EFFECTIVE LEARNING WITH HETEROGENOUS, NOISY, MULTI-RELATIONAL HEALTHCARE DATA

by

Devendra Singh Dhami

APPROVED BY SUPERVISORY COMMITTEE:

Sriraam Natarajan, Chair

Vibhav Gogate

Nicholas Ruozzi

Gautam Kunapuli

David Page

This dissertation is dedicated to my parents and my wife

who have been the pillars of strength throughout my ardous journey

of realizing an important dream

EFFECTIVE LEARNING WITH HETEROGENOUS, NOISY, MULTI-RELATIONAL HEALTHCARE DATA

by

DEVENDRA SINGH DHAMI, BE, MS

DISSERTATION

Presented to the Faculty of

The University of Texas at Dallas

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY IN

COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT DALLAS

May 2020

ACKNOWLEDGMENTS

First and foremost, I wish to convey my deepest gratitude and a very special thanks to my advisor, Dr. Sriraam Natarajan, for his unending support and guidance. He is the utmost reason that I got interested in the field of artificial intelligence and machine learning as well as research in general. My first brush with machine learning was his "Applied Machine Learning" class I took in 2014 in Indiana University and even now remains the favorite class of my educational career. His passion for teaching and research has been one of the foremost reasons for me pursuing a career in academia. The most important principle that I have learned from him is that *"although having good papers is important, being a good scholar is more important"*. He has also taught me to take failure in the correct spirit, to learn from it and finally convert the failure into success. I intend to emulate these principles in all my future endeavors.

If there is someone who has contributed the most in making my journey a success and supporting me throughout this bumpy ride, they are my parents and my wife. My parents are my constant source of strength. My father, a PhD himself, is someone I have grown up looking up to and emulating and my decision of pursuing a PhD was a step in that direction. I have seen my mother work very hard without expecting anything in return and this selflessness is something that I have tried and incorporate in my life, professional and otherwise. My wife has been a constant inspiration for me and has been with me in both thick and thin. She has stuck with me and lifted me from the darkest of times and has always reminded me that there is no replacement for hard work and is an unsung hero of this dissertation.

I would like to thank all of my advisory committee members, Dr. Sriraam Natarajan, Dr. Vibhav Gogate, Dr. Nicholas Ruozzi, Dr. Gautam Kunapuli and Dr. David Page, without whose help, support and guidance this dissertation would not have been possible. I would like to say a special thanks to Dr. Gautam Kunapuli; from discussing new research ideas for hours on end in his office to solving existing bottlenecks in the ongoing research, I have learnt a lot from him and collaborated

with him on multiple papers. His guidance has been one of the major contributors to their acceptance and publication in important conferences.

I would like acknowledge my peers in the StARLinG Lab (at UTD), who before being my colleagues, are my closest friends. They have always been by my side, both personally and professionally, giving me vital feedback on the papers that were submitted as well as several personal matters. Aside from those, we have had the opportunity to have very motivating research discussions and to collaborate on several research ventures.

I would also like to acknowledge Dr. David Page (Duke University) with whom I have had the honor of collaborating on various research papers and his feedback on all these research problems have been of tremendous help. I have studied and followed his work extensively which has always served as a great source of inspiration for me. He has always been appreciative of the work and thought process being put in the research papers and has always acted as a morale booster.

I would also like to thank Dr. Ameet Soni (Swarthmore College) who has always been a patient listener and played a big role in getting my first research work published. Whenever I have spoken to him I have come out richer with new ideas. He is very enthusiastic about solving new problems and I have tried to incorporate that enthusiasm during all the research papers, right from the concept induction till the submission of the paper.

I would like to thank all the staff of the Computer Science Department at The University of Texas at Dallas for giving me one less reason to worry about and helping me with the administrative process in the smoothest fashion, be it for travel or for academic formalities. I would also like to thank my advisor and the department for giving me the opportunity to speak at the UTD machine learning conference on a couple of occasions. These opportunities gave me a lot of exposure about facing a big crowd and helped me a lot in networking with like minded people from the academic community as well as from the industry. It helped me gain a deeper respect for the strides machine learning is taking in enhancing the everyday life of people around the world and also gave a deeper

understanding of the importance of keeping academia and industry abreast of each other's advances. A special mention is necessary for the staff and professors at IUB as well as the city of Bloomington, Indiana, where I spent more than half of my graduate life. It is a beautiful place to live in and Indiana University and especially Bloomington will always hold a special place in my heart.

Finally, I gratefully acknowledge the support of National Institute of Health (NIH) grant no. R01 GM097628 and DARPA Minerva award FA9550-19-1-0391. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the NIH, DARPA or the US government.

March 2020

EFFECTIVE LEARNING WITH HETEROGENOUS, NOISY, MULTI-RELATIONAL HEALTHCARE DATA

Devendra Singh Dhami, PhD
The University of Texas at Dallas, 2020

Supervising Professor: Sriraam Natarajan, Chair

With the rise in development of machine learning models, a lot of progress has also been made in their applications to real world problems. Healthcare forms one of the most critical area for machine learning research as it directly impacts the life of the general population. Although a lot of work has been done and is continuing being done in the area of machine learning for healthcare, a variety of open problems on handling the underlying noisy structure of the data and multi-modality of data, still exist.

Classical machine learning requires the data to be in the form of a flat feature vector but with real world data and especially healthcare, this is seldom the case. The data is almost always multi-modal i.e. consists of different data types such as, relational (graph), images and text, to name a few. Naturally, the machine learning models being developed for healthcare are thus expected to take advantage of the varying types of data provided to it and learn more effectively.

Naive solutions take advantage of a large amount of available data to learn robust models. The fallacy of such models is generalization, since in healthcare domain the model can be posed with unseen tasks, such as identifying a new strain of virus, identifying a new drug etc., which can result in its failure. Thus taking advantage of human experts becomes an important part while developing models for such high impact tasks. Humans and machines can work in unison to handle the problem of generalization and can learn from one another.

This dissertation explores these challenges in depth and tries to address them individually before presenting a unified view of how to overcome these challenges within a machine learning framework. We analyse and present detailed solutions to each challenge and also show how they are interconnected. We understand the importance of learning machine learning models from different modalities of data and gain insights about why moving from a propositional setting to a more general relational setting is important. We also depict the importance of human experts and how they can be crucial in high impact tasks such as healthcare. One of the chapters clearly shows that even if the machine learning models are adversarial among themselves, using human expert as an ally in the adversarial setting can help in learning far more effective models. Thus this dissertation proposes several methods to overcome the most glaring problems in developing machine learning models for healthcare tasks and develop effective models. It also outlines several challenges that lie ahead and need to be overcome in order to realize the complete potential of the changes machine learning can bring in healthcare.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Machine learning has gone through significant transformations and has been given different definitions and different interpretations from its advent to the present time. Tom Mitchell describes machine learning as a *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"* (Mitchell, 1997). Right from 1952 when Arthur Samuel coined the term machine learning while creating a computer program for playing checkers to the the recent successes in self driving cars, there has been a huge advancement in the field of machine learning.

This emergence of machine learning has had a great deal of effect on taking major steps in solving several real world problems ranging from oil spill detection in satellite images (Kubat et al., 1998; Solberg et al., 1999), traffic accident prevention (Hu and Zheng, 2009; Tango and Botta, 2013), spam filtering and phishing attacks detection (Guzella and Caminhas, 2009; Salihovic et al., 2018) to several health care related tasks such as cancerous tumor detection (Havaei et al., 2017), Alzheimer's detection (Donini et al., 2016; Natarajan et al., 2014) and mortality prediction (Motwani et al., 2016; Deprez et al., 2017). With an exponential rise in the amount of data and knowledge available, the need for developing effective machine learning models to tackle real world problems such as heart attack (Jin et al., 2009; Thirumalai et al., 2017), ER admissions (Hong et al., 2018; Rojas et al., 2018) and mortality rate (Taylor et al., 2016; Veith and Steele, 2018), to name a few, is of paramount importance.

Although the amount of available data has been rising exponentially with each passing year, application of machine learning has not been on the desired scale due to several reasons such as:

1. Most real world data is relational in nature and representing such data as a flat feature vector, as required by classical machine learning, is often accompanied by loss of information. Thus, building a machine learning model with such ***multi-relational*** data is highly desirable.

2. Abstracting down from the representation issues, real world data is ***noisy*** i.e. there exist several spurious relations as well as missing data which can lead to underwhelming performance by a machine learning model.

3. The modality of data available, for the same real world task to be solved, can be highly ***heterogeneous*** in nature i.e the data modality can vary widely from imaging data to graph based data to a simple propositional format. Using these different modalities effectively and more importantly in unison to learn a machine learning model is still an open problem.

4. Human experts can bring in a wealth of information which has shown to improve the training of machine learning models. As Tom Mitchell writes, *"Bias free learning is futile"* (Mitchell, 1997), we plan to include human expert to augment machine learning models with knowledge (bias) to learn more effectively.

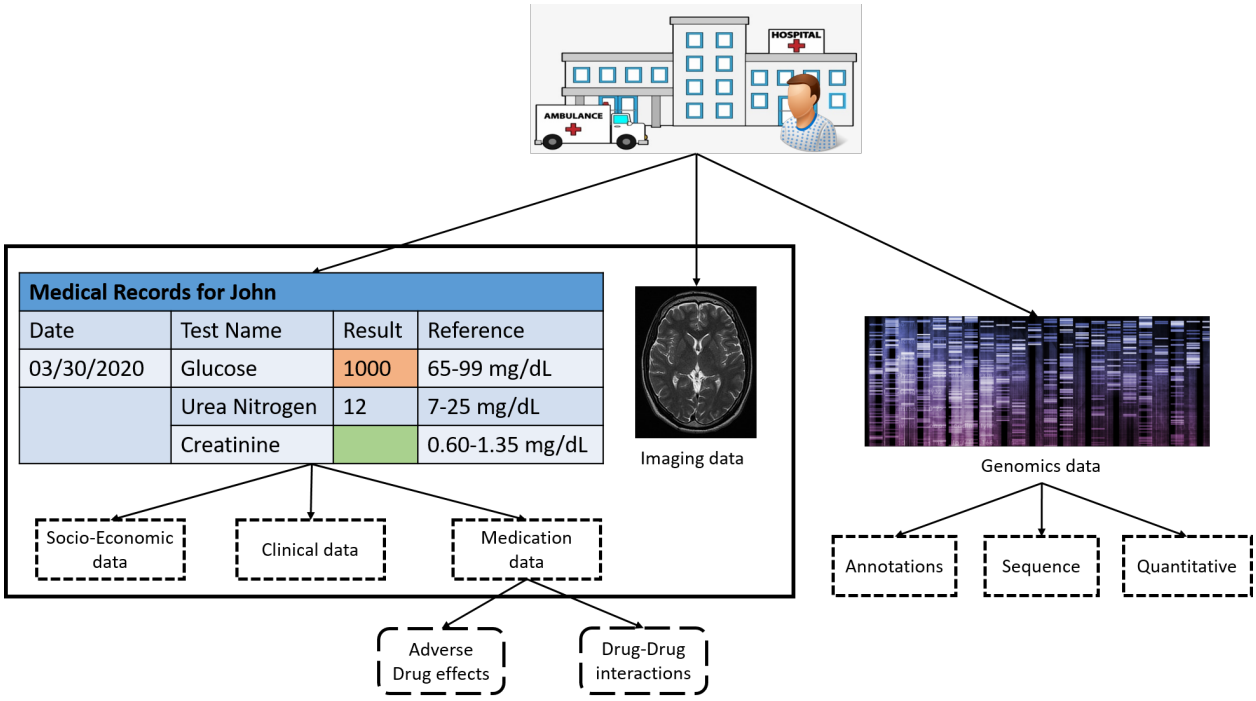


Figure 1.1: Example of heterogeneous, noisy and multi-relational data specified here in the domain of healthcare. The glucose value is clearly incorrectly entered, creatinine is a missing value (noisy data) and the patient data has multiple modalities (heterogeneous).

Given: Noisy, heterogeneous, multi-relational healthcare data.

To Do: Developing effective machine learning algorithms while exploiting human expertise.

To demonstrate the above roadblocks in application of machine learning methods to real world problems, consider a healthcare motivated scenario below:

Scenario 1: *Consider a patient, John, who is admitted to a hospital and is complaining of tremors in the fingers of his hand, having difficulty in sitting and walking in a correct posture. He is also complaining of suffering episodes of a chronic, abnormal stillness where there is a complete loss of movement. The physician have an initial diagnosis of Parkinson's disease and order MRI scans and several tests such as the Montreal Cognitive Assessment (MoCA), and tests to asses the motor functions such as speech, facial expression and tremors in the limbs (motor-UPDRS). The hospital also asks his children to fill out a form that gives personal details about John including his medical history and his smoking and drinking habits. The physician also collect oral swabs from him to generate his genomic profile for their records. Typically, using such information of a* ***heterogeneous*** *nature is vital for a human expert (physician) diagnose and detect diseases and prescribe medications or develop treatment plans. Such observations are at varied levels of representation; from low-level propositional values of clinical tests to structured databases containing* ***multi-relational*** *data with entities, relations and hierarchies. Also, they could be* ***noisy*** *due to possible errors in observation, data-entry process or implicit lack of samples.*

Figure 1.1 clearly depicts the multiple types of data that can be collected from a single patient in a single visit. This data can be divided into three categories at a high level: (1) medical data, (2) imaging data and (3) genomic data. Considerable amount of research in developing machine learning models for health care has resulted in several predictive models being developed (Choi et al., 2016, 2017; Weiss et al., 2012; Liu et al., 2019).

