the continuity of the map or how well an input signal preserves the local continuity of the map [43]. When the first and second best-matching units to object o_k are adjacent in the map space, then o_k is said to preserve local map continuity and if they are not adjacent then there is a topological error. To evaluate the overall topology of the map the proportion of input signals for which the first and second best-matching units are not adjacent is measured (3.7) [43]. A lower error yields a better map and topology.

$$te_c = \frac{1}{n}\sum_{k=1}^{n} adj(o_k) \tag{3.7}$$

where

$$adj(o_k) = \begin{cases} 1, if\ the\ first\ and\ second\ BMUs\ are\ not\ adjacent \\ 0, otherwise \end{cases}.$$

Another matric for measuring topology preservation in crisp SOM is discussed in [56]. The metric is said to be topology preserving if for any x_i, if x_j is the kth nearest neighbor of x_i, then w_j is the kth nearest neighbor of w_i.

The concept of first and second BMUs is not applicable to FSOM since every unit i is a BMU of every object o_k with a degree u_{ik}. A possible workaround is to harden the fuzzy partition produced by FSOM to find the BMU then compute the topographic error as in (3.7). Another approach is to consider the two neurons in which o_k has the highest membership as the first and second BMUs. However, neither of these two approaches exploits the membership grade of FSOM. Therefore, a new formulation to measure the local continuity of the map in FSOM is needed to evaluate its goodness and the topology preservation, which is the topic of the next section.

3.4. Topology Preservation in RFSOM

In RFSOM every neuron is a BMU of every object with a varying degree of membership. Regardless, both the crisp and fuzzy SOM should preserve the topology. Therefore, every pattern presented to RFSOM is also expected to preserve the local continuity of the map. One can consider the first and second neuron with the highest membership to o_k as best and second winning neurons, w_k and w_j. However, this flawed strategy uses only two neurons and discards all other neurons despite the fact other neurons might have high membership to o_k. Relying on two neurons can only give us a false sense of the map continuity. Consider a scenario where the first and second neurons with the highest memberships to o_k, w_k and w_j are immediate neighbors, but the neuron with the third highest membership to o_k is distant from w_k and w_j. A better approach is to use the membership values and utilize all neurons when measuring the topology

preservation of RFSOM. More specifically, by looking at the differences of the membership values between the neurons and their immediate neighbors we can make a conclusion on how well the local topology of the map is preserved.

For any given object o_k in RFSOM, we expect neurons with high firing strength to o_k to be concentrated in one region (H region). Also, not all neurons have the same firing strength, as we go further away from the H region, the membership values start to diminish gradually. If the correct data topology is discovered by RFSOM, the H region corresponds to the catchment basin or part of it where o_k belongs the most. In such case, we say that o_k preserves the local continuity of the map. On the other hand, if the neurons of high membership to object o_k are scattered throughout the map or if no H region is identified then the object fails to preserve the topology of the map. For exemplification, Fig. 3.1a shows the topographic map for Hepta dataset [41] and Fig. 3.1b shows the H region for some input pattern.

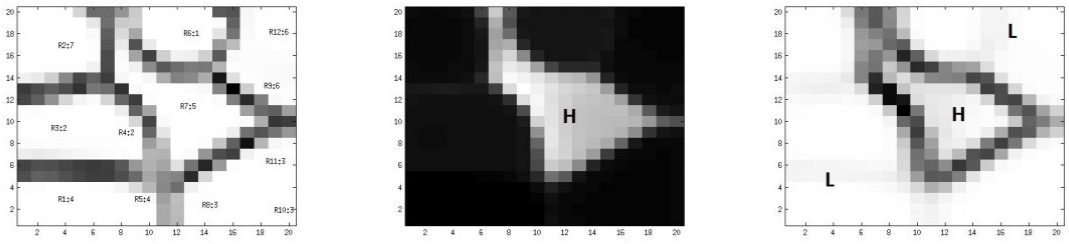
In order to assess how well an object o_k preserves the local continuity of the map we first need to compute the HL-matrix. HL-matrix has the same dimensions as the topographic map and c neurons. A topology preserving HL-matrix includes two main regions, the H region which contains the neurons with high membership to object o_k and the L region containing the rest of the neurons which have low membership values to o_k, as shown in Fig. 3.1c. Observe that the HL-matrix of o_k represents a snapshot of the U-matrix (Fig. 3.1a). Adjacent neurons in regions H and L should have similar membership values to o_k. Hence, the difference in the membership values between a neuron i and its immediate neighbors $N(i)$ should be very small with exception to the bordering neurons that separate the H and L regions as shown in Fig. 3.1c. For a given object o_k we first

compute its HL-matrix where the value at every neuron's coordinate is computed as follows

$$HL(i) = \sum_{j \in N(i)} |u_{ik} - u_{jk}| \tag{3.8}$$

$HL(i)$ corresponds to the sum of differences between the membership u_{ik} and the memberships of o_k in $N(i)$. Then that difference is projected on top of the grid position of every neuron. This process is performed for every input pattern. For a small topographic error the value for every neuron $HL(i)$ should be as small as possible, which means that the neuron i and its neighbors $N(i)$ have very similar memberships to the given input pattern.

For an object to preserve the local topology it is imperative that we identify a single region labeled H. Failure in identifying a single region H will cause the topographic error to increase and possibly reaching its maximum value. This technique is stricter than the topographic error in (3.7). Here we want to ensure that two adjacent neurons have similar membership to o_k, which is somewhat similar to (3.7), but in addition we would like to ensure that o_k preserves the local continuity within a specific region of the map.



(a) Hepta dataset RFSOM map (b) H region of random point (c) HL-matrix of a random point

Fig. 3.1. Topology preservation for some random point o_k

For more accurate evaluation of the topology preservation it is recommended that we normalize the HL-matrix as follows

$$NHL(i) = \frac{HL(i)}{\sum_{j=1}^{c} HL(j)} \tag{3.9}$$

We give two important reasons for this normalization: first, it sets an upper bound on the topographic error, similar to (3.7) the maximum error is 1. Second, normalization is crucial when comparing the topographic errors across different maps. Once the normalized HL-matrix is computed, the final topographic error of a single object o_k will depend on the neurons identified in the region labeled H. The error is simply the sum of values enclosed in the H region of the NHL-matrix (3.10). As the values in the H region get smaller, so does the topographic error. Meaning that adjacent neurons in the H region share similar memberships to o_k.

$$te_f(k) = \sum_{i \in H} NHL(i) \tag{3.10}$$

The final topographic error of the map is computed as the average topographic error overall the objects as

$$te_f = \frac{1}{n} \sum_{k=1}^{n} te_f(k)\,. \tag{3.11}$$

We would like to point out few remarks about the proposed measure (3.11): first, the only way for a map to result in a zero topographic error is when the values in the H region are equal to zero. In other word, when neuron $i \in H$ and its neighbors $N(i)$ have an identical membership to o_k. Second, an HL–matrix may not contain a unique H region. In this situation the topographic error can reach its maximum, which is the sum of all values in the NHL-matrix ($te_f = 1$).

Fig. 3.2 summarizes the process described above. Starting with the input data X or D, FSOM/RFSOM outputs the 2D map and the $c \times n$ partition matrix U. For every column in U, which corresponds to some pattern k, we can visualize the distribution of the memberships across the map. The distribution will give us a nice visual of which area on the map corresponds to the neuron that have firing strength to the input pattern. Then, we compute the HL matrix, which segments the neurons that have high firing strength from the ones having low membership values. Finally, based on the neurons in the H region we compute the topographic error for the kth patterns, followed by the computation of the overall topographic error.

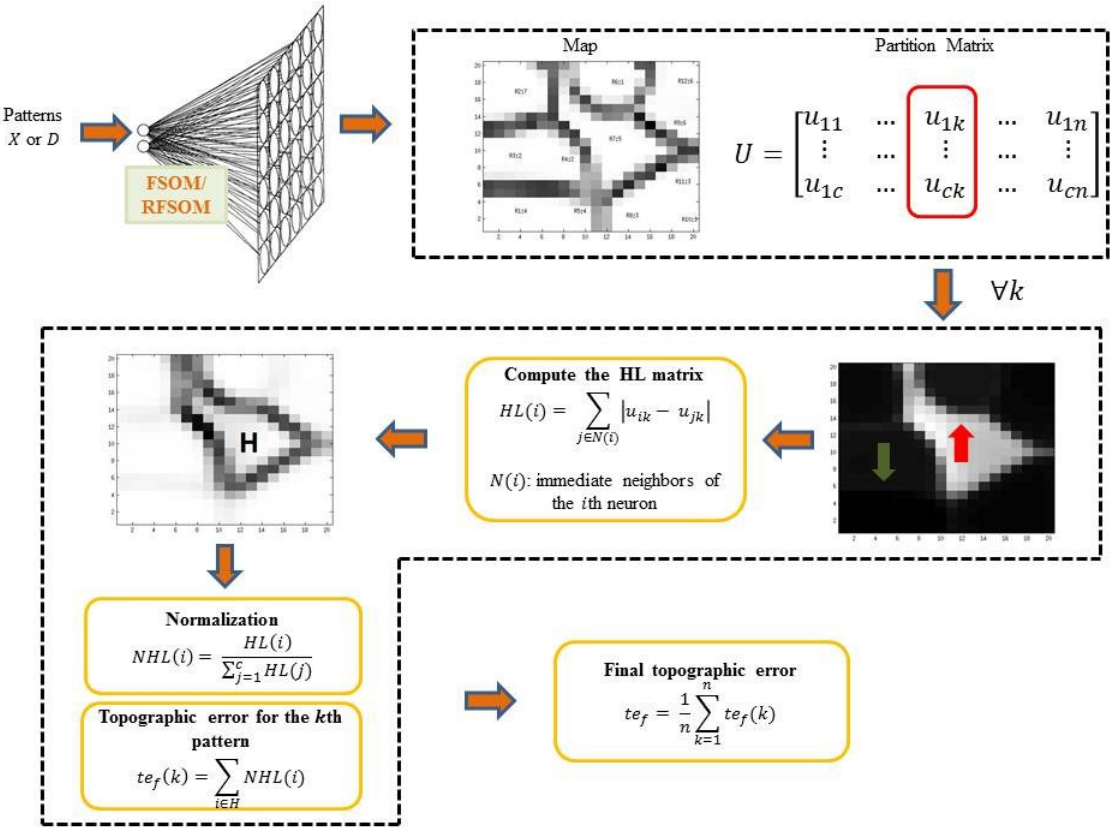


Fig. 3.2. Topology preservation in FSOM/RFSOM using image segmentation

The approach presented so far suffers from some limitations and drawbacks: (a) It is computational expensive since it relies on image segmentation. For every pattern k we have to perform image segmentation to identify the region containing the neurons with high firing strength to k. The more input patterns we have and the large the map is, the more time it will take to compute the final topographic error. In many cases it can take longer than the actual running time of the FSOM/RFSOM algorithm. (b) Image segmentation cannot always be reliable. In fact, the reliability depends on how noisy or not noisy the map produced by FSOM/RFSOM is. Some resulting maps can be very noisy, which makes it harder for the image segmentation algorithms to identify the region with high membership values. (c) It is threshold based. Image segmentation algorithms can be very sensitive and can cause over segmentation of the map if we ran the algorithm directly on the FSOM/RFSOM map. Therefore, we have to set a threshold on the image before segmentation. The hope is the threshold we choose we will allow the segmentation algorithm to identify the region were the neurons with high firing strength are located. Sometimes we might overestimate or underestimate the threshold resulting in an inaccurate topographic error. (d) The proposed error does not use all the neurons to compute the topographic error, which contradicts with the whole idea of having a membership function. Currently, only those neurons within some defined regions are used to measure the error, while in fact we all neurons should contribute according to their membership function. Due to those reasons we propose an alternative way to measure the topographic error in fuzzy SOM algorithms.

The alternative approach is more intuitive and simpler than the image segmentation based approach. Fig. 3.3 demonstrates how an input pattern preserves the

map continuity in fuzzy SOM algorithms. As the distance between any two neurons i and j increases, we expect the two neurons to represent different patterns. Hence, for some input pattern k the difference between u_{ik} and u_{jk} should be get larger as the distance between neurons i and j increases. On the other hand, if the neurons p and q are close to each other, then we expect the membership values u_{pk} and u_{qk} to similar.

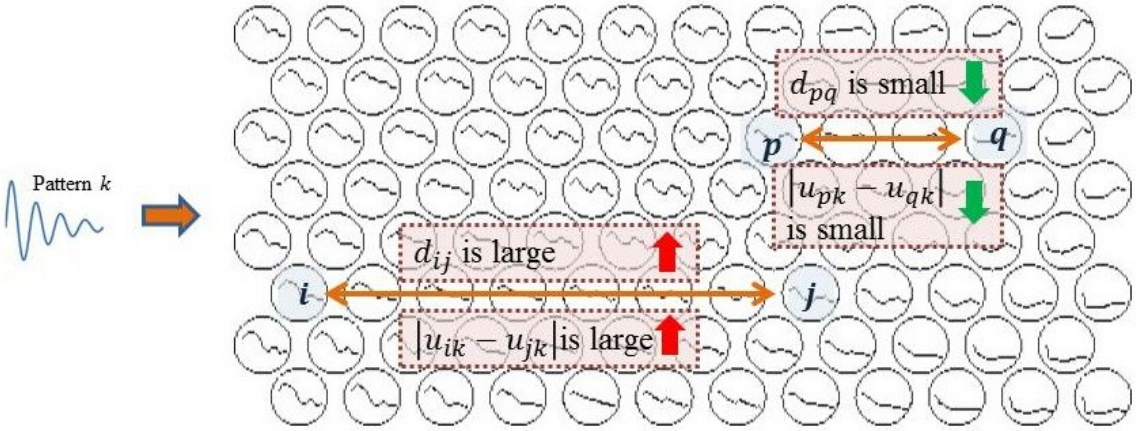


Fig. 3.3. A topology preserving input pattern k

So, in order to compute the fuzzy topographic error for some pattern k we first compute the spreading of the membership value among all neurons relative to k using (3.12).

$$(\Delta u)_{ijk} = \frac{|u_{ik} - u_{jk}|}{d_{ij}};\ \forall\ 1 \le i, j \le c \tag{3.12}$$

This gives us the error that every pair of neurons has contributed to k, which we will use to compute the overall topographic error for pattern k as:

$$te_f(k) = \frac{c(c-1)}{2} \sum_{i=1}^{c} \sum_{j=1}^{c} (\Delta u)_{ijk}\,. \tag{3.13}$$

Eq. (3.13) is repeated for every k, which results in a final topographic error:

$$te_f = \frac{1}{n}\sum_{k=1}^{n} te_f(k) \tag{3.14}$$

This technique is summarized in Fig. 3.4.

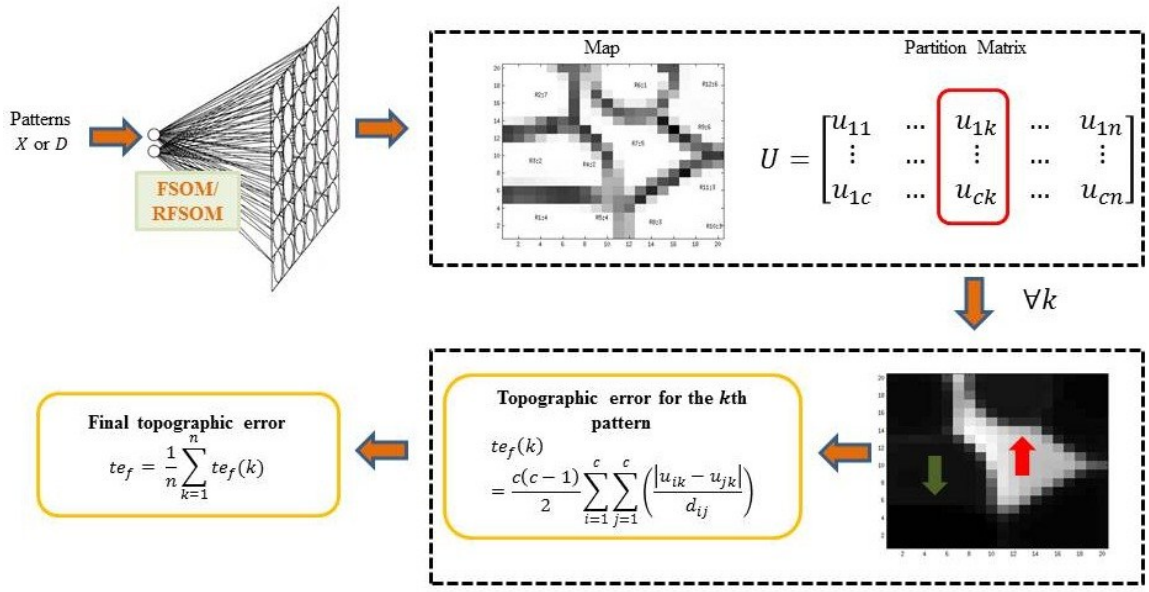


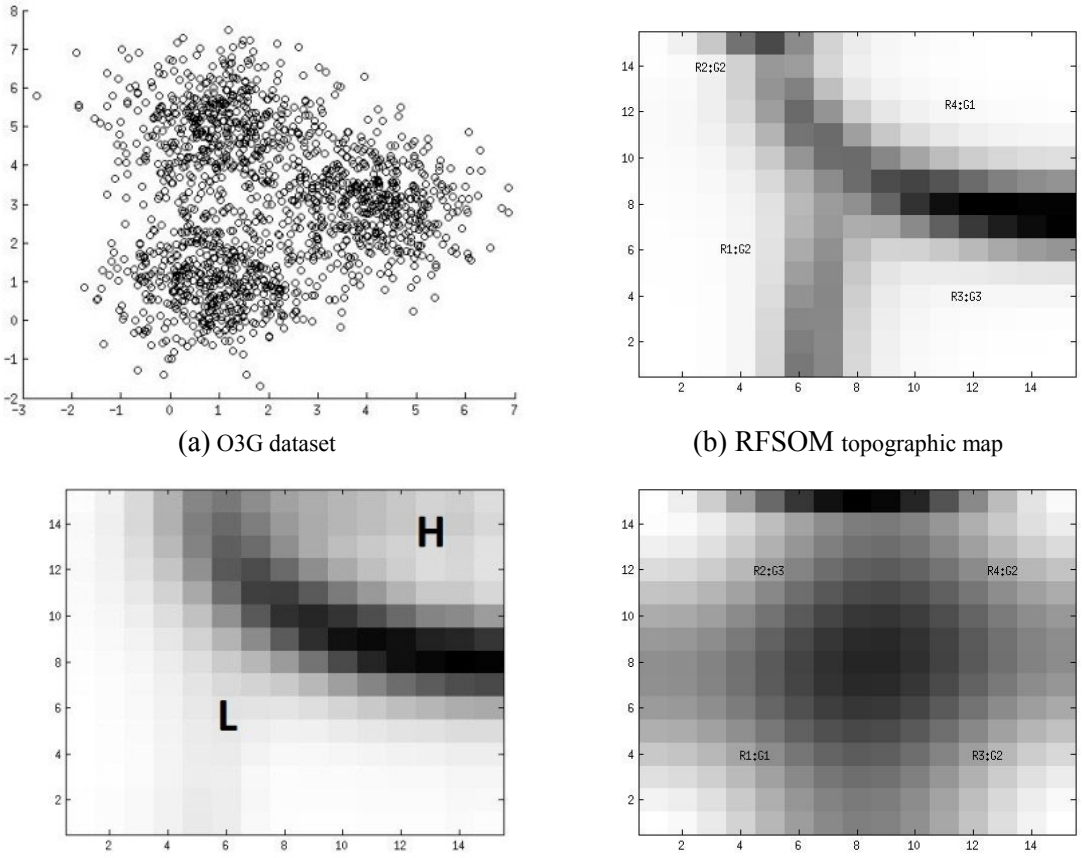
Fig. 3.4. Topology preservation in FSOM/RFSOM process

3.5. Experimental Results

3.5.1. Fuzzy Topographic Error on O3G

The overlapping three Gaussian (O3G) dataset contains three clusters of size 500 each (Fig. 3.5a). Clusters in O3G have larger variance which causes overlapping. We setup RFSOM with initial (σ_0), final neighborhood radius (σ_f), initial fuzzifier (q_0), final fuzzifier (q_f), map dimensions and number of epochs to be 2, 0.5, 1, 2, 15×15 and 10, respectively. The resulting topographic map is shown in Fig. 3.5b.

From Fig. 3.5c it is clear that the HL-matrix for some given pattern contains the two H and L regions, which is an indication that it preserves the local continuity of the map.

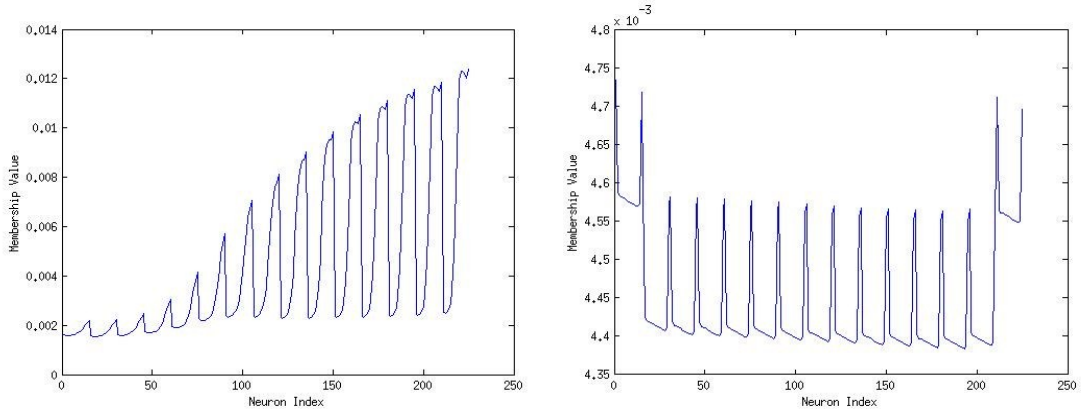


(a) O3G dataset (b) RFSOM topographic map

(c) HL-matrix for a random point (d) Non-topology preserving map

Fig. 3.5. Topology preservation from the O3G

In a topology preserving map, such as the one in Fig. 3.1c, the membership u_{ik} is expected gradually increase while approaching the H region and neurons with the highest membership should be located within the H region as demonstrated in Fig. 3.6a. On the other hand, a non-topology preserving map as in Fig. 3.5d we see a more chaotic membership values among the neurons (Fig. 3.6b) causing te_f to increase. It could also mean that the four regions or corners in Fig. 3.5d are wrapped around to form one region representing all input patterns, failing to preserve the topology.



(a) Topology preserving membership signal (b) Non-topology preserving membership signal

Fig. 3.6. The membership values for a random point

Now, let us compare te_c and te_f for the maps in Fig. 3.1b and Fig. 3.1d. If we compute te_c for the map in Fig. 3.5b (Table 3.1), where the two neurons with the highest membership value to an input pattern are used as the first and second BMU, we find it higher than the te_c in Fig. 3.5d (Table 3.1). On the contrary, te_f has increased from 0.32 in Fig. 3.1b to $te_f = 1$ in Fig. 3.1d. In this scenario te_f reveals more information about the goodness of the map resulted from RFSOM since we probably expect Fig. 3.5b to be more topology preserving than Fig. 3.5d.

Table 3.1. Behaviour of te_c and te_f when varying σ_0

Map	$\boldsymbol{\sigma_0}$	$\boldsymbol{te_c}$	$\boldsymbol{te_f}$
Fig. 3.5b	2	0.021 (0.006)	0.32 (0.03)
Fig. 3.5d	4	0.004 (0.004)	1 (0)

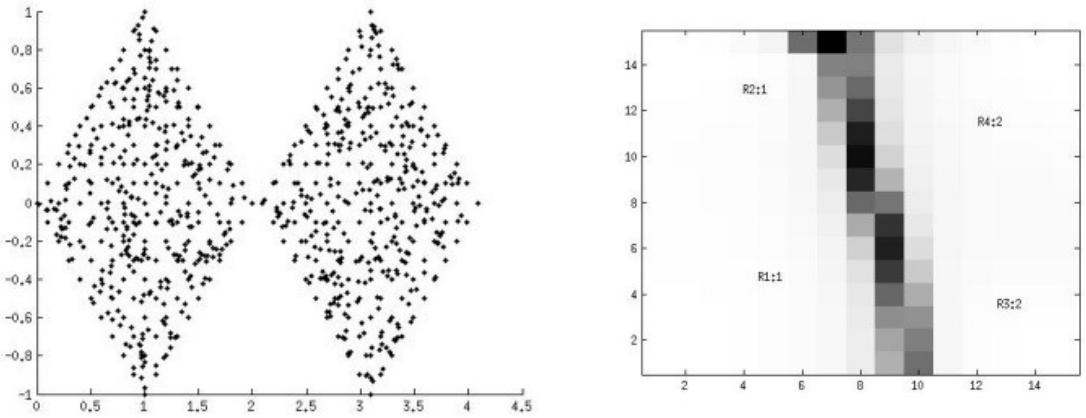
If we instead use the second approach for measure the topographic error we will find that $te_f = 0.0003$, a very small number compared to the image segmentation based approach. That is because the two approaches are searching for different criteria. The image segmentation based approach attempts to find a specific region/catchment basin where the neurons has a strong firing strength to the input pattern. Then it measures if

that region preserves the continuity of the map (local continuity). The second approach attempts to compute a global continuity of the map by using all neurons.

3.5.2. **Fuzzy Topographic Error and Map Dimensions**

In this experiment we will use the Two Diamonds dataset from the Fundamental Clustering Problem Suite (FCPS), which contains 800 data points [41] as shown in Fig. 3.7a. On this dataset we will show how the map dimensions can have an influence on the topographic error. Same parameters used on the O3G dataset will be used for the Two Diamonds with exception to the map dimensions which is set it be 20×20. The resulting topographic map is shown in Fig. 3.7b.

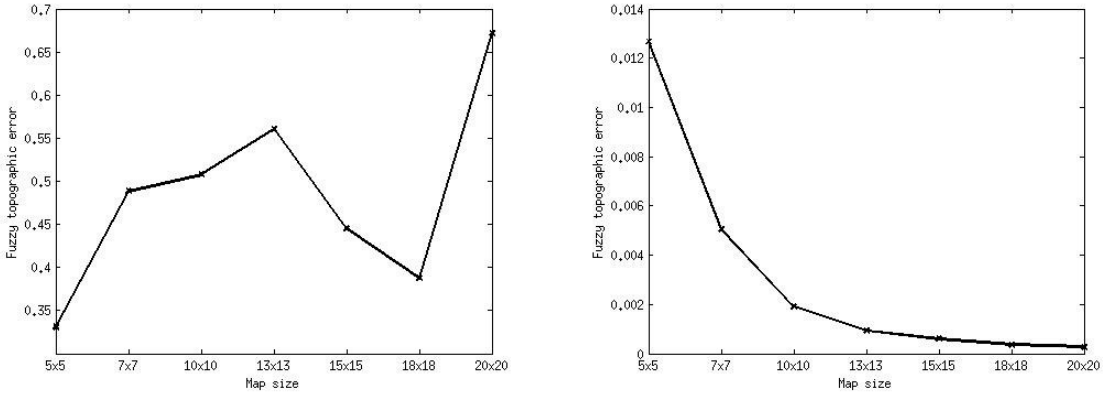
A smaller map of size 10×10 was also produced for the Two Diamonds dataset. It is not shown since it is very similar to the map in Fig. 3.7b. We found the overall topological error of the 20×20 map measured to be 0.33. As the map size increases it is likely that the H region increases which in some cases causes an increase in the membership variance among adjacent neurons. On the contrary, 10×10 map might have lower variance in the memberships among neighboring neurons in the H region and hence a lower topographic error (overall topographic error is 0.28). Notice that as the map size increases the topographic error increases (Fig. 3.8a). Therefore, it is important to choose a map size suitable for the dataset.



(a) Two diamonds dataset (b) RFSOM map

Fig. 3.7. Topology preservation vs. map size

How does the second approach behave as the map size varies? As the map size increases we expect the fuzzy membership values to get distributed across a larger number of neurons. Hence, causing the membership values to get smaller and smaller as the number of neurons gets bigger. Also, as the map gets larger, the distance among the neurons will increase, therefore, the ratio of the change in membership and distance (3.12) will get smaller causing the topographic error to decrease. On the other hand, as the map gets smaller, the membership values will get distributed on smaller number of neurons and the distances in a smaller map are smaller than those of a bigger map, hence, we expect a bigger topographic error. This behavior is shown in Fig. 3.8b.



(a) Image segmentation based error (b) Second approach

Fig. 3.8. Fuzzy topographic error vs. map size

3.6. Conclusion

In this chapter we presented a method for measuring the topology preservation in fuzzy self-organizing maps. The newly proposed topographic error relies on the membership distribution on the map and in some sense is an extension to the crisp topographic error. The assumption is that adjacent neurons should have similar memberships to a given object o_k. In addition, we presented the HL-matrix. A topology preservation HL-matrix for a given o_k contains two regions, the H region that encompasses the neurons with high membership to o_k and the L region which contains the low membership neurons to o_k. In the results different scenarios were presented to demonstrate how the topographic error behaves when varying the map dimensions. We observed that the topographic error in FSOM tends to be higher than the standard topographic error used in SOM.

One drawback of the proposed measure is its dependence on the map dimensions. For instance, as the map dimensions or size increases so does the topographic error. To overcome this problem, one is expected to specify a map dimension that is suitable to the input dataset. The dependency of the topographic error on the SOM parameters is not necessarily a bad thing. On the contrary, a high topographic error is an indication that the map is not optimal and the parameters require tuning. However, additional experiments are needed to study the influence of other parameters such as the neighborhood size and the fuzzifier, in addition to the map dimensions, on the proposed topographic error.

The second approach attempts to find solutions to some of the drawbacks found in the image segmentation approach. We longer need to depend on the image segmentation, which can be unreliable and can causes unpredictable results. Also, instead of measuring

the error based on only a subset of the neurons, which defeats the purpose of using the fuzzy membership values, the second approach utilizes all neurons when computing the error.

In the last two chapters we discussed the relational self-organizing maps, derived based on the RFCM formulation. From the experimental results we saw that RFSOM work. But have you wondered what would happen if the relational data matrix is not Euclidean? Would RFCM and its derivative algorithms fail? And would it always fail if the relational matrix is not Euclidean? We will try to answer some of these questions in the next chapter.