1.1 Learning from noisy, heterogeneous data

As depicted in figure 1.1, data collected from a patient can be, and generally is, heterogeneous in nature. Since classical machine learning models cannot handle these multiple modalities simultaneously, the data needs to be converted to a single format (such as extracting propositional features from images (Förstner, 1994; Chen et al., 2016) or extracting euclidean embeddings from graphical data (Bordes et al., 2011; Nickel et al., 2016)) which can then be fed to a machine learning model in different ways such as:

1. Learning multiple kernels from each of the heterogeneous data set and then combine them together to obtain a prediction using multiple kernel learning (Das et al., 2010; Senechal et al., 2012; Mariette and Villa-Vialaneix, 2018).

2. Combining the heterogeneous data sets in a Bayesian framework (Troyanskaya et al., 2003; Sokolov et al., 2013).

3. Using multi-view learning which is a machine learning framework that considers learning with multiple views to improve the overall generalization performance of a machine learning model (Sun, 2013; Zhao et al., 2017; Christoudias et al., 2012).

1.2 Learning from multi-relational data

As mentioned earlier, classical machine learning models require the data to be in a flattened propositional tabular format whereas most of the real world data is relational consisting of entities and relations between entities. Statistical relational learning (Getoor and Taskar, 2007; Raedt et al., 2016) combines the power of probabilistic models to capture uncertainty with logic/relational models to take advantage of the rich domain structure to handle complexity and is used for working with relational data directly without converting it to a propositional format that leads to loss of information. Several models have been developed such as markov logic networks (Richardson and

Domingos, 2006) (for undirected graphs), relational dependency networks (Neville and Jensen, 2007a) (for directed graphs) that can handle relational data. Moreover, boosted versions of these models have also been developed (Natarajan et al., 2015; Khot et al., 2013) to learn more complex relational features resulting in more effective models.

Another set of models map the relations and entities in the relational data into a low level representation based on translational distance models (Xiao et al., 2016; He et al., 2015) or compositional operators for entities and relations (Yang et al., 2015; Nickel et al., 2016) to perform various relational learning tasks such as link prediction and node classification.

1.3 Dissertation Statement

Different types of data can reveal different type of properties and patterns about the given task and thus it is very important to use these different set of data in order to learn better and more effective models. Our objective is to develop models that can take advantage of noisy, heterogeneous multi-relational data while using additional knowledge from the human (preferably an expert) for data generation in data scarce (due to privacy/access/cost issues) and knowledge rich domains.

1.4 Dissertation Contributions

A general overview of the type of the data handled in this thesis is shown in figure 1.2. This dissertation makes the following contributions:

I. ***Develop an interpretable machine learning model for predicting Parkinson's disease in patients using only the clinical data, also leveraging expert advice for selecting the best features for the models to learn from.***

II. ***Develop methods that take advantage of different types of data within the medication data (figure 1.1), such as***

   (i) Relationships between drugs and the corresponding enzymes,targets and transporters.

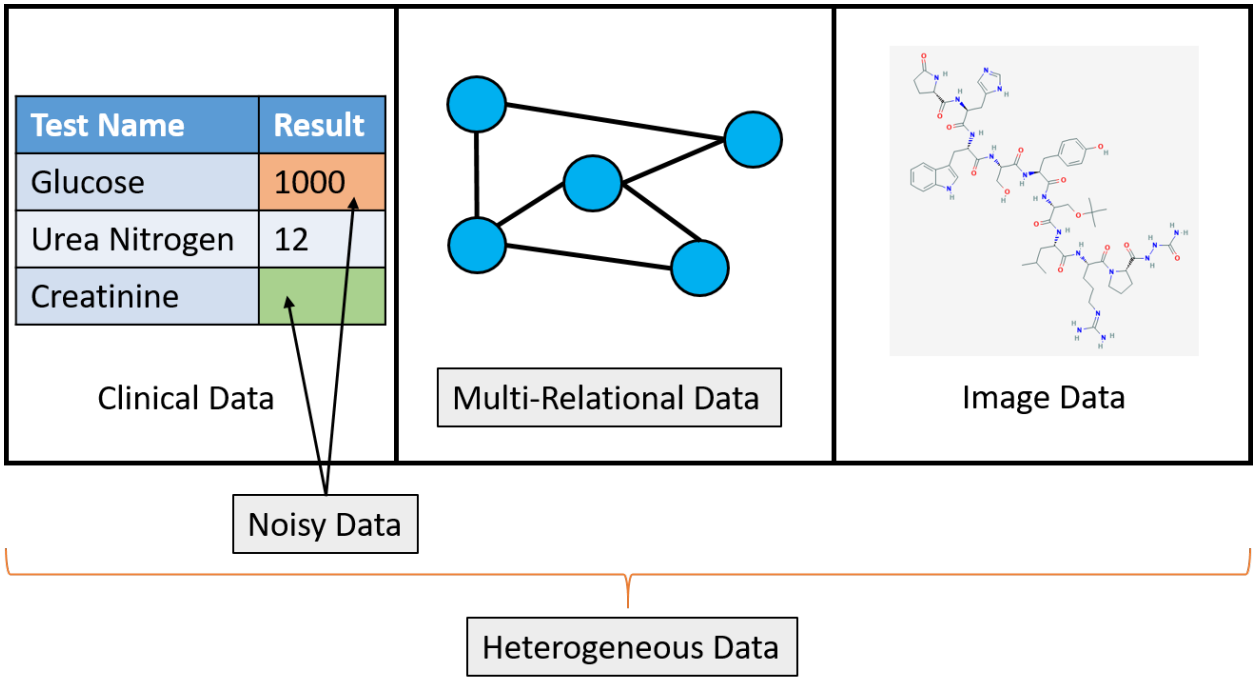


Figure 1.2: Example of the different modalities of data considered in this thesis.

(ii) Drug structure similarity data, such as molecular fingerprints and string representation of the drug molecular structure.

(iii) Drug structure images.

to predict and discover novel drug-drug interactions (DDIs) along with learning an effective representation for the different types of data. One major advantage of our methods is the ability to discover potential DDIs i.e. methods capable of knowledge refinement as well as knowledge discovery.

III. ***Develop methods to take advantage of local neighborhood structure in multi-relational data to avoid extracting similarity measures from given multi-relational data since using such similarities for model construction can result in a loss of information.***

1.5 Dissertation Outline

The dissertation has been divided into 3 high-level parts. Part I outlines our approaches to learn from noisy, heterogeneous data, Part II outlines our approaches to learn from multi-relational data and Part III highlights our approach to construct synthetic heterogeneous healthcare data which can otherwise be costly and difficult to access and obtain.

Chapter 2 presents the required technical background and related work for different problems that have been tackled in this thesis. It also positions our work in the context of recent advances in the current research directions. First we present the motivations behind solving problems in healthcare domain that more often than not consists of noisy and heterogeneous data and also discuss existing research. Next, we present the existing research on methods using multi-relational data to avoid information loss in various real world domains. We also present related literature on human knowledge augmented learning and deep adversarial networks to motivate the requirement of a generative model for healthcare related data generation.

Part I

Chapter 3 presents our first approach that uses only clinical data for identifying Parkinson's patients. The work takes advantage of an expert during the feature selection phase to determine the most important factors involved in patient identification and then uses a gradient boosting technique to come up with the an interpretable machine learning model for predicting Parkinson's disease in patients.

Chapter 4 presents our approach to learn from heterogeneous data to predict as well as *discover* new drug-drug interactions. Several machine learning models have been developed for predicting drug-drug interactions that take advantage of the wealth of bio-medical literature to mine these interactions but multitudes of data of different modalities are generally ignored. This chapter details

a novel method for combining different modalities together resulting in a better predictive model that is also capable of discovering new drug-drug interactions.

Part II

Chapter 5 describes a learning algorithm for relational local neighborhood-based approach that takes advantage of the fact that local relationships of entities in a graph can be generalized to a global representation of relationship *between* the entities. The main idea behind this model is that "nodes with shared neighbors are more likely to be similar and thus are more likely to have a link between them". We show that structure learning (here learning first order rules) for positive and negative examples in the data separately, using a kernel density estimation method, can result in a better exploitation of the search space.

Chapter 6 extends the framework of graph convolutional networks (GCN) to relational domains. Our framework ROCGCN initially uses a density estimation method to learn first order rules from positive and negative data separately and the creates the feature matrix by counting the number of satisfied groundings for each query example with respect to each first order rule learned. The distance matrix is learned by taking a pairwise euclidean distance between the features. Our work uses the distance matrix in GCN training instead of an adjacency matrix as usually required. Thus the learned rules are used as the observed layer leading to more abstract and rich latent layers.

Part III

Chapter 7 augments the generative machine learning framework of generative adversarial networks with human (expert) knowledge to generate synthetic data sets from real medical data sets keeping the intrinsic relationships between the individual features intact. Since access to real medical data is not readily available to researchers due to the presence of unique constraints on secondary use, our framework can generate synthetic data faithfully to overcome this problem.

CHAPTER 2

TECHNICAL BACKGROUND

This chapter provides required background on the various problems that this dissertation aims to address. We first describe the various approaches that have been taken in the literature for handling noisy, heterogeneous data along with some basic concepts that are essential in understanding the challenges involved in learning with such data with the focus on various healthcare tasks. Next, we present the extensive literature on learning from relational data, why is learning in relational domains difficult and how they form a major focus of this dissertation. Finally, we present and analyze the existing research that closely relates to the approaches discussed in the subsequent chapters.

2.1 Learning from Noisy and Heterogeneous data

Vikram Sarabhai, the father of the Indian space program had famously said *"He who can listen to the music in the midst of noise can achieve great things"*. This is inherently true in machine learning since most of the (real-world) data is highly noisy which can be due to several reasons (see figure 1.2). To understand these reasons more clearly, consider an extension of scenario 1:

Scenario 2: *After data has been collected from John the next step is to feed it into an electronic health record. The physician who is tasked with this data entry has been on call for the last 36 hours and thus, naturally, is very tired. While entering the test results into the system, a value for the glucose test was entered as 1000 instead of the true value of 100 and the value of creatinine was not entered thereby resulting in a missed value.*

The scenario above depicts a couple of most common ways in which the real data can be noisy, namely, missing data and incorrectly entered values. A large amount of work exists that deals with the noisy data (Yang et al., 2004; Sebban and Janodet, 2003; Van Hulse and Khoshgoftaar, 2009; Feng et al., 2018) in several noisy domains such as aerial imaging (Mnih and Hinton, 2012),

sentiment analysis (Gamon, 2004; Barbosa and Feng, 2010) and healthcare (Gamberger et al., 2000; Li et al., 2016).

Along with dealing with such noisy data, another big challenge that mars the large scale adaptation of machine learning methods in real world domains is that of heterogeneity or multi-modality of the available data for a single task. This challenge was depicted in scenario 1 where multiple types of data such as clinical data, image data and genomic data could be collected for a single patient in a single visit. A separate sub-area of machine learning called multimodal machine learning (Baltrušaitis et al., 2018) exists that deals with heterogeneous data for a single task in machine learning and consists of several classes of approaches ranging from combining or fusing multi-modal data effectively for taking advantages of information contained in all modalities (Srivastava and Salakhutdinov, 2012; Dähne et al., 2015; Ding et al., 2014) to aligning sub-elements from each modality with each other to abstract out more complex relationship among different modalities also referred to as manifold alignment (Wang and Mahadevan, 2009; Cui et al., 2014).

2.1.1 Drug-Drug Interaction Prediction and Discovery from Noisy, Heterogeneous Data

In our work for drug-drug interaction (DDI) prediction and discovery, that we define in chapter 4, we use three different kinds of data, namely, (1) graph data in the form of relationships of various drugs with different enzymes, transporters and targets, (2) molecular data in form of features and fingerprints and (3) image data for molecular structure of drug and learn a single representation of these modalities using an optimization procedure similar to multiple kernel learning (Gönen and Alpaydın, 2011). We now present a clinical and algorithmic background for the DDI problem.