CHAPTER 4

IMPROVEMENTS TO THE RELATIONAL FUZZY c-MEANS CLUSTERING ALGORITHM

Relational fuzzy c-means (RFCM) is an algorithm for clustering objects represented in a pairwise dissimilarity values in a dissimilarity data matrix D. RFCM is dual to the fuzzy c-means (FCM) object data algorithm when D is a Euclidean matrix. When D is not Euclidean, RFCM can fail to execute if it encounters negative relational distances. To overcome this problem we can Euclideanize the relation D prior to clustering. There are different ways to Euclideanize D such as the β-spread transformation, where some constant is added to the off-diagonal elements of D. There are at least four alternatives to the β-spread method. In this article we compare five methods for Euclideanizing D to $\widetilde{D}$. The quality of $\widetilde{D}$ for our purpose is judged by the ability of RFCM to discover the apparent cluster structure of the objects underlying the data matrix D. Our main conclusion: the subdominant ultrametric transformation is a clear winner, producing much better partitions of $\widetilde{D}$ than the other four methods. This leads to a new algorithm we call the improved RFCM (iRFCM).

4.1. Introduction

Consider a set of objects $O = \{o_1, \cdots, o_n\}$, where the goal is to group them into c natural groups. Objects can be described by feature vectors $X = \{x_1, \ldots, x_n\} \in \mathbb{R}^p$ such that x_i is an attribute vector of dimension p representing object o_i. Alternatively, objects can be represented using a pairwise relationship. The relationships are stored in a relational matrix R, where $R = [r_{ij}]$ measures the relationship between o_i and o_j. If R is a

dissimilarity relation denoted by $D = [d_{ij}]$, then it must satisfy the following three conditions:

$$d_{ii} = 0 \qquad \text{for } i = 1, \cdots, n; \tag{4.1a}$$

$$d_{ij} \geq 0 \qquad \text{for } i = 1, \cdots, n \text{ and } j = 1, \cdots, n; \text{ and} \tag{4.1b}$$

$$d_{ij} = d_{ji} \qquad \text{for } i = 1, \cdots, n \text{ and } j = 1, \cdots, n, \tag{4.1c}$$

where condition (4.1a) is self-dissimilarity, (4.1b) is non-negativity and (4.1c) is symmetry. A well-known relational clustering algorithm that is suitable for clustering objects described by D is the relational fuzzy c-means (RFCM) proposed in [11] (Algorithm 4.1). RFCM, the relational dual of the FCM algorithm, takes an input dissimilarity matrix D and outputs a fuzzy partition matrix $U \in M_{fcn}$, where

$$M_{fcn} = \left\{ U \in \mathbb{R}^{c \times n} \middle| u_{ik} \in [0,1], \sum_{k=1}^{n} u_{ik} > 0, \sum_{i=1}^{c} u_{ik} = 1, \forall\, 1 \leq i \leq c \; and \; 1 \leq k \leq n \right\} \tag{4.2}$$

The duality relationship between RFCM and FCM is based on the squared Euclidean distance or 2-norm that defines the dissimilarity d_{ij} between two feature vectors x_i and x_j describing o_i and o_j and the dissimilarity between the cluster center v_i and o_j. In other words, RFCM assumes that

$$D = [d_{ij}] = \left[\left\| x_i - x_j \right\|_2^2 \right] \tag{4.3}$$

The relation $D = [d_{ij}]$ is Euclidean if there exists feature vectors $X = \{x_1, \ldots, x_n\} \in \mathbb{R}^p$ with an embedding dimension $p < n$, such that for all i, j $d_{ij} = \left\| x_i - x_j \right\|_2^2$. When D is Euclidean, it has a realization in some Euclidean space. In this

case, RFCM and FCM will produce the same partition of relational and feature vector representation of the data. If D is not Euclidean, RFCM will still find clusters in any D whose entries satisfy (4.1) as long as it can execute, but in this case it is possible for RFCM to experience an execution failure. This happens when the relational distances between prototypes and objects $d_{R,ik}$ in equation (4.4) become negative for some i and k (Algorithm 4.1, line 6). Another important observation about RFCM is that it expects *squared* dissimilarities D. If the dissimilarities are not squared, meaning that we have $\sqrt{D}$ instead of D such that $\sqrt{D} = D^{1/2} = \left[\sqrt{d_{ij}}\right]$, then the dissimilarities must be squared before clustering using RFCM so that D is the Hadamard product $D = \left(\sqrt{D}\right)^2$. Throughout this chpater D is assumed to contain *squared* dissimilarities.

Non-Euclidean Relational Fuzzy *c*-Means (NERFCM), repairs RFCM "on the fly" with a self-healing property that automatically adjusts the values of $d_{R,ik}$ and the dissimilarities in D in case of failure [12]. The self-healing property is based on the β-spread, which works by adding a positive constant β to the off-diagonal elements of D. In fact, there exists β_0 such that the β-spread transformed matrix D_β is Euclidean for all $\beta \geq \beta_0$. The parameter β controls the amount spreading and must be as small as possible to minimize unnecessary dilation that distorts the original D, which in turn may result in the loss of cluster information. The exact value of β_0 is the largest positive eigenvalue of the matrix PDP, where $P = I - \frac{1}{n}(\mathbf{1}\mathbf{1}^T)$. Eigenvalue computation is avoided by the self-healing module, which is invoked during execution only when needed. When activated, this module adjusts the current D by adding a minimal β-spread to its all off-diagonal elements.

An alternative to using NERFCM is to transform the matrix D by a mapping that converts it to Euclidean form (we call this operation "Euclideanizing D"), and then running RFCM on the Euclideanized matrix $\widetilde{D}$. This approach guarantees that RFCM will not fail since $\widetilde{D}$ is already Euclidean. There are at least five ways to Euclideanize D, including the β-spread transformation. In addition to the β-spread transformation, this chpater will study the other four Euclideanization approaches indicated under option 1 in Fig. 4.1. We defer the possible improvement of the self-healing module with these alternative strategies (option 2 in Fig. 4.1) to later study. So this chpater is about improving RFCM using option 1, hence iRFCM. A companion paper will consider improving NERFCM using option 2, hence iNERFCM. The RFCM algorithm is listed as pseudocode in Algorithm 4.1.

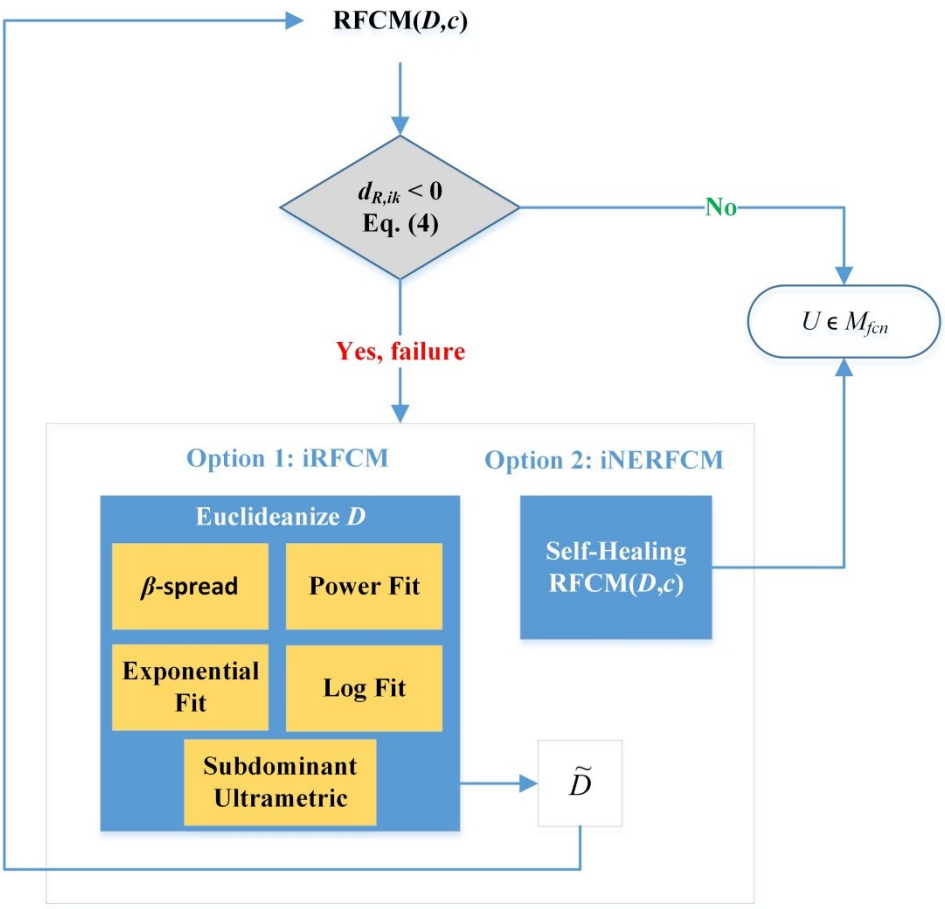


Fig. 4.1. Possible solutions RFCM can utilize when input D is non-Euclidean

Algorithm 4.1. Relational fuzzy c-means (RFCM) [11]

Input: D, c, fuzzifier $m > 1$ (default $m = 2$), t_{max} (default $t_{max} = 100$), ε (default $\varepsilon = 0.0001$)

Output: U, V_R

Initialize: step $= \varepsilon$, $t = 1$

Relational cluster centers $V_R^0 = \left(v_{R,1}^0, v_{R,2}^0, \cdots, v_{R,c}^0\right)$, $v_{R,i}^0 \in \mathbb{R}^n$

Note: we use c randomly chosen rows of D as initial centers.

while $t \leq t_{max}$ and step $\geq \varepsilon$

$$d_{R,ik} = \left(Dv_{R,i}^{t-1}\right)_k - \frac{1}{2}\left(v_{R,i}^{t-1}\right)^T Dv_{R,i}^{t-1}; \text{ for } 1 \leq i \leq c \text{ and } 1 \leq k \leq n \tag{4.4}$$

for $k = 1$ *to* n

if $d_{R,ik} \neq 0$ for all i

$$u_{ik} = 1 \Big/ \left(\frac{d_{R,ik}}{\sum_{j=1}^{c} d_{R,ik}}\right)^{\frac{1}{m-1}}; \forall i \tag{4.5}$$

else

Set $u_{ik} > 0$ for $d_{R,ik} = 0$, $u_{ik} \in [0, 1]$ and $\sum_{j=1}^{c} u_{jk} = 1$

end if

end for

$$v_{R,i}^t = (u_{i1}^m, \cdots, u_{in}^m) / \sum_{k=1}^{n} u_{ik}^m; \text{ for } 1 \leq i \leq c \tag{4.6}$$

$$\text{step} \leftarrow \max_{\substack{1 \leq i \leq c \\ 1 \leq j \leq n}} \left\{\left|V_R^{(t)} - V_R^{(t-1)}\right|\right\}$$

$t \leftarrow t + 1$

end while

4.2. Euclidean Distance Matrices (EDM) and the iRFCM Algorithm

Given a dissimilarity matrix D it is known that

D is a Euclidean distance matrix (EDM) $\Leftrightarrow W(D_{0.5})$ is positive semi-definite (*p.s.d*) (4.7)

where

$$W(D_{0.5}) = PD_{0.5}P \tag{4.8}$$

P is the centering matrix defined as

$$P = I - \frac{1}{n}(\mathbf{1}\mathbf{1}^T) \tag{4.9}$$

I is the identity matrix and $D_{0.5}$ is defined as

$$D_{0.5} = \begin{cases} -1/2\, D & \text{for squared dissimilarities} \\ -1/2\left(\sqrt{D}\right)^2 & Otherwise \end{cases} \tag{4.10}$$

In (4.10) and below, $\left(\sqrt{D}\right)^2$ is the Hadamard square of $\sqrt{D}$. The trick in using (4.8) is knowing if the dissimilarities are squared as in D or not squared as in $\sqrt{D}$, which determines which case of (4.10) to use. This is a question we cannot answer; rather the answer depends on one's knowledge of how the dissimilarities were computed.

The number of strictly positive eigenvalues of $W(D_{0.5})$ gives the maximum number of the embedding dimensions $p < n$ for the realization of D [57], [58]. If $W(D_{0.5})$ is not *p.s.d* then D can be Euclideanized to $\widetilde{D}$ by making $W(D_{0.5})$ *p.s.d* using the following general transformation

$$W\left(\widetilde{D}\right) = W(D_{0.5}) + \gamma W(\Delta_{0.5}), \tag{4.11}$$

where γ is some positive constant,

$$\Delta = \left\{\left[\delta_{ij}\right] \middle| \delta_{ij} = 0\ \forall i = j, \delta_{ij} \geq 0\ \forall i \neq j \text{ and } 1 \leq i, j \leq n\right\} \tag{4.12}$$

and $\Delta_{0.5}$ is computed the same way as $D_{0.5}$ using (4.10). Eq. (4.11) implies that the Euclideanized $\widetilde{D}$ is given by

$$\widetilde{D} = D + \gamma\Delta \tag{4.13}$$

Table 4.1 lists the five transformations that carry non-Euclidean D's into Euclidean $\widetilde{D}$'s that we will consider in this chapter.

Table 4.1. Transformations of $D \to \widetilde{D}$

Name	Formula	Reference	Eqn.
β-Spread	$\Delta^{\beta} = \mathbf{1}\mathbf{1}^T - I;\ \beta > 0$	[12], [57], [59]	(4.14a)
Subdominant Ultrametric (SU)	$\Delta^{SU} = \left[\delta_{ij}^{SU}\right]$ $\delta_{ij}^{SU} = max\{d_{v_k v_{k+1}} \in P_k \mid P_k = (i = v_0, v_1, \cdots, v_k, v_{k+1} = j) \in MST(D)\}$ $MST(D)$ is the minimum spanning tree of D (Fig. 4.2)	[60], [61]	(4.14b)
Power Fit (PF)	$\Delta^{PF} = D^{\alpha};\ 0 < \alpha \leq 1$	[59], [62]	(4.14c)
Exponential Fit (EF)	$\Delta^{EP} = \left(\mathbf{1}\mathbf{1}^T - e^{-\alpha\sqrt{D}}\right)^2;\ \alpha > 0$	[59], [62]	(4.14d)
Log Fit (LF)	$\Delta^{LF} = \left(\log_2\left(M + \left(\sqrt{D}\right)^{\alpha}\right)\right)^2;\ \ 0 < \alpha \leq 1$	[62]	(4.14e)

Given that D is not Euclidean and Δ has an Euclidean representation of dimension $n-1$, the goal is to find a positive constant γ to Euclideanize D. In the β-spread case (4.14a) we can find the exact γ, which is $\gamma = -2\lambda$, where λ is the smallest eigenvalue of $W(D_{0.5})$ at (4.8). Bénasséni [59] generalized this concept by incorporating additional choices of Δ. To understand Bénasséni's generalization we rewrite $W(\Delta_{0.5})$ in terms of its eigendecomposition $W(\Delta_{0.5}) = V\big(W(\Delta_{0.5})\big) \cdot \Lambda\big(W(\Delta_{0.5})\big) \cdot V\big(W(\Delta_{0.5})\big)^T$ where $\Lambda\big(W(\Delta_{0.5})\big)$ is the $(n-1) \times (n-1)$ diagonal matrix of the non-zero eigenvalues of $W(\Delta_{0.5})$ and $V\big(W(\Delta_{0.5})\big)$ is the corresponding $n \times (n-1)$ matrix of the normalized eigenvectors. Since $\Lambda\big(W(\Delta_{0.5})\big)$ is positive definite, the minimum constant γ that makes (4.11) *p.s.d* is $\gamma = -\lambda(D_{0.5}, \Delta_{0.5})$ where $\lambda(D_{0.5}, \Delta_{0.5})$ is given by

$$\lambda(D_{0.5}, \Delta_{0.5}) = \lambda_{min}\Big(\Lambda\big(W(\Delta_{0.5})\big)^{-1/2} \cdot V\big(W(\Delta_{0.5})\big)^T \cdot W(D_{0.5}) \cdot V\big(W(\Delta_{0.5})\big) \cdot \Lambda\big(W(\Delta_{0.5})\big)^{-1/2}\Big) \tag{4.15}$$

A more detailed proof can be found in [59].

In (4.14a) the same constant is added to all the off-diagonal dissimilarity values in D. In some cases adding the same constant to all of the off-diagonal elements can cause a lot of distortion and a large correspondingly discrepancy between D and $\widetilde{D}$, causing $\widetilde{D}$ to lose the original structure of the data. This distortion can propagate into the RFCM clustering algorithm, causing a loss in the original cluster information. This is a very serious concern when $d_{ij} \in [0,1]$ and the additive constant γ is large (an example of this will be shown in the results section). To alleviate this problem, we can use one of the other choices of Δ listed in Table 4.1, such as the subdominant ultrametric (SU).

The SU of D, denoted as Δ^{SU}, is derived from the minimum spanning tree of D, MST(D). Recall that D represents an undirected graph whose vertices are the objects described by D. A length d_{ij} is assigned to each edge (i,j). To determine Δ^{SU}, construct MST(D) T, such as the one shown in Fig. 4.2. T may not be uniquely determined if some edges have identical weights, but Δ^{SU} is unique and does not depend on any particular choice of T [60]. We use Prim's algorithm to determine the MST [63].

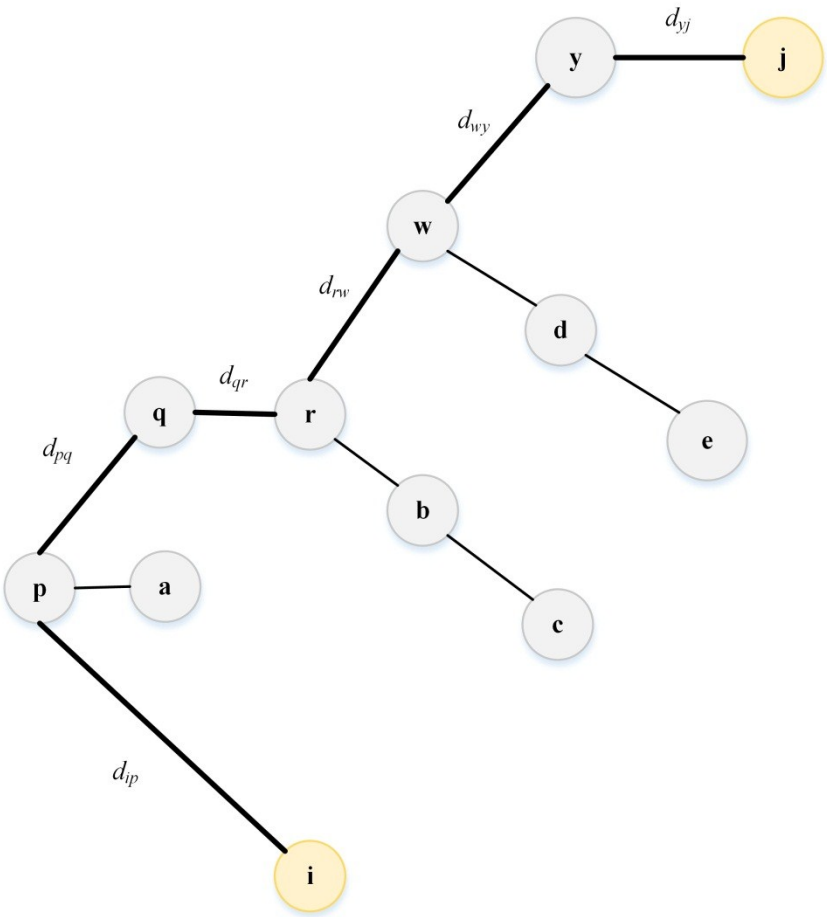


Fig. 4.2. Example of a minimum spanning tree

Eq. (4.14b) states that the SU distance between i and j, δ_{ij}^{SU}, is the maximum weight along the path $P_k = (i = v_0, v_1, \cdots, v_k, v_{k+1} = j)$ connecting objects i and j. In Fig. 4.2 there are six edges between i and j (bolded color). The first edge (i, p) has weight d_{ip}, second edge (i, q) has weight d_{pq}, etc. The SU distance between i and j for the particular MST in Fig. 4.2 is then given by the edge with the highest weight

$$\delta_{ij}^{SU} = max\{d_{ip}, d_{pq}\ d_{qr}, d_{rw}, d_{wy}, d_{yj}\} = d_{ip}$$

Unlike the other transformations we will discuss later, the SU distance is a function of the original dissimilarities and its objective is to maximize the distance between any two

objects. Holman in [61] proved that Δ^{SU} on n objects is Euclidean with $n-1$ dimensions. Once Δ^{SU} is computed we can find γ, where $\gamma = -\lambda(D_{0.5}, \Delta^{SU}_{0.5})$.

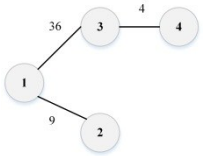
The following squared dissimilarity matrix

$$D = \begin{bmatrix} 0 & 9 & 36 & 81 \\ 9 & 0 & 49 & 36 \\ 36 & 49 & 0 & 4 \\ 81 & 36 & 4 & 0 \end{bmatrix}$$

is not *p.s.d* since the eigenvalues of $W(D_{0.5}) = PD_{0.5}P$ at (4.8) are $\{-11.31, 0, 15.77, 49.28\}$.

Using the SU we can Euclideanize D, which first involves computing the MST of D that will be used to compute Δ^{SU}.

$$D = \begin{bmatrix} 0 & 9 & 36 & 81 \\ 9 & 0 & 49 & 36 \\ 36 & 49 & 0 & 4 \\ 81 & 36 & 4 & 0 \end{bmatrix} \xrightarrow{MST(D)} \quad \xrightarrow{\Delta^{SU}} \begin{bmatrix} 0 & 9 & 36 & 36 \\ 9 & 0 & 36 & 36 \\ 36 & 36 & 0 & 4 \\ 36 & 36 & 4 & 0 \end{bmatrix}$$



To compute the smallest eigenvalue $\lambda(D_{0.5}, \Delta^{SU}_{0.5})$ in (4.15) we first compute $W(\Delta^{SU}_{0.5}) = P\Delta^{SU}_{0.5}P$, where $\Delta^{SU}_{0.5} = -1/2\,\Delta^{SU}$.

$$W(\Delta^{SU}_{0.5}) = \begin{bmatrix} 3/4 & -1/4 & -1/4 & -1/4 \\ -1/4 & 3/4 & -1/4 & -1/4 \\ -1/4 & -1/4 & 3/4 & -1/4 \\ -1/4 & -1/4 & -1/4 & 3/4 \end{bmatrix} \times \begin{bmatrix} 0 & -4.5 & -18 & -18 \\ -4.5 & 0 & -18 & -18 \\ -18 & -18 & 0 & -2 \\ -18 & -18 & -2 & 0 \end{bmatrix}$$

$$\times \begin{bmatrix} 3/4 & -1/4 & -1/4 & -1/4 \\ -1/4 & 3/4 & -1/4 & -1/4 \\ -1/4 & -1/4 & 3/4 & -1/4 \\ -1/4 & -1/4 & -1/4 & 3/4 \end{bmatrix}$$

$W(D_{0.5})$ is computed the same way. To save space we do not show the eigenvalues and eigenvectors of $W(\Delta^{SU}_{0.5})$, but once computed, we will have $\Lambda\big(W(\Delta_{0.5})\big)$, 3×3 diagonal matrix of non-zero eigenvalues and $V\big(W(\Delta_{0.5})\big)$, 3×4 normalized eigenvectors.

Inserting $V\left(W(\Delta_{0.5})\right)$, $\Lambda\left(W(\Delta_{0.5})\right)$ and $W(D_{0.5})$ in (4.15) gives $\lambda(D_{0.5}, \Delta_{0.5}^{SU}) = -3.84$. Then with D, Δ^{SU} and γ, where $\gamma = -\lambda(D_{0.5}, \Delta_{0.5}^{SU}) = 3.84$, in (4.13) results in Euclidean form of D realized by the SU transformation,

$$\widetilde{D} = \begin{bmatrix} 0 & 43.55 & 174.19 & 219.19 \\ 43.55 & 0 & 187.19 & 174.19 \\ 174.19 & 187.19 & 0 & 19.35 \\ 219.19 & 174.19 & 19.35 & 0 \end{bmatrix}.$$

We can verify that $\widetilde{D}$ is Euclidean by computing the eigenvalues of $W\left(\widetilde{D}_{0.5}\right)$, $\{0, 0, 31, 173.41\}$, which indicates that $W\left(\widetilde{D}_{0.5}\right)$ is *p.s.d.*

The third choice of Δ (4.14c) belongs to the family of power functions parameterized by α. Using a transformation based on the power fit (PF) involves a smaller distortion to the original dissimilarities D compared to the β-spread transformation. According to Bénasséni [59] there exists some real constant α_0 such that D^{α} is Euclidean for $\alpha \leq \alpha_0$. Notice that for any $d_{ij}^{\alpha} > 0$, as $\alpha \to 0$, then d_{ij}^{α} tends monotonically to 1. In other words, if $d_{ij} > 0$ for all i, j and $i \neq j$ the $\lim_{\alpha \to 0} D^{\alpha} = \Delta^{\beta}$, where Δ^{β} is given in (4.14a).

The exponential fit (EF) Δ^{EF} (4.14d) was first mentioned in Dattoro [62] to show that some nonlinear compositions of EDMs are also EDMs. Bénasséni [59] used this transformation to Euclideanize D. Similar to Δ^{PF}, Δ^{EF} is a function of α and the limit property of (4.14d) states that as the $\lim_{\alpha \to \infty}\left(\mathbf{1}\mathbf{1}^T - e^{-\alpha\sqrt{D}}\right)^2 = \Delta^{\beta}$ if $d_{ij} > 0$ for all i, j and $i \neq j$. Bénasséni [59] shows that there exists α_0 such that for $\alpha \geq \alpha_0$, Δ^{EF} is Euclidean.

Another nonlinear composition of EDM proposed in Dattoro [62] that can also yield EDM and that can be used in the Euclideanization of D is the log fit (LF) given in (4.14e).

The last three transformations in Table 4.1 are parametric and hence require finding a value α that makes D Euclidean. In this chapter a greedy search for α was performed for these three transformations. There may be a more efficient approach for finding α, but this is beyond the scope of this dissertation and will be addressed in future work. What follows is the iRFCM algorithm that incorporates the transformations mentioned above.

Algorithm 4.2. Improved relational fuzzy c-means (iRFCM)

1 **Input:** D, c, Δ, $m > 1$ (default $m = 2$), t_{max} (default $t_{max} = 100$), ε (default $\varepsilon = 0.0001$)
2 **Output:** U, V_R
3 **Initialize:** $D_{0.5} = -1/2\, D$
4 $W(D_{0.5}) = PD_{0.5}P$

5 **if** $W(D_{0.5})$ is not *p.s.d*
6 $\Delta_{0.5} = -1/2\, \Delta$, where $\Delta \in \{\Delta^{\beta}, \Delta^{SU}, \Delta^{PF}, \Delta^{EF}, \Delta^{LF}\}$
7 $\gamma = -\lambda(D_{0.5}, \Delta_{0.5})$
8 $\widetilde{D} = D + \gamma\Delta$
9 $D \leftarrow \widetilde{D}$
10 **endif**

11 U, V_R = RFCM(D, c, m, t_{max}, ε)

We have listed Algorithms 4.1 and 4.2 for the fuzzy case ($m > 1$). There are also hard and possibilistic versions of RFCM, the relational hard c-means (RHCM) [11] and the relational possibilistic c-means (RPCM) [35]. Algorithm 4.2 will also generalize them by replacing line 11 in Algorithm 4.2 with your choice of RHCM or RPCM assuming the appropriate changes are made to Algorithm 4.1.

4.3. Experimental Results

4.3.1. **Example 1. Mutation Data**

The Mutation dataset [64] contains 20 objects each representing a cytochrome (the names of the animal organisms are listed in Fig. 4.4). Fitch and Margoliash [65] recount the history of the work in [9] in a delightful one page essay that is historically charming. The distance between two cytochromes is defined as the minimum number of nucleotides that must be altered in order for the gene of one of the cytochromes to code for the other [64]. Squaring the distances in $\sqrt{D_{mut}}$ given in the lower half of Table 4.3 in [64] leads to the conclusion that D_{mut} is not an EDM. However, when D_{mut} is submitted to RFCM, we find experimentally that it does not fail, even though the theory predicts possible execution failure. So, this dataset is ideal for studying how much distortion each of the five transformations in Table 4.1 will introduce into the clusters detected by RFCM on D_{mut}. Fitch *et. al.* [64] visualize the data using the phylogenetic tree shown in Fig. 4.3.

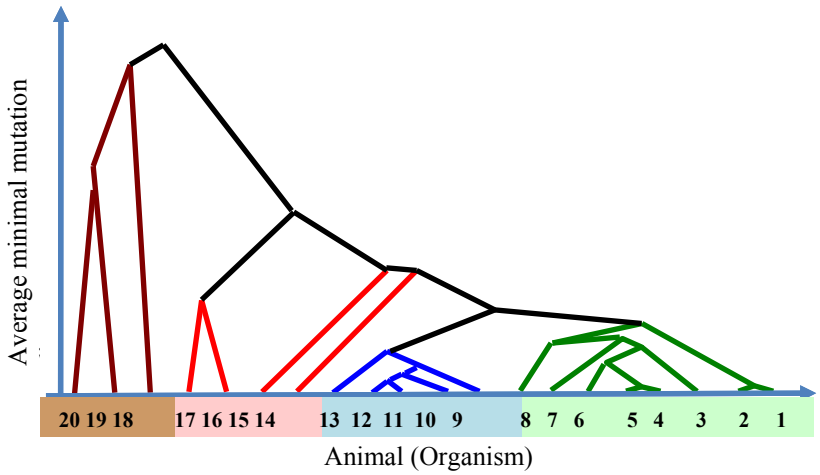


Fig. 4.3. Mutation phylogeny tree (Fig. 2 in [64])

We can also visualize the structure of the data using the Improved Visual Assessment of Tendency (iVAT) algorithm [66] as in Fig. 4.4. The four darkest diagonal sub-blocks in the image of Fig. 4.4 correspond to the three singletons Mold, Yeast and Fungus and a larger block containing the other 17 objects. Thus, the four clusters most strongly suggested by Fig. 4.4 are $\{1-17\}$, $\{18\}$, $\{19\}$, $\{20\}$. This agrees exactly with the clusters that would be obtained by cutting the tree in Fig. 4.3 at $c = 4$. We will (arbitrarily) call this partition the ground truth U_{GT} 4-partition of D_{mut}.