The interactions of a drug can be specified in two ways: (1) the drug has an adverse effect on the human body, called adverse drug events (ADEs), and (2) the drug interacts with another drug to cause an adverse effect called drug-drug interactions (DDIs). Most recent research has focused on finding ADEs from text. Different approaches have been taken in order to identify and discover ADEs in the machine learning community, especially from the natural language processing (NLP)

perspective. Chee et al. (2011) make use of ensemble classifiers to extract ADEs, while Liu and Chen (2013) used transductive SVMs to extract ADEs from online health forums. Gurulingappa et al. (2012) use NLP with support vector machines (SVMs) to extract ADEs from MEDLINE casereports. Karlsson et al. (2013) and Page et al. (2012) perform ADE information extraction from EHR data. More recently, Kang et al. (2014) took a knowledge-based approach for extracting ADEs from bio-medical text by using a concept recognition module identifying drugs and consequent adverse effects in sentences and also consists of a knowledge-base module to decide if a relation exists between the drugs and effects, while Natarajan et al. (2017) use Markov logic networks for adverse drug event extraction from text.

In comparison to the extraction of ADEs, the problem of DDI prediction and discovery has received far less attention, although similarity-based methods have proven to be very popular. The problem of DDI discovery is a pairwise classification task, which lends itself very well to kernel-based methods (Shawe-Taylor and Cristianini, 2004). Kernels are naturally suited to representing pairwise similarities, and are constructed directly from the data vectors during pre-processing. Most similarity-based methods for DDI discovery/prediction also use text sources such as biomedical research literature as the underlying data source, and construct NLP-based kernels from these medical documents (Segura-Bedmar et al., 2011; Chowdhury and Lavelli, 2013). Our work differs considerably from such approaches as we do not restrict ourselves to corpus-based NLP kernels for similarity, but rather *focus on molecular and structural similarities*. It should be noted, however, that our framework can easily support such NLP-based similarities as it is designed to work with such heterogeneous similarity measures, including NLP, and will be an interesting next step.

Fusing multiple kernels has also been studied as a viable approach for DDI discovery. Chowdhury and Lavelli (2011) combined linguistic and NLP kernels for the *DDIExtraction2011* challenge (Segura Bedmar et al., 2011). While this work used multiple similarities (kernels), they were all constructed from the same data, making their approach homogeneous. The work of Cheng and Zhao (2014) is closest to our heterogeneous approach; they consider four types of drug-drug

similarities (phenotypic, therapeutic, structural and genomic). However, a significant difference from our approach is that they treat pairwise similarities directly as features for use with standard machine-learning models such as SVMs and k-nearest neighbor classifiers. This approach destroys the structural and neighborhood information inherent in drug-drug similarity matrices; this means that their model does not capture the true complexity of the DDI manifold. Our method differs from Cheng et al's as we combine heterogeneous similarities *jointly* and (locally) optimally, rather than combining kernels into a single feature set and running propositional classifiers.

It should also be noted that other multiple kernel approaches *do not learn relative weights of similarities*, that is, kernel combination is not a part of the learning process and is performed *a priori* using fixed weights. This is a significant difference, as our approach learns a kernel as well as relative weights between similarities to show which ones have the most influence on the final kernel. Molecular structure similarity analysis has been studied in the context of DDIs before, where Vilar et al. (2012) used SMILES code and MACCS fingerprints, with a matrix multiplication method thresholded by a Tanimoto coefficient cutoff to predict new DDIs. Similarity-based kernels were also used in the different task of drug-target interactions prediction (Ding et al., 2013). The work of Tatonetti et al. (2011), Thomas et al. (2011) and Percha et al. (2012) are also relevant, though applied to drug-target interaction prediction since their prediction task is very close to ours and some of their features constructed from adverse event reporting system can be seamlessly integrated into our work.

From a machine-learning standpoint, our work is closely related to multiple kernel learning, which combines the power of multiple kernels together to learn a linear or non-linear kernel combination. The work of Lanckriet et al. (2004) optimizes over a linear combination of multiple kernels through semidefinite programming. In this seminal work, Lanckriet et al. (2004), test their method on two data sets and demonstrate that learning a combination of kernels is indeed better than learning single kernels for classification. Bach et al. (2004) built upon this work and proposed more efficient algorithms for multiple kernel learning. Sonnenburg et al. (2006) further generalized

the formulation in (Lanckriet et al., 2004) by posing the multiple kernel learning problem as an semi-infinite linear program that is easier to solve. In recent years, multiple kernel learning has also been extended to multi-class problems (Zien and Ong, 2007) and localized kernels (Gönen and Alpaydin, 2008), where kernels are learned more precisely using the local information available. Other methods such as those proposed by Rakotomamonjy et al. (2007) and Varma and Babu (2009) led to efficiency improvements.

These and other methods are discussed in Gönen and Alpaydın (2011). These approaches all rely on the fact that multiple kernel learning can be equivalently cast in terms of the SVM dual; thus these approaches are used for *individual classification* of training examples. Our framework is considerably different, however, as we are interested in *pairwise classification* of training examples to identify interactions.

Our framework, relies on *kernel alignment*, which serves to regularize a kernel learning problem. Specifically, we seek to learn a *single kernel* from multiple similarities by aligning the kernel with the labels (Cortes et al., 2012) as well as local neighborhood (Hoi et al., 2007). At a high level, our approach seeks to perform manifold regularization (Belkin et al., 2004) and alignment, to fuse information from various similarity measures into one kernel.

2.2 Learning from Relational data

Although extracting similarity measures from given data and using the similarities for model construction has been used widely in machine learning, most of the real world data is relational in nature. Representing such data in form of flat feature vectors results in loss of information. Examples include important relationships not being captured while the transformation process at one end to the creation of spurious relationships in the transformed data at the other end, which can adversely impact the task at hand. For instance, *since we cannot pre-determine the number of chemical and biological compounds that are related to or lead to an interaction of a drug (i.e. there is no pre-determined process that can be generalized to every drug thereby introducing* ***uncertainty****),*

it becomes difficult if not impossible to represent them as fixed length feature vectors. Even if we are able to obtain this feature vector representation, it is with respect of each drug which makes it difficult to ***generalize*** *over the various type of reactions that might happen with other drugs.* This presents us with two primary problems of a flat feature vector representation of relational domains, namely *linkage* and *autocorrelation* (Jensen and Neville, 2002).

To deal with relational data directly and avoid the above mentioned problems, we need to handle the uncertainty and generalizability issues and to do so we turn to two of the oldest fields of study within mathematics and computer science, *probability* and *first-order logic*. Statistical Relational Learning (SRL) (Getoor and Taskar, 2007; Raedt et al., 2016) harnesses the power of probability (that handles noise and uncertainty) and first-order logic (that handles generalizability by capturing the rich underlying relational structure). Although an already vast yet still evolving field, SRL models can be categorized based on the underlying structure, whether it is undirected or directed. Undirected SRL models include Markov logic networks (Richardson and Domingos, 2006; Singla and Domingos, 2005) with the structure learning (Kok and Domingos, 2005) and parameter learning (Lowd and Domingos, 2007) and the respective boosted model, MLN-Boost (Khot et al., 2011). Directed SRL models include relational dependency networks (Neville and Jensen, 2007b) and the respective boosted model, RDN-Boost (Natarajan et al., 2012a). Recently, a lot of effort is also taking place to combine deep learning with relational methods to take advantage of the scalability of deep learning models and interpretibility of relational models (Sourek et al., 2015; Kaur et al., 2017a; Kazemi and Poole, 2018a; Kaur et al., 2019).

2.2.1 Structure Learning for Local Neighborhood Models from Multi-Relational Data

Graph data is naturally multi-relational and algorithms that can directly operate on the graph data without resorting to similarity generation are highly desirable. In our work we take advantage of the underlying local-neighborhood structure while dealing with such graph data and learn a representation for the graph entities based on their local neighborhoods. Learning embeddings from

a given knowledge graph, or **graph embedding**, is a well-studied problem in machine learning. A large body of recent work in this area can be categorized broadly by the underlying approaches: matrix factorization, deep learning, edge reconstruction, graph kernels and generative models. These approaches have been extensively surveyed recently; see for instance, Nickel et al. (2016), Wang et al. (2017a) and Cai et al. (2018). In general, Gaifman models tend to scale better than many such approaches to higher-arity relations and target-query complexity (Niepert, 2016) owing to their local view and incorporation of count-based features, as opposed to the global view of (say) neural network or factorization methods, which are forced to look at the entire graph to construct effective embeddings. While highly effective, a key drawback of these approaches is their inability to incorporate new data easily, often requiring a new model to be trained.

Bordes et al. (2011) use a neural network architecture to create embeddings from any knowledge base. In more recent work (Dettmers et al., 2018), deep learning has been employed to learn such embeddings. Other recent methods such as HOLE (Nickel et al., 2016) uses circular correlation of the vector representations of entities to create holographic embeddings for binary relational data and Poincare embeddings (Nickel and Kiela, 2017) embeds the underlying knowledge graph into a Poincare ball.

Relational Learning Relational learning is the task of predicting values of a relation, given a relational database of entities and the observed relations among the given entities. One of the most important tasks in relational learning is that of *link prediction* which determines whether a relation (link) exists between entities based on the given relational database. Taskar et al. (2004) use Markov Logic Network to predict links between entities in relational domains with a focus on collective link classification. Martínez et al. (2017) and Al Hasan and Zaki (2011) present a comprehensive survey on link prediction problems in complex networks and social networks respectively. Deep learning methods have also been used for link prediction problems especially using graph neural networks (Harada et al., 2018). Chuan et al. (2018) present a metric learning based method for link prediction in online social networks. More such recent methods are surveyed in Goyal and Ferrara (2018).

Structure Learning Structure learning has been a well-studied problem in graphical models and can be defined as is the problem of estimating a graph structure i.e. model learning, that can principally summarize the dependencies in a given data set. For our purpose, structure learning (in relational domains and probabilistic graphical models) can be interpreted as learning first-order rules from the given graph and/or the data. Structure learning for Bayesian networks has been especially well studied and various methods exist such as genetic algorithms (Larrañaga et al., 1996), linear programming (Jaakkola et al., 2010) and constraints (De Campos et al., 2009). For undirected graphical models, Kok and Domingos (2005) optimize a weighted likelihood function while searching through a space of clauses to learn a Markov logic network structure. Natarajan et al. (2012a) present an end-to-end algorithm i.e structure and parameter learning algorithms for boosting relational dependency networks. Several other methods also exist such as using L_1 regularization (Lee et al., 2007), hypergraph lifting (Kok and Domingos, 2009), relational logistic regression (Ramanan et al., 2018) and decision trees (Lowd and Davis, 2010). A more comprehensive survey for structure learning in graphical models is provided in Zhou (2011).

2.2.2 Learning Relational Graph Convolutional Networks

Recently, several successful methods for learning embeddings of large knowledge bases have been developed. They have been motivated through the inevitability of learning and reasoning about various entities, their attributes and relations present in the knowledge bases. Several of these approaches such as TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransG (Xiao et al., 2016) and KG2E (He et al., 2015), to name a few, can be grouped into translational distance models that focus on minimizing a distance based function under some constraints or using regularizing factors between entities and relations. More recent approaches propose to extend these translation approaches by embedding the knowledge graphs into more complex spaces such as the hyperbolic space (Balazevic et al., 2019; Kolyvakis et al., 2019) and the hypercomplex space (Zhang et al., 2019; Sun et al., 2019). Another important class of approaches such as RESCAL (Nickel et al.,

2011), DistMult (Yang et al., 2015) and HolE (Nickel et al., 2016) focus on various compositional operators for the entities and relations in the knowledge graph. A more complete survey on the different types of knowledge graph embeddings is given in (Wang et al., 2017b).

While several methods have focused on applying neural networks to extract relational embeddings on large networks such as knowledge graphs, a potential limitation of much of this line of research is that the inherent semantic structure of the network is not exploited. To overcome this limitation, graph convolutional networks (GCNs) (Kipf and Welling, 2017) were proposed that generalized neural network models to multi-relational, graph-structured data sets. A GCN (Kipf and Welling, 2017) generalizes neural network models to multi-relational, graph-structured data sets where each convolution layer in the GCN applies a graph convolution i.e. a spectral filtering of the graph signal (the feature matrix of the graph) via the Graph Fourier Transform.

Similar to the convolution operators in convolutional neural networks (CNNs) that extract locally stationary features in the inputs data, GCNs utilize the graph convolution operator defined with respect to the adjacency matrix $\mathcal{A}$ to extract local features from a semantic point of view. The fundamental difference between CNNs and GCNs is that, while CNNs apply spatial convolutional filters, GCNs apply spectral convolutional filters or similar aggregational filters (Hamilton et al., 2017; Ying et al., 2018) to incorporate the neighborhood information in the underlying model which leads to a better generalization. Similar to a CNN which can, and generally has, multiple convolutional layers, successive application of graph convolution is possible, with each application (layer) producing a successively more informative node embeddings, while possibly simplifying the network structure.

While successful, GCNs still have a limitation in that they cannot directly be applied on multi-relational data/networks. To alleviate this, a recent extension to the relational data for GCNs was proposed (Schlichtkrull et al., 2018) where a latent representation of the entities are explicitly constructed and a tensor factorization then exploits these representations for the prediction tasks. We take an alternative approach – we consider the more recent successes inside SRL (Khot et al.,

2014; Lao and Cohen, 2010a) to develop novel combinations of the entities and their relationships to construct richer latent representations. As we demonstrate empirically, this leads to superior predictive performance.

While our work can be used for entity attribute prediction and link prediction, we focus on the more interesting link prediction task to demonstrate the utility of our approach.

2.3 Faithfully Generating Clinical Data

Generative Aversarial Networks While GANs have proven to be powerful frameworks for estimating generative distributions, convergence dynamics of a naive mini-max algorithm for learning GANs has been shown to be unstable. Some recent learning approaches, among many others, augment learning either via statistical relationships between the true and the learned generative distribution such as Wasserstein-1 distance (Arjovsky et al., 2017), MMD (Li et al., 2015) etc. or via spectral normalization of the parameter space of the generator model (Miyato et al., 2018) which controls the generator distribution from drifting too far.

Most of the work in the space of GAN models has been dedicated towards applying them to image data. Since the generator and discriminator used in GANs are (deep) neural networks which have proven to be very efficient in computer vision tasks, using GANs on such tasks seem a natural approach although GANs have been previously applied to generating medical records. Choi et al. (2017) propose an approach that extends the GANs models to be applied to patient's electronic health record (EHR) data to generate synthetic patient records. The authors present a model that is similar to (Li et al., 2015) in nature as it is a combination of auto-encoders and GANs but instead of learning in the auto-encoder space, the auto-encoder assists GANs to learn in the true data space by decoding the generated distribution into synthetic EHR records. The authors also propose mini-batch averaging to prevent mode collapse, a recurrent problem in training of GANs.

Most of the architectures proposed for GANs make use of neural networks for the discriminator and generator models. Esteban et al. (2017) use recurrent neural networks, specifically long-short

term memory networks, for the discriminator and generator models and focus their approach on medical data-sets. They propose two models, Recurrent GAN (RGAN) and Recurrent Conditional GAN (RCGANs) that make use of recurrent neural network (long-short term memory networks, to be specific) for the discriminator and generator models and focus their approach on medical data-sets. RCGAN and RGAN differ in the number of inputs each of the model takes as RGAN takes a unique random seed at every (temporal) input to produce the synthetic data, whereas RCGAN takes additional inputs that condition the produced outputs. The authors also propose two interesting ways to evaluate the GAN output, namely, train on synthetic, test on real (TSTR) and train on real, test on synthetic (TRTS). These are important since the real quality of learned examples can be judged if the features learned in one distribution can be generalized to the other distribution.

As mentioned above, privacy is an important issue in the sharing of medical data sets for research purposes. Some previous approaches have been proposed that reserve privacy while generating synthetic patient records (Howe et al., 2017; Beaulieu-Jones et al., 2019; Jordon et al., 2019). Howe et al. (2017) propose DataSynthesizer, a tool to generate synthetic data from sensitive real medical data with strong privacy guarantees. The tool works in three different modes and an important functionality of each mode is that special care has been taken to preserve privacy of sensitive medical data by adding Laplacian noise while constructing the distribution. Since the noise is known, the resulting attributes can be de-noised to obtain the true values. Most of these models added random noise based on differential privacy i.e. no one patient (example) should have a significant influence on the learned model. The random noise added is directly proportional to the effect an example has on the learned model.

Human Expert Advice based Models In his seminal monograph, "The Need for Biases in Learning Generalizations" Mitchell shows that inductively biasing learners is necessary in order achieve true generalization over new instances (Mitchell, 1980). One, albeit traditional, view of such inductive bias is of "generality via simplicity" where regularization strategies aim to control

the complexity of a model. However, another form of inductive bias is incorporating "knowledge of the domain". The goal of a system that implements such a bias (with domain knowledge typically obtained from human domain experts) is to learn a model that is consistent with the training examples *as well as with known facts about the domain*.

One typical approach to incorporate human guidance in model learning is by providing *rules over training examples and features*. Some of the earliest such approaches were explanation-based learning (EBL-NN, (Shavlik and Towell, 1989)) or ANNs augmented with symbolic domain rules (KBANN, (Towell and Shavlik, 1994)). Substantial research in this area have studied various techniques of leveraging domain knowledge for optimal model generalization, such as, polyhedral constraints as in the case of knowledge-based SVMs, (Cortes and Vapnik, 1995; Schölkopf et al., 1998; Le et al., 2006; Fung et al., 2002; Kunapuli et al., 2010)), *preferences* which has been studied extensively within the preference-elicitation frameworks (Braziunas and Boutilier, 2006; Kunapuli et al., 2013b) or *qualitative constraints* such as monotonicities and synergies (Yang and Natarajan, 2013) or *quantitative relationships and constraints* (Ganchev et al., 2010). Our knowledge augmented learning framework most closely resembles qualitative constraints in spirit.

While widely successful in building optimally generalized models in presence of systematic noise (or sample biases), knowledge-based approaches have mostly been explored in the context of discriminative modeling. In the generative setting, a recent work extends the principle of posterior regularization from bayesian modeling to deep generative models in order to incorporate structured domain knowledge (Hu et al., 2018). Traditionally, knowledge based generative learning have been studied as a part of learning probabilistic graphical models with structure/parameter priors (Mansinghka et al., 2006).

PART I

LEARNING FROM NOISY, HETEROGENEOUS DATA

CHAPTER 3

LEARNING FROM CLINICAL DATA WITH EXPERT ADVICE

Parkinson's, a progressive neural disorder, is difficult to identify due to the hidden nature of the symptoms associated. This difficulty in identification of progressive neural disorders can result in the full fledged development of the disorder in patients. Since these disorders have no cure, the only method to prevent the onset of such disorders is early diagnosis can significantly improve the quality of life for the patients. This chapter presents our work (Dhami et al., 2017) that considers the data collected as part of the Parkinson's Progression Marker Initiative (PPMI) (Marek et al., 2011) and aims to predict the presence of Parkinson's disease in a patient based on clinical data. There exist few important challenges for this task such that the data is inherently noisy and that there seem to be no real strong indicator that explains the progression clearly.

3.1 Introduction

We consider the problem of predicting the incidence of Parkison's disease, a progressive neural disorder. Specifically, we consider the data collected as part of the Parkisons Progression Marker Initiative (PPMI) and aim to predict if a subject has Parkinson's based on clinical data - particularly, motor assessments (motor functions) and non-motor assessments (neurobehavioral and neuropsychological tests). One of the most important challenges for this task is that there appears to be no real strong indicator that explains the progression clearly (Marek et al., 2011).

Our hypothesis, that we verify empirically in this work is that instead of considering a small set of strongly influencing risk factors, it might be more effective to consider a large set of weakly influencing risk factors. To enable learning with this large set of weak influences, we adapt the recently successful gradient-boosting algorithm (Friedman, 2001) for this task. We exploit the use of a domain expert in identifying the right set of features and consider learning from the longitudinal data. Unlike standard methods that require projecting the data to a feature vector format (using what

are called aggregation or propositionalization methods), our proposed approach models the data faithfully using time as a parameter of a logical representation. In addition, use of the state-of-the-art boosting technique allows for learning flexible models. We evaluate our proposed approach on $\approx$ 1200 patients from the PPMI study and demonstrate the effectiveness and efficiency of the proposed approach.

In this chapter, we make the following key contributions: we consider the challenging task of predicting Parkinson's from 37 different risk factors (features in machine learning terminology). These features were chosen by interacting with the domain expert. The key advantage of this approach is that it models the underlying data faithfully without converting the data into a standardized feature vector format. We evaluate our approach on the real data from PPMI study where we compare our proposed method against several baselines.

The rest of the chapter is organized as follows: after introducing the necessary background, we present the boosting for Parkinson's approach. Then we present detailed experimental evaluation before concluding the chapter by outlining areas of future research.

3.2 PPMI study

Parkinson's Progression Markers Initiative(PPMI) is an observational study with the main aim of identifying features or biomarkers that impact Parkinson's disease progression. Marek et al. (Marek et al., 2011) defines one of the goals of the PPMI study as recognizing key features that will help in identifying the progression of the disease. Since Parkinson's is a gradually increasing disease and does not have a specific known cause, any pointers about the features that can help in identifying the progression can be critical to developing treatment for the condition. The study collects data using a variety of clinical and imaging assessments [1]. Figure 3.1 shows the structure of the PPMI study.

The collected data can be divided broadly into four distinct categories:

[1]The data can be accessed from the website of Laboratory of NeuroImaging (LONI) (https://ida.loni.usc.edu)

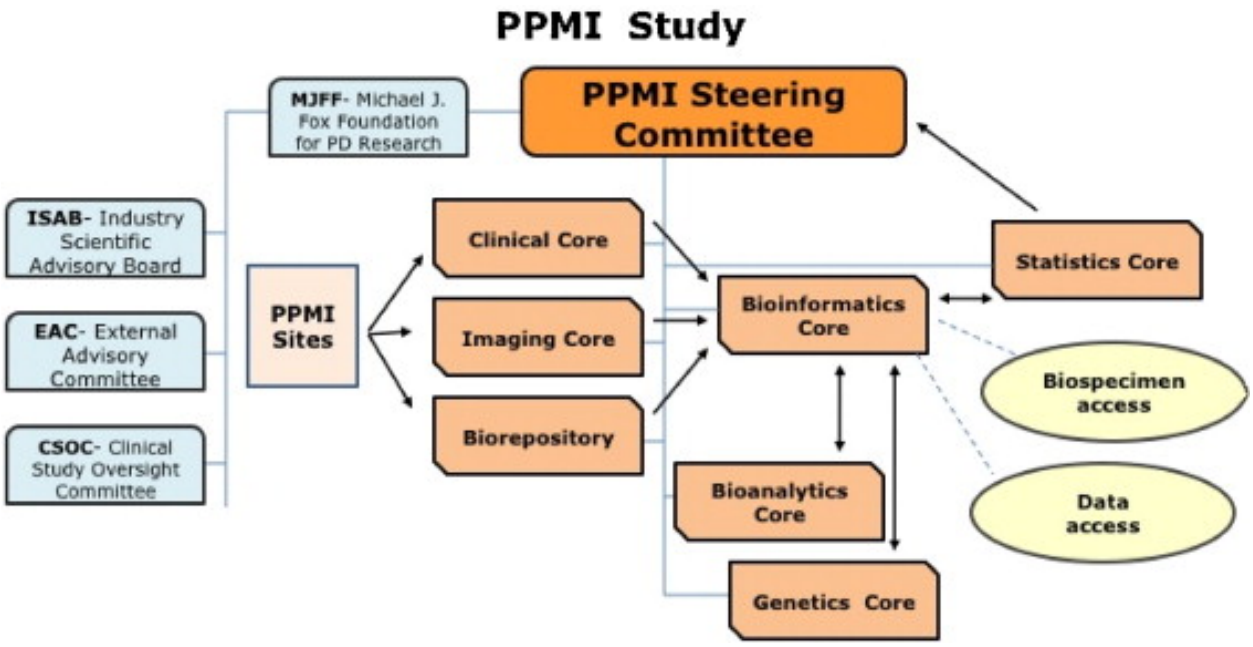


Figure 3.1: Marek et al. (2011) describe the study organisation

1. Imaging data: MRI and DaTSCAN images of a subject. DaTSCAN images are constructed as the including criteria of the subject since these images can confirm or refute a doctor's initial diagnosis whereas MRI of the subject is taken during the baseline visit after the inclusion in the study.

2. Clinical data: medical history and various clinical assessments carried out during the study.

3. Biospecimens: physical samples taken from the subject's body, e.g. urine, plasma, blood and serum.

4. Subject demographic data: race, gender and ethinicity of the subject.

In our work we focus primarily on the clinical data which mainly consists of 2 types of assessments:

1. Motor assessments: everyday motor function characteristics.

2. Non-motor assessments: neurobehavioral and neuropsychological tests.

Since Parkinson's mainly affect the motor system (i.e. the part of the nervous system associated with movement) and the initial symptoms are mostly movement related, using motor assessment data seems a natural approach. A subset of the clinical data is selected based on the expert input after which a set of 37 features are obtained. These can be broadly defined in these categories:

1. Motor-UPDRS (consists of 34 features). (Goetz et al., 2008)

2. Montreal Cognitive Assessment (MoCA), a non-motor assessment.

   The MoCA score of a patient for a single visit.

   Difference in MoCA scores of the patient from the last visit.

3. Total UPDRS score.

MoCA consists of series of questions assessing various parameters of a subject such as the visual capability, capacity of recognizing objects, the attention span and memorizing words to name a few. Each of the questions are scored with the total score being 30. A subject with score of ≥ 26 is considered to be normal.