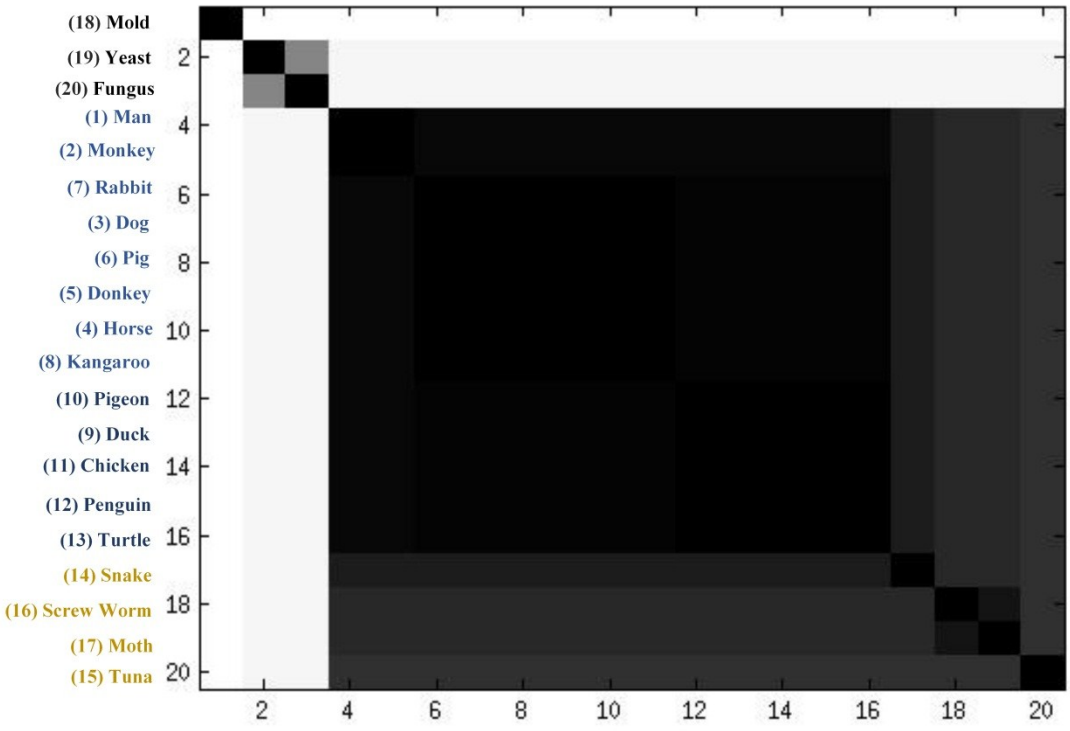


Fig. 4.4. iVAT image of the *squared* Mutation data D_{mut}

Since RFCM and iRFCM produce fuzzy partitions, we need a way to convert them to crisp partitions in order to compare them with our U_{GT} partition at $c = 4$. Here we use the standard *hardening* scheme, i.e., the maximum membership in each column of fuzzy partition U is replaced by 1, and the remaining $c - 1$ values become 0's. RFCM was applied to D_{mut} for the fuzzifier values $m = 1.05$ and $m = 2$. At $m = 1.05$, we are (almost) seeing the results of running RHCM on this data.

Table 4.2 lists the U_{GT}, the ground truth partition of $\sqrt{D_{mut}}$, and also the hardened 4-partitions of D_{mut} at these two values ($m = 1.05$ and $m = 2$). The five transformation methods are identified by the inducing matrix $D_{mut,0.5}$, which is used in equation (4.10) to realize the EDM built with D. There are 3 mismatched labels between U_{GT} and $U_{D_{mut}}$

at $m = 1.05$, and 9 label differences at $m = 2$. As expected, this confirms that at $m = 2$, memberships of the 20 organisms are much more widely distributed across the 4 clusters than at $m = 1.05$.

Table 4.2. Hardened 4-partitions found by RFCM(D_{mut}) and iRFCM($\widetilde{D}_{mut}$)

*Cluster 1 +Cluster 2 ●Cluster 3 ■Cluster 4

			$m = 1.05$						$m = 2$					
o_i	Organi	U_{GT}	$U_{D_{mut}}$	$U_{\Delta^{SU}}$	$U_{\Delta^{\beta}}$	$U_{\Delta^{PF}}$	$U_{\Delta^{EF}}$	$U_{\Delta^{LF}}$	$U_{D_{mut}}$	$U_{\Delta^{SU}}$	$U_{\Delta^{\beta}}$	$U_{\Delta^{PF}}$	$U_{\Delta^{EF}}$	$U_{\Delta^{LF}}$
1	Man	*	*	*	*	*	*	*	*	*	*	+	+	*
2	Monke	*	*	*	*	*	*	*	*	*	*	+	+	*
3	Dog	*	*	*	*	*	*	*	*	*	*	*	*	*
4	Horse	*	*	*	*	*	*	*	*	*	*	*	*	*
5	Donke	*	*	*	*	*	*	*	*	*	*	*	*	*
6	Pig	*	*	*	*	*	*	*	*	*	*	*	*	*
7	Rabbit	*	*	*	*	*	*	*	*	*	*	*	*	*
8	Kangar	*	*	*	*	*	*	*	*	*	*	*	*	*
9	Pigeon	*	*	*	*	*	*	*	●	*	●	●	●	●
10	Duck	*	*	*	*	*	*	*	●	*	●	●	●	●
11	Chicke	*	*	*	*	*	*	*	●	*	●	●	●	●
12	Pengui	*	*	*	*	*	*	*	●	*	●	●	●	●
13	Turtle	*	*	*	*	*	*	*	●	*	●	●	●	●
14	Tuna	*	*	*	*	*	*	*	●	●	●	+	+	●
15	Snake	*	*	*	*	*	*	*	●	●	●	+	+	*
16	Fly	*	●	*	*	●	●	●	*	●	+	+	+	+
17	Moth	*	●	*	*	●	●	●	●	●	+	+	+	+
18	Mold	+	+	+	+	+	+	+	+	+	■	■	■	■
19	Yeast	●	■	●	●	■	■	■	■	■	■	■	■	■
20	Fungus	■	■	■	■	■	■	■	■	■	■	■	■	■

Both RFCM runs group $\{1 - 8\}$ together, and a quick look at the tree in Fig. 4.3 confirms this as a primary structure in the data. Visual acuity makes it hard to see this in Fig. 4.4, but the pixels corresponding to these 8 organisms are identified and grouped together along the vertical axis in Fig. 4.4 too. So, this inference is consistent with both visual representations of D_{mut}, and with our intuition about what clusters "should be" in the data, based on our everyday notions about classes of animals.

How much are RFCM partitions of D_{mut} distorted when iRFCM is applied to $\widetilde{D}_{mut}$? We can get a somewhat surprising picture of what this type of feature extraction does by comparing the 5 Euclideanized data results to the results for D_{mut}. For example, compare the columns for D_{mut} and $\widetilde{D}_{mut}$ using (4.13) with Δ^{SU} and Δ^{β}. At $m = 1.05$, there are 3 disagreements between the hardened labels of $U_{D_{mut}}$ and the 4-partitions $U_{\Delta^{SU}}$ and $U_{\Delta^{\beta}}$ -BUT- $U_{\Delta^{SU}}$ and $U_{\Delta^{\beta}}$ both match U_{GT} perfectly! So, this is an instance where feature extraction does its job for clustering, by improving the results obtained by the same algorithm on the transformed data. The other three partitions obtained at $m = 1.05$ are identical to $U_{D_{mut}}$. An interesting conundrum: the transforms that preserve the cluster structure in D_{mut} don't yield the best matches to the ground truth.

At $m = 2$, fuzziness increases, memberships are more distributed, and there are five different hardened partitions available. The best match partition to U_{GT} is $U_{\Delta^{SU}}$ (5 disagreements), the minimum distortion of $U_{D_{mut}}$ by Euclideanization is realized by $U_{\Delta^{\beta}}$ (3 disagreements). Another thing we can notice in Table 4.2 is that the PF and EF methods offer identical interpretations of this data for both values of m.

Now that we have presented the various clustering results, let's take a closer look at how every choice Δ affected the original dissimilarities. Fig. 4.5 shows the mean and standard deviation of the transformed dissimilarities $\widetilde{D}_{mut}$ computed for every Δ. The β-spread, PF, EF and LF all have mean μ and standard deviation σ that are very close to the original dissimilarities. The SU on the other hand, resulted in the highest mean $\mu = 7573$ and standard deviation $\sigma = 9511$. This is expected from the SU as it amplifies the dissimilarities using the minimum spanning tree. However, despite the large spreading caused by the SU, it has shown to provide better results as we will see in later

experiments. Please be careful to distinguish the two effects of Euclideanizing D_{mut} we have studied in this example: Fig. 4.5 is about the distortion of D_{mut}, whereas Table 4.2 is concerned with the distortion of RFCM partitions $U_{D_{mut}}$. Evidently the β-spread transform causes the least distortion from the input data (and this seems confirmed by the resultant partition information in Table 4.2). But the SU provides the best U_{GT} matches. Later examples will corroborate our early belief that Δ^{SU} yields the most reliable Euclideanization of D from the clustering point of view.

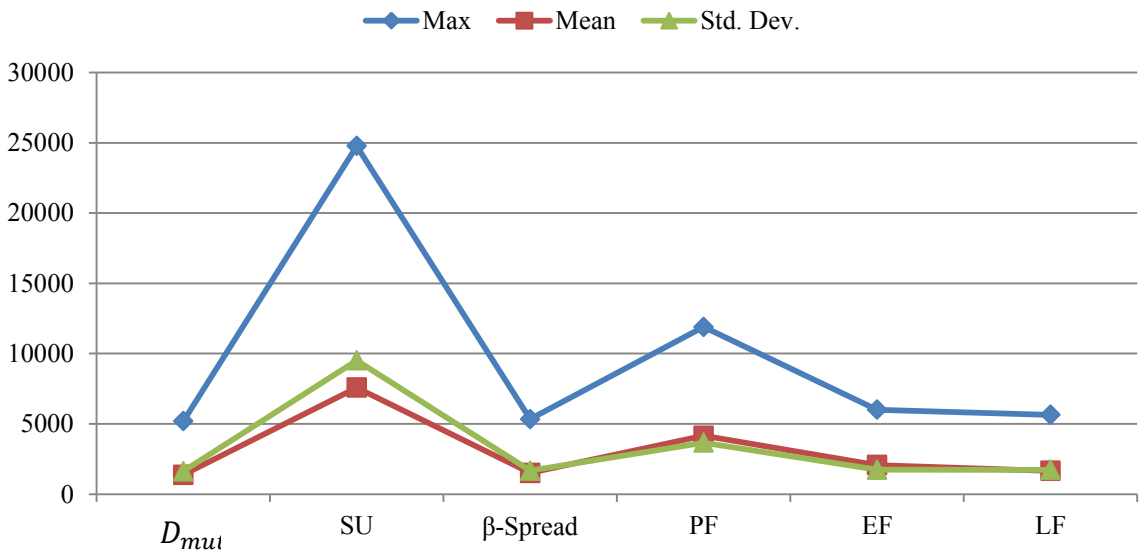


Fig. 4.5. The max, mean and standard deviation of the elements in $\tilde{D}_{mut}$ for the 5 input matrices compared to D_{mut}.

4.3.2. **Example 2. GDP194 Data**

The GDP194 dataset contains 194 sequences of human gene products and was obtained from ENSEMBL 2009 [51]. The relational data of the gene products was computed using a fuzzy measure similarity, which is based on Sugeno's λ measure [52]. The GDP194 characteristics are shown in Table 4.3, where we see that the data contains three classes and hence we will use iRFCM with $c = 3$.

Table 4.3. Characteristics of the GDP194 dataset

ENSEMBL Family ID	F_i = Protein Family	Gene Symbols	No. of Genes	No. of Sequences
339	Myotubularin	MTMR1÷4, MTMR1÷4	7	21
73	Receptor Precursor	FGFR1÷4, RET, TEK, TIE1	7	87
42	Collagen Alpha Chain	COL1A2, COL21A2, COL24A2, COL27A2, COL2A1, COL3A1, COL4A1, COL4A2, COL4A3, COL4A6, COL5A3, COL9A1, COL9A2	13	86

GPD194 as used here is represented by a matrix $\sqrt{D_{194}}$ of (unsquared) dissimilarity data, which was built from the similarity data such that $\sqrt{d_{ij}} = 1 - s_{ij}$, where s_{ij} is the fuzzy similarity between gene products i and j. We then squared the values, obtaining D_{194}, and computed the eigenvalues of the matrix $W\left(D_{194,0.5}\right)$ defined by (4.8). This matrix is not *p.s.d.* (there are 12 negative eigenvalues), so it is possible that RFCM will fail to execute. But, unlike D_{mut} in Example 1, which was also not *p.s.d.* but for which RFCM ran anyway, here RFCM experiences execution failure after encountering 27 negative relational distances appearing during the first iteration. At this point, we have the two options shown in Fig. 4.1: Euclideanize D using the 5 transformations in Table 4.1, or alteration of RFCM with self-healing. Since this chapter is about option 1, we clustered the data using iRFCM with $m = 2$, $c = 3$ and the five choices of Δ (Table 4.1). Let U denote the fuzzy 3-partition produced by iRFCM on $\widetilde{D}_{194}$ made with the SU, Fig. 4.6b is a visual representation of U made with an induced dissimilarity image $D(U)$. $D(U)$ is given by

$$D(U) = 1 - \frac{U^T U}{\max\limits_{\forall i,j}\{U^T U\}} \tag{4.16}$$

where $(U^T U)_{ij} = \sum_{k=1}^{c} u_{ki} u_{kj}$ is the coupling of objects i and j overall c clusters. The theory underlying (4.16) and several other examples of the use of this induced dissimilarity measure appear in [67]. The SU view in Fig. 4.6b is clearly superior to the partitions produced by the other four methods. Part of the SU performance is attributed to its attempt to maximize the distance between any two objects by taking the longest edge along the path connecting them, thus causing a larger separation among the objects.

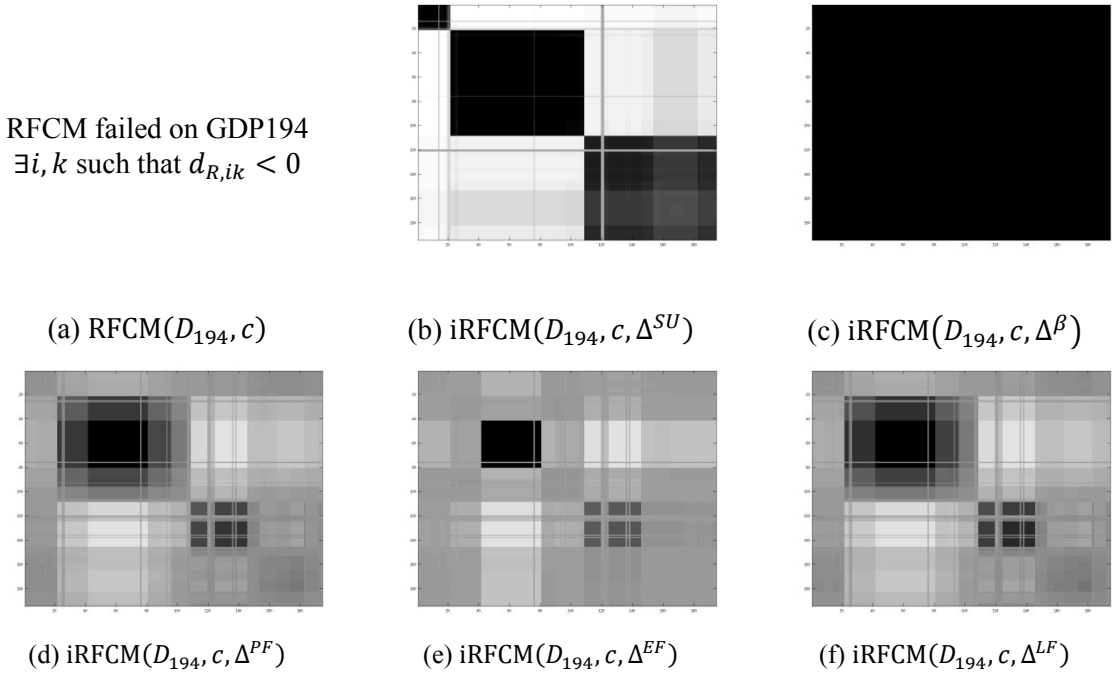
It was reported in [34], [52] that the third family, the collagen alpha chain, is divided into three subgroups: fibril forming collagens, type IV collagens, and fibril associated collagens with interrupted triple helices. Those groups are visible in Fig. 4.6b, in the lower right corner.

The β-spread transformation result - the all black image in view 4.6c - is very interesting. The β-spread is widely cited approach in the literature for Euclideanizing D, but for clustering it is not clear that this is the best choice. The induced partition dissimilarity in Fig. 4.6c shows no clusters. The dissimilarities in the GDP194 are bounded such that $0 \leq d_{ij} \leq 1$, so adding a large constant to all of the off-diagonal dissimilarities can distort the structure of the data. Subsequently, this causes a large difference between D_{194} and $\widetilde{D}_{194}$. In this case adding the constant, $\beta = 17.28$, to the dissimilarities makes it harder for iRFCM to distinguish the objects and hence iRFCM assigns a membership $u_{ik} \approx 1/c$, where $c = 3$ to every object. This is one interpretation for the black image in Fig. 4.6c - the image we call the "black image of death."

The PF image in Fig. 4.6d of the dissimilarity induced by the iRFCM partition of D_{194} using (4.16) suggests that the data contain two compact clusters. The first cluster which is the first block along the diagonal corresponds to the 87 sequences in the receptor

precursor family. The second cluster corresponds to the first sub-cluster identified in the SU case, which is the fibril forming collagens family.

The EF in Fig. 4.6e suggests a different interpretation of this data. The most notable cluster is the black block in Fig. 4.6e corresponding to a subgroup of the receptor precursor family, which are the sequences having the gene FGFR2. The lighter color block contains the sequences in fibril forming collagens subgroup. The LF in Fig. 4.6f has some resemblance to Fig. 4.6d.



(a) RFCM(D_{194}, c) (b) iRFCM$(D_{194}, c, \Delta^{SU})$ (c) iRFCM$\left(D_{194}, c, \Delta^{\beta}\right)$

(d) iRFCM$(D_{194}, c, \Delta^{PF})$ (e) iRFCM$(D_{194}, c, \Delta^{EF})$ (f) iRFCM$(D_{194}, c, \Delta^{LF})$

Fig. 4.6. The induced dissimilarity images $D(U)$ produced from clusterings of the GDP194 dataset using iRFCM with different choices of Δ and $c = 3$

There are two take-away messages from this example: (i) the SU is clearly the *best* way to convert D_{194} to $\widetilde{D}_{194}$ using (4.13) to preclude execution failure of RFCM; and (ii) the β-spread is clearly the *worst* of the five methods considered here.

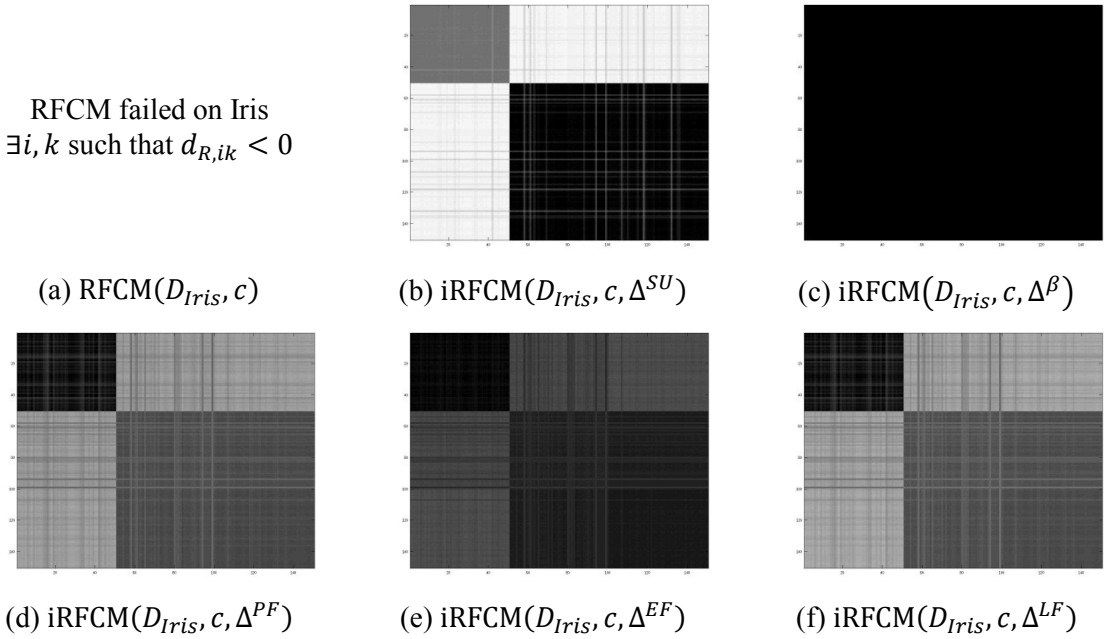
4.3.3. **Example 3. Iris Data**

Anderson's Iris data X_{Iris}, collected by Anderson in 1935 comprises $n = 150$ feature vectors in $p = 4$ dimensions [68], [69]. Each vector in Iris has one of three (crisp) physical labels corresponding to the Iris subspecies it belongs to: Setosa, Versicolor, or Virginica. This famous data set has probably appeared in more clustering papers than any other dataset on the planet. Perhaps the most interesting property of X_{Iris} is that this data has 3 classes (physically labeled) representing the ground truth, but only 2 computer point of view clusters. Let's find out what iRFCM thinks.

We begin with X_{Iris}, and construct from it the matrix D_{Iris}. The ij-th entry of this matrix is the square of the sup norm between x_i and x_j, i.e., $d_{ij} = \|x_i - x_j\|_{sup}^2$. The matrix $W(D_{Iris,0.5})$ is indefinite (it has 73 eigenvalues < 0 and 77 eigenvalues ≥ 0), so D_{Iris} is not Euclidean. Similar to Example 2, we find that RFCM ($m = 2$, $c = 3$) fails to execute directly on D_{Iris}, so we compute $\widetilde{D}_{Iris}$ via (4.13) with the five transformations at (4.14) which make it Euclidean. The largest negative eigenvalue of $W(D_{Iris,0.5})$ is 16.977, so adding this value to all of the off-diagonal elements of D_{Iris}, as the β-spread does at (4.14a), makes it Euclidean.

Fig. 4.7 displays visual representations of the five partitions obtained by iRFCM using the five Euclideanized versions of D_{Iris} and the induced dissimilarity matrix $D(U)$ at (4.16). First, note that three of the five results (views b, d, and f) strongly support the conclusion that iRFCM thinks there are only $c = 2$ clusters in Iris, even though we ran the algorithm with $c = 3$. If you look carefully at Fig. 4.7e, you will see that the EF also supports this, but with much less assurance. This is consistent with our view of Iris. The β-spread partition again produces the black image of death in Fig. 4.7c. Another

observation is that except for the β-spread, the first 50 objects received high membership values in the first cluster and low, but almost equal memberships in the second and third clusters, while the last 100 objects received high, but almost equal membership values in the second and third clusters. Overall, it certainly appears the SU again offers the best way to extend RFCM when an execution failure occurs due to a non-Euclidean input.



(a) RFCM(D_{Iris}, c) (b) iRFCM$(D_{Iris}, c, \Delta^{SU})$ (c) iRFCM$\left(D_{Iris}, c, \Delta^{\beta}\right)$

(d) iRFCM$(D_{Iris}, c, \Delta^{PF})$ (e) iRFCM$(D_{Iris}, c, \Delta^{EF})$ (f) iRFCM$(D_{Iris}, c, \Delta^{LF})$

Fig. 4.7. The induced dissimilarity images $D(U)$ produced from clusterings of the Iris dataset using iRFCM with different choices of Δ and $c = 3$

4.4. Conclusion and Discussion

RFCM is a popular algorithm for (fuzzily) clustering objects described by a dissimilarity data matrix D. But since RFCM is the relational dual of FCM, execution of the algorithm is guaranteed only when the dissimilarities in D have a Euclidean representation with an embedding dimension $p < n$. If D is not Euclidean then the duality relation will be violated and most importantly the distances $d_{R,ik}$ can become negative. There are two options to circumvent this problem. Option 2 in Fig. 4.1 advocates the use of a self-healing RFCM such as NERFCM, which adjusts the

dissimilarities and the distances "on the fly," and only when needed, if a negative distance is encountered. The second choice (Option 1 in Fig. 4.1) is to Euclideanize D prior to running RFCM. This second strategy is the one pursued here, leading to a new algorithm, iRFCM.

Five different choices of Δ were used to Euclideanize D prior to clustering. Computationally, the easiest transformation to use is the β-spread. In the β-spread approach, the same constant is added to all off-diagonal dissimilarities. If the dissimilarities are small and the constant is large, as in the GDP194 data, the original structure of the data gets distorted, and with that one can lose the cluster information. Our examples suggest that the β-spread mapping delivers good news, and bad news. The good news: it minimizes the distortion between D and $\widetilde{D}$; the bad news is that it seems to maximize the distortion between the partitions U_D and $U_{\widetilde{D}}$. On the other hand, the SU transformation seems have the best performance when visualized using the induced partition dissimilarity. The three parametric based transformations, viz., the PF, EF and LF, have varying performance, but the main limitation of the parametric functions is finding an optimal α_0 that can Euclideanize D and produce reasonable partitions of the data. In this chapter we took a simple approach and directly searched for α_0. Determining an optimal value of α for iRFCM clustering using these three transformations is a challenging and important problem that we defer to a future investigation.

Every Δ produces a different dataset that somewhat resembles the original dissimilarities. We have witnessed in the results that different choices of Δ have resulted in different clusterings. The main difference that separates the five methods into three types is the effect that they have on the original object distances. The SU distance

between two objects is the maximum dissimilarity between those objects in D. This is the only transformation among the five that uses original data values (as opposed to transformed ones); thus, the positions of the objects are not changed to achieve Euclideanization. The PF, EF and LF mappings are all parametric, and all replace the original dissimilarities with new ones. In terms of object locations, this amounts to rearranging the underlying realization of the objects to make it Euclidean. Thus, these three transformations distribute the spread. Finally, the β-spread is the most disruptive of the five. Adding the largest negative eigenvalue of $W(D_{0.5})$ to all of the off-diagonal entries of D amounts to spreading (literally) the objects by a fixed, maximal amount, so the original dissimilarities in this conversion are all gone.

Some limitations emerge from Euclideanizing D prior to clustering. First, it is not very scalable. It definitely works for small datasets, but as n increases so does the time needed to Euclideanize D. It will require a large amount of time to compute the SU distance, which involves the construction of the minimum spanning tree. Second, as n increases, the time to compute the smallest eigenvalue of $W(D_{0.5})$ will also increase. Recall that $W(D_{0.5})$ always has a zero eigenvalue, and many of the non-zero eigenvalues are close to zero. Actually getting the eigenvalues becomes a numerically intractable problem due to instability and scalability as n increases. Large-scale parallel eigensolvers based on Message Passing Interface (MPI) exist for large matrices, but they were tested on matrices with a maximum order of less than 1 million [70]. In the age of very large data we need tools that scale to matrices at the order of billions, such as the one based on MapReduce and Hadoop proposed in [70]. The situation becomes even more computationally expensive when we search for α that makes D Euclidean because for

every α, we have to evaluate whether Δ is Euclidean or not. Third, the original dissimilarities get distorted and can lose their original structure when a constant is added, which was made abundantly clear in the β-spread case. A possible and a more scalable solution to this is to use a different approach such as self-healing RFCM (NERFCM), where the dissimilarities are transformed "on the fly", only if needed, and only a small constant is added to keep the discrepancy between D and $\widetilde{D}$ to the minimum, a topic that will be discussed further in future work.

In the next two chapters we will switch from clustering to classification, where we used random forest and ontologies to predict the risk of future diseases using a Medicare dataset.

CHAPTER 5

PREDICTING DISEASE RISKS FROM HIGHLY IMBALANCED DATA USING RANDOM FOREST

We present a method utilizing Healthcare Cost and Utilization Project (HCUP) dataset for predicting disease risk of individuals based on their medical diagnosis history. The presented methodology may be incorporated in a variety of applications such as risk management, tailored health communication and decision support systems in healthcare. We employed the National Inpatient Sample (NIS) data, which is publicly available through Healthcare Cost and Utilization Project (HCUP), to train random forest classifiers for disease prediction. Since the HCUP data is highly imbalanced, we employed an ensemble learning approach based on repeated random sub-sampling. This technique divides the training data into multiple sub-samples, while ensuring that each sub-sample is fully balanced. We compared the performance of support vector machine (SVM), bagging, boosting and RF to predict the risk of eight chronic diseases. We predicted eight disease categories. Overall, the RF ensemble learning method outperformed SVM, bagging and boosting in terms of the area under the receiver operating characteristic (ROC) curve (AUC). In addition, RF has the advantage of computing the importance of each variable in the classification process. In combining repeated random sub-sampling with RF, we were able to overcome the class imbalance problem and achieve promising results. Using the national HCUP data set, we predicted eight disease categories with an average AUC of 88.79%.

1.1. Background

The reporting requirements of various US governmental agencies such as Center for Disease Control (CDC), Agency for Health Care Quality (AHRQ) and US Department of Health and Human Services Center for Medicare Services (CMS) have created huge public datasets that, we believe, are not utilized to their full potential. For example, CDC (www.cdc.gov) makes available National Health and Nutrition Examination Survey (NHANES) data. Using NHANES data, Yu et al. [71] predicts diabetes risk using an SVM classifier. CMS (www.cms.gov) uses the Medicare and Medicaid claims to create the minimum dataset (MDS). Herbert et al. [72] uses MDS data to identify people with diabetes. In this chapter we use the National Inpatient Sample (NIS) data created by AHRQ (www.ahrq.gov) Healthcare Utilization Project (HCUP), to predict the risk for eight chronic diseases.

Disease prediction can be applied to different domains such as risk management, tailored health communication and decision support systems. Risk management plays an important role in health insurance companies, mainly in the underwriting process. Health insurers use a process called underwriting in order to classify the applicant as standard or substandard, based on which they compute the policy rate and the premiums individuals have to pay. Currently, in order to classify the applicants, insurers require every applicant to complete a questionnaire, report current medical status and sometimes medical records, or clinical laboratory results, such as blood test, etc. By incorporating machine learning techniques, insurers can make evidence based decisions and can optimize, validate and refine the rules that govern their business. For instance, Yi et al [71], applied

association rules and SVM on an insurance company database to classify the applicants as standard, substandard or declined.

Another domain where disease prediction can be applied is tailored health communication. For example, one can target tailored educational materials and news to a subgroup, within the general population, that has specific disease risks. Cohen et al. [73], discussed how tailored health communication can motivate cancer prevention and early detection. Disease risk prediction along with tailored health communication can lead to an effective channel for delivering disease specific information for people who will be likely to need it.

In addition to population level clinical knowledge, de-identified public datasets represent an important resource for the clinical data mining researchers. While full featured clinical records are hard to access due to privacy issues, de-identified large national public dataset are readily available [74]. Although these public datasets don't have all the variables of the original medical records, they still maintain some of their main characteristics such as data imbalance and the use of controlled terminologies (ICD-9 codes).