Unified Parkinson Disease Rating Scale (UPDRS) is a rating scale used for determining the extent of Parkinson's disease progression in a subject. Each assessment in a UPDRS scale ranges from 0 to 4, with 0 being the absence of an abnormal behavior and 4 representing severe abnormal behavior with respect to the assessment. Motor-UPDRS refers to the value of motor assessments in the UPDRS scale. Total UPDRS score refers to the sum of all the assessments of the 34 features used.

3.3 Functional Gradient-Boosting

Dependency networks (Heckerman et al., 2001) are graphical models that learn a joint distribution over the variables using independently learned conditional distributions for each variable in the model. These models have proved to be quite effective in the task of probabilistic inference.

Relational dependency networks (Neville and Jensen, 2007a) extend the dependency networks to a relational setting where the joint distribution is learned as a product of conditional distributions over ground atoms. In our work, we are interested in estimating the conditional distribution - $P(par|\mathbf{x})$ where $\mathbf{x}$ represents the set of motor and non-motor assessments (called features in machine learning terminology) and par denotes the incidence of Parkinson's disease for the particular patient. Given this aim, one could apply any machine learning algorithm for learning this distribution. We focus on the gradient-boosting technique which has had success in medical applications (Natarajan et al., 2012, 2013; Weiss et al., 2012).

Gradient-boosting is a gradient-ascent technique performed on the functional space. Standard gradient-descent techniques maximize an objective such as log-likelihood or mean-squared error and derive the gradients w.r.t the parameters that are being learned. They typically start with initial parameters θ_0 and compute the gradient (δ_1) of an objective function w.r.t. θ_0. The obtained gradient is used to update the parameters by adding the gradient to the previous parameters (i.e. the updated parameter $\theta_1 = \theta_0 + \delta_1$) and this process is repeated until convergence. For probabilistic models, an alternate method is that of gradient-boosting that represents the conditional distributions using a functional representation, for instance a *sigmoid* function. Then the gradients are calculated w.r.t to this function. For instance, one could represent $P(par|\mathbf{x}) = \frac{e^{\psi(y|\mathbf{x})}}{1+e^{\psi(y|\mathbf{x})}}$.

Now, the gradients of the loglikelihood can be taken w.r.t ψ. Friedman (2001) suggested that instead of obtaining the gradients w.r.t the global function ψ, one could obtain the gradient for each example $\langle \mathbf{x}_i, y_i \rangle$, where y denotes the target i.e., presence of Parkinson's. The key idea is that while these set of gradients are an approximation of the overall gradient, they are, generally, good approximations because the direction of these gradients point to the same direction as that of the true gradient.

This functional gradient of likelihood w.r.t. ϕ for each example i is

$$\frac{\partial log(P(y_i|\mathbf{x}_i)}{\partial \phi(y_i|\mathbf{x}_i)} = I(y_i = 1) - P((y_i = 1|\mathbf{x}_i) \tag{3.1}$$

where I is the indicator function (equal to 1 if $y_i = 1$ and equal to 0 if $y_i = 0$). The gradient is the difference between the true label and the current prediction and is positive for positive examples and negative for negative examples. In simpler terms, the negative examples are pushed towards 0 and the positive examples are pushed to 1 resulting in a well defined decision boundary.

The approach proposed by Friedman (2001) represents the objective function as a regression function of the given examples, $\phi(\mathbf{x})$ and the gradients are calculated with respect to this regression function. These gradients are termed as functional gradients and analogous to the parametric gradient descent, the functional gradient descent's final function is the sum of the initial regression function with all the functional gradients i.e. after n steps $\phi_n(x) = \phi_0(x) + \delta_1(x) + \delta_2(x) + ... + \delta_n(x)$.

Functional gradient descent fits a regression function $\hat{\phi}(m)$, (generally regression trees) to the gradients δ_m at each step m. The final model is given by $\phi_m = \phi_0 + \hat{\delta_1} + \hat{\delta_2} + \hat{\delta_3} + ... + \hat{\delta_m}$ i.e. the sum over the regression trees. Due to this sequential nature of learning models based on the previous iteration, functional gradient ascent is also known as functional-gradient boosting (FGB). Natarajan et al. (2012b) extend FGB to the relation setting to overcome the assumption of a propositional representation of the data in standard FGB and we employ this relational version, relational functional gradient boosting (RFGB) as our method. A standard objective function used in RFGB is the log-likelihood and the probability of an example is represented as a sigmoid over the ϕ function. The above process can be shown in figure 3.2.

We use a predicate logic representation for modeling examples as groundings/instantiations (e.g. *mcstest(3001,302,"29")*) of the features. The ϕ function is represented by relational regression trees (RRT) (Blockeel and De Raedt, 1998) which uses the relational features as input to the trees. The key reason for using this relational version is that this particular method has an efficient search strategy that can be easily guided by the domain expert who could provide preference information. These preferences have been shown to be particularly useful in presence of noisy data (Odom et al., 2015; Odom and Natarajan, 2018).

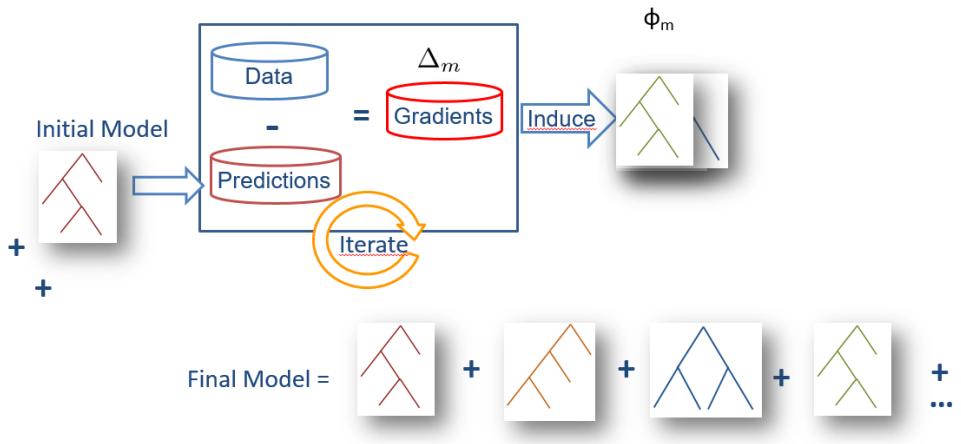


Figure 3.2: Relational Functional Gradient Boosting.

3.4 Proposed Approach

The aim of the PPMI study is to determine the most predictive features for the disease progression, the involvement of a domain expert during the feature selection process becomes beneficial. Our specific aim is to predict the incidence of Parkinson's in a patient. Since the data is longitudinal in nature, it is important that we model time faithfully. We view time as a special type of relation and hence we create features in the predicate logic format as *feature_name(patient id, time, feature value)*. This allows time to be an argument of the predicate which indicates the number of days after the first record, the given study data was included. We take the start date of the study to be day 0. As an example consider that the records of patient A were entered on January 1, 2016 and these are the oldest records for any patient that can be found in the study. Also say, the records of patient B were entered on January 1, 2017. The variable time for patient A will then be assigned as 0 and for patient B will be assigned as 365.

Algorithm 1 shows our proposed approach. The first step of the process starts with the correlation analysis of the raw data obtained. The raw clinical data consists of 81 features. A correlation matrix is constructed with each entry representing the correlation coefficient (we use the *Pearson correlation coefficient*) between each variable and the others. The 50 features with low mutual

correlation and high correlation to the class label are selected. The second step consists of the expert evaluating the obtained features giving us a further pruned set of 34 features and thus the final data to be used for the classification task. The key reason for considering a relational representation is two fold:

1. Relational models allow learning using the natural representation of the data.

2. Data is longitudinal i.e. a patient has multiple entries in the data. Propositional classifiers have limitations in learning such data (require aggregators) (Perlich and Provost, 2006).

Algorithm 1 BoostForParkinson's

function BOOSTPARK($Data$)
  $C = CalcCorr(Data)$
  $E = Expert(C)$
  $M = FGBoost(Tar, F, E)$ ▷ Tar is the target predicate and F is the set of features in E
end function
---
function FGBOOST($Tar_f, F, Data$)
  $Mod_0^f = InitialFunction$ ▷ f denotes index of current target
  for $1 \leq s \leq |S|$ **do** ▷ S is the number of gradient steps
    $EG_f = GenExamples(f; Mod_{s-1}^f, Data)$ ▷ Mod is the current model
    $\Delta_m(f) = FitRelRegressTree(EG_f, f)$ ▷ Fit trees to the functional gradient
    $Mod_s^f = Mod_{s-1}^f + \Delta_m(f)$ ▷ Updating the model
  end for
  $P(X_f = x_f | Ne(x_f)) \propto Mod_S^f(x_f)$ ▷ $Ne(x_f)$ represents the neighbors of x_f that influence x_f
  return Mod ▷ Return the final model
end function
---
function CALCCORR($Data$)
  for $1 \leq i \leq |IF|$ **do** ▷ IF are the initial features
    $corCoeff(i) = PCC(1 : IF \neq i)$ ▷ PCC refers to the calculation of Pearson correlation coefficint
    $corMatrix.add(corCoeff(i))$
  end for
  ReducedFeatureSet = LeastCorrelatedFeatures(corMatrix)
  return ReducedFeatureSet
end function

The learner is provided with the training data which it uses to learn a relational regression tree. The last step is the prediction phase where the learned model can be queried to return the probability

Table 3.1: Summary of classifiers.

Classifier	Summary
BoostPark10	Functional gradient boosting with 10 RRT's
BoostPark20	Functional gradient boosting with 20 RRT's
LR-min, LR-max and LR-mean	Logistic regression on the min, max and mean aggregation of propositional data
GB-min, GB-max and GB-mean	Gradient boosting with tree depth of 5 on the min, max and mean aggregation of propositional data
SVM-L-min, SVM-L-max and SVM-L-mean	Support vector machines with linear kernel on the min, max and mean aggregation of propositional data
SVM-P-min, SVM-P-max and SVM-P-mean	Support vector machines with polynomial kernel on the min, max and mean aggregation of propositional data
SVM-RBF-min, SVM-RBF-max and SVM-RBF-mean	Support vector machines with radial basis function kernel on the min, max and mean aggregation of propositional data

of the target being true given the evidence. Since all the evidence is observed, inference requires simply querying all the relational regression trees, summing up their regression values and returning the posterior estimates i.e. the probability that the given test example belongs to the positive class.

Interpretability: One key limitation of the proposed approach is the interpretability of the final model. While each boosted tree in itself is interpretable, given that they are not learned independently of each other other makes the model difficult to interpret. To make the model comprehensible, we take an approximate approach that we call the Craven approach (Craven and Shavlik, 1996) which was originally developed for making neural networks interpretable. The key idea is to relabel the training data based on the boosted model that we have learned and then train an overfitted tree to this labeled data. The intuition is that this new large tree will represent the decisions made by the original set of trees due to its performance on the training data. Recall that our original training data consists of Boolean labels (Parkinson's vs negative). But the relabeled data consists of regression values that are being learned in the new tree. Hence, the resulting tree is closer to the original learned model as we show in our experiments.

3.5 Experiments

In our empirical evaluations, we aim to explicitly ask the following questions:

Q1: How effective is the feature selection with expert in predicting Parkinson's?
Q2: Given the longitudinal nature of the data, is our method more effective than using standard classifiers in this prediction task?
Q3: Do the initial number of regression trees makes any difference in the learning of the model?
Q4: Does the method of aggregation make a difference in learning a model?

We compare our method, BoostPark, to three propositional classifiers: Logistic Regression, Gradient-Boosting and Support Vector Machines. The propositional data is aggregated using three aggregator functions: min, max and mean over time. Table 3.1 gives an overview of all the classifiers considered in our work.

Our data consists of records for 1194 patients, with 378 positive examples (i.e. Parkinson's patients) and 816 negative examples. Regression trees are learned on the given data which form the training model. The data is split into training and testing data with a ratio of $60\% - 40\%$. The training data has 226 positive and 489 negative examples whereas the testing data has 152 positive and 327 negative examples. We perform 10-fold cross validation and present the results.

Table 3.2: Results for BoostPark with and without expert advice

	With Expert				Without Expert			
Classifier	Accuracy	AUC-ROC	AUC-PR	F1 score	Accuracy	AUC-ROC	AUC-PR	F1 score
BoostPark10	0.889	0.973	0.937	0.808	0.854	0.932	0.9	0.797
BoostPark20	0.901	0.977	0.947	0.851	0.881	0.94	0.87	0.832

Table 3.2 shows the result of learning from 50 features obtained after correlation and 37 features after the expert advice. This helps us in answering **Q1** affirmatively. Feature selection with expert is effective in our classification task.

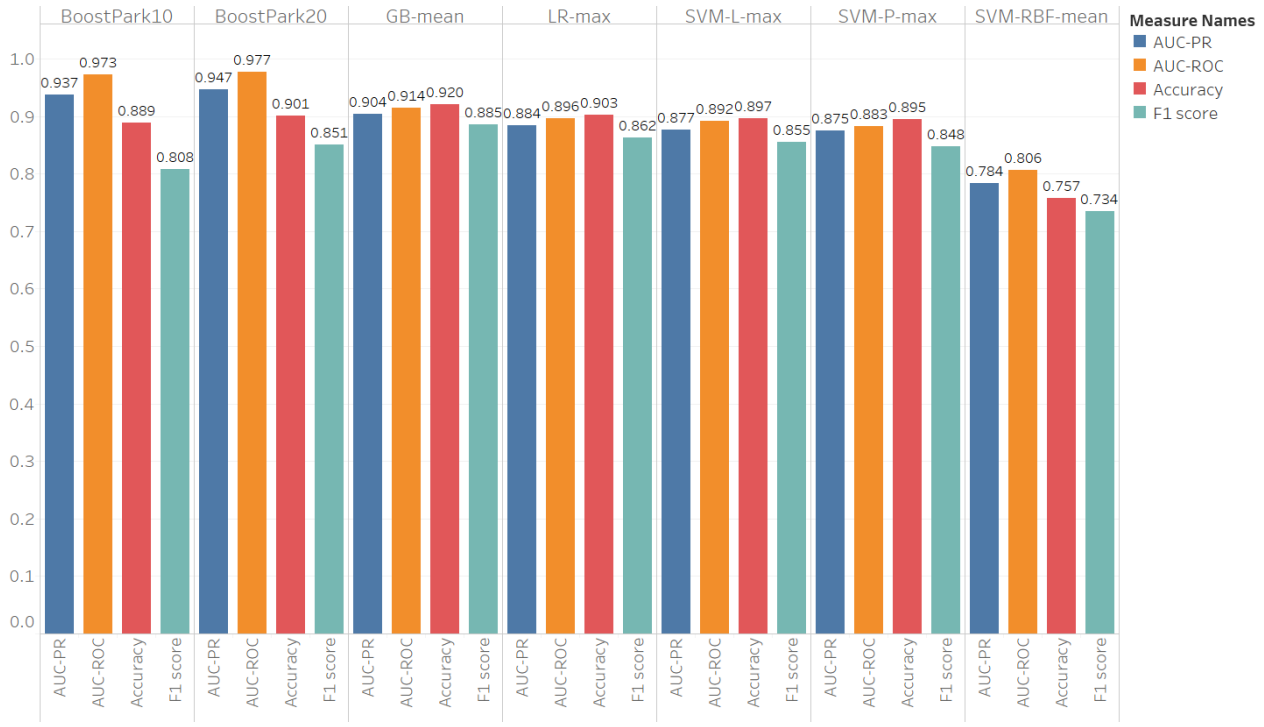
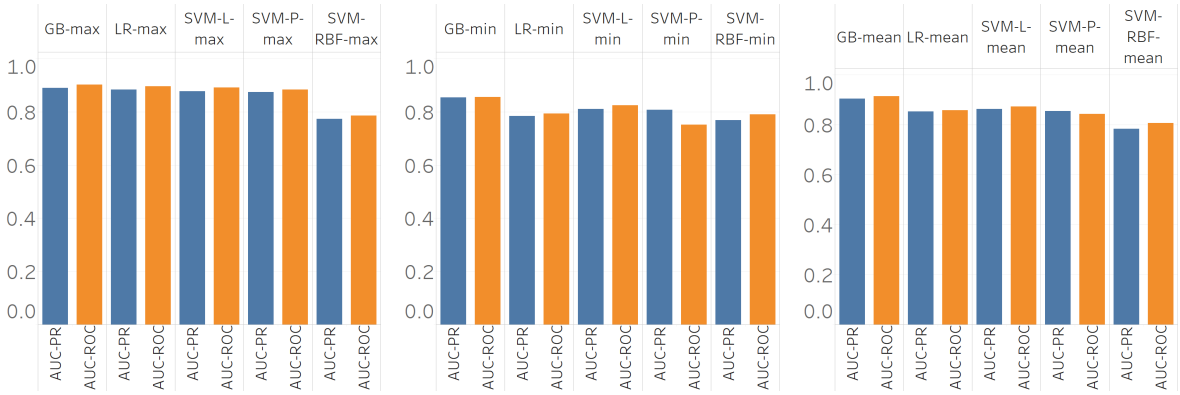


Figure 3.3: Classifier results. Only the best classifiers among the aggregators are shown.

Since we aggregate the propositional data using 3 aggregators, the best performing aggregator for all the propositional classifiers is selected and compared to our methods BoostPark10 and BoostPark20. BoostPark10 and BoostPark20 perform considerably better than the propositional counterparts in terms of AUC-ROC and AUC-PR and performs equally well in terms of accuracy. This helps answer **Q2** positively. Our method is more effective than the standard classifiers in this prediction task.

As mentioned in the previous section, we obtain an approximate combined tree. Specifically, we combine two different number of trees - 10 and 20 trees - and these are presented in figure 3.5 and figure 3.6 respectively. As seen from the resulting trees, a few of the features do remain the same, most evidently the feature constituting the root of the tree, although some features do change. Thus **Q3** can be answered affirmatively. The initial number of regression trees makes some difference in the learning of the model. However, the prediction ability of both the trees are similar. This indicates that there could be multiple ways in which the risk factors could interact to yield similar



(a) Aggregator = max (b) Aggregator = min (c) Aggregator = mean

Figure 3.4: Propositional classifier performance on aggregated data.

performance results. This also validates our claim that most risk factors have a weak influence on the target disease. The similarity in the trees also indicate the consistency of our learning algorithm in extracting these predictive features.

Figure 3.4 presents the performance of all the propositional algorithms on all the aggregators. As we can see, the *max* aggregator generally performs better than the other two aggregators. This result answers **Q4**. The method of aggregation does make a difference in learning a propositional model.

3.6 Conclusion

Identifying important features responsible for the progression in Parkinson's disease in a patient remains a compelling challenge. We use a human domain expert to guide our method with identifying a relatively large set of influencing risk factors. We then present a learning method that can consider this large set of weak influences in learning a probabilistic model. We evaluated our results on the PPMI data and demonstrated that our learning approach significantly outperforms the standard machine learning classifiers. Since Parkinson's is a progressive disease, developing robust temporal models for this task remains an interesting challenge. Extending our learning algorithm to handle hybrid temporal models will allow for modeling the continuous data more faithfully. Finally,

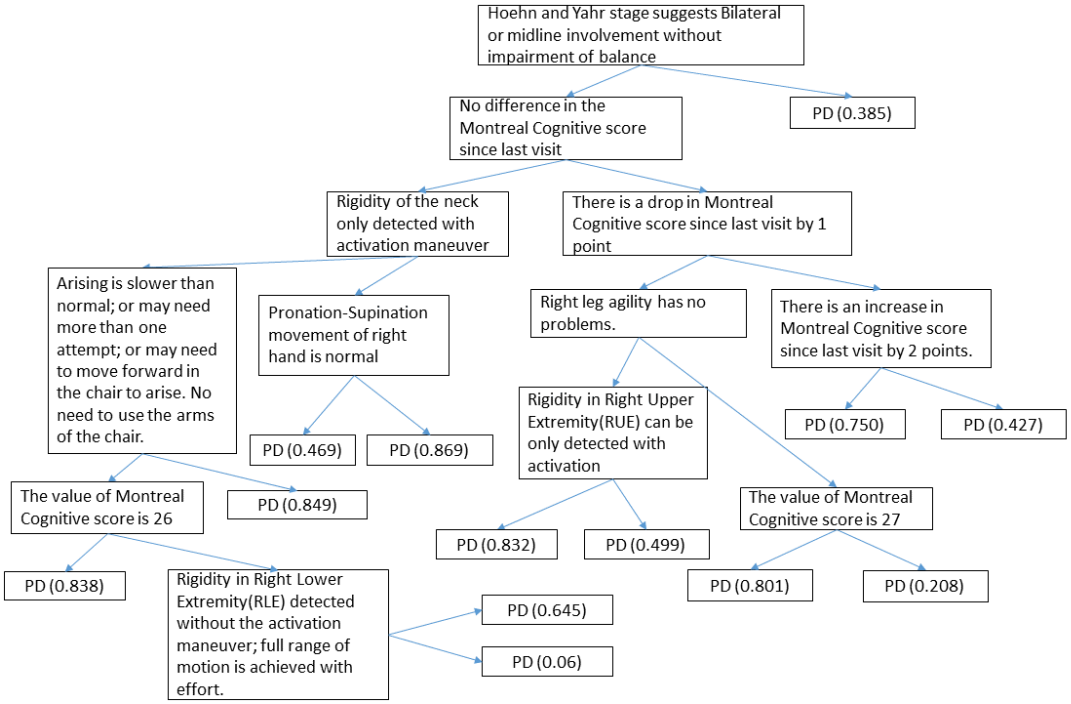


Figure 3.5: Combined tree learnt with BoostPark10.

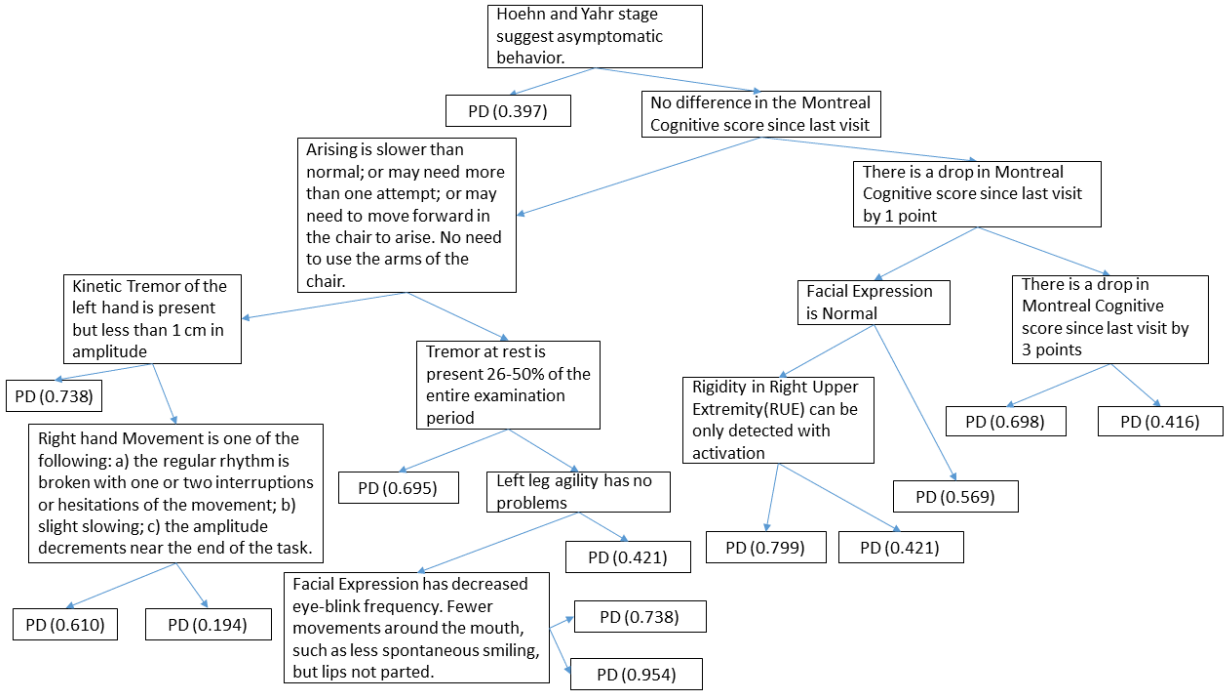


Figure 3.6: Combined tree learnt with BoostPark20.

scaling up the learning algorithm to learn from a broader set of population data rather than be limited by a single study remains an interesting open problem.

CHAPTER 4

LEARNING FROM HETEROGENEOUS SIMILARITIES

In this chapter, we present our similarity function-based approach (Dhami et al., 2018) for predicting and discovering potential drug-drug interaction (DDIs) by combining two different kinds of data, namely, graph data in the form of relationships of various drugs with different enzymes, transporters and targets and molecular data in form of features and fingerprints. The idea of our approach is not just to predict such DDIs but also to discover potential new interactions by incorporating elements of both classification and similarity based algorithms.

4.1 Introduction

Drug-drug interactions (DDIs) occur when multiple medications are co-administered and can potentially cause adverse effects on the patients. DDIs have emerged, around the world, as a major cause of hospital admissions, rehospitalizations, emergency room visits, and even death (Becker et al., 2007). These numbers are even more stark among older adults, who are more likely to be prescribed multiple medications; the study by Becker et al. (2007) which identifies that the elderly have an increased risk factor of as much as 8.5 times over the general population. Consequently, DDIs contribute to increased hospital stays and increasing costs of health care, even though up to 50% of these adverse drug effects (ADEs) are preventable (Gurwitz et al., 2003).