Several machine learning techniques were applied to healthcare data sets for the prediction of future health care utilization such as predicting individual expenditures and disease risks for patients. Moturu et al. [75], predicted future high-cost patients based on data from Arizona Medicaid program. They created 20 non-random data samples, each sample with 1,000 data points to overcome the problem of imbalanced data. A combination of undersampling and oversampling was employed to a balanced sample. They used a variety of classification methods such as SVM, Logistic Regression, Logistic

Model Trees, AdaBoost and LogitBoost. Davis et al. [76], used clustering and collaborative filtering to predict individual disease risks based on medical history. The prediction was performed multiple times for each patient, each time employing different sets of variables. In the end, the clustering results were combined to form an ensemble. The final output was a ranked list of possible diseases for a given patient. Mantzaris et al. [77], predicted Osteoporosis using Artificial Neural Network (ANN). They used two different ANN techniques: Multi-Layer Perceptron (MLP) and Probabilistic Neural Network (PNN). Hebert et al. [72], identified persons with diabetes using Medicare claims data. They ran into a problem where the diabetes claims occur too infrequently to be sensitive indicators for persons with diabetes. In order to increase the sensitivity, physician claims where included. Yu et al. [71], illustrates a method using SVM for detecting persons with diabetes and pre-diabetes.

Zhang et al. [78], conducted a comparative study of ensemble learning approaches. They compared AdaBoost, LogitBoost and RF to logistic regression and SVM in the classification of breast cancer metastasis. They concluded that ensemble learners have higher accuracy compared to the non-ensemble learners.

Together with methods for predicting disease risks, in this chapter we discuss a method for dealing with highly imbalanced data. We mentioned two examples [72], [75] where the authors encountered class imbalanced problems. Class imbalance occurs if one class contains significantly more samples than the other class. Since the classification process assumes that the data is drawn from the same distribution as the training data, presenting imbalanced data to the classifier will produce undesirable results. The data set we use in this work is highly imbalanced. For example, only 3.59% of the patients have

heart disease, thus it is possible to train a classifier with this data and achieve an accuracy of 96.41% while having 0% sensitivity.

5.2. Data

The Nationwide Inpatient Sample (NIS) is a database of hospital inpatient admissions that dates back to 1988 and is used to identify, track, and analyze national trends in health care utilization, access, charges, quality, and outcomes. The NIS database is developed by the Healthcare Cost and Utilization Project (HCUP) and sponsored by the Agency for Healthcare Research and Quality (AHRQ) [10]. This database is publicly available and does not contain any patient identifiers. The NIS data contains discharge level information on all inpatients from a 20% stratified sample of hospitals across the United States, representing approximately 90% of all hospitals in the country [74]. The five strata for hospitals are based on the American Hospital Association classification. HCUP data from the year 2005 will be used for the experiments.

The data set contains about 8 million records of hospital stays, with 126 clinical and nonclinical data elements for each visit (Appendix II). Nonclinical elements include patient demographics, hospital identification, admission date, zip code, calendar year, total charges and length of stay. Clinical elements include procedures, procedure categories, diagnosis codes and diagnosis categories. Every record contains a vector of 15 diagnosis codes. The diagnosis codes are represented using the *International Classification of Diseases, Ninth Revision, Clinical Modification* (ICD-9-CM). The International Statistical Classification of Disease is designed and published by the World Health Organization (WHO). The ICD-9 codes are alphanumeric codes, 3-5 characters long and used by hospitals, insurance companies and other facilities to describe health

conditions of the patient. Every code represents a disease, condition, symptom, or cause of death. There are numerous codes, over 14,000 ICD-9 codes and 3,900 procedures codes.

In addition, every record contains a vector of 15 diagnosis category codes. The diagnosis categories are computed using the Clinical Classification Software (CCS) developed by HCUP in order to categorize the ICD-9 diagnosis and procedure codes. CCS collapsed these codes into a smaller number of clinically meaningful categories, called diagnosis categories. Every ICD-9 code has a corresponding diagnosis category and every category contains a set of ICD-9 codes. We denote each of the 259 disease categories by a value in the range [1, 259]. In Fig. 5.1 we show an example of disease category ("Breast cancer") and some of the ICD-9 codes included in it (174.0 – "Malignant neoplasm of female breast", 174.1– "Malignant neoplasm of central portion of female breast", 174.2– "Malignant neoplasm of upper-inner quadrant of female breast", 233.0-"Carcinoma in situ of breast and genitourinary system").

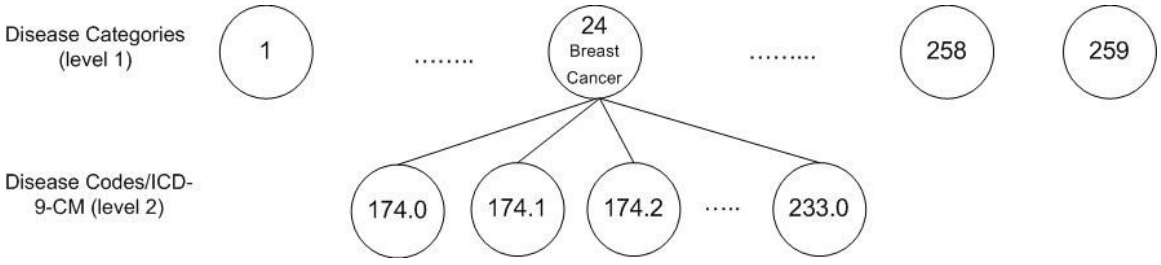


Fig. 5.1. Disease codes and categories hierarchical relationship

Demographics such as age, race and sex are also included in the data set. Fig. 5.2 shows the distribution of patients across age, race and sex.

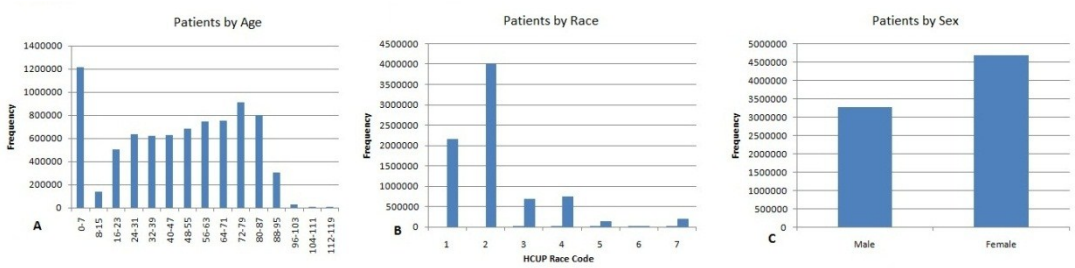


Fig. 5.2. Distribution of patient across age, race and sex

The 2005 HCUP data set is highly imbalanced. The imbalance rate ranges across diseases from 0.01-29.1%. 222 diagnosis categories occurs in less than 5% of the patients and only 13 categories occur in more than 10% of the patients. In Table 5.2 we show the top 10 most prevalent disease categories and in Table 5.3 we show some of the rarest diseases present in the 2005 HCUP data set.

Table 5.1. The 10 most prevalent diseases categories

Disease Category	Prevalence
Hypertension	29.1%
Coronary Atherosclerosis	27.65%
Hyperlipidemia	14.46%
Dysrhythmia	14.35%
Other Circulatory Diseases	12.02%
Diabetes mellitus no complication	12%
Anemia	11.93%

Table 5.2. Some of the most imbalanced diseases categories

Disease Category	Percent of Active class
Male Genital Disease	0.01%
Testis Cancer	0.046%
Encephalitis	0.059%
Aneurysm	0.74%
Breast Cancer	1.66%
Peripheral Atherosclerosis	3.16%
Diabetes Mellitus w/complication	4.7%

One limitation of the 2005 HCUP data set is the arbitrary order in which the ICD-9 codes and disease categories were listed. The codes were not listed in the chronological order according to the date they were diagnosed. Also, the data set does not provide anonymous patient identifier, which could be used to check if multiple records belong to the same patient or to determine the elapsed time between diagnoses.

5.3. Methods

5.3.1. Data Pre-processing

The data set was provided in a large ASCII file containing the 7,995,048 records. The first step was to parse the data set, randomly select *N* records and extract a set of relevant features. Every record is a sequence of characters that are not delimited. However, the data set instructions specifies the starting column and the ending column in the ASCII file for each data element (length of data element). HCUP provides a SAS program to parse the data set, but we chose to develop our own program to perform the parsing.

Feature Selection

For every record, we extracted the age, race, sex and 15 diagnosis categories. Every record is represented as a $d = 262$ dimensional feature vector. Features 1-259 are binary, one for each disease category. The remaining three features are age, race and sex. We denote the samples that contain a given disease category as "active" and the remaining ones as "inactive". The active and inactive data samples are defined only from the point of view of the disease being classified. A snippet of the data set is presented in Table 5.4. For example, in Table 5.4, sample 1 is active for disease category 50, while sample *N* is inactive.

Table 5.3. Sample Dataset, the shaded column represents the category to predict

	Cat. 1	Cat. 2	Cat. 3	….	Cat. 50	….	Cat. 257	Cat. 258	Cat. 259	Age	Race	Sex
Patient 1	0	0	0	….	1	….	0	1	1	69	3	0
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
Patient *N*	1	0	0	….	0	….	1	0	0	55	1	1

While, in general, using only disease categories may not lead to a valid disease prediction, the approach presented in this chapter needs to be seen in the larger context of our TigerPlace eldercare research [3]. By integrating large public data sets (such as the one used in this chapter) with monitoring sensors and electronic health records (EHR) data, we can achieve the required prediction precision for an efficient delivery of tailored medical information.

5.3.2. **Learning from Imbalanced Data**

A data set is class-imbalanced if one class contains significantly more samples than the other. For many disease categories, the unbalance rate ranges between 0.01-29.1% (that is, the percent of the data samples that belong to the active class). For example, (see Table 5.3) only 3.16% of the patients have Peripheral Atherosclerosis. In such cases, it is challenging to create an appropriate testing and training data sets, given that most classifiers are built with the assumption that the test data is drawn from the same distribution as the training data [79].

Presenting imbalanced data to a classifier will produce undesirable results such as a much lower performance on the testing that on the training data. Among the classifier

learning techniques that deal with imbalanced data we mention oversampling, undersampling, boosting, bagging and repeated random sub-sampling [80], [81]. In the next section we describe the repeated random sub-sampling method that we employ in this work.

5.3.3. **Repeated Random Sub-Sampling**

Repeated random sub-sampling was found to be very effective in dealing with data sets that are highly imbalanced. Because most classification algorithms make the assumption that the class distribution in the data set is uniform, it is essential to pay attention to the class distribution when addressing medical data. This method divides the data set into active and inactive instances, from which the training and testing data sets are generated. The training data is partitioned into sub-samples with each sub-sample containing an equal number of instances from each class, except for last sub-sample (in some cases). The classification model is fitted repeatedly on every sub-sample and the final result is a majority voting over all the sub-samples.

In this chapter we used the following repeated random sub-sampling approach. For every target disease we randomly choose N samples from the original HCUP data set. The N samples are divided into two separate data sets, N_1 active data samples and N_0 inactive data samples, where $N_{1+} N_{0=} N$. The testing data will contain 30% active samples N_1 (TsN_1) while the remaining 70% will be sampled from the N_0 (TsN_0) inactive samples. The 30/70 ratio was chosen by trial-and-error. The training data set will contain the remaining active samples (TrN_1) and inactive samples (TrN_0).

Since the training data is highly imbalanced ($TrN_1 << TrN_0$), the TrN_0 samples are partitioned into NoS training sub-samples, where NoS is the ratio between TrN_0 and TrN_1.

Every training sub-sample has equal number of instances of each class. The training active samples (TrN_1) are fixed among all the training data sub-samples, while the inactive samples will be sampled without replacement from TrN_0. There will be *NoS* sub-samples to train the model on. Eventually, every inactive sample in the training data is selected once, while every active sample is selected *NoS* times. After training the model on all the sub-samples, we employ a "majority voting" approach to determine the final class memberships (see Algorithm 5.1). A diagram describing the process of RF and the sub-sampling procedure is presented in Fig. 5.3.

Algorithm 5.1. Repeated Random Sub-Sampling

1 *TsN* = total number of samples in the testing data
2 N_0 = number of inactive samples

3 N_1 = number of active samples
4 *N* = total number of samples in the data set, where $N=N_0+N_1$

5 Generate testing data
6 Randomly select *TsN1* samples from *N1* , where *TsN1*=0.3**N1*
7 Randomly select *TsN0* samples from *N0*, where *TsN0=TsN - TsN1*

9 *Ts* = *TsN0* samples + *TsN1* samples (*Ts* = testing data)
10
11 Generate training data
12 Contains *TrN1* samples, *TrN1* = remaining *N1* samples after generating testing data
13 Contains *TrN0* samples, *TrN0* = remaining *N0* samples after generating testing data

14 *NoS = TrN0/TrN1*

15 **for** *s* = 1 to *NoS* **do**
16 Generate the *s* training data sub-sample
17 *TrSS0* = Randomly select *TrN1* (number of training active samples) samples from *TrN0* inactive samples without replacement (Guarantees full balance of the training data sub-samples, except for the last sub-samples, in some cases)
18 *TrSS = TrSS0 + TrN1*
19 ys(x) = classifer(*TrSS*, *Ts*) (Predicted class labels for *Ts* using sub-sample *TrSS*)
20 **endfor**

21 y(x) = majority voting $\{y_s(x)\}_1^{NoS}$ (Final predicted class is majority voting over all sub-samples)

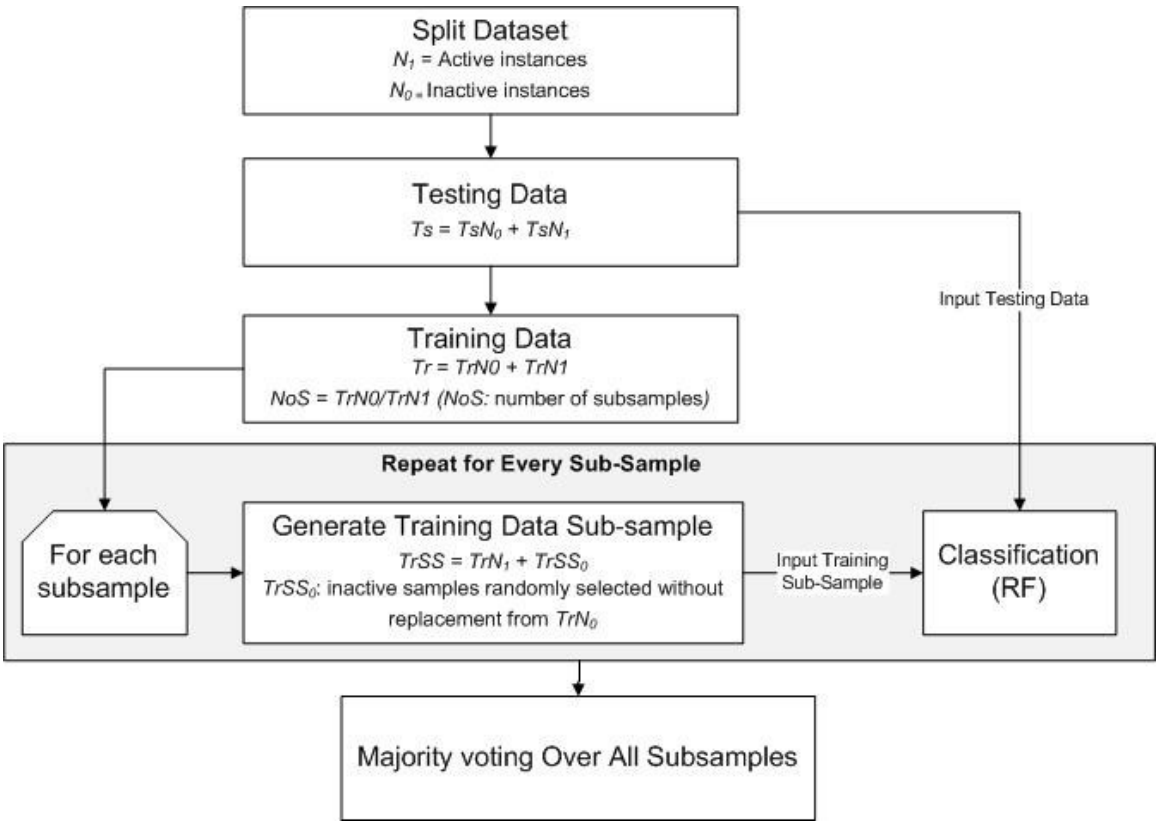


Fig. 5.3. Flow diagram of random forest and sub-sampling approach

5.3.4. **Random Forest**

RF is an ensemble learner, a method that generates many classifiers and aggregates their results. RF will create multiple classification and regression (CART) trees, each trained on a bootstrap sample of the original training data and searches across a randomly selected subset of input variables to determine the split. CARTs are binary decision trees that are constructed by splitting the data in a node into child nodes repeatedly, starting with the root node that contains the whole learning sample [82]. Each tree in RF will cast a vote for some input *x*, then the output of the classifier is determined by majority voting

of the trees (algorithm 5.2). RF can handle high dimensional data and use a large number of trees in the ensemble. Some important features of RF are [83]:

- It has an effective method for estimating missing data.
- It has a method, weighted random forest (WRF), for balancing error in imbalanced data.
- It estimates the importance of variables used in the classification.

Algorithm 5.2. Random Forest for Classification

```
1   ntree = number of trees to be generated
2   N = number of samples in the data set

3   for t = 1 to ntree do
4       Generate bootstrap sample Z of size N from the original data - with replacement
5       for each bootstrap Z grow a classification tree

6         for i = 1 to NumberOfNodes do
7             randomly sample mtry variables from M variables
8             choose best split among the sampled variables (bagging is special case of RF and
              obtained whenmtry = M)
9         endfor
10
11        yt(x) = class prediction of the tth tree
12  endfor

13  Yrf (x) = majority voting {y_t(x)}^ntree_1   (Final predicted class is majority voting over all trees
    in RF)
```

Chen et al. [84], compared WRF and balanced random forest (BRF) on six different and highly imbalanced data sets. In WRF, they tuned the weights for every data set, while in BRF, they changed the votes cutoff for the final prediction. They concluded that BRF is computationally more efficient than WRF for imbalanced data. They also found that WRF is more vulnerable to noise compared to BRF. In this chapter, we used RF without tuning the class weights or the cutoff parameter.

5.3.5. **Splitting Criterion**

Like CART, RF uses the Gini measure of impurity to select the split with the lowest impurity at every node [85]. Gini impurity is a measure of the class label distribution in the node. The Gini impurity takes values in [0, 1], where 0 is obtained when all elements in a node are of the same class. Formally, the Gini impurity measure for the variable $X=\{x_1, x_2, \ldots, x_j\}$ at node t, where j is the number of children at node t, N is the number of samples, n_{ci} is the number of samples with value x_i belonging to class c, a_i is the number of samples with value x_i at node t, then the Gini impurity is given by [82]

$$I(t_{x_i}) = 1 - \sum_{c=0}^{C} \left(\frac{n_{ci}}{a_i}\right)^2 \tag{5.1}$$

The Gini index of a split is the weighted average of the Gini measure over the different values of variable X, which is given by

$$Gini(t, X) = \sum_{i=1}^{j} \frac{a_i}{N} I(t_{x_i}) \tag{5.2}$$

The decision of the splitting criterion will be based on the lowest Gini impurity value computed among the m variables. In RF, each tree employs a different set of m variables to construct the splitting rules.

5.3.6. **Variable Importance**

One of the most important features of RF is the output of the variable importance. Variable importance measures the degree of association between a given variable and the classification result. RF has four measures for the variable importance: raw importance score for class 0, raw importance score for class 1, decrease in accuracy and the Gini index. To estimate variable importance for some variable j, the out-of-bag (OOB)

samples are passed down the tree and the prediction accuracy is recorded. Then the values for variable j are permuted in the OOB samples and the accuracy is measured again. These calculations are carried out tree by tree as the RF is constructed. The average decrease in accuracy of these permutations is then averaged over all the trees and is used to measure the importance of the variable j. If the prediction accuracy decreases substantially, then that suggests that the variable j has strong association with the response [86]. After measuring the importance of all the variables, RF will return a ranked list of the variable importance.

Formally, let β_t be the OOB samples for tree t, $t \in \{1,..., ntree\}$, y'_i is the predicted class for instance i before the permutation in tree t and $y'_{i,\alpha}$ is the predicted class for instance i after the permutation. The variable importance VI for variable j in tree t is given by

$$VI_j^t = \frac{\sum_{i=1}^{N} \beta_t I(y_i = y_i'^t)}{|\beta_t|} - \frac{\sum_{i=1}^{N} \beta_t I(y_i = y_{i,\alpha}'^t)}{|\beta_t|} \tag{5.3}$$

The raw importance value for variable j is then averaged over all trees in the RF.

$$VI_j = \frac{\sum_{t=1}^{ntree} VI_m^t}{ntree} \tag{5.4}$$

The variable importance used in this chapter is the Mean Decrease Gini (MDG), which is based on the Gini splitting criterion discussed earlier. The MDG measure the decrease ΔI (5.1) that results from the splitting. For two class problem, the change in I (5.6) at node t is defined as the class impurity (5.5) minus the weight average of Gini measure (5.2) [19, 20].

$$I(t) = 1 - \sum_{c=0}^{C} \left(\frac{n_i}{N}\right)^2 \tag{5.5}$$

The decrease in Gini impurity is recorded for all the nodes *t* in all the trees (*ntree*) in RF for all the variables and Gini Importance (*GI*) is then computed as [87]

$$GI = \sum_{ntree} \sum_{t} \Delta I(t) \tag{5.6}$$

5.3.7. **Classification with Repeated Random Sub-Sampling**

Training the classifier on a data set that is small and highly imbalanced will result in unpredictable results as discussed in earlier sections. To overcome this issue, we used repeated random sub-sampling. Initially, we construct the testing data and the *NoS* training data sub-samples. For each disease, we train *NoS* classifiers and test all of them on the same data set. The final labels of the testing data are computed using a majority voting scheme.

5.3.8. **Model Evaluation**

To evaluate the performance of the RF we compared it to SVM on imbalanced data sets for eight different chronic diseases categories. Two sets of experiments were carried out:

Set I: We compared RF, boosting, bagging and SVM performance with repeated random sub-sampling. Both classifiers were fitted to the same training and testing data and the process was repeated 100 times. The ROC curve and the average AUC for each classifier were calculated and compared. To statistically compare the two ROC curves we employed an Analysis of Variance (ANOVA) approach [88] where the standard deviation of each AUC was computed as:

$$SE = \sqrt{\frac{\theta(1-\theta) + (C_p - 1)(Q_1 - \theta^2) + (C_n - 1)(Q_2 - \theta^2)}{C_p C_n}}, \quad (5.7)$$

where Cp, Cn, θ are the number positive instances, negative instances and the AUC, respectively and

$$Q_1 = \frac{\theta}{(2-\theta)}, Q_2 = \frac{2 * \theta^2}{(1+\theta)}. \quad (5.8)$$

Set II: In this experiment we compare RF, bagging, boosting and SVM performance without the sampling approach. Without sampling the data set is highly imbalanced, while sampling should improve the accuracy since the training data sub-samples fitted to the model are balanced. The process was again repeated 100 times and the ROC curve and the average AUC were calculated and compared.

5.4. Results

We performed the classification using R, which is open source statistical software. We used R Random Forest (randomForest), bagging (ipred), boosting (caTools) and SVM (e1071) packages. There are two parameters to choose when running a RF algorithm: the number of trees (*ntree*) and the number of randomly selected variables (*mtry*). The number of trees did not significantly influence the classification results. This can be seen in Fig. 5.4, where we ran RF for different *ntree* values to predict breast cancer. As we see from Fig. 5.4, the sensitivity of the classification did not significantly change once *ntree*>20. The number of variables randomly sampled as candidates at each split (*mtry*) was chosen as the square root of the number of features (262 in our case),

hence *mtry* was set to 16. Palmer et al. [89] and Liaw et al. [90] also reported that RF is usually insensitive to the training parameters.

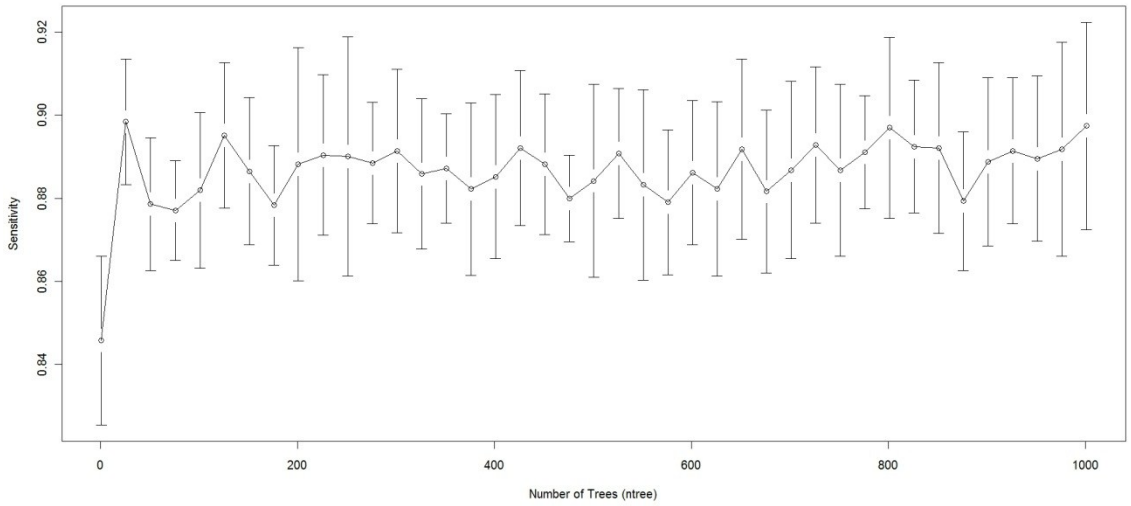


Fig. 5.4. RF behaviour when the number of trees (ntree) varies

For SVM we used a linear kernel, termination criterion (*tolerance*) was set to 0.001, *epsilon* for the insensitive-loss function was 0.1 and the regularization term (*cost*) was set to 1. Also, we left bagging and boosting with the default parameters.

We randomly selected N=10,000 data points from the original HCUP data set. We predicted the disease risks on 8 out of the 259 disease categories. Those categories are: breast cancer, type 1 diabetes, type 2 diabetes, hypertension, coronary atherosclerosis, peripheral atherosclerosis, other circulatory diseases and osteoporosis.

5.4.1. **Result set I: Comparison of RF, bagging, boosting and SVM**

RF, SVM, bagging and boosting classification were performed 100 times and the average area under the curve (AUC) was measured. The repeated random sub-sampling approach has improved the detection rate considerably. On seven out of eight disease

categories RF outperformed the other classifiers in terms of AUC (Table 5.5). In addition to ROC comparison, we used ANOVA [88] as mentioned earlier to statistically compare the ROC of boosting and RF, since both of these classifiers scored the highest in terms of AUC. ANOVA results comparing RF ROC and boosting ROC are summarized in Table 5.6. The lower the *p* value is the more significant the difference between the ROCs is. The results of ANOVA test tells us that although RF outperformed boosting in terms of AUC, that performance was only significant in three diseases only (high prevalence diseases). The possible reason for performance difference insignificance for the other 5 diseases (mostly low prevalence diseases) might be the low number of active samples available in our sampled dataset. For example, for breast cancer we would have about 166 cases available.

Table 5.4. RF,SVM, bagging and boosting performance in terms of AUC on eight disease categories

Disease	RF	SVM	Bagging	Boosting
Breast cancer	**0.9123**	0.9063	0.905	0.8886
Diabetes no complication	**0.8791**	0.8417	0.8568	0.8607
Diabetes	**0.94317**	0.9239	0.9294	0.9327
Hypertension	**0.9003**	0.8592	0.8719	0.8842
Coronary Atherosclerosis	**0.9199**	0.8973	0.887	0.9026
Peripheral	**0.9095**	0.8972	0.8967	0.9003
Other Circulatory	**0.7899**	0.7591	0.7669	0.7683
Osteoporosis	**0.87**	0.867	0.8659	0.8635

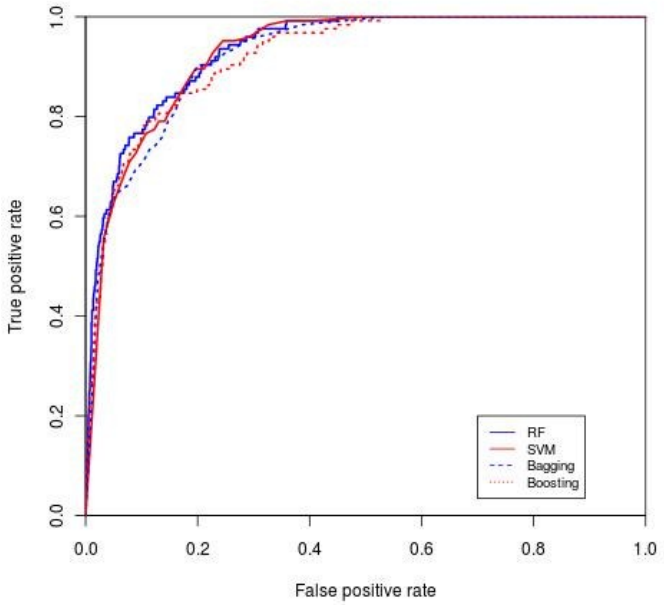


Fig. 5.5. ROC curve for diabetes mellitus

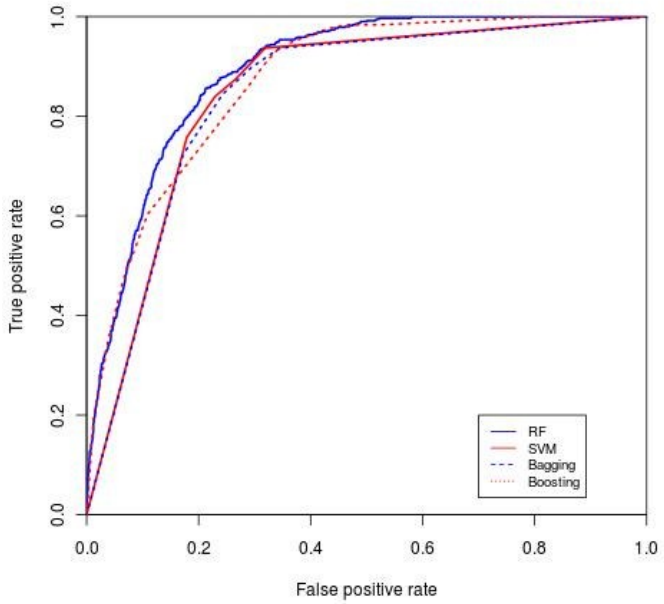


Fig. 5.6. ROC curve for hypertension

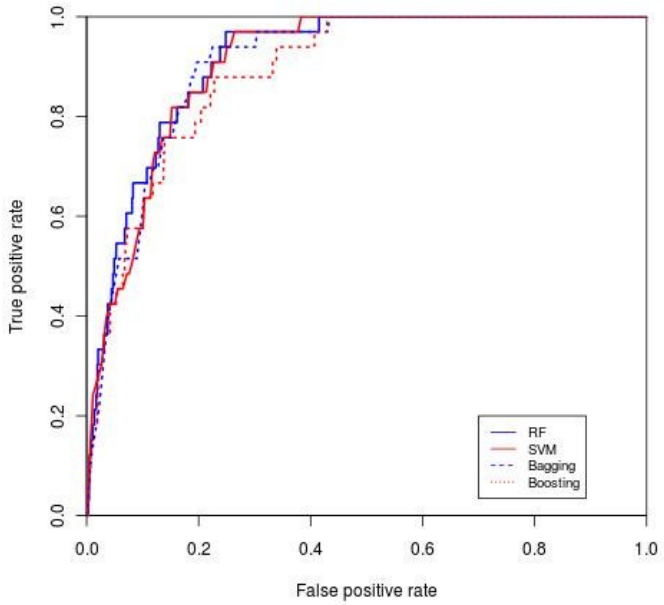


Fig. 5.7. ROC curve for breast cancer

Table 5.5. Statistical comparison of RF and boosting ROC curves, the lower the value the more significant the difference is

Disease	p **−value**
Breast cancer	0.8057
Diabetes no complication	0.3293
Diabetes with/complication	0.6266
Hypertension	0.2
Coronary Atherosclerosis	0.2764
Peripheral Atherosclerosis	0.8203
Other Circulatory Diseases	0.566
Osteoporosis	0.908

We compared our disease prediction results to the ones reported by other authors. For instance, Yu et al. [71], describes a method using SVM for detecting persons with diabetes and pre-diabetes. They used data set from the National Health and Nutrition Examination Survey (NHANES). NHANES collects demographic, health history, behavioural information and it may also include detailed physical, physiological, and laboratory examinations for each patient. The AUC for their classification scheme I and

II was 83.47% and 73.81% respectively. We also predicted diabetes with complications and without complications and the AUC values were 94.31% and 87.91% respectively (Diabetes without complication ROC curve in Fig. 5.5).

Davis et al. [76] used clustering and collaborative filtering to predict disease risks of patients based on their medical history. Their algorithm generates a ranked list of diseases in the subsequent visits of that patient. They used an HCUP data set, similar to the data set we used. Their system predicts more than 41% of all the future diseases in the top 20 ranks. One reason for their low system performance might be that they tried to predict the exact ICD-9 code for each patient, while we predict the disease category.

Zhang et al. [78] performed classification on breast cancer metastasis. In their study, they used two published gene expression profiles. They compared multiple methods (logistic regression, SVM, AdaBoost, LogitBoost and RF). In the first data set, the AUC for SVM and RF was 88.6% and 89.9% respectively and for the second data set 87.4% and 93.2%. The results we obtained for breast cancer prediction for RF were 91.23% (ROC curve in Fig. 5.7).

Mantzaris et al [77] predicted osteoporosis using multi-layer perceptron (MLP) and probabilistic neural network (PNN). Age, sex, height and weight were the input variables to the classifier. They reported a prognosis rate on the testing data of 84.9%. One the same disease, we reported an AUC for RF of 87%.

One of the important features of the RF approach is the computation of the importance of each variable (feature). We used Mean Decrease Gini (Eq. 5.5, 5.6, 5.7) measure to achieve the variable importance (Table 5.7). Variables with high importance have strong association with the prediction results. For example, Mantzaris et al. [77]

mentioned that osteoporosis (row 8) is more prevalent in people older than 50 and occurs in women more than men and that agrees with the first and forth important variables (age and sex) reported by RF (Table 5.7). Another example is diabetes with complication (row 3) that often presents with fluid-electrolyte imbalance and it's incidence is inversely correlated with a normal pregnancy.

Table 5.6. Top four most importance variable for the eight disease categories

Disease	Variable	Variable 2	Variable 3
1.Breast cancer	Age	Sex	Secondary malignant Secondary
2. Diabetes no complication	Age	Hypertension	Hyperlipidemia
3. Diabetes with/complication	Age	Normal pregnancy	Fluid-electrolyte Imbalance
4. Hypertension	Age	Hyperlipidemia	Diabetes without compl.
5. Coronary atherosclerosis	Age	Hypertension	Hyperlipidemia
6. Peripheral atherosclerosis	Age	Coronary Atherosclerosis	Hypertension
7. Other circulatory diseases	Age	Dysthymia	Anemia
8. Osteoporosis	Age	Race	Hypertension

5.4.2. **Result set II: Sampling vs. non-sampling**

In this section we show that classification with sampling outperforms standalone classifiers on the HCUP data set (Table 5.8). RF, bagging, boosting and SVM with sampling have higher ROC curves and reaches a detection rate of 100% faster than the standalone classifiers. For demonstration purposes, we included the comparisons for RF with and without sampling for three disease categories, breast cancer, other circulatory diseases and peripheral atherosclerosis (ROC curve in Fig. 5.8-5.10). Table 5.8 describes the results for the non-sampling classification for the four mentioned classifiers.

Table 5.7. RF, SVM, bagging and boosting performance without sub-sampling in terms of AUC on eight disease categories

Disease	RF	SVMSVM	Bagging	Boosting
Breast cancer	**0.8793**	0.5	0.5085	0.836
Diabetes no	**0.8567**	0.5	0.4749	0.8175
Diabetes	**0.9084**	0.648	0.4985	0.8278
Hypertension	**0.8893**	0.6908	0.4886	0.8515
Coronary	**0.9193**	0.6601	0.4945	0.8608
Peripheral	**0.8872**	0.5	0.4925	0.8279
Other Circulatory	**0.7389**	0.5	0.4829	0.6851
Osteoporosis	0.7968	0.5	0.4931	**0.8561**

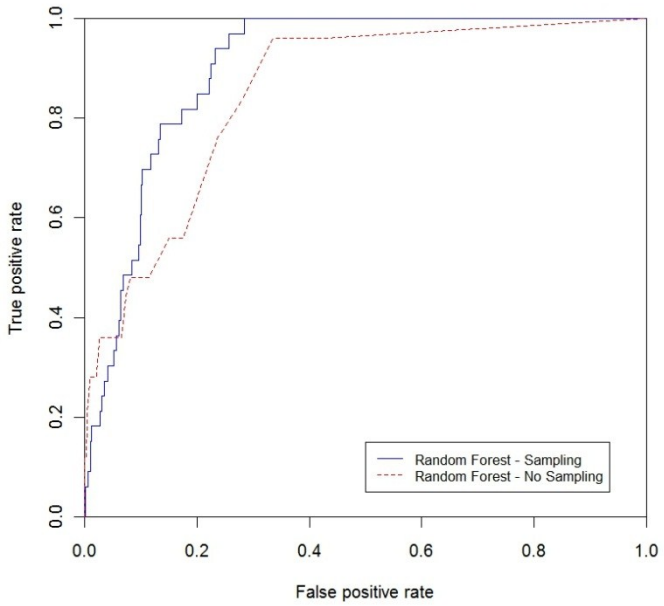


Fig. 5.8. ROC curve for breast cancer (sampling vs. non-sampling)

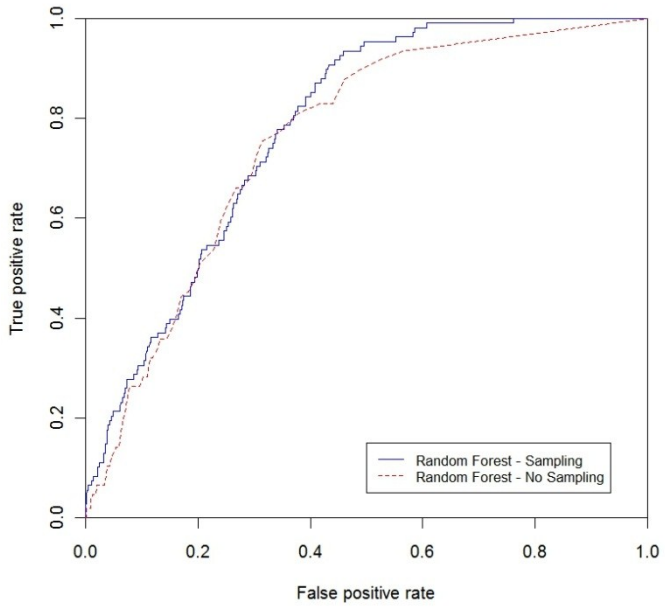


Fig. 5.9. ROC curve for other circulatory diseases (sampling vs. non-sampling)

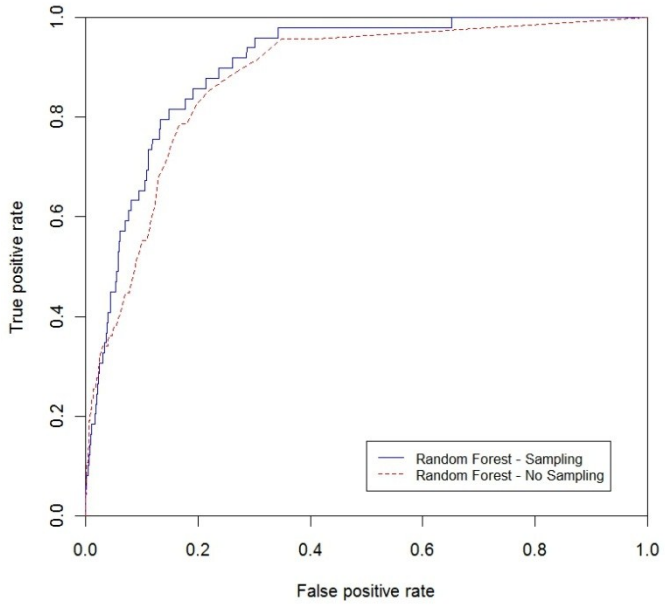


Fig. 5.10. ROC curve for peripheral atherosclerosis (sampling vs. non-sampling)

5.5. Conclusions and Future Work

Disease prediction is becoming an increasingly important research area due to the large medical datasets that are slowly becoming available. While full featured clinical records are hard to access due to privacy issues, de-identified large public dataset are still a valuable resource for at least two reasons. First, they may provide population level clinical knowledge. Second, they allow the data mining researcher to develop methodologies for clinical decision support systems that can then be employed for electronic medical records. In this study, we presented a disease prediction methodology that employs random forests (RF) and a nation-wide de-identified public dataset (HCUP). We show that, since no national medical warehouse is available to date, using nation-wide datasets provide a powerful prediction tool. In addition, we believe that the presented methodology can be employed with electronic medical records, if available.

To test our approach we selected eight chronic diseases with high prevalence in elderly. We performed two sets of experiments (set I and set II). In set I, we compared RF to other classifiers with sampling, while in set II we compared RF to other classifiers without sub-sampling. Our results show that we can predict diseases with an acceptable accuracy using the HCUP data. In addition, the use of repeated random sub-sampling is useful when dealing with highly imbalanced data. We also found that incorporating demographic information increased the area under the curve by 0.33-10.1%. .

In this chapter we used the NIS dataset (HCUP) created by AHRQ. Few researchers have utilized the NIS dataset for disease predictions. The only work we found on disease prediction using NIS data was presented by Davis et al. [76], in which clustering and collaborative filtering was used to predict individual disease risks based on

medical history. In this work we provided extensive proof that RF can be successfully used for disease prediction in conjunction with the HCUP dataset.

Some of the limitations of our approach come from limitations of the HCUP data set such as the arbitrary order of the ICD-9 codes and lack of patient identification. For example, since the ICD-9 codes are not listed in chronological order according to the date they were diagnosed, we inherently use future diseases in our prediction. This explains, in part, the high accuracy of our prediction. In addition, the HCUP data set does not provide anonymous patient identifier, which can be used to check if multiple records belong to the same patient and to estimate the time interval between two diagnoses. Hence we might use the data for the same patient multiple times. Additionally, the data set does not include the family history; rather it includes the individual diagnosis history which is represented by the diseases categories.

The accuracy achieved in disease prediction is comparable or better than the previously published results. The average RF AUC obtained across all disease was about 89.05% which may be acceptable in many applications. Additionally, unlike many other published results were they focus on predicting one specific disease, our method can be used to predict the risk for any disease. Finally, we consider the results obtained with the proposed method adequate for our intended use, which is tailored health communication.

The classification discussed in the section relies on representing each patient by a crisp feature vector. A patient either has a disease or not. We can change this by exploiting the hierarchical relationship among ICD-9 codes and instead represent each patient with a fuzzy feature vector, which is the discussion of the next chapter.

CHAPTER 6

IMPROVING DISEASE PREDICTION USING ICD-9 ONTOLOGICAL FEATURES

Disease prediction has become important in a variety of applications such as health insurance, tailored health communication and public health. Disease prediction is usually performed using publically available datasets such as HCUP, NHANES or MDS that were initially designed for reporting or cost evaluation but not for prediction. In these datasets, medical diagnoses are traditionally arranged in "diagnose-related groups" (DRGs). In this chapter we compare the disease prediction based on crisp DRG features with the results obtained employing a new set of features that consist of the fuzzy membership of patient diagnoses in the DRG groups. The fuzzy membership features were computed using an ICD-9 ontological similarity approach. The prediction results obtained on a subset of 30,000 patients from the 2005 HCUP data representing three diseases (diabetes, atherosclerosis and hypertension) using two classifiers (random forest and SVM) show a significant (about 10%) improvement in the area under the ROC curve (AROC).

6.1. Introduction

Disease prediction is employed in different domains such as risk management, tailored health communication and public health. Risk management plays an important role in health insurance industry, mainly in the underwriting process. Health insurers use a process called underwriting in order to classify the applicant as standard or substandard, based on which they compute the policy rate and the premiums individuals have to pay [71].

Another domain where disease prediction may be applied is tailored health communication. For example, we can target specific medical educational materials and news to a subgroup within the general population that has a high predicted risk for a given disease. Cohen et al [73] discussed how tailored health communication for cancer patients can motivate cancer prevention and early detection. Disease risk prediction along with tailored health communication represents an effective preventive medicine method that may lead in the long-run to a reduction in the cost of medical care.

The reporting requirements of various US governmental agencies such as Center for Disease Control (CDC), Agency for Health Care Quality (AHRQ) and US Department of Health and Human Services Center for Medicare Services (CMS) have created huge public datasets that, we believe, are not utilized to their full potential. For example, CDC (www.cdc.gov) makes available National Health and Nutrition Examination Survey (NHANES) data. Using NHANES data, Yu et al. [91] predicts diabetes risk using an SVM classifier. CMS (www.cms.gov) uses the Medicare and Medicaid claims to create the minimum dataset (MDS). Herbert et al. [72] uses MDS data to identify people with diabetes. In this chapter we use the National Inpatient Sample (NIS) data created by AHRQ (www.ahrq.gov) Healthcare Utilization Project (HCUP), to predict the risk for three diseases: diabetes, atherosclerosis and hypertension. To compute the disease risk we use a new set of ICD-9 features based on ontological similarity between the ICD-9 diagnoses contained in a DRG and the ICD-9 diagnoses of the patient. We compare this approach with the prediction of the same diseases described in [92].

The remainder of this chapter is structured thusly: in Section 6.2 we describe the ICD-9 medical taxonomy together with the similarity measure used in the feature

extraction process. In Section 6.3 we describe the proposed ontological feature extraction algorithm together with a brief description of the classifiers employed, in Section 6.4 we show some results obtained on a subset of 2005 HCUP patient dataset and in Section 6.5 we provide some conclusions and ideas for future research.

6.2. ICD-9 Medical Taxonomy and Similarity Measure

International classification of diseases-version 9 (ICD-9) is a diagnose coding system used in hospitals for data retrieval and billing purposes. Every code represents a disease, condition, symptom, or cause of death. However, from our point of view ICD-9 represents an ontology, i.e. a controlled vocabulary overlaid with a "is-a" term hierarchy. The controlled vocabulary allows for detection of synonymy when two diagnoses are compared. The hierarchy (tree) structure allows for assessing the semantic similarity between diagnoses. A part of the ICD-9 tree is shown in Fig. 6.1.

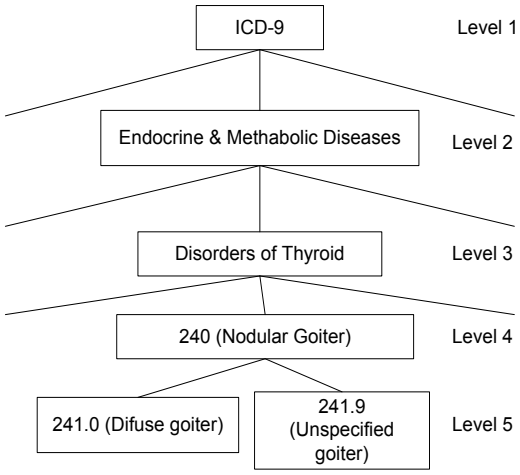


Fig. 6.1. Partial view of the ICD-9 hierarchy

From Fig. 6.1 we see that diagnoses 241.0 and 241.9 are semantically related although they are syntactically (string-wise) different. Unfortunately, the ICD-9 hierarchy

has only 5 levels. This will have an impact on the granularity of the term similarity as many pairs of terms will have the same similarity coefficient. However, even with this low granularity the impact on prediction performance is significant.

In our view, the hierarchical structure of the ICD-9 ontology represents the knowledge of the medical field as viewed by the domain experts (physicians). The key of our approach is to use the domain knowledge (hierarchy) in computing patient similarities. Given two patients described by a sequence of ICD-9 diagnoses, T_{ij}, we first consider the problem of computing the association (seen as fuzzy membership), $s(T_1, T_2)$, between two terms (diagnoses) T_1 and T_2.

There are many algorithms for defining term similarity in a taxonomy (see Chapter 2 in [93]). One way of computing term similarity is to assign each term T_i weights based on its importance, IMP, within the ontology. As a consequence, two patients are more similar if they both have the same rare (in the database) disease (say cystic fibrosis) than if they both have flu. The term importance can be computed (see Chapter 2 in [93]) using path-based, depth-based, density-based, information content-based approaches. In this work we use a depth-based approach. The importance IMP, of a term in the ICD-9 taxonomy is computed as $IC = 1 - 1/n$ where $n = \{1,2,3,4,5\}$ is the level of the term within the hierarchy. For example, the IMP of diagnosis code 241 (Goiter, level 4) is $IMP(241) = 1 - 1/n = 1 - 1/4 = 0.75$. For consistency, we consider $IMP(level\ 5) = 1$. Now, returning to the problem from the beginning of this paragraph, the similarity of two diagnosis terms, $s(T_1, T_2)$, is defined as:

$$s(T_1, T_2) = IMP(NCA(T_1, T_2)) \tag{6.1}$$

where NCA=”nearest common ancestor” of the two terms in the ontology.

In Fig. 6.1, the nearest common ancestor (NCA) of 241.0 and 241.9 is 241, and so, the similarity between the two diagnoses is $s(241.0,214.9) = IMP(241) = 0.75$. This is clearly the simplest approach and it is only used here for illustrative purposes.

For two sets of ICD-9 terms, $P_1 = \{T_{11}, \dots, T_{1n}\}$and $P_2 = \{T_{21}, \dots, T_{2m}\}$ we can define a variety of similarities (see Chapter 2 in [93] for details). In this chapter we consider the following simple formula:

$$s(P_1, P_2) = \max_{i,j}\{T_{1i}, T_{2j}\} \tag{6.2}$$

6.3. Study Methodology

To better understand the feature extraction process we first describe the 2005 HCUP dataset used in this chapter (denoted henceforth HCUP2005).

6.3.1. The HCUP2005 dataset

The Nationwide Inpatient Sample (NIS) is a database of hospital inpatient admissions that dates back to 1988 and it is used to identify, track, and analyze national trends in health care utilization, access, charges, quality, and outcomes. The NIS database is developed by the Healthcare Utilization Project (HCUP) and sponsored by the AHRQ. This database is publicly available and does not contain any patient identifiers. The database contains discharge level information on all inpatients from a 20% stratified sample of hospitals across the United States, representing approximately 90% of all US hospitals [74]. HCUP data from the year 2005, denoted as HCUP2005, will be used in this chapter.

The data set contains 7,995,048 hospital stays and 126 clinical and nonclinical data elements for each hospital stay. Nonclinical elements include patient demographics,

hospital identification, admission date, zip code, calendar year, total charges and length of stay. Clinical elements include procedures, procedure categories, diagnosis codes and diagnosis categories. Every record contains a vector of 15 ICD-9 diagnosis codes. In addition, every record contains a vector of 15 diagnosis category codes (DRGs). The diagnosis categorization is performed using the Clinical Classification Software (CCS) developed by HCUP. There are numerous ICD-9 codes, over 14,000 codes; CCS collapsed these codes into a smaller number of clinically meaningful DRGs. There are 259 diagnosis categories in the HCUP2005 dataset, every category is denoted by a value in the range 1-259. Demographics such as age, race and sex are also included in our data set and used predicting the three medical conditions.

The prevalence in the HCUP2005 dataset of three diseases used in this chapter (hypertension, diabetes mellitus and breast cancer) is given in Table 6.1 below. As we see from Table 6.1, some diseases like testis cancer might not have a sufficient number of samples for training a classifier even on such a large dataset.

Table 6.1. The prevalence of three diseases in the HCUP2005 dataset

Disease	Prevalence
Hypertension	29.1%
Diabetes mellitus, no	12%
Coronary Atherosclerosis	27.65%
Testis Cancer	0.046%

6.4. ICD-9 based Ontological Features

To predict a disease, we extract from HCUP2005 a random set of N patients, $N/2$ with the disease and $N/2$ without it. For each patient, P_i with $i = \{1, \dots, N\}$, we used from HCUP2005 the following variables: age, race, sex, 15 ICD-9 codes $\left(P_{i,ICD9}\right)$ and 15 diagnosis categories $\left(P_{i,DRG}\right)$. As mentioned before, there are 259 DRGs, DRG_j with $j = \{1, \dots, 259\}$, and every group contains a set of ICD-9 codes,

$DRG_j = \{ICD9_{j1}, \dots, ICD9_{jk}\}$. In [5] we represented each patient P_i using a feature vector $x_i^{crisp} \in \mathbf{R}^p$, with $P = 262$ dimensions. Features 1-259 (DRG related) were computed as:

$$x_{ip}^{crisp} = \begin{cases} 1 & P_{i,ICD9} \cap DRG_p \neq \emptyset \\ 0 & P_{i,ICD9} \cap DRG_p = \emptyset \end{cases}. \quad (6.3)$$

Essentially, since the DRGs were computed for us, the feature vector had an 1 in position p if DRG_p was contained in the diagnoses set of patient i,$P_{i,DRG}$. As a result, each feature vector contained at most 15 ones (the number of DRGs stored per patient) which was a rather sparse representation. The last 3 features (260, 261, 262) were sex, age and race, respectively.

In this chapter we propose to compute the 259 diagnose related features using a fuzzy membership in each DRG_j, i.e. a value between 0 and 1 that represents the similarity between a diagnose and DRG_j. The proposed features will be calculated as:

$$x_{ip} = s(P_{i,ICD9}, DRG_p); p \in \{1, \dots, 259\}, \quad (6.4)$$

where the above similarity, s, is computed using formula (6.2). Features with index 260, 261 and 262 are similar to the ones we used in [92], i.e. age, race and sex.

6.4.1. **Example 1**

Consider a patient with the following set of diagnoses $P = \{682.6, 486, 453.42, 493.92, 41.11, 250.00, 782.3, 278.01\}$. The crisp [92] and the fuzzy (ontological) features (index 1-259) related to this patient are shown in Fig. 6.2.

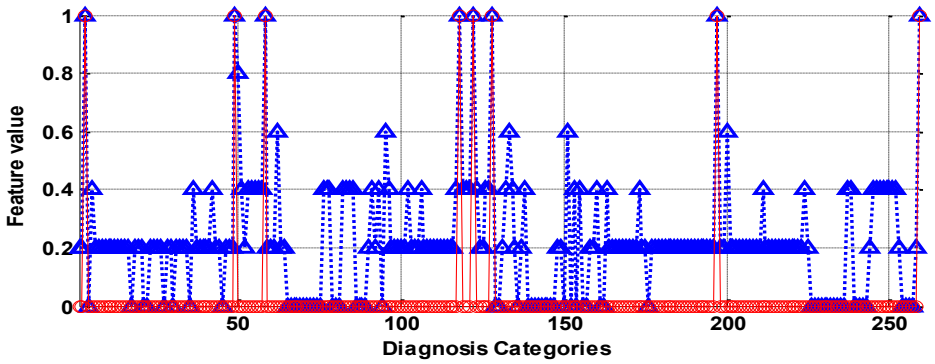


Fig. 6.2. The crisp (red) and fuzzy ontological (blue) features for patient P from example 1

We note that the ontological features coincide with the crisp ones for DRGs that contain one of the ICD-9 codes from the diagnoses set (in number of 8). An example is DRG_{49} ("diabetes mellitus without complications") that contains the ICD-9 code 250.00 ("Diabetes mellitus without complication type II or unspecified type not stated as uncontrolled"). However, there are other indices where the ontological features have a high value. Take for example DRG_{50} ("diabetes mellitus with complications") that contains among others the ICD-9 code 250.03 ("Diabetes mellitus without complication type I uncontrolled"). Since P does not contain this code, $x_{50}^{crisp} = 0$. However, since $s(250.00,250.03) = 0.8$, $x_{50} = 0.8$, the related ontological feature is greater than zero. Aside from the fact that by using the relations from the ICD-9 taxonomy we provide a better representation of the diagnoses set, we also account for the uncertainty of the coding process itself (known to be somewhat unreliable).

6.5. Classifiers used

In this chpater we present experiments performed with two classifiers, random forests (RF) [83] and support vector machines (SVM) [94].

RF is an ensemble learner, a method that generates many classifiers and aggregates their results. RF adds a layer of randomness to bagging by building large collection of de-correlated trees. RF will create multiple CART-like trees, each trained on a bootstrap sample of the original training data and searches across a randomly selected subset of input variables to determine the split. Each tree in RF will cast a vote for some input x, then the output of the classifier is determined by majority voting of the trees. Since the focus of this chapter is on features rather than on classifiers themselves, we refer the reader to chapter 5 for more details on RF and SVM.

We performed the classification using R, which is an open source statistical software. We used R *randomForest* and *SVM* (e1071) packages. The parameters to the RF were as follows: number of trees (ntree) was set to 500. Overall, the number of trees didn't seem to influence the classification results. The number of variables randomly sampled as candidates at each split (mtry) is equal to the square root of the number of features. Since in our case we have 262 features, mtry was consequently set to 16.

For SVM we used a linear kernel, termination criterion (tolerance) was set to 0.001, epsilon for the insensitive-loss function was 0.1 and the regularization term (cost) was 1.

6.6. Experiments

We tested both classifiers, RF and SVM, with $N = 10{,}000$ patients extracted from HCUP2005, $N_d = 5{,}000$ that had the disease and $N_n = 5{,}000$ that didn't. Out of the 10,000 samples, 7,000 were used for training and 3,000 for testing. Obviously, we excluded the target disease from the diagnosis set of the N_d patients that had it. We performed the same experiment for three diseases (first 3 lines in Table 6.1):

hypertension, diabetes mellitus and arteriosclerosis. The results obtained are showed in the next section.

6.7. Results

The classification results, area under the curve (AROC) and receiver operating characteristic (ROC) curve, for diabetes are shown in Table 6.2 and Fig. 6.3.

Table 6.2. AROC results for diabetes prediction

	Crisp Features	Fuzzy Features
RF	0.9524	0.9996
SVM	0.8567	0.981

From Table 6.2 we see that the use of fuzzy features lead to a important (4-13%) improvement in the ROC curves for both classifiers. This can be also observed in Fig. 6.3, below.

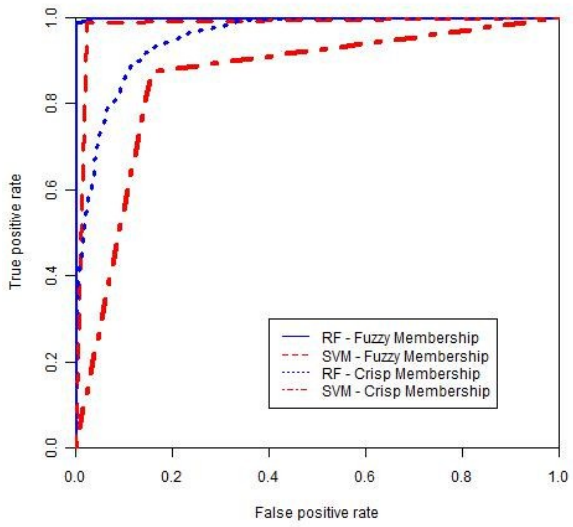


Fig. 6.3. ROC curves for diabetes prediction obtained with random forest (blue) and SVM (red).

The AROC improvement was smaller for RF than for SVM, since RF had already a good prediction performance dues to its builtin feature selection property.

The results obtained for atheoriosclerosis prediction are shown in Table 6.3 and Fig. 6.4.

Table 6.3. AROC results for artheriosclerosis prediction

	Crisp Features	Fuzzy Features
RF	0.9647	0.9995
SVM	0.8833	0.9737

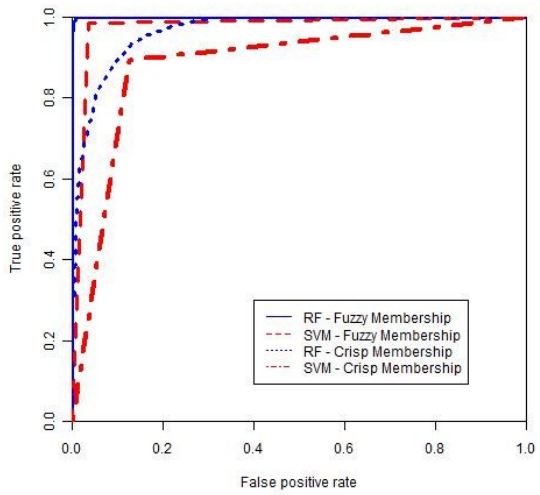


Fig. 6.4. ROC curves for ateriosclerosis prediction obtained using random forest (blue) and SVM (red)

For the atherioscelrosis prediction, too, we obtained a significant performance improvement (3-9%) when fuzzy features are used.

The results obtained for hypertension prediction are shown in Table 6.4 and Fig. 6.5. Again, a notable AROC improvement (5-13%) is obtained by using the fuzzy features instead of the crisp ones.