While regulatory agencies such as the U. S. Food and Drug Administration have rigorous approval processes for new drugs, controlled clinical trials do not always uncover all possible drug interactions. For example, the last stage of the FDA approval process involves a Phase III clinical trial, which typically enrolls 1000–5000 individuals, while the drug may be prescribed to millions of patients after approval. In addition to clinical trials, *in vitro* and *in vivo* experiments are also used to identify DDIs. However, these approaches are highly labor-intensive, costly and time-consuming. Another factor is that many known DDIs involve medications such as anti-inflammatories or anticoagulants, which are prescribed for common and chronic conditions. Other

confounding factors that make studying DDIs a difficult challenge include dosage variations and demographic variability.

All of these challenges have led to a shift in research towards *in silico* approaches that leverage advances in AI and machine learning (Percha and Altman, 2013) for DDI discovery. These approaches for DDI can be viewed in one of two ways:

1. The feature-based view, which roughly categorizes the approaches based on the type of DDI features used. These are either text-based (which involves the analysis of abstracts or EHRs) or structure-based (which involve the study of chemical, molecular and pharmacological properties). Our approach is structure-based.

2. The algorithm-based view distinguishes between approaches as classification (which treat DDI discovery as a binary classification problem) and clustering (which assume that similar drugs may interact with a same drug). Our approach is a hybrid of both these paradigms.

In prior work, our group has performed ADE discovery and subgroup discovery from Electronic Health Records (EHR) (Page et al., 2012) and text-mining of medical journal abstracts (Odom et al., 2015). These approaches address the problem of *post-marketing surveillance*, that is, they seek to exploit the new information available after a drug has been approved and has been prescribed to larger, more diverse populations. In this work, we address *pre-trial discovery*, that is, we reframe the problem as one of studying drug-drug interactions, rather than taking a single drug and finding adverse events associated with it. The primary motivation is to preemptively identify potential DDI and ADE risks during drug design. As we show in this work, our novel formulation incorporates elements of *both classification and similarity based algorithms*, which improves discovery as well as explainability. The result is a kernel that we call SKID^3 (**S**imilarity-based **K**ernel for **I**dentifying **D**rug-**D**rug interactions and **D**iscovery).

Our problem setting differs from current approaches in three significant ways, that motivated us to develop SKID^3:

- A majority of current work focuses on drug-interaction discovery through *information extraction*, specifically through text mining. These approaches attempt to identify drug interactions from various unstructured text-based sources such as biomedical journals and semi-structured sources such as EHRs.

  We approach the problem by looking at structured sources of information for insights into drug interactions. Specifically, we characterize drug similarities using different properties of drugs such as molecular structure and pharmacological interaction pathways. This allows us to pose the DDI discovery problem as a structure prediction (Bakir et al., 2007) task.

- The approaches that do use structural information generally aim to extract explicit vector representations of properties such as 3-d structure, which allows the application of off-the-shelf machine-learning techniques such as support vector machines and kernel learning (Shawe-Taylor and Cristianini, 2004).

  We, instead, analyze structural similarities between drugs in ways that are natural and intuitive to their representation (such as random walks on chemical interaction pathways), rather than forcing an artificial and uninterpretable embedding in a vector space.

- Finally, many current approaches focus on a single type of interaction or similarity, whether it is discovered from text sources or from structural analysis. This is a significant drawback, as this analysis approach ignores the diverse pharmacological facets to drug-drug interaction to look at one (or a few) interaction types in isolation.

  We develop a general and extensible framework that admits heterogeneous characterizations arising from any source including text-based, molecular structure, pharmacological, phenotypic, genomic, therapeutic similarities. This allows us to exploit diverse characterizations of drug similarities from various perspectives, fusing them into one coherent, interpretable model.

We make the following contributions with our proposed solution to address the above limitations:

- We characterize molecular similarity between two drugs using a novel approach: *knowledge-refined random walks* to measure the *reachability* of one drug from another; reachability informs the intuition that drugs that are more reachable are more interactive. As far as we are aware, this is the first work on exploiting bias knowledge to characterize drug similarities for DDI discovery.

- We develop a novel framework that combines multiple similarity measures into unified kernel that exploits and fuses their potential. In addition to our novel reachability measure, we also use four other measures that capture molecular and chemical similarities through SMILES strings and MACCS fingerprints.

- We formulate DDI as a *kernel-learning problem* that fuses heterogeneous similarity measures. Our formulation enables us to treat each similarity as a different *view of drug interactions*. By fusing similarities from different sources, our formulation aims to reconcile various (molecular, pharmacological etc.,) views into a single model. Further, our formulation incorporates terms to capture both individual as well as neighborhood interactions, leading to greater robustness.

- From a machine-learning standpoint, our formulation is general in that it admits a *variety of regularization and loss functions*. In this work, we show our approach for a specific formulation that attempts to simultaneously align the optimal kernel with the heterogeneous similarity measures as well as predict the drug-drug interactions.

- From an optimization standpoint, our formulation is a *bilinear program*, which is a non-convex optimization problem. We illustrate an alternating minimization approach for solving this problem; this approach identifies robust and relevant local solutions for DDI discovery and scales well with the underlying drug database size.

- Our empirical evaluation on a data set constructed from DrugBank uncovers previously known drug-drug interactions with high accuracy. Furthermore, a closer inspection of "false positives" and "false negatives" identified by SKID^3 reveals that it has identified drug-drug interactions, missing from DrugBank, but existing in other independent sources. This clearly demonstrates the potential of our approach to perform DDI discovery. More specifically, it also offers us a path forward: DDI discovery via active learning with semi-supervised data, which is the real-world problem setting.

The complete pipeline of our method is shown in figure 4.1. The rest of this chapter is organized as follows: after reviewing related work in the next section, we define the problem of DDI prediction/discovery. We then present similarity measures and formulate kernel learning for DDI discovery. Next, we present our comprehensive experimental evaluation before concluding the chapter by motivating interesting research directions.

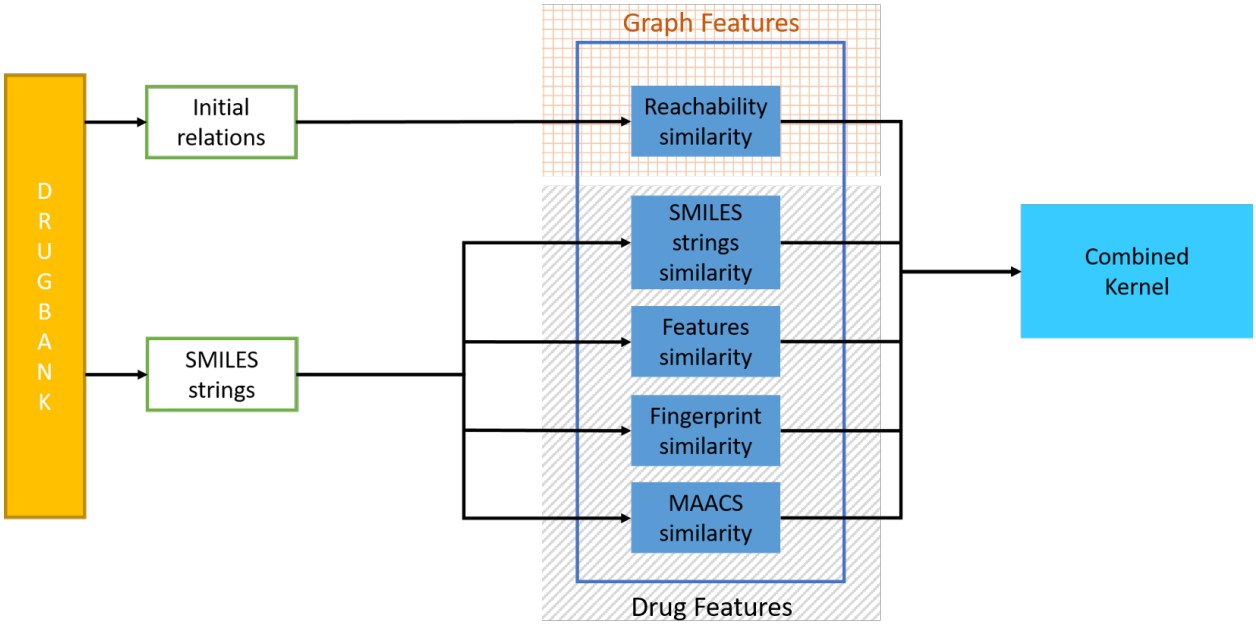


Figure 4.1: Complete pipeline for creation of SKID^3

4.2 Drug-Drug Interactions

Before we define the DDI task, we give terse definitions of various entities involved when two or more drugs interact. The *target* or drug target is the protein modified by the drug in order to achieve the desired effect once the drug is administered to the body. *Enzymes* are catalysts that accelerate biological reactions, while *transporters* are proteins that help drugs reach the intended target (Nigam, 2015), and also help in determining whether the drug will be absorbed, distributed or eliminated.

DDIs can be either *synergistic* (positive, and help increase the effect of the drugs) or *antagonistic* (negative, cause serious side effects). We do not differentiate between these two types of interactions. DDIs themselves can be inherently classified into two categories (August et al., 1997):

- **Pharmacokinetic:** This is the effect that a drug goes through when administered, for example, it is absorbed or metabolized. In case of DDIs, pharmacokinetic refers to the (synergystic or antagonistic) effect of one drug on the other drug's absorption, distribution, metabolism and excretion when administered simultaneously or within a short time span of one another.

- **Pharmacodynamic:** This is the effect that body goes through when a drug is administered. In case of DDIs, pharmacodynamic refers to the effect of one drug on another drug when they are operating on the same target or even different targets, but with similar behaviour towards the different targets i.e do they inhibit the tendency of the the target to act which can cause an unwanted interaction.

The pharmacokinetic category consists of metabolism interactions like enzyme inhibitors and substrates. Target, enzyme and transporter inhibitors are chemical molecules that bind to the target (or enzyme, or transporter resp.), and inhibit its activity. Enzyme/transporter substrates are molecules which react with the enzyme/transporter, and are converted into different molecules called products. The pharmacodynamic category, on the other hand, occurs due to the agonists and

antagonists. *An agonist binds to a target, and evokes a response, while an antagonist binds to the target and inhibits a response.*

We build our approach based on these two categories with the motivation that if two drugs interact, then there should exist a "path of relationships" describing the molecular and structural properties of the drugs, especially when there is an interaction. Thus, we extract relations as shown in Table 4.1 from the DrugBank database, whose general schema is shown in figure 4.2. These relations ensure that we are in the domain of pharmacokinetic and pharmacodynamic categories of the DDIs. Another motivation for using these relations is that the effect of enzymes on DDIs, especially the cytochrome P450, have been well studied extensively in medical literature (Guengerich, 1997; Ogu and Maxa, 2000). Thus, the use of such relations becomes natural in DDI prediction, and can be considered a form of domain expertise.

Table 4.1: Initial relations

Initial Relations
EnzymeSubstrate(drug, enzyme)
EnzymeInhibitor(drug, enzyme)
EnzymeInducer(drug, enzyme)
TargetSubstrate(drug, target)
TargetAntagonist(drug, target)
TargetInducer(drug, target)
TargetAgonist(drug, target)
TransporterSubstrate(drug, transporter)
TransporterInhibitor(drug, transporter)
TransporterInducer(drug, transporter)

4.3 Kernel Learning for Drug Drug Interactions

In classical multiple kernel learning (Bach et al., 2004; Lanckriet et al., 2004), kernels are typically constructed in two different ways. First, multiple kernels can be constructed from the same data source (homogeneous), or from different data sources (heterogeneous). These multiple kernels are then combined in a linear or non-linear fashion. It is important to note that in such a multiple-kernel

learning setting, the assumption is that we have the complete data vectors $\mathbf{x}_i$ from which we can construct multiple kernels. Our method diverges considerably from this representation since we do not have an explicit representation or embedding of the drugs. Instead, we have several different similarity measures, from which we construct a single kernel for our prediction/discovery task.