Table 6.4. AROC results for hypertension prediction

	Crisp Features	Fuzzy Features
RF	0.9454	0.9991
SVM	0.8537	0.989

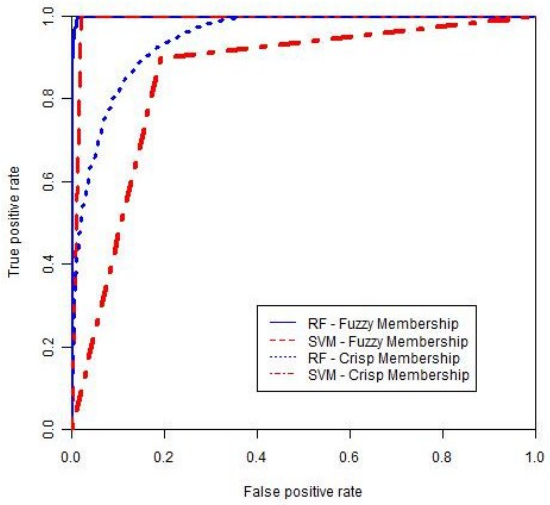


Fig. 6.5. ROC curves for hypertension prediction obtained using random forest (blue) and SVM (red)

6.8. Conclusion

In this chpater we presented a method for disease predicting using large public medical datasets. Disease prediction is important in a variety of applications such as health insurance, tailored health communication and public health. The presented method is based on employing ICD-9 diagnostic groups (DRGs) and demographics variables in conjunction with classification algorithms, such as SVM and RF. As opposed to using a crisp DRG membership for the ICD-9 codes, we introduced a novel fuzzy membership computed based on ICD-9 ontological similarity. The results presented on three different diseases and two classifiers show that the fuzzy features lead to an important improvement (between 3 and 13%) in prediction performance. The improvement is due to the fact that the fuzzy features capture the relationships between the DRG groups in the process of feature extraction.

This chapter ends the discussion on the disease classification topic and in the next chapter we move to a new topic related to NLP and ontologies and its application the nursing informatics.

CHAPTER 7

QUANTIFYING CARE COORDINATION DOSE USING NATURAL LANGUAGE PROCESSING AND DOMAIN SPECIFIC ONTOLOGY

The focus of this chapter is to quantify care coordination. It describes a method that employs Natural Language Processing (NLP) aided with a domain specific ontology to guide the extraction of care coordination activities and the focus upon which the specific activity was performed. Using the extracted activities, we evaluated the amount of care coordination received by every patient. We compared two groups of patients: Aging in Place (AIP) who received enhanced care coordination and Home Healthcare (HHC) who received traditional care. A care coordination ontology was built from the Omaha Case Management category. From the parsed notes of every patient, we mapped the extracted activities to the ontology. Based on the extracted activities, profiles were computed for each Omaha problem and patient. Using these profiles, we computed the care coordination dose for all patients. Constructing and testing the profiles was performed using 139,173 notes. Patients were tracked from the time they were admitted to AIP or HHC until they were discharged. We found that patients in AIP received a higher dose than HHC in most problems, with larger doses being given in AIP than in HHC in all four Omaha categories. We found “Communicate” and “Manage” activities are widely used in care coordination. That confirmed the expert hypothesis that care coordinators spent most of their time communicating about their patients and managing problems. Overall, nurses performed care coordination in both AIP and HHC, but the aggregated dose across Omaha problems and categories is larger in AIP.

7.1. Background

Coordination of healthcare services is vital for older adults who are extraordinarily vulnerable to the effects of illness, cognitive decline, disability, poverty, and limited social support [95]. Care coordination, which includes transitional care services, is seen as a way to improve healthcare, resulting in improved health, and reduced costs [96]. Over the last several decades, a number of care coordination models have been proposed, including transitional care models and Aging in Place [97]–[99]. Benefits of nurse care coordination include reductions in emergency room visits, increased patient survival post-hospitalization, fewer readmissions, reduced costs, and increased transitional care safety [100], [101]. As care coordination becomes more widely accepted and reimbursed, it will become increasingly important to be able to measure activities used in nurse care coordination.

Although the number of care coordination programs is growing and care coordination is generally viewed positively, there remain significant problems with care coordination measurement, including the identification of specific activities that constitute care coordination, and determining how much care coordination (dose) was delivered to each patient. A recent review of 96 measurement instruments reported that 88% of the care coordination measures relied on survey methods, of which 93% measured the aspects of communication, and 81% focused on the transfer of information [102]. However, measures that depend on survey methods do not fully capture the detailed processes used in care coordination or the activities used to coordinate care. Even well-defined frameworks, such as the Agency for Healthcare Research and Quality (AHRQ) Care Coordination Measures Atlas [103], do not describe detailed activities, nor

suggest specific measures directly linked to activities documented by practicing care coordinators.

The current state of the art in care coordination measurement relies on structured data fields, and custom built tools and surveys to capture the work of care coordinators in an abstract way. However, nurses documentation in electronic medical records (EMR) includes structured data, and narrative notes (free text) that describe care coordination activities. These narrative notes describe the work of care coordinators from the care coordinator's perspective.

It is time consuming and labor intensive to analyze narrative notes using traditional qualitative methods, and the number of notes that can be analyzed using such methods are limited. The use of natural language processing (NLP) and domain ontologies, can overcome this limitation due to its ability to mine large amounts of unstructured narrative notes. Many medical domain specific ontologies can be used to mine data, such as the Unified Medical Language System (UMLS) [104], OpenGALEN [105], SNOMED [106], in addition to biomedical terminology in general ontologies, such as WordNet [107]. In some cases, general ontologies may not be the right choice, since they are broad and lack domain specific concepts. This known problem has led researchers to build domain-specific ontologies. A domain-specific ontology containing concepts within a certain a field, nursing care coordination in our case, is a way to store specialized knowledge of a certain domain. An example of such ontology is an antimicrobial prescribing that contains 199 classes. In [108] Personally Created Cognitive Artifact (PCCAT) studies were used to identify and codify the knowledge nurses use in

assessing, diagnosing, planning, implementing, and evaluating patient care needs. The study of the PCCAT ultimately resulted in the development of a taxonomy and ontology.

The study and measurement of care coordination is relatively new, with the majority of research studies about care coordination occurring in the last decade [96]. This chapter reports on the development and use of a domain-specific ontology, which was built expressly to measure care coordination. It presents an approach for extracting care coordination activities from narrative notes, building profiles that describe care coordination activities, and finally using the profiles to quantify the care coordination dose received by each patient.

7.2. Methods

7.2.1. Setting and Sample

This study employed an analysis of an EMR data from 217 patients who were admitted to a home healthcare agency for enhanced care coordination through AIP [99], [101] and 691 who received traditional HHC without enhanced care coordination. Nurses in both AIP and HHC documented patient interventions in an EMR that used the standardized Omaha System, a taxonomy of nursing care that includes problem classification and intervention schemes. Institutional Review Board approval was obtained prior to the start of the study.

7.2.2. Dataset Description

In every patient visit, nurses identified, assessed, and documented patients' healthcare problems. The EMR had structured inputs allowing nurses to select from 42 Omaha problems, four Omaha categories; 1) Health Teaching, Guidance and

Counseling, 2) Treatments and Procedures, 3) Case Management, 4) Surveillance, and 75 Omaha interventions [109]. In addition to the predefined dropdown menus and checkboxes, nurses had the option of inputting free text narratives. Nurses used the comment box to document activities or findings that were not easily described in structured data.

The data contain a total of 139,173 narrative notes for the two groups of patients divided into four Omaha categories as shown in Table 7.1. Although, the majority of narrative notes were written in the category of *Surveillance*, experts who are doctoral prepared nurses and a social worker believed the Omaha *Case Management* category contained activities specific to care coordination. In order to obtain activities specific to care coordination, the Case Management category was used to extract care coordination activities, build the ontology, and construct the profiles. The number of interventions for each problem is similar in both groups, ranging between 3-5 interventions per problem (Table 7.1). Additionally, nurses in AIP documented more problems. In the case management category, the average AIP patient had about 40 narrative notes, compared to 11 notes in HHC.

Table 7.1. Characteristics of the dataset by Omaha category and group (AIP, $n = 217$; HHC, $n = 691$)

Category	Group	No. of Notes	Avg. No. Notes per Patient	Avg. No. of Problems per Patient	Avg. No. of Interventions per Problem
Health teaching, guidance and counseling	AIP	7,020	39	3.97	5.05
	HHC	21,047	33	3.35	5.22
Treatments and procedures	AIP	9,156	53	3.15	4.04
	HHC	17,593	29	2.47	4.04
Case management	AIP	6,311	40	3.17	3.83
	HHC	4,727	11	1.96	3.06
Surveillance	AIP	34,298	158	8.46	4.08
	HHC	39,021	58	5	3.95

A sample of the dataset for one patient is shown in Table 7.2, which contains eight short notes documented under case management category for the health care supervision problem.

Table 7.2. A sample patient dataset

Note No.	Intervention	Narrative note
1	Transportation	Informed client transportation had been arranged for 09-04-01 appoints…
2	Medical dental case	Client has a scheduled appointment with Doctor… on 07-19-01 at 2:15 PM. Client was informed about appointment.
3	Medical dental care	Contacted Doctor… office to confirmed appoint. on 12-22-00.
4	Medical dental care	Orders received for new wound care to the feet daily…
5	Transportation	Client was assisted down stairs per … staff to meet Bus yesterday to be taken to Doctor's… appoint…
6	Transportation	Attempted to reach …to arrange transportation for Doctor's appoint. on 12-22-00. No answer.
7	Transportation	…Arrangements for transportation had previously been made per…staff.
8	Transportation	Client was assisted down stairs per … staff to meet bus yesterday to be taken to Doctor's. appoint…

7.2.3. Care Coordination Ontology

A major goal of this study was to extract from nursing notes concepts of importance to care coordination in order to quantify care coordination dose. A care

coordination ontology was built to guide the concepts extraction, and those concepts were then used to build care coordination patient activity profiles and determine care coordination dose. Building a domain-specific ontology is an iterative process that is started by identifying the corpus of text on which the ontology will be derived. Only the 11,038 narratives under the Omaha category Case Management were used in the construction of the domain-specific ontology of care coordination. After dividing each narrative note into tokens or words using the NLTK tokenizer [110], we identified 16,000 terms. In order to identify terms that were most important to care coordinators, a frequency distribution was computed. The terms with frequency greater than 100 were provided to experts, including care coordinators and social workers, to help in conceptualizing the ontology.

A top-down approach to building the ontology was adopted. Five top level concepts were identified by the experts: 1) care coordination *activities* contained action verbs used by nurses when coordinating care, 2) care coordination *foci* represented the objects the activities acted upon, 3) *actors* contained people who interacted with care coordinators, 4) *problems* described specific patient problems identified by the care coordinator, and 5) *places* included locations where patients' resided when they received care. Candidate terms identified from Case Management were added to the appropriate class in the ontology. This process resulted in about 900 ontology concepts. Protégé [111], an open source software, was used for editing and modeling the ontology.

At this stage, the ontology had redundant, misplaced, misinterpreted concepts and missing synonyms; therefore, refinement and pruning was necessary. In order to address these issues, the context in which the terms were used in the narrative notes had to be

understood. For this task, clinicians on the research team who had extension care coordination experience examined every concept individually. They analyzed each concept by reading the notes in which the concept was found to identify the context of use including (a) what the concept referenced, (b) who were the actors involved, and (c) why was it relevant to care coordination. This process was performed iteratively, and it reduced the size of the ontology to 394 concepts. Of these 394 concepts, 66 were classified as care coordination activities, 156 as coordination focus, and the remaining concepts were distributed across the other three classes (Table 7.3). The top level concepts of care coordination activities and focus are shown in Fig. 7.1. Interested readers are encouraged to download the full ontology hosted on BioPortal (http://bioportal.bioontology.org/ontologies/NCCO).

Table 7.3. Summary of the ontology

Class	Number of Child Nodes[a]	Maximum Depth
Activities	66	3
Administer	1	
Assess	4	
Assist	1	
Attempt	1	
Communicate	16	
Identify	1	
Instruct	1	
Manage	34	
Monitor	5	
Obtain	1	
Order	1	
Foci	164	6
Ability	1	
Access	1	
Adherence	1	
Appointment	1	
Appropriateness	1	
Care	77	
Documentation	12	
Follow-up	1	
Information	3	
Resource	13	
Services	46	
Supervision	1	
Transportation	5	
Understanding	1	
Problems	91	6
Actors	54	3
Places	19	3

[a] The count includes the parent node.

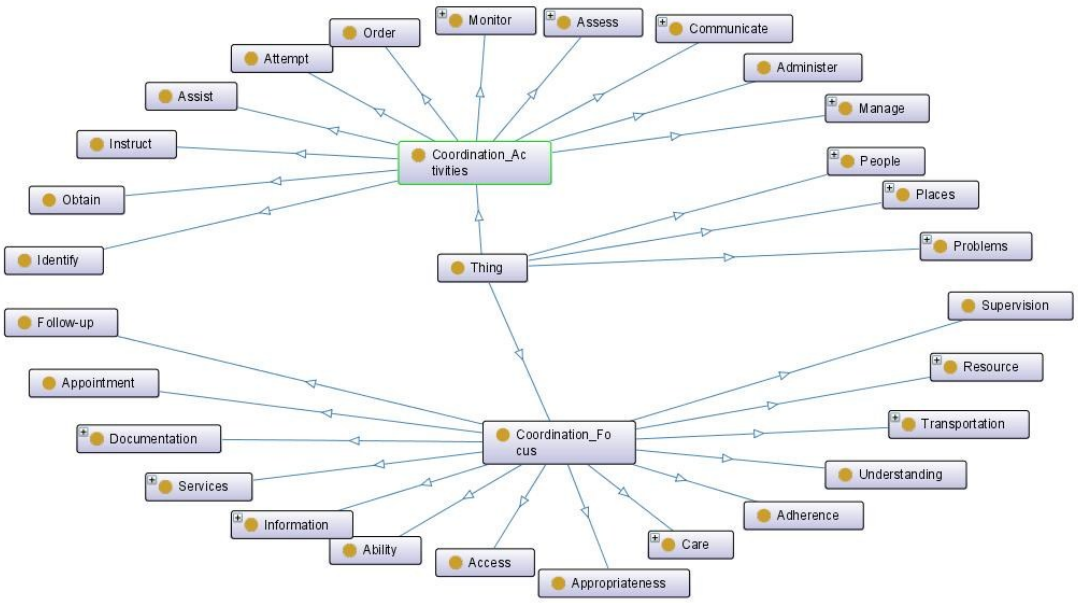


Fig. 7.1. Top level concepts of care coordination activities and focus

7.2.4. **Mining Nurses Narratives**

The ontology was then used to mine nurses notes and develop problem profiles describing care coordination activities. Each problem profile describes care coordination activities pertinent to a specific Omaha problem. There are N_A coordination activities as show in Fig. 7.1, where $N_A = 11$ and the activities are indexed from 1 to 11 as follows

$$A = \{Administer,\ Assess,\ Assist,\ Attempt,\ Communicate, Identify,$$
$$Instruct,\ Manage, Monitor, Obtain, Order\ \}.$$

However, the activities alone are not informative because they were out of context. For example the activity adjustment does not mean much by itself, but when combined with the foci of medication it describes work done by care coordinators to adjust medications. Not every activity was of relevance to the target problem. For that reason, the profile contains only those activities that co-occur with at least one care coordination

focus from our ontology. Also, in order to simplify the process, we used the top level concepts from coordination activities and focus (Fig. 7.1, Table 7.3). Meaning that all child nodes were collapsed under one of the activities in *A*. For example, "Adjust" is a child node of "Manage" (which has 34 child nodes) and "Medication" is a child node of "Care" (which has 77 child nodes).

We used a simple method for pairing care coordination activities and foci. That is, we paired every activity with all the child nodes of coordination focus. After performing sentence boundary disambiguation, tokenization, and stemming [110] on the narrative notes using NLTK, the text was searched using an activity recognizer based on regular expression. For instance, the following narrative contained one activity "continue medication". The output of the activity recognizer is the tagged narrative as shown below.

> *"Continues [ACTIVITY] to take her pain medication [FOCUS] prior to scheduled SNV's / dressing changes which are very painful for client. Continues to have leg spasms and tensed body with pain, and cries out with pain at times. Continues to report pain is like someone flaying her with a knife and # 10 pain rating on pain scale of 1-10."*

Notice that the term "Continues" in the second sentence will not contribute any information to the profile for two reasons. First, it did not co-occur with a focus and second it had already been counted once in the previous sentence. It is also possible that in a given note an activity co-occurs with more than one focus. In such case, only the focus that is closest to the activity is used, while the others are discarded. For instance, the following narrative has the activity "Take" followed by two foci which are

"Medication" and "Therapy". Hence, the focus "Therapy" will be ignored since "Medication" is closer to the activity "Take".

"Instructed not to allow pain to become severe and best to take [ACTIVITY] pain medication [FOCUS] one hour prior to therapy."

Clearly, this approach is simple but the results were promising. Using the search technique described above and with the aid of the ontology, the patient and problem profiles were computed as described in the next section.

7.2.5. **Problem Profiles**

In the Case Management category, nurse care coordinators used 32 ($N_{OP} = 32$), out of 42 possible problems. Every Omaha problem, denoted as i where $1 \leq i \leq N_{OP}$, under which some care coordination activities are documented consisted of a set of patients, $S_i = \{C_j\}$, who are identified as having this particular problem. $|S_i|$ is the number of patients having problem i. Sentence boundary disambiguation, tokenization, and stemming were performed on the notes associated with every intervention for patient $C_j \in S_i$. Then with the aid of the care coordination ontology, the occurrences of the top level of coordination activities in the notes were extracted. If we assume that a nurse used N_I^{ij} unique interventions to manage patient j who has problem i, then we can represent that patient problem in a matrix W_I^{ij} which has N_I^{ij} rows (one row for each intervention) and N_A columns (one column for every activity).

$$W_I^{ij} = \begin{bmatrix} w_{ij1}^{1} & \cdots & w_{ij1}^{a} & \cdots & w_{ij1}^{N_A} \\ w_{ijk}^{1} & \cdots & w_{ijk}^{a} & \ldots & w_{ijk}^{N_A} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ w_{ijN_I^{ij}}^{1} & \cdots & w_{ijN_I^{ij}}^{a} & \cdots & w_{ijN_I^{ij}}^{N_A} \end{bmatrix} \tag{7.1}$$

where the element at position (k, a) is $w_{ijk}^{a} \in [0\ ,1]$. If the care coordinator used the ath activity in the kth intervention then $w_{ijk}^{a} = 1$, otherwise $w_{ijk}^{a} = 0$. Formally,

$$w_{ijk}^{a} = \begin{cases} 1, if\ \exists A_a\ in\ I_{ijk} \\ 0, otherwise \end{cases} \tag{7.2}$$

Thusly, a patient-problem profile is constructed

$$P_{ij} = \left[w_{ij}^{1}, w_{ij}^{2}, \cdots, w_{ij}^{N_A}\right] \tag{7.3}$$

where w_{ij}^{a} represents the unique number of interventions that contained at least one occurrence of the activity A_a. As the activity A_a occurs in more interventions, the more significance it has in Omaha problem i for patient j. In other words, the patient profile for this problem is the column sum of W_I^{ij}, where w_{ij}^{a} is given by

$$w_{ij}^{a} = \sum_{k=1}^{N_I^{ij}} w_{ijk}^{a} \tag{7.4}$$

Using this process, profiles were constructed for all patients $C_j \in S_i$, where every patient profile P_{ij} is represented as a numerical feature vector of length N_A. Based on the individual patient profiles computed at (7.3) and (7.4), the general profile for problem i was computed. The general profile P_i is defined as the medoid profile among all patients $C_j \in S_i$. The medoid profile represents the patient that is closest to all other profiles in problem i. Since every patient has a numerical profile as in (7.3), we compute the Euclidean distance among all patients in problem i. The patient that minimizes the distance to all other profiles is the medoid that is selected to represent problem profile P_i. Formally, the profile P_i is

$$V_i = \min_{1 \le j \le |S_i|} \sum_{k=1}^{|S_i|} \left\| P_{ij} - P_{ik} \right\|_2^2 \tag{7.5}$$

where V_i is the index to the patient whose profile had the minimum distance to all other profiles. Then the profile P_i is presented as

$$P_i = P_{i,V_i} = \left[w_{i,V_i}^1, w_{i,V_i}^2, \cdots, w_{i,V_i}^{N_A} \right] \Leftrightarrow \left[w_i^1, w_i^2, \cdots, w_i^{N_A} \right] \tag{7.6}$$

which is again a numerical feature vector of length N_A that is the most representative of all patients in problem i. Fig. 7.2 summarizes the flow process which was used to extract the activity-focus pair from the narrative notes.

Example 1: Patient profile

Consider the sample patient data shown earlier in Table 7.2, a single patient with narrative notes associated with Omaha problem health care supervision in the case management category. The notes are also associated with two different interventions: transportation (T) and medical dental care (D). The resulting matrix W_I^{ij} is

$$W_I^{ij} = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix},$$

where the first row corresponds to intervention T, which contains the activities $P_{ij,T} = \{Assist, Attempt, Communicate, Manage\}$ and the second row is D intervention profile, $P_{ij,D} = \{Communicate, Manage, Order\}$.

Using (7.3) and (7.4), the resulting patient profile for healthcare supervision problem is

$$P_{ij} = [0,0,1,1,2,0,0,2,0,0,1]$$

or

$$P_{ij} = \{Assist, Attempt, 2 \times Communicate, 2 \times Manage, Order\}$$

P_{ij} shows that the activities "Assist", "Attempt" and "Order" were used in one intervention, "Communicate" and "Manage" were used in two interventions. The above process was performed for every patient and problem to calculate the profiles, which were used in quantifying the coordination dose.

7.2.6. **Care Coordination Dose**

Computing the care coordination dose is central to the goals of the study. To quantify the care coordination dose, patients were followed from their date of admission to AIP or HHC for 360 days, until their death, or until the end of the study (whichever condition occurred first). The narrative notes documented during that period for each patient was extracted and parsed, and finally used to compute a patient profile for every problem. Both the individual patient problem profiles and the general problem profiles represent the foundation for quantifying the dose. This section describes how to compute a numerical care coordination dose for every patient in every problem, and how to aggregate individual doses to compute a final care coordination dose.

When computing the dose for problem i, only those activities where $w_i^a > 0$ will be used in the calculation (only the activities that appear in the problem profile). The coordination dose of patient j in problem i is computed as a function of the profiles P_{ij} (7.3) and P_i (7.6) as follows

$$D_{ij} = \frac{\sum_{a=1}^{N_A} \delta_i^a w_{ij}^a}{\sum_{a=1}^{N_A} w_i^a}, \tag{7.7}$$

where

$$\delta_i^a = \begin{cases} 1, & if\ w_i^a > 0 \\ 0, & otherwise \end{cases}, \tag{7.8}$$

$w_{ij}^{a} \in P_{ij}$ and $w_{i}^{a} \in P_{i}$. If activity A_a occurs more frequently in the patient profile P_{ij} compared to the problem profile P_i $(w_{ij}^{a} > w_{i}^{a})$ then it is possible for $D_{ij} > 1$, which indicates that the patient received a high dose.

The cumulative or aggregated dose overall Omaha problems within some Omaha category for patient j is simply the sum of all individual problem doses for that patient and is given by

$$D_j = \sum_i D_{ij}. \tag{7.9}$$

Example 2: Patient dose

Given the profile for healthcare supervision $P_i = [0,0,0,0,1,0,0,0,0,1,0]$, δ_i, where $\delta_i = [0,0,0,0,1,0,0,0,0,1,0]$ and the patient profile for health care supervision from example 1, $P_{ij} = [0,0,1,1,2,0,0,2,0,0,1]$, the care coordination dose in healthcare supervision for that particular patient is computed using (7.7) as follows

$$D_{ij} = \frac{\delta_i^5 * w_{ij}^5 + \delta_i^{10} * w_{ij}^{10}}{w_i^5 + w_i^{10}} = \frac{2+0}{2} = 1$$

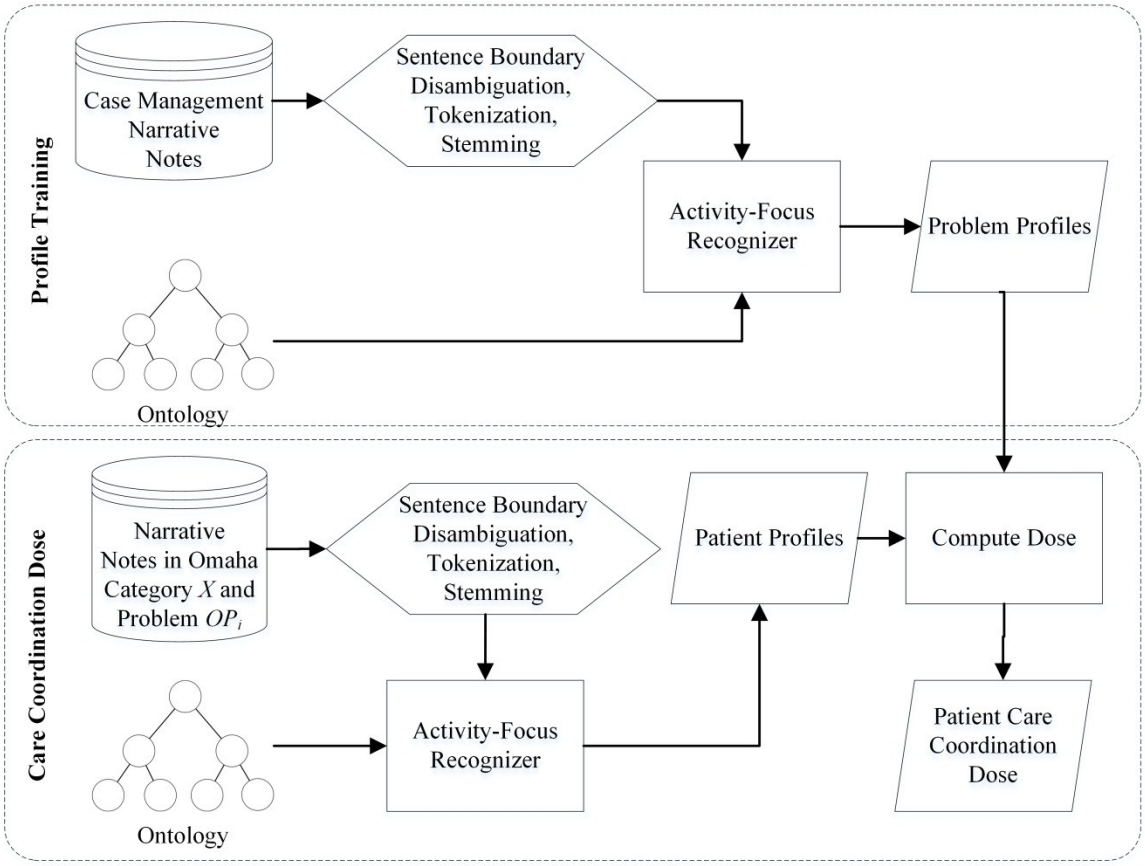


Fig. 7.2. A flow diagram of the activity-focus extraction process

7.3. Results

Based on patient problem profiles and general problem profiles, the care coordination dose for AIP and HHC patients were computed. From the problem profiles the activities "Communicate" and "Manage" were identified as the most widely used activities in care coordination, and appeared in 23 and 29 problem profiles, respectively, as shown in Fig. 7.3. This finding was not surprising to the clinical experts. In fact, it confirmed their hypothesis that nurse care coordinators spent most of their time communicating about their patients and managing their problems. On the other hand, "Assist" appeared in three profiles (role change, caretaking/parenting, and other

physiological), “Order” occurred in two profiles, “Assess”, “Instruct”, “Monitor” and “Obtain” appeared in one profile, while the activities “Administer”, “Attempt” and “Identify” are not represented in any of the 32 profiles. This finding is because those activities have very low prevalence among the patients, or they did not co-occur with coordination foci. Additionally, profiles are composed of the most commonly occurring activities, so not all activities in the ontology are represented in the profiles.

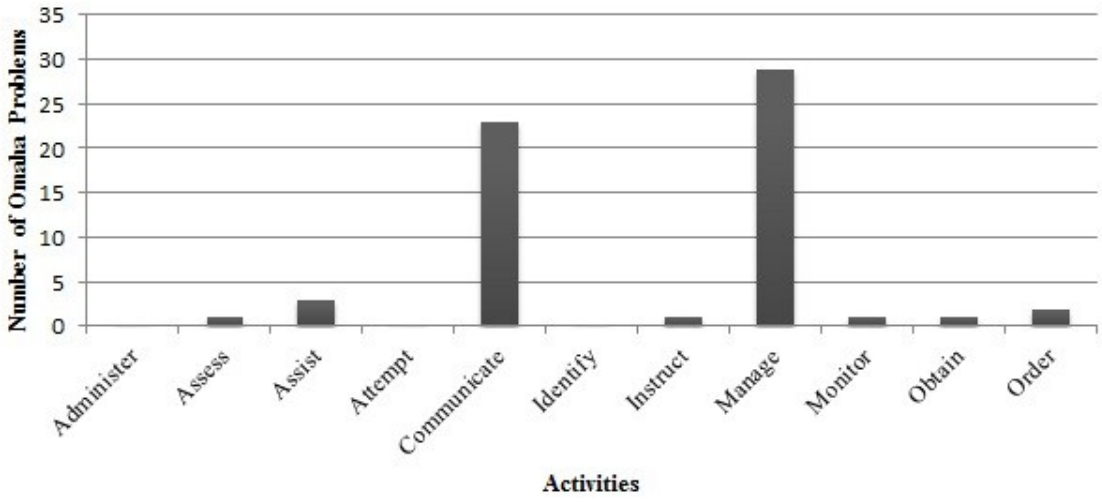


Fig. 7.3. Number of problem profiles representing each activity

To understand if the activities accurately described care coordination, it was necessary to also extract the focus of the activity. Fig. 7.4 displays the top 20 most occurring activity-focus pairs in AIP and HHC. These activities are related to communication and management, both of which had a large number of child nodes, 16 and 34, respectively. Overall, more care coordination activities were documented in AIP than HHC. Sixty-seven percent of activities that involve communication about durable medical equipment occurred in AIP and about 92% of medication management takes

place in AIP. Only two activities ("manage care" and "manage documentation") among the 20 considered occurred more often in HHC than in AIP.

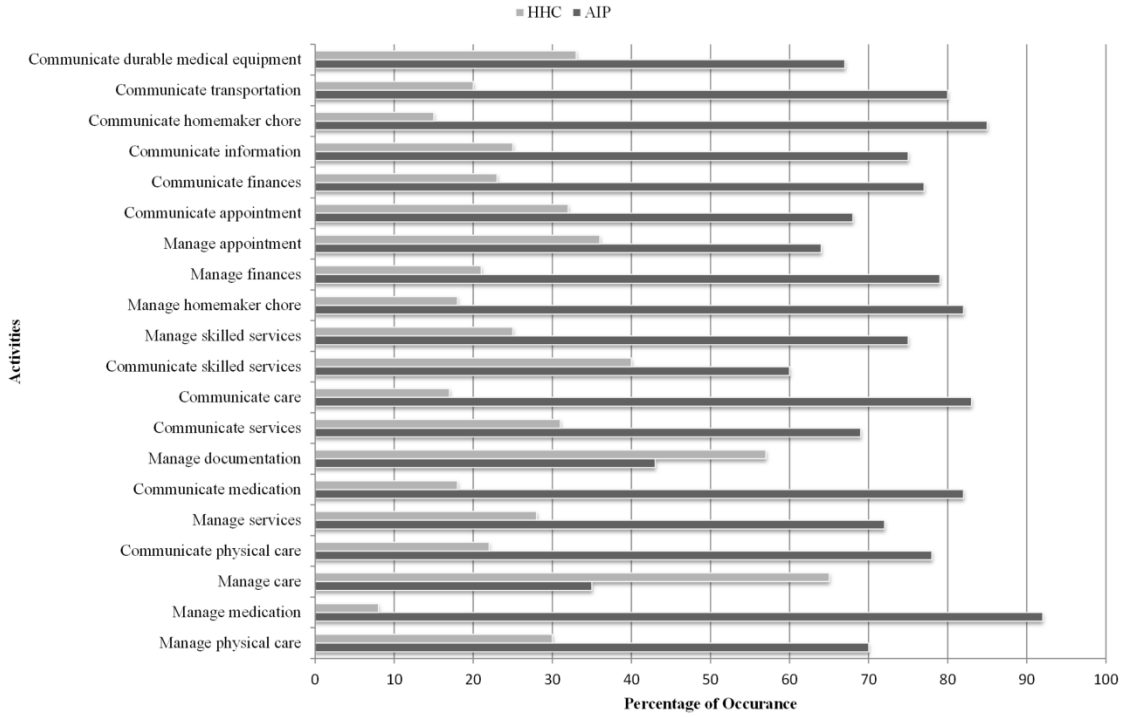


Fig. 7.4. Top 20 most frequent activity-focus pairs in Case Management and the percentage of occurrence in every group

There are also a greater number of unique activities documented in AIP than in HHC, which shows that care coordinators in AIP use various techniques to manage patients' care. Fig. 7.5 shows the number of unique activities documented in AIP and HHC for every Omaha problem in the Case Management category. We clearly see more diverse activities used in AIP. The Caretaking/parenting Omaha problem contains the most diverse activities, where 103 unique activities were used by care coordinators in AIP, while only 16 were used in HHC. Similarly, Circulation, Healthcare Supervision, Medication Regimen, Mental Health and Neuro-Muscular Skeletal show a large difference in the number of activities documented in both groups. There are also some

Omaha problems where HHC has no documented activities at all such as Cognition, Family Planning, and Pain. There is a handful of Omaha problems, where HHC patients have more activities, such as Communication with Community Resources, Digestion-Hydration and Personal Care.

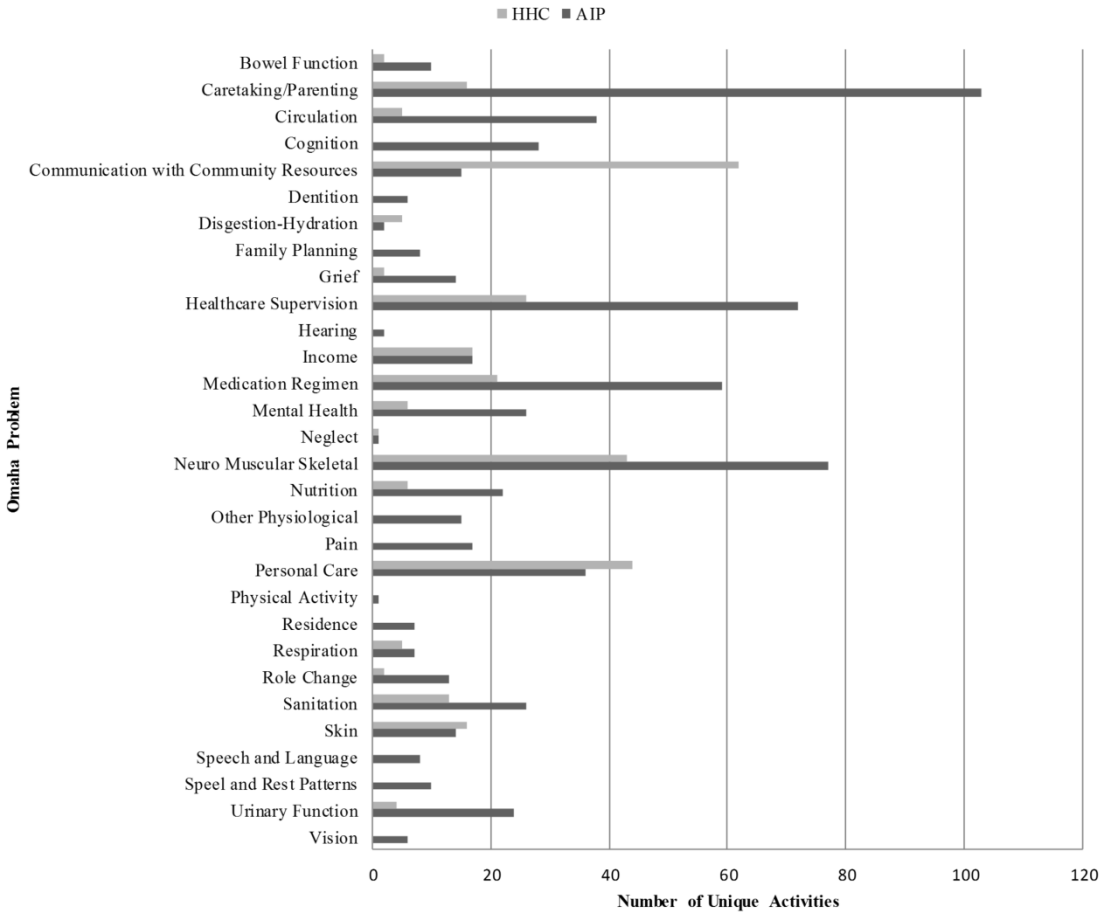


Fig. 7.5. Number of unique activity-focus pairs by Omaha problem in Case Management

Table 7.4 reports the different doses of care coordination used for AIP and HHC patients as measured in the case management category. This method detected differences between the AIP and HHC care coordination doses, with AIP having higher average doses of care coordination in all but three problems (Communication with Community

Resources, Skin, and Healthcare Supervision). A higher average dose for HHC patients can be expected in Communication with Community Resources and Skin Omaha problems, since more activities occur in HHC as shown in Fig. 7.5.

To further validate the problem profiles, we computed the dose for the patients using the same time period in the remaining three Omaha categories: Health Teaching, Guidance and Counseling, Treatments and Procedures, and Surveillance. Using the problem profiles that were initially developed in the case management category, we computed the aggregated dose for every patient within every category. The results are presented in the lower part of Table 7.4. We discovered that Care Coordination is not exclusive to the Case Management category; rather it exists in the other three categories. Also, the aggregated dose of care coordination in all categories is higher in AIP than HHC.

Table 7.4. Care coordination dose by problem and category in AIP and HHC groups

	Omaha Problem	Average Dose		Median Dose		Max. Dose	
		AIP	HHC	AIP	HHC	AIP	HHC
Dose by Omaha Problem in Case Management							
Environmental domain							
OP_1	Income	0.9	0.73	1	0.5	1	1
Psychosocial domain							
OP_4	Communication with community resources	0.64	0.67	1	0.5	1	1
OP_7	Mental health	1	1	1	1	2	1
OP_8	Caretaking/parenting	0.8	0.56	0.83	0.5	2	1
Physiological domain							
OP_{16}	Skin	0.71	1.13	1	1	1	2
OP_{17}	Neuro-muscular skeletal	0.81	0.54	1	0.5	1.5	1
OP_{19}	Circulation	0.88	0.4	1	0.5	1	0.5
OP_{22}	Urinary function	0.83	0.75	1	0.75	1.5	1
Health-related behaviors domain							
OP_{24}	Nutrition	0.86	0.67	1	0.5	1	1
OP_{27}	Personal care	0.95	0.91	1	1	2	2
OP_{30}	Healthcare supervision	0.53	0.66	0.5	0.5	1	1.5
OP_{31}	Medication regimen	0.86	0.71	1	0.5	1	1
Aggregated Dose by Omaha Category							
Health teaching, guidance and counseling		1.39	0.99	1	1	6	4
Treatments and procedures		1.21	0.92	1	1	4	2
Case management		2.2	1.15	1.5	1	11.53	3
Surveillance		1.4	0.9	1	1	7	3

7.4. Discussion and Conclusion

This chapter presented a novel approach to the measurement of care coordination dose. Care coordination relies heavily on communication with patients, family members, and healthcare team members, all of which is usually detailed in narrative notes. Using 139,173 narrative notes for building activity profiles, we measured the care coordination dose in both AIP and HHC for every Omaha problem. The dose is similar at the problem level, but the aggregated dose is higher in AIP. This finding is most likely because AIP used a greater number of problems, and nurse care coordinators in AIP documented more information in the narrative notes.

This is the first kind of work to describe the development of an ontology of care coordination that was subsequently coupled with NLP techniques to extract care coordination activities from free text. This work also describes a unique way to quantify (dose) how much care coordination patients received.

There are limitations to our method of measuring care coordination. The first limitation comes from the simplistic pairing of activities and foci. Not all pairs of activity focus are acceptable, for example, instead of "Adjust Note" using "Document Note" seems more appropriate. Alternatively, one can set rules for pairing the activity and focus, but that is a tedious process and not scalable as more concepts are added to the ontology. A better solution could be to use more sophisticated NLP techniques, such as part-of-speech tagging and chunking, to identify the activity (verb) and it is focus (noun). Another possible solution is to build a context free grammar specifically tailored for nurses' narratives.

Another limitation of our approach is the use of regular expression to extract the activity and focus. One problem of this approach is that if two activities and one focus occur in the same sentence, then both activities will be extracted along with the same focus, while in fact one activity should be linked to the focus and second activity should simply be ignored. A possible solution is to use NLP techniques as discussed earlier.

Also due to the large number of activities and foci in the ontology, we decided to represent the profiles using only the 11 main activities. The child nodes were collapsed under their corresponding parent activities. This technique allows the profiles to be more readable, compact, and easier to visualize. The drawback to this approach is that the profile is less granular and contextual. To demonstrate, suppose two profiles contain the

parent activity "communicate"; on the surface, these two profiles might look the same. But in reality, these two profiles are different because the first profile contains two of "communicate" child nodes, "report" and "call", while the other profile contains child nodes "discuss" and "ask".

That said, we believe that the method and the results presented will encourage the development of more specific and better approaches. For instance, the care coordination ontology, along with the extracted activity-focus pairs, can provide the foundation for building care coordination content free grammar that can describe the nursing language used in care coordination and be used in parsing activities more accurately.

This chapter presented the techniques used in parsing the narratives notes, building patient profiles, problem profiles, and deriving the dose of care coordination. We identified that "Communicate" and "Manage" activities are widely used in care coordination; confirming the expert hypothesis that nurse care coordinators spent most of their time communicating about their patients and managing problems. Overall, nurses in both AIP and HHC preformed care coordination, but the aggregated dose across Omaha problems and categories is larger in AIP.

CHAPTER 8

PERSPECTIVES AND OPEN PROBLEMS

With the advancement of technology, especially the widespread mobile devices and sensors that are available at low cost; we are collecting more data than ever. As products become more user centric, commercial companies have come to realize the value of data in the age of personalization and customization of products and services for their users and customers. Tailoring products became more widespread with the introduction of mobile devices. Also, the integration of products and services has worked greatly for the benefit of the users.

Fig. 8.1 gives an idea of the amount of data we produce [112]. We send millions of emails/second, 20 hours of video upload/minute, 50 million tweets/day, etc. With this huge amount of data traditional tools for data analysis are no longer sustainable and new, scalable and reliable algorithms are needed to handle the massive amount of data.

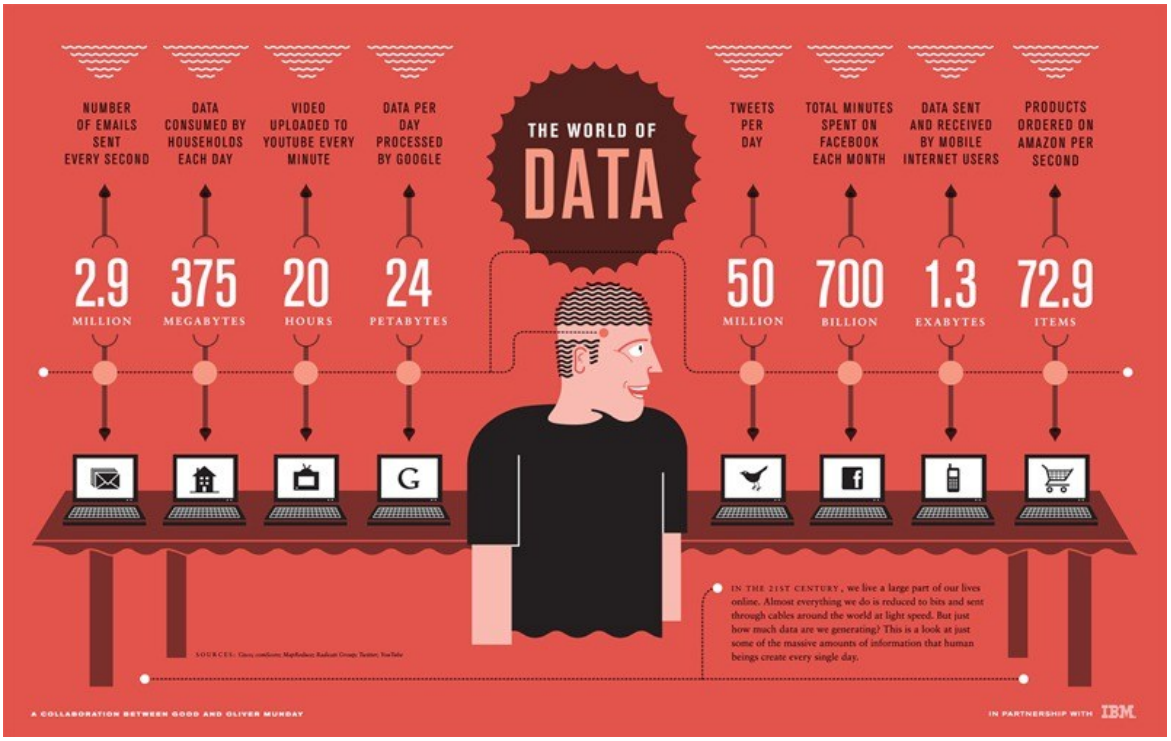


Fig. 8.1. The world of data [112]

Many of the pattern recognition tasks require cluster analysis that work on large data. There are many proposed object-based clustering algorithm that are designed to scale for large datasets, viz. *random sampling* and *extended* FCM [113], *single-pass* FCM [114], *online* FCM [115], *bit-reduced* FCM [116] and *kernel* based FCM [113].

Extending relational clustering algorithm for large datasets is by far much less popular than object-based clustering and that is probably because more object datasets exist than relational data. At the time of this writing we searched Google scholar for the terms (clustering “large data”), which returned about 124,000 results, while searching for (clustering “large relational data”) or (clustering “large dissimilarity data”) returned only about 200 results. Among those 200 results, we found few large relational data analysis

and clustering algorithms proposed such as eNERF [117], VAT for big data (bigVAT) [118] and topographic maps for large dissimilarity datasets [53].

8.1. Improved Non-Euclidean Relational Fuzzy c-Means (iNERF)

The current formulation of the iRFCM proposed in chapter 4 makes it hard to cluster datasets beyond $n > 1000$ for many reasons. The most obvious reason is the computational complexity of computing the eigenvalues of the dissimilarity matrix D. But it is even more computationally expensive if we choose a transformation such as the subdominant ultrametric, where we are required to compute the minimum spanning tree of D and the ultrametric distance. I performed a small experiment to show the running time for computing the minimum spanning tree, MST, and the subdominant ultrametric, SU, distance as n increases as shown in Fig. 8.2. Notice for SU we only show the running time for $n \leq 1000$. It was not feasible to show the running time for $n > 1000$ simply because the algorithm took a very long time to run and it was forced to terminate. We ran the experiments on a Dual Four Core XEON E5-2609 (2.4GHz, 10M cache) with 64GB, DDR3 RDIMM Memory, 1600MHz.

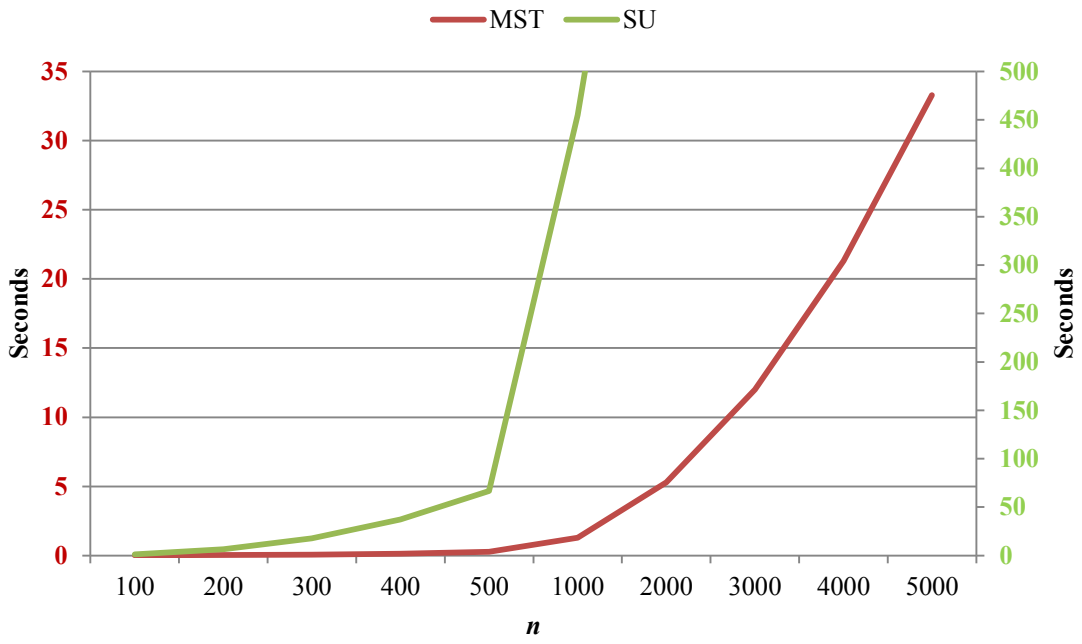


Fig. 8.2. Running time of the MST and SU as n increases

If we choose other types of transformation, we will face a new problem, we have to search for the correct parameter α that makes D Euclidean. These are some of the challenges iRFCM faces when dealing with large dissimilarity matrices. Hence, a solution is needed to make iRFCM work for datasets containing more than a handful of objects. An algorithm to alleviate this problem is the iNERF. The iNERF would be an extension to the well-known NERFCM algorithm [12], which avoids computing the eigenvalues, instead it computes an underestimated smallest eigenvalue which is then added to the off-diagonal elements of D. In addition to the NERFCM approach, the iNERF will contain additional pseudo-transformations that are to approximate the actual transformations that are carried out in iRFCM: the subdominant ultrametric, power fit, log fit and the exponential fit.

We have also seen the "black image of death" in chapter 4. The cause of this image is probably the amount of spreading we exercise on the matrix. The larger the additive constant the more spread out the objects become to the extent that the original dissimilarity matrix D gets distorted and we lose the original structure of the data causing the clustering algorithm to have difficulty finding the clusters. The "black image of death" only occurred in the β-spread transformation. In fact, β-spread is the most vulnerable to this problem because in this transformation an additive constant is added to all of the off-diagonal elements of D unlike the other four transformations.

We (James Bezdek, James Keller, Mihail Popescu and I) thought about this problem further and one possible solution is to avoid global spreading of the objects, like what we did with the five transformations. Instead, we should apply local transformation on only those objects that causes the algorithm to fail. This is our next mission, which will be followed by adapting NERFCM for big data.

8.2. Cluster Analysis for Big Relational Data

Datasets can be very large and they may exist in a high dimensional space. Through my collaboration with the School of Medicine and School of Nursing, it became more evident how unscalable the existing clustering algorithms are. How can one cluster 97,000 patients, where every patient is represented by a set of Activity of Daily (ADL) scores? Once has to compute the pairwise distances among patients to perform clustering. This process will result in a $97{,}000 \times 97{,}000$ relational data matrix. Large relational data introduces some problems to the clustering algorithms:

- Increases the computational complexity: as the number of objects and dimensions increase so does the computational complexity of the algorithm. In some cases,

one can use sampling or dimensionality reduction. However, the surge in the computational complexity is inevitable as we will produce and analyze more and more data.

- Increases the memory usage: this is a very critical issue as long as the cost of manufacturing memory stays high. No matter how much memory we have the datasets are always increasing in size, which means we have to upgrade the memory constantly. Take for example the ADL dataset, to accommodate relational matrix in memory one need about 35GB of RAM. Increasing the memory and disk space is merely a solution, it is a way to avoid the solution and ignore the problem. Here we are assuming that the data is loadable (Fig. 8.3). Hence, it can fit in memory. But if the data is unloadable (Fig. 8.3) then other techniques has to be developed.
- Increases the data sparsity: the sparsity of the data increases exponentially with the dimensionality of the input space. Meaning the data becomes less dense and objects become equidistant from one another [119], [120]. If there are no two objects that are close to each other then it is hard to find any clusters. In such case it is likely that clustering algorithms will fail to find clusters.

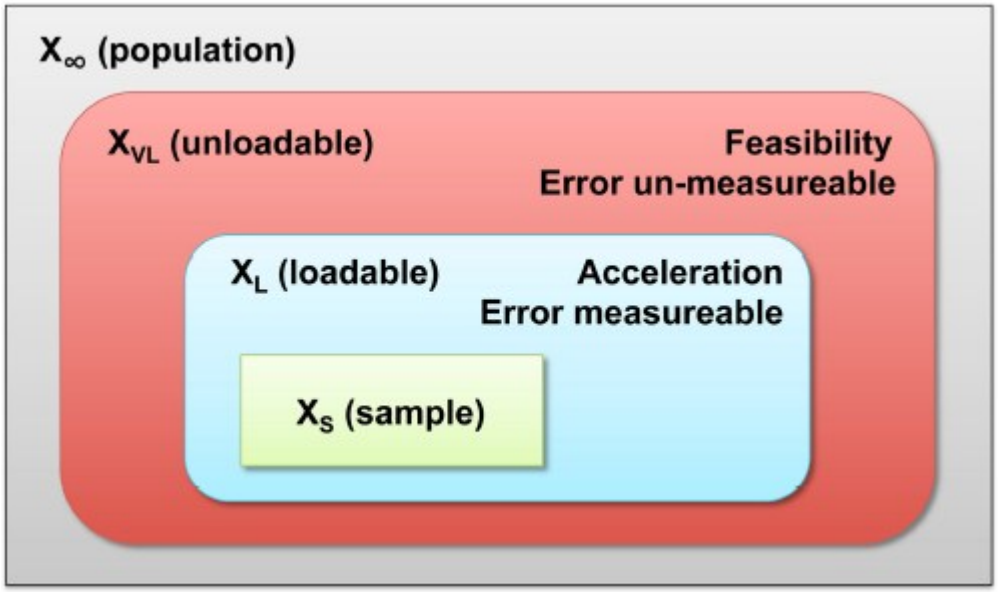


Fig. 8.3. Population X_{∞}, and samples X_{VL}, $X_L$and X_s [113]

8.2.1. **iNERFCM for Big Data**

That said, iNERF needs to scale for large and unloadable datasets. For unloadable relational data we can use random sampling and extension, or we can process the relational data in blocks separately and the last stage would be combining the results of all the blocks to come up with a final partition U.

8.2.2. **RFSOM for Big Data**

For RFSOM to create an impact, be scalable and more usable, it needs to handle large dataset. Numerous methods are proposed to handle large relational datasets for topographic maps [53], [121]. The authors in [121] carried on an experiment to cluster 77,977 protein sequences. The neurons were presented in vectorial form of 400 dimensions since the protein sequences can be converted into 400 dimensional dipeptide histograms. Those histograms were used to train the SOM and once the neurons were labeled by the protein sequences, the vectorial representation is abandoned. By not going

further into the experiment, one can see how this technique is not purely relational and it relies on vector representation of the objects. In [53] the authors propose a K-approximations algorithm to cluster large datasets. SOM is provided with a patch, P_t , which represents part of the relational matrix, R. The idea is to add the prototypes from processing the previous patch P_{t-1} to the current patch P_t. Every prototype is presented with k closest objects; therefore for c neurons we have $K = c * k$ approximations which play the role of a compressed representation of the already seen data points. N_{t-1} stores the indices of the K-approximations, $r(N_{t-1})$ denotes the inter-distances of points from the K-approximations produced at time $t-1$, $r(N_{t-1}, P_t)$ is the pairwise distances among the K-approximations resulted from the patch at $t-1$ and current patch. Note that $r(N_{t-1}, P_t)$ is computed on demand using some similarity measure. At time t a new relational matrix is created.

$$P^* = \begin{bmatrix} r(N_{t-1}) & r(N_{t-1}, P_t) \\ r(N_{t-1}, P_t)^T & P_t \end{bmatrix} \tag{8.1}$$

There are some potential drawbacks of this technique. The number of approximations depends on various things that must be kept in mind to provide a scalable SOM: it depends on the map size. SOM is not a clustering algorithm where one can provide c number of clusters ($c \ll n$). In fact, for best results it is recommended to initialize SOM with a large number of neurons to better reflect the topology of the data. As c increases, the objects will be distributed on a larger lattice and it is likely that the number of objects listed under a winning neuron will decrease. And as c increases so does the number of approximations that is required to compress the current patch. In this scenario the K-approximation will fail, in other words as $\lim_{c\to\infty} K = \lim_{c\to\infty} c * k = \infty$.

Example:

Assume RSOM is initialized with c neurons, where $c = 400$, and every neuron is approximated using k nearest objects and let $k = 5$. This results in $K = 400 * 5 = 2{,}000$ approximations. Also assume that the patch size is 1,000 objects. This means that at every iteration a new patch, P^*, of size $3{,}000 \times 3{,}000$ is created. Instead, if we use SOM with 4,000 neurons ($c = 4{,}000$), then the new patch size will be $21{,}000 \times 21{,}000$. As we can see this is not really a scalable algorithm even if we set $k = 1$. Consider WebSOM which is used to cluster 1,124,134 documents using 104,040 neurons [24]. K-approximations would certainly fail to cluster those documents. Note that the authors in [24] used a smaller map of size 18×45 to estimate a larger map of size 204×512.

Second, generating approximations for the empty or interpolating neurons only increases the size of the current patch. Interpolating neurons border the clusters and no objects are listed under those neurons. Thus, the interpolating neurons can be ignored which will decrease the patch size. Ignoring interpolating neurons might work for the RSOM since some neurons do not have receptive fields. However, this is not the case in RFSOM because none of the neurons are empty as discussed in chapter 4.

APPENDIXES

Appendix I. Distance between relational prototypes in proof

This proof is based on the weights computed in FSOM. Given the following weight update equation for FSOM, which we also presented in 2.13

$$m_i = \sum_{k=1}^{n}\sum_{j=1}^{c} u_{jk} h_{ij} x_k \Big/ \sum_{k=1}^{n}\sum_{j=1}^{c} u_{jk} h_{ij}$$

And the RFSOM coefficients update equation 2.16

$$\alpha_i = \sum_{j=1}^{c} h_{ij} u_j \Big/ \sum_{j=1}^{c}\sum_{k=1}^{n} h_{ij} u_{jk}$$

We will proof that $\|m_i - m_j\|^2 = \alpha_i^t R \alpha_j - \frac{\alpha_i^t R \alpha_i}{2} - \frac{\alpha_j^t R \alpha_j}{2}$

Proof:

$$\|m_i - m_j\|^2 = \left\| \frac{\sum_{a=1}^{n}\sum_{d=1}^{c} u_{da}^q h_{id} x_a}{\sum_{a=1}^{n}\sum_{d=1}^{c} u_{da}^q h_{id}} - \frac{\sum_{b=1}^{n}\sum_{e=1}^{c} u_{eb}^q h_{je} x_b}{\sum_{b=1}^{n}\sum_{e=1}^{c} u_{eb}^q h_{je}} \right\|^2$$

$$= \frac{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\left(u_{eb}^{q}h_{ie}\right)x_a^t x_b}{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\left(u_{eb}^{q}h_{ie}\right)}$$

$$- 2\frac{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\left(u_{eb}^{q}h_{je}\right)x_a^t x_b}{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\left(u_{eb}^{q}h_{je}\right)}$$

$$+ \frac{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\left(u_{eb}^{q}h_{ie}\right)x_a^t x_b}{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\left(u_{eb}^{q}h_{ie}\right)}$$

$$+ \frac{\sum_{a=1}^{n}\sum_{d=1}^{c}\left(u_{da}^{q}h_{id}\right)\|x_a\|^2}{\sum_{a=1}^{n}\sum_{d=1}^{c}\left(u_{da}^{q}h_{id}\right)}$$

$$- \frac{\sum_{a=1}^{n}\sum_{d=1}^{c}\left(u_{da}^{q}h_{id}\right)\|x_a\|^2}{2\sum_{a=1}^{n}\sum_{d=1}^{c}\left(u_{da}^{q}h_{id}\right)}$$

$$+ \frac{\sum_{a=1}^{n}\sum_{d=1}^{c}\left(u_{da}^{q}h_{id}\right)\|x_a\|^2}{2\sum_{a=1}^{n}\sum_{d=1}^{c}\left(u_{da}^{q}h_{id}\right)}$$

$$+ \frac{\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{eb}^{q}h_{je}\right)\|x_b\|^2}{\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{eb}^{q}h_{je}\right)}$$

$$- \frac{\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{eb}^{q}h_{je}\right)\|x_b\|^2}{2\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{eb}^{q}h_{je}\right)}$$

$$+ \frac{\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{eb}^{q}h_{ie}\right)\|x_b\|^2}{2\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{eb}^{q}h_{je}\right)}$$

$$= \frac{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\|x_a - x_b\|^2\left(u_{eb}^{q}h_{je}\right)}{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\left(u_{eb}^{q}h_{je}\right)}$$

$$- \frac{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\|x_a - x_b\|^2\left(u_{eb}^{q}h_{ie}\right)}{2\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{id}\right)\left(u_{eb}^{q}h_{ie}\right)}$$

$$- \frac{\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{jd}\right)\|x_a - x_b\|^2\left(u_{eb}^{q}h_{je}\right)}{2\sum_{a=1}^{n}\sum_{d=1}^{c}\sum_{b=1}^{n}\sum_{e=1}^{c}\left(u_{da}^{q}h_{jd}\right)\left(u_{eb}^{q}h_{je}\right)}$$

$$= \alpha_i^t R\alpha_j - \frac{\alpha_i^t R\alpha_i}{2} - \frac{\alpha_j^t R\alpha_j}{2}$$

Appendix II. HCUP data variables

Complete list of 126 HCUP data elements. The elements marked with "*" (rows 33-47) are the ones used in the classification as input variables.

1	AGE	Age in years at admission
2	AGEDAY	Age in days (when age < 1 year)
3	AMONTH	Admission month
4	ASOURCE	Admission source (uniform)
5	ASOURCEUB92	Admission source (UB-92 standard coding)
6	ASOURCE_X	Admission source (as received from source)
7	ATYPE	Admission type
8	AWEEKEND	Admission day is a weekend
9	DIED	Died during hospitalization
10	DISCWT	Weight to discharges in AHA universe
11	DISPUB92	Disposition of patient (UB-92 standard coding)
12	DISPUNIFORM	Disposition of patient (uniform)
13	DQTR	Discharge quarter
14	DRG	DRG in effect on discharge date
15	DRG18	DRG, version 18
16	DRGVER	DRG grouper version used on discharge date
17	DSHOSPID	Data source hospital identifier
18	DX1	Principal diagnosis
19	DX2	Diagnosis 2
20	DX3	Diagnosis 3
21	DX4	Diagnosis 4
22	DX5	Diagnosis 5
23	DX6	Diagnosis 6
24	DX7	Diagnosis 7
25	DX8	Diagnosis 8
26	DX9	Diagnosis 9
27	DX10	Diagnosis 10
28	DX11	Diagnosis 11
29	DX12	Diagnosis 12
30	DX13	Diagnosis 13
31	DX14	Diagnosis 14
32	DX15	Diagnosis 15
*33	DXCCS1	CCS: principal diagnosis
*34	DXCCS2	CCS: diagnosis 2
*35	DXCCS3	CCS: diagnosis 3
*36	DXCCS4	CCS: diagnosis 4

*37	DXCCS5	CCS: diagnosis 5
*38	DXCCS6	CCS: diagnosis 6
*39	DXCCS7	CCS: diagnosis 7
*40	DXCCS8	CCS: diagnosis 8
*41	DXCCS9	CCS: diagnosis 9
*42	DXCCS10	CCS: diagnosis 10
*43	DXCCS11	CCS: diagnosis 11
*44	DXCCS12	CCS: diagnosis 12
*45	DXCCS13	CCS: diagnosis 13
*46	DXCCS14	CCS: diagnosis 14
*47	DXCCS15	CCS: diagnosis 15
48	ECODE1	E code 1
49	ECODE2	E code 2
50	ECODE3	E code 3
51	ECODE4	E code 4
52	ELECTIVE	Elective versus non-elective admission
53	E_CCS1	CCS: E Code 1
54	E_CCS2	CCS: E Code 2
55	E_CCS3	CCS: E Code 3
56	E_CCS4	CCS: E Code 4
57	FEMALE	Indicator of sex
58	HOSPID	HCUP hospital identification number
59	HOSPST	Hospital state postal code
60	KEY	HCUP record identifier
61	LOS	Length of stay (cleaned)
62	LOS_X	Length of stay (as received from source)
63	MDC	MDC in effect on discharge date
64	MDC18	MDC, version 18
65	MDNUM1_R	Physician 1 number (re-identified)
66	MDNUM2_R	Physician 2 number (re-identified)
67	NDX	Number of diagnoses on this record
68	NECODE	Number of E codes on this record
69	NEOMAT	Neonatal and/or maternal DX and/or PR
70	NIS_STRATUM	Stratum used to sample hospital
71	NPR	Number of procedures on this record
72	PAY1	Primary expected payer (uniform)
73	PAY1_X	Primary expected payer (as received from source)
74	PAY2	Secondary expected payer (uniform)
75	PAY2_X	Secondary expected payer (as received from source)
76	PL_UR_CAT4	Patient Location: Urban-Rural 4 Categories
77	PR1	Principal procedure

78	PR2	Procedure 2
79	PR3	Procedure 3
80	PR4	Procedure 4
81	PR5	Procedure 5
82	PR6	Procedure 6
83	PR7	Procedure 7
84	PR8	Procedure 8
85	PR9	Procedure 9
86	PR10	Procedure 10
87	PR11	Procedure 11
88	PR12	Procedure 12
89	PR13	Procedure 13
90	PR14	Procedure 14
91	PR15	Procedure 15
92	PRCCS1	CCS: principal procedure
93	PRCCS2	CCS: procedure 2
94	PRCCS3	CCS: procedure 3
95	PRCCS4	CCS: procedure 4
96	PRCCS5	CCS: procedure 5
97	PRCCS6	CCS: procedure 6
98	PRCCS7	CCS: procedure 7
99	PRCCS8	CCS: procedure 8
100	PRCCS9	CCS: procedure 9
101	PRCCS10	CCS: procedure 10
102	PRCCS11	CCS: procedure 11
103	PRCCS12	CCS: procedure 12
104	PRCCS13	CCS: procedure 13
105	PRCCS14	CCS: procedure 14
106	PRCCS15	CCS: procedure 15
107	PRDAY1	Number of days from admission to PR1
108	PRDAY2	Number of days from admission to PR2
109	PRDAY3	Number of days from admission to PR3
110	PRDAY4	Number of days from admission to PR4
111	PRDAY5	Number of days from admission to PR5
112	PRDAY6	Number of days from admission to PR6
113	PRDAY7	Number of days from admission to PR7
114	PRDAY8	Number of days from admission to PR8
115	PRDAY9	Number of days from admission to PR9
116	PRDAY10	Number of days from admission to PR10
117	PRDAY11	Number of days from admission to PR11
118	PRDAY12	Number of days from admission to PR12

119	PRDAY13	Number of days from admission to PR13
120	PRDAY14	Number of days from admission to PR14
121	PRDAY15	Number of days from admission to PR15
122	RACE	Race (uniform)
123	TOTCHG	Total charges (cleaned)
124	TOTCHG_X	Total charges (as received from source)
125	YEAR	Calendar year
126	ZIPInc_Qrtl	Median household income quartile for patient's ZIP Code

BIBLIOGRAPHY

[1] A. Pantelopoulos and N. G. Bourbakis, "A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis," *IEEE Trans. Syst. Man, Cybern. Part C (Applications Rev.*, vol. 40, no. 1, pp. 1–12, Jan. 2010.

[2] "Apple's iWatch: what features are on our wishlist?," *The Guardian*, 04-Feb-2014.

[3] M. Skubic, G. Alexander, M. Popescu, M. Rantz, and J. Keller, "A smart home application to eldercare: current status and lessons learned.," *Technol. Health Care*, vol. 17, no. 3, pp. 183–201, Jan. 2009.

[4] D. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *AAAI workshop on knowledge discovery in databases*, 1994, pp. 359–370.

[5] J. J. Jiang and D. W. Conrath, "Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy," p. 15, Sep. 1997.

[6] L. Engelman and J. Hartigan, "K-means clustering," *W. J. Dixon, al.BMDP Stat. Softw.*, pp. 464–473, 1981.

[7] P. Rousseeuw and L. Kaufman, "Finding groups in data: An introduction to cluster analysis," *John, John Wiley Sons*, 1990.

[8] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Comput. Geosci.*, vol. 10, no. 2–3, pp. 191–203, Jan. 1984.

[9] R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering," *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 2, pp. 98–110, May 1993.

[10] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, Sep. 1967.

[11] R. J. Hathaway, J. W. Davenport, and J. C. Bezdek, "Relational duals of the c-means clustering algorithms," *Pattern Recognit.*, vol. 22, no. 2, pp. 205–212, Jan. 1989.

[12] R. J. Hathaway and J. C. Bezdek, "Nerf c-means: Non-Euclidean relational fuzzy clustering," *Pattern Recognit.*, vol. 27, no. 3, pp. 429–437, Mar. 1994.

[13] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit.*

Lett., vol. 31, no. 8, pp. 651–666, 2010.

[14] I. J. Sledge, J. C. Bezdek, T. C. Havens, and J. M. Keller, "Relational Generalizations of Cluster Validity Indices," *IEEE Trans. Fuzzy Syst.*, vol. 18, no. 4, pp. 771–786, Aug. 2010.

[15] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemom. Intell. Lab. Syst.*, vol. 2, no. 1–3, pp. 37–52, Aug. 1987.

[16] M. Davison, "Introduction to multidimensional scaling and its applications," *Appl. Psychol. Meas.*, 1983.

[17] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction.," *Science*, vol. 290, no. 5500, pp. 2319–23, Dec. 2000.

[18] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding.," *Science*, vol. 290, no. 5500, pp. 2323–6, Dec. 2000.

[19] M. Belkin and P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation," *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, Jun. 2003.

[20] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[21] L. Churilov and A. Flitman, "Towards fair ranking of Olympics achievements: the case of Sydney 2000," *Comput. Oper. Res.*, vol. 33, no. 7, pp. 2057–2082, Jul. 2006.

[22] W. Kurdthongmee, *Applications of Self-Organizing Maps*. InTech, 2012.

[23] G. Stegmayer, M. Gerard, and D. Milone, "Data Mining Over Biological Datasets: An Integrated Approach Based on Computational Intelligence," *IEEE Comput. Intell. Mag.*, vol. 7, no. 4, pp. 22–34, Nov. 2012.

[24] S. Kaski, T. Honkela, K. Lagus, and T. Kohonen, "WEBSOM – Self-organizing maps of document collections," *Neurocomputing*, vol. 21, no. 1–3, pp. 101–117, Nov. 1998.

[25] R. D. Pascual-Marqui, A. D. Pascual-Montano, K. Kochi, and J. M. Carazo, "Smoothly distributed fuzzy c-means: a new self-organizing map," *Pattern Recognit.*, vol. 34, no. 12, pp. 2395–2402, 2001.

[26] S.-C. Chi, R.-J. Kuo, and P.-W. Teng, "A fuzzy self-organizing map neural

network for market segmentation of credit card," in *SMC 2000 Conference Proceedings. 2000 IEEE International Conference on Systems, Man and Cybernetics. "Cybernetics Evolving to Systems, Humans, Organizations, and their Complex Interactions" (Cat. No.00CH37166)*, vol. 5, pp. 3617–3622.

[27] P. Vuorimaa, "Fuzzy self-organizing map," *Fuzzy Sets Syst.*, vol. 66, no. 2, pp. 223–231, Sep. 1994.

[28] M. Sap and E. Mohebi, "Hybrid Self Organizing Map for Overlapping Clusters," *Springer-Verlag Proc. CCIS*, 2008.

[29] E. Mohebi and M. N. M. Sap, "Hybrid Kohonen Self Organizing Map for the Uncertainty Involved in Overlapping Clusters Using Simulated Annealing," in *2009 11th International Conference on Computer Modelling and Simulation*, 2009, pp. 53–58.

[30] E. C.-K. Tsao, J. C. Bezdek, and N. R. Pal, "Fuzzy Kohonen clustering networks," *Pattern Recognit.*, vol. 27, no. 5, pp. 757–764, May 1994.

[31] M. Karabulut and T. İbrikci, "A fuzzy self-organizing map algorithm for biological pattern recognition," *Expert Syst.*, p. no–no, Oct. 2010.

[32] A. Golli, B. Conan-Guez, and F. Rossi, "A self-organizing map for dissimilarity data," *Classif. Clust. Data Min. Appl.*, pp. 61–68, 2004.

[33] A. Hasenfuss and B. Hammer, "Relational topographic maps," *Adv. Intell. Data Anal. VII*, 2007.

[34] T. C. Havens, J. M. Keller, and M. Popescu, "Computing with words with the ontological self-organizing map," *Fuzzy Syst. IEEE Trans.*, vol. 18, no. 3, pp. 473–485, 2010.

[35] I. Sledge, J. Bezdek, T. Havens, and J. Keller, "A relational dual of the fuzzy possibilistic c-means algorithm," in *International Conference on Fuzzy Systems*, 2010, pp. 1–9.

[36] A. Ultsch, "Emergence in self-organizing feature maps," ... *Self-Organizing Maps (WSOM'07), Bielefeld, ...*, 2007.

[37] M. Khalilia and M. Popescu, "Fuzzy relational self-organizing maps," in *2012 IEEE International Conference on Fuzzy Systems*, 2012, pp. 1–6.

[38] T. Kohonen, "The self-organizing map," *Neurocomputing*, 1998.

[39] T. Heskes, "Self-organizing maps, vector quantization, and mixture

modeling.," *IEEE Trans. Neural Netw.*, vol. 12, no. 6, pp. 1299–305, Jan. 2001.

[40] F. Moutarde and A. Ultsch, "U*F clustering: a new performant 'cluster-mining' method based on segmentation of Self-Organizing Maps," in *Workshop on Self-Organizing Maps (WSOM'2005)*, 2005.

[41] A. Ultsch, "Clustering with SOM: U* C," *Proc. Work. Self- Organ. Maps*, pp. 75–82 , 2005.

[42] E. Arsuaga Uriarte and F. Díaz Martín, "Topology preservation in SOM," *Int. J. Math. Comput. Sci.*, 2005.

[43] K. Kiviluoto, "Topology preservation in self-organizing maps," in *Proceedings of International Conference on Neural Networks (ICNN'96)*, vol. 1, pp. 294–299.

[44] A. Ultsch, "Maps for the visualization of high-dimensional data spaces," *Proc. Work. Self Organ. Maps*, 2003.

[45] J. Vesanto and E. Alhoniemi, "Clustering of the self-organizing map.," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 586–600, Jan. 2000.

[46] J. A. F. Costa and M. L. de Andrade Netto, "A new tree-structured self-organizing map for data analysis," in *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No.01CH37222)*, vol. 3, pp. 1931–1936.

[47] J. A. F. Costa and M. L. de Andrade Netto, "Cluster analysis using self-organizing maps and image processing techniques," in *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028)*, vol. 5, pp. 367–372.

[48] F. Meyer, "Topographic distance and watershed lines," *Signal Processing*, vol. 38, no. 1, pp. 113–125, Jul. 1994.

[49] L. Shafarenko, M. Petrou, and J. Kittler, "Automatic watershed segmentation of randomly textured color images.," *IEEE Trans. Image Process.*, vol. 6, no. 11, pp. 1530–44, Jan. 1997.

[50] C. Almanac, "Congressional Quarterly Almanac," *Washington, DC*, 1985.

[51] T. J. P. Hubbard, B. L. Aken, S. Ayling, B. Ballester, K. Beal, E. Bragin, S. Brent, Y. Chen, P. Clapham, L. Clarke, G. Coates, S. Fairley, S. Fitzgerald, J. Fernandez-Banet, L. Gordon, S. Graf, S. Haider, M. Hammond, R. Holland, K. Howe, A. Jenkinson, N. Johnson, A. Kahari, D. Keefe, S.

Keenan, R. Kinsella, F. Kokocinski, E. Kulesha, D. Lawson, I. Longden, K. Megy, P. Meidl, B. Overduin, A. Parker, B. Pritchard, D. Rios, M. Schuster, G. Slater, D. Smedley, W. Spooner, G. Spudich, S. Trevanion, A. Vilella, J. Vogel, S. White, S. Wilder, A. Zadissa, E. Birney, F. Cunningham, V. Curwen, R. Durbin, X. M. Fernandez-Suarez, J. Herrero, A. Kasprzyk, G. Proctor, J. Smith, S. Searle, and P. Flicek, "Ensembl 2009.," *Nucleic Acids Res.*, vol. 37, no. Database issue, pp. D690–7, Jan. 2009.

[52] M. Popescu, J. M. Keller, and J. A. Mitchell, "Fuzzy measures on the gene ontology for gene product similarity," *{IEEE/ACM} Trans. Comput. Biol. Bioinforma.*, pp. 263–274, 2006.

[53] B. Hammer and A. Hasenfuss, "Topographic mapping of large dissimilarity data sets.," *Neural Comput.*, vol. 22, no. 9, pp. 2229–84, Sep. 2010.

[54] P. Corsini, B. Lazzerini, and F. Marcelloni, "A new fuzzy relational clustering algorithm based on the fuzzy C-means algorithm," *Soft Comput.*, vol. 9, no. 6, pp. 439–447, Apr. 2004.

[55] J. C. Bezdek, E. C.-K. Tsao, and N. R. Pal, "Fuzzy Kohonen clustering networks," in *[1992 Proceedings] IEEE International Conference on Fuzzy Systems*, pp. 1035–1043.

[56] J. C. Bezdek and N. R. Pal, "An index of topological preservation and its application to self-organizing feature maps," in *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)*, vol. 3, pp. 2435–2440.

[57] T. Cox and M. Cox, *Multidimensional scaling*, 2nd ed. Chapman and Hall, 2000.

[58] K. V. Mardia, J. T. Kent, and J. M. Bibby, *Multivariate analysis, probability and mathematical statistics*. London Academic, 1979.

[59] J. Benasseni, M. B. Dosse, and S. Joly, "On a General Transformation Making a Dissimilarity Matrix Euclidean," *J. Classif.*, vol. 24, no. 1, pp. 33–51, Jun. 2007.

[60] S. Sattath and A. Tversky, "Additive similarity trees," *Psychometrika*, vol. 42, no. 3, pp. 319–345, Sep. 1977.

[61] E. Holman, "The relation between hierarchical and Euclidean models for psychological distances," *Psychometrika*, 1972.

[62] J. Dattorro, *Convex Optimization & Euclidean Distance Geometry*. Meboo

Publishing, 2005.

[63] R. Prim, “Shortest connection networks and some generalizations,” *Bell Syst. Tech. J.*, 1957.

[64] W. Fitch and E. Margoliash, “Construction of phylogenetic trees,” *Science (80-.).*, 1967.

[65] W. Fitch and E. Margoliash, “This week’s citation classic,” *Curr. Contents*, no. 27, 1988.

[66] L. Wang, U. Nguyen, J. Bezdek, C. Leckie, and K. Ramamohanarao, “iVAT and aVAT: enhanced visual analysis for cluster tendency assessment,” *Adv. Knowl. Discov. Data Min.*, vol. 6118, pp. 16–27, 2010.

[67] J. Huband and J. Bezdek, “VCV2–Visual cluster validity,” *Comput. Intell. Res. Front.*, 2008.

[68] E. Anderson, “The Irises of the Gaspe Peninsula,” *Bull. Am. Iris Soc.*, vol. 59, pp. 2 – 5, 1935.

[69] J. C. Bezdek, J. M. Keller, R. Krishnapuram, L. I. Kuncheva, and N. R. Pal, “Will the real iris data please stand up?,” *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 3, pp. 368–369, Jun. 1999.

[70] U. Kang, B. Meeder, and C. Faloutsos, “Spectral analysis for billion-scale graphs: Discoveries and implementation,” *Adv. Knowl. Discov. Data Min.*, vol. 6635, pp. 13–25, 2011.

[71] Y. Tan and G.-J. Zhang, “The application of machine learning algorithm in underwriting process,” in *2005 International Conference on Machine Learning and Cybernetics*, 2005, pp. 3523–3527 Vol. 6.

[72] P. L. Hebert, L. S. Geiss, E. F. Tierney, M. M. Engelgau, B. P. Yawn, and A. M. McBean, “Identifying persons with diabetes using Medicare claims data,” *Am. J. Med. Qual.*, vol. 14, no. 6, p. 270, 1999.

[73] E. L. Cohen, C. A. Caburnay, D. A. Luke, S. Rodgers, G. T. Cameron, and M. W. Kreuter, “Cancer coverage in general-audience and Black newspapers.,” *Health Commun.*, vol. 23, no. 5, pp. 427–35, Sep. 2008.

[74] Agency for Healthcare Research and Quality, “Healthcare Cost and Utilization Project (HCUP).” 2003.

[75] S. T. Moturu, W. G. Johnson, and H. Liu, “Predicting Future High-Cost Patients: A Real-World Risk Modeling Application,” in *2007 IEEE*

International Conference on Bioinformatics and Biomedicine (BIBM 2007), 2007, pp. 202–208.

[76] D. A. Davis, N. V Chawla, N. Blumm, N. Christakis, and A. L. Barabási, "Predicting individual disease risk based on medical history," in *Proceeding of the 17th {ACM} conference on Information and knowledge management*, 2008, pp. 769–778.

[77] D. H. Mantzaris, G. C. Anastassopoulos, and D. K. Lymberopoulos, "Medical disease prediction using Artificial Neural Networks," in *2008 8th IEEE International Conference on BioInformatics and BioEngineering*, 2008, pp. 1–6.

[78] W. Zhang, F. Zeng, X. Wu, X. Zhang, and R. Jiang, "A Comparative Study of Ensemble Learning Approaches in the Classification of Breast Cancer Metastasis," in *International Joint Conference on Bioinformatics, Systems Biology and Intelligent Computing, 2009. {IJCBS'09}*, 2009, pp. 242–245.

[79] F. Provost, "Machine learning from imbalanced data sets 101," *… AAAI'2000 Work. Imbalanced Data Sets*, 2000.

[80] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study.," *Intell. data Anal.*, 2002.

[81] J. Quinlan, "Bagging, boosting, and C4. 5," *Proc. Natl. Conf. Artif. …*, 1996.

[82] L. Breiman, *Classification and regression trees*. Chapman & {Hall/CRC}, 1984.

[83] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[84] C. Chen, A. Liaw, and L. Breiman, "Using random forest to learn imbalanced data," *Univ. California, Berkeley*, 2004.

[85] L. Breiman and others, "Manual-Setting Up, Using, and Understanding Random Forests V4.0," *At ftp://ftp. stat. berkeley. edu/pub/users/breiman*, 2003.

[86] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The elements of statistical learning: data mining, inference and prediction," *Math. Intell.*, vol. 27, no. 2, pp. 83–85, Jun. 2005.

[87] B. H. Menze, B. M. Kelm, R. Masuch, U. Himmelreich, P. Bachert, W. Petrich, and F. A. Hamprecht, "A comparison of random forest and its Gini

importance with standard chemometric methods for the feature selection and classification of spectral data," *{BMC} Bioinforma.*, vol. 10, no. 1, p. 213, 2009.

[88] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognit.*, vol. 30, no. 7, pp. 1145–1159, 1997.

[89] D. S. Palmer, N. M. O'Boyle, R. C. Glen, and J. B. O. Mitchell, "Random forest models to predict aqueous solubility," *J. Chem. Inf. Model*, vol. 47, no. 1, pp. 150–158, 2007.

[90] A. Liaw and M. Wiener, "Classification and Regression by randomForest," *R news*, 2002.

[91] W. Yu, T. Liu, R. Valdez, M. Gwinn, and M. J. Khoury, "Application of support vector machine modeling for prediction of common diseases: the case of diabetes and pre-diabetes," *BMC Med. Inform. Decis. Mak.*, vol. 10, no. 1, p. 16, 2010.

[92] M. Khalilia, S. Chakraborty, and M. Popescu, "Predicting disease risks from highly imbalanced data using random forest.," *BMC Med. Inform. Decis. Mak.*, vol. 11, no. 1, p. 51, Jan. 2011.

[93] M. Popescu and D. Xu, *Data mining in biomedicine using ontologies*. 2009.

[94] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[95] R. Johnson and J. Wiener, "A profile of frail older Americans and their caregivers," 2006.

[96] P. F. Cipriano, K. Bowles, M. Dailey, P. Dykes, G. Lamb, and M. Naylor, "The importance of health information technology in care coordination and transitional care," *Nurs. Outlook*, vol. 61, no. 6, pp. 475–489, Nov. 2013.

[97] M. D. Naylor, D. A. Brooten, R. L. Campbell, G. Maislin, K. M. McCauley, and J. S. Schwartz, "Transitional care of older adults hospitalized with heart failure: a randomized, controlled trial.," *J. Am. Geriatr. Soc.*, vol. 52, no. 5, pp. 675–84, May 2004.

[98] E. A. Coleman, C. Parry, S. Chalmers, and S.-J. Min, "The care transitions intervention: results of a randomized controlled trial.," *Arch. Intern. Med.*, vol. 166, no. 17, pp. 1822–8, Sep. 2006.

[99] K. D. Marek, L. Popejoy, G. Petroski, D. Mehr, M. Rantz, and W.-C. Lin,

"Clinical outcomes of aging in place.," *Nurs. Res.*, vol. 54, no. 3, pp. 202–11.

[100] M. Camicia, B. Chamberlain, R. R. Finnie, M. Nalle, L. L. Lindeke, L. Lorenz, D. Hain, K. D. Haney, N. Campbell-Heider, K. Pecenka-Johnson, T. Jones, N. Parker-Guyton, G. Brydges, W. T. Briggs, M. C. Cisco, C. Haney, and P. McMenamin, "The value of nursing care coordination: A white paper of the American Nurses Association," *Nurs. Outlook*, vol. 61, no. 6, pp. 490–501, Nov. 2013.

[101] K. D. Marek, L. Popejoy, G. Petroski, and M. Rantz, "Nurse Care Coordination in Community-Based Long-Term Care," *J. Nurs. Scholarsh.*, vol. 38, no. 1, pp. 80–86, Mar. 2006.

[102] E. M. Schultz, N. Pineda, J. Lonhart, S. M. Davies, and K. M. McDonald, "A systematic review of the care coordination measurement landscape.," *BMC Health Serv. Res.*, vol. 13, no. 1, p. 119, Jan. 2013.

[103] K. McDonald, "Care coordination measures atlas," 2010.

[104] O. Bodenreider, "The Unified Medical Language System (UMLS): integrating biomedical terminology.," *Nucleic Acids Res.*, vol. 32, no. Database issue, pp. D267–70, Jan. 2004.

[105] A. L. Rector, J. E. Rogers, P. E. Zanstra, and E. Van Der Haring, "OpenGALEN: open source medical terminology and tools.," *AMIA Annu. Symp. Proc.*, p. 982, Jan. 2003.

[106] K. A. Spackman, K. E. Campbell, and R. A. Côté, "SNOMED RT: a reference terminology for health care.," *Proc. Am. Med. Informatics Assoc. Annu. Fall Symp.*, pp. 640–4, Jan. 1997.

[107] G. A. Miller, "WordNet: a lexical database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995.

[108] S. McLane, A. Esquivel, and J. P. Turley, "Developing a taxonomy and an ontology of nurses' patient clinical summaries.," *Stud. Health Technol. Inform.*, vol. 146, pp. 352–7, Jan. 2009.

[109] "Omaha System Intervention Scheme," 2013. [Online]. Available: http://www.omahasystem.org/interventionscheme.html. [Accessed: 30-Jan-2013].

[110] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. O'Reilly Media, Inc., 2009.

[111] M. Musen, M. Crubézy, R. Fergerson, and N. Noy, "The Protégé ontology editor and knowledge acquisition system," 2009.

[112] "IBM Unveils Breakthrough Software to Exploit Big Data," *Business Day*, Business Day, 2014.

[113] T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall, and M. Palaniswami, "Fuzzy c-Means Algorithms for Very Large Data," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, Dec. 2012.

[114] P. Hore, L. O. Hall, and D. B. Goldgof, "Single Pass Fuzzy C Means," in *2007 IEEE International Fuzzy Systems Conference*, 2007, pp. 1–7.

[115] P. Hore, L. O. Hall, D. B. Goldgof, Y. Gu, A. A. Maudsley, and A. Darkazanli, "A Scalable Framework For Segmenting Magnetic Resonance Images.," *J. Signal Process. Syst.*, vol. 54, no. 1–3, pp. 183–203, Jan. 2009.

[116] S. Eschrich, L. O. Hall, and D. B. Goldgof, "Fast accurate fuzzy clustering through data reduction," *IEEE Trans. Fuzzy Syst.*, vol. 11, no. 2, pp. 262–270, Apr. 2003.

[117] J. C. Bezdek, R. J. Hathaway, J. M. Huband, C. Leckie, and R. Kotagiri, "Approximate clustering in very large relational data," *Int. J. Intell. Syst.*, vol. 21, no. 8, pp. 817–841, Aug. 2006.

[118] J. M. Huband, J. C. Bezdek, and R. J. Hathaway, "bigVAT: Visual assessment of cluster tendency for large data sets," *Pattern Recognit.*, vol. 38, no. 11, pp. 1875–1886, 2005.

[119] *Advanced Techniques in Knowledge Discovery and Data Mining*. Springer, 2005, p. 272.

[120] A. Rajaraman and J. D. Ullman, *Mining of Massive Datasets*. Cambridge University Press, 2011, p. 326.

[121] T. Kohonen and P. Somervuo, "How to make large self-organizing maps for nonvectorial data," *Neural Networks*, vol. 15, no. 8–9, pp. 945–952, Oct. 2002.

CODE LISTING

RFSOM

RFSOM MATLAB toolbox is available on github repository.

URL: https://github.com/mohammedkhalilia/SOM

The online repository includes the source code, documentation and datasets. Furthermore, it does not only implement RFSOM, but also includes the following algorithms:

1. Online SOM
2. Batch SOM
3. Fuzzy Batch SOM
4. Relational SOM
5. Relational Fuzzy SOM

iRFCM

iRFCM MATLAB toolbox is available on github repository.

URL: https://github.com/mohammedkhalilia/iRFCM

Also, iRFCM repository includes source code, documentation and datasets. The repository implements both RFCM and iRFCM.

VITA

Mohammed A. Khalilia

The author, Mohammed A. Khalilia, was born in Nablus, Palestine in 1982. He attended the University of Missouri from 2001 to 2006 and received a Bachelor of Science in Computer Science in 2006. Immediately following his graduation from Mizzou, he started working toward a PhD in Computer Science at the University of Missouri-Columbia as well.

His research interests include theory and applications of computational intelligence – including fuzzy sets and neural networks – pattern recognition, natural language processing, informatics and ontologies. His work experience includes teaching assistance positions in courses such as Java, Operating Systems, Engineering of the World Wide Web, Senior Capstone I and II. In 2011 he worked as a graduate research assistant (GRA) on a grant, titled "Small Computational Algorithms for Predictive Health Assessment," funded by the National Science Foundation, with Dr. Mihail Popescu as principle investigator. Then in 2011-2014, he worked as a GRA with the principle investigator Lori Popejoy on a National Institute of Health funded grant, titled "Care Coordination For Older Adults: Process, Outcomes, and Cost."

Mohammed is currently a health information analyst at the University of Missouri, Institute for Clinical and Translational Sciences. This is part of a $13.3 million grant, funded by the federal government, titled "Leveraging Information Technology for Hi-Tech and Hi-Touch" Care.