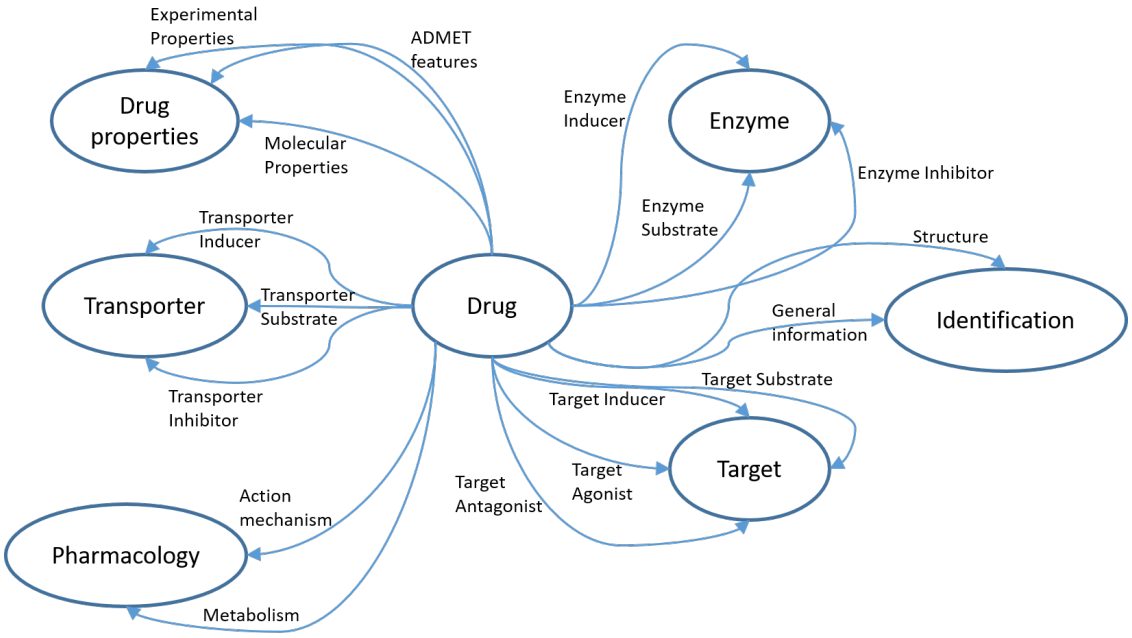


Figure 4.2: A general schema representation of the DrugBank database

4.3.1 Drug-Drug Similarity Measures

Graph feature: Reachability

A key component describing drug-drug interactions is the charaterization of how two drugs react with each other. This is captured using a *directed graph of known chemical reactions* between drugs and enzymes, transporters etc. using ADMET (absorption, distribution, metabolism, excretion and toxicity) features. The idea of *reachability* follows from the intuition that two drugs are likely to interact with one another if one is reachable from the other via one or more paths in an ADMET knowledge graph.

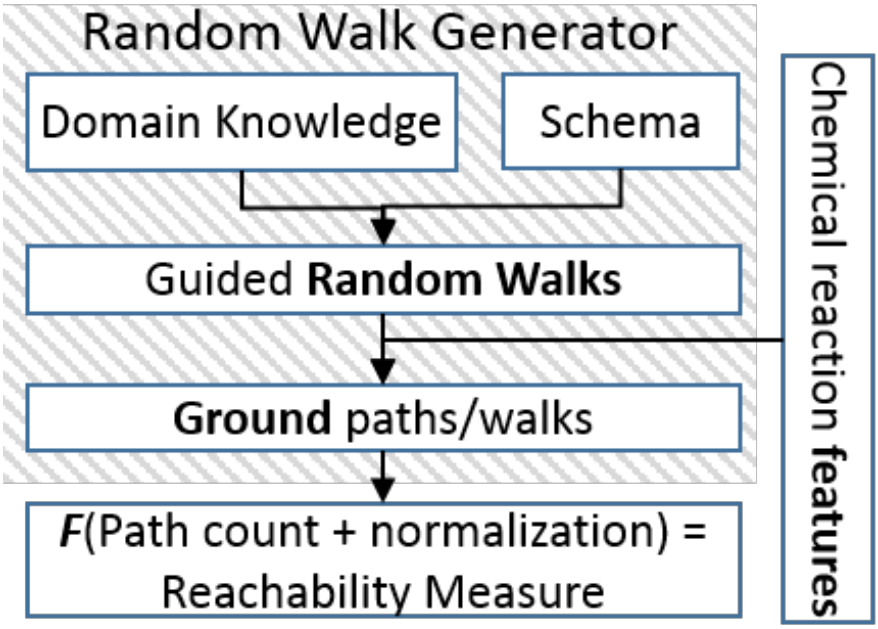
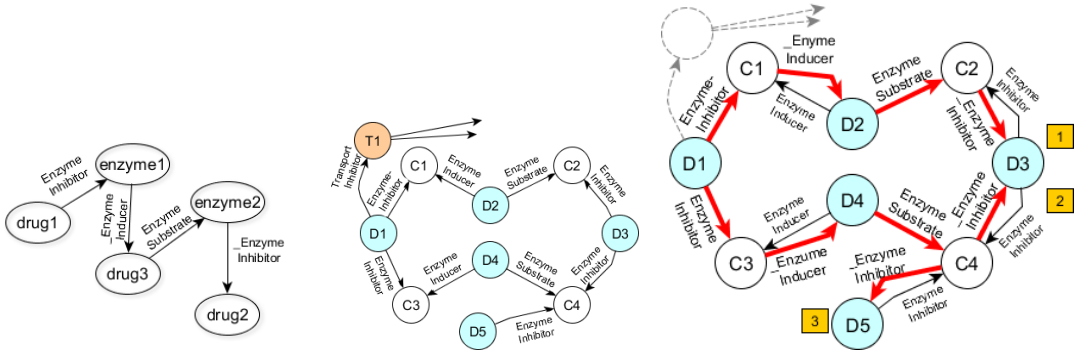


Figure 4.3: Reachability measure generation

Table 4.2: Domain knowledge

EnzymeInhibitor(drug, enzyme) ∧ _EnzymeInducer(enzyme, drug)
EnzymeInhibitor(drug, enzyme) ∧ _TransporterInhibitor(transporter, drug)
EnzymeInhibitor(drug, enzyme) ∧ _EnzymeInhibitor(enzyme, drug)
EnzymeInhibitor(drug, enzyme) ∧ _EnzymeInhibitor(enzyme, drug) ∧ EnzymeInhibitor(drug, enzyme)

While there exist numerous approaches in graph theory for reachability analysis on graphs (Lü and Zhou, 2011; Taskar et al., 2004), our choice is guided by the fact that we operate on multi-relational, directed, relatively sparse graphs involving several thousands of entities/nodes representing drugs, enzymes, targets etc. An iterative search within such a large graph may be intractable. We are inspired by the success of randomized approaches in computational statistics and the seminal work on the path ranking algorithm (PRA, (Lao and Cohen, 2010b)). These approaches show that random walks on a knowledge graphs can be used to generate robust predictive models for relation extraction and reachability analysis. We adapt a similar approach to construct our reachability measure. The estimation of reachability between 2 drugs in a given drug pair proceeds as follows (Figure 4.3):



(a) Parameterized Walk '$\mathbb{W}$' (b) Network of reactions '$\mathcal{G}$' (c) Identified paths/instances in the graph

Figure 4.4: Instantiation process of a parameterized random walk $\mathbb{W}$ (left) is equivalent to sub-graph matching for a given motif. The graph $\mathcal{G}$ (middle) shows a part of the chemical reaction network (D_x, C_x & T_x indicate drugs, enzymes and transporters resp.). The rightmost figure shows how 3 different instances/paths (marked in **red**) have been identified that satisfy $\mathbb{W}$.

(a) ***Preprocessing***: A knowledge graph is constructed for known chemical reactions using ADMET features.

(b) ***Guided (Parameterized) Random Walk Generation***: Parameterized random walks are sequences of relations with shared arguments, where the arguments are entity classes (not entity instances) starting and ending in the `drug` entity. Essentially, parameterized random walks are paths in the relational schema of chemical reactions (Figure 4.2). Similar to PRA (Lao and Cohen, 2010b), our random-walk generation allows for walking against the implicit direction of the relation. Thus, the relations prefixed with _ represent the inverse of a given relation. An example of a random walk through an ADMET graph looks like: `TargetInhibitor(d0, t0)` $\wedge$ `_TargetInhibitor(t0, d1)` $\wedge$ `TransporterSubstrate(d1, t2)` $\wedge$ `_Transporter Inhibitor(t2, d3)`. We impose certain restrictions on the walks, including disallowing same relation types from following each other (a relation and its inverse are considered different types). We generate several random walks of varying length. Guidance is induced via refining the parameterized walks using domain knowledge (US Food and Drug Administration, 2012) (Table 4.2) that indicate certain types of chemical reactions (or a

series), which when present in the walks, increases the likelihood of an interaction between the two drugs at the start and end of the path.

(c) ***Instantiation***: Instantiation of a parameterized walk, $\mathbb{W}$, is the process of finding all possible paths, satisfying $\mathbb{W}$, that exist in the network of chemical reactions $\mathcal{G}$ between two drugs of a given pair $\langle \mathbf{d}_1, \mathbf{d}_2 \rangle$ (Figure 4.4). If we consider paths as subgraphs, and $\mathbb{W}$ a *motif*, then set of instances $\mathcal{I}_{\langle \mathbf{d}_1, \mathbf{d}_2 \rangle} = \{ \forall g \mid g \subseteq \mathcal{G}, g \models \mathbb{W}, \mathbf{d}_1 \wedge \mathbf{d}_2 \in g \}$. Searching for the set of instances is a combinatorially hard problem ($\#P$-complete). We exploit the power of graph databases to compute this. The network of reactions is represented as an RDF[1] graph and the parameterized walks are posed as SPARQL queries (Grobe, 2009).

(d) ***Measure/Score generation***: The reachability measure is generated for every drug pair $\langle \mathbf{d}_1, \mathbf{d}_2 \rangle$ by obtaining the cardinality (count) of the instance set $\mathcal{I}_{\langle \mathbf{d}_1, \mathbf{d}_2 \rangle}$.

Table 4.3: Few of the groundings generated for the random walk `TransporterSubstrate(d0, t1)` $\wedge$ `Transporter(t1, d1)` $\wedge$ `TargetInhibitor(d1, e1)` $\wedge$ `Enzyme(e1, d2)`

Pravastatin,Multidrug resistance protein 1,Acetaminophen,H synthase 1,Hydromorphone
Metoprolol,Multidrug resistance protein 1,Diclofenac,H synthase 2,Ibuprofen
Venlafaxine,Multidrug resistance protein 1,Acetylsalicylic acid,H synthase 1,Diphenhydramine
Cephalexin,Solute carrier family 22 member 6,Naproxen,H synthase 1,Zolpidem
Levothyroxine,Solute carrier organic anion transporter family member 1C1,Diclofenac,H synthase 1,Hydromorphone

Drug Feature: Similarities based on SMILES and SMARTS strings

The simplified molecular-input line-entry system (SMILES) is a commonly-used specification for describing chemical and molecular structure using ASCII strings. The SMILES arbitrary target specification (SMARTS) is an extension of SMILES that is also commonly used for specifying molecular sub-structures precisely. We extract four similarity measures based on molecular and chemical properties of the drug (specified by SMILES and SMARTS strings) using the package

[1]The Resource Description Framework (RDF) was developed by the WWW Consortium (W3C) for knowledge representation and management on the web.

`rdkit`[2]. We compute four similarity measures from SMILES strings (Anderson et al., 1987), which have been previously proven useful in various bio-computing tasks (Helma et al., 2004; Arimoto et al., 2005; Cao et al., 2012):

(S1) Molecular **Feature Similarity** (FS) compares the chemical properties of two drugs using 19 features extracted from their SMILES strings. These features include the number of valence electrons, number of aromatic rings and number of hydrogen donors and receptors, which are important for determining the reactiveness of a molecule. We use the Jaccard distance between all features as the similarity between two drugs.

(S2) SMILES **String Similarity** (SS) is the similarity between the SMILES strings themselves, which is calculated using edit distance between the strings.

(S3) **Molecular Fingerprint** (FP) similarity is computed between the fingerprints, which are bit-string representations of the molecular structure.

(S4) **Molecular ACCess System** (MACCS) keys are a particular type of fingerprint generated from SMARTS strings. Similarities on MACCS are commonly used in the drug discovery domain, though they have been proven to be useful on the DDI domain as well (Vilar et al., 2014).

4.3.2 Notation and Problem Description

Before describing our approach in detail, we formalize our notations. Given a drug database with N drugs, our goal is to discover whether a pair of drugs $\mathbf{d}_i$ and $\mathbf{d}_j$ **interact** with each other. Recall that we do not distinguish between synergistic and antagonistic interactions. Let all possible drug pairs in the database be $\mathcal{P} = \{ (\mathbf{d}_i, \mathbf{d}_j) \mid 1 \leq i, j \leq N \}$. We use the short-hand notation ij to denote the drug-drug pair $(\mathbf{d}_i, \mathbf{d}_j)$. As mentioned previously, our problem setting is considerably

[2] `http://www.rdkit.org/`

different from the classical multiple kernel learning framework. We do not attempt to construct an explicit vector representation or embedding of a drug $\mathbf{d}_i$. Instead, given N drugs, we construct M pairwise similarity matrices S_m, for $m = 1, \ldots, M$. As described above, these similarities can be constructed using various drug properties that represent the potential for interactions such as molecular structure, pharmacological attributes etc. Since these "similarities" represent potential for interactions, they can also be constructed from natural language text extracted from such diverse sources as electronic health records (Page et al., 2012) and journal abstracts (Odom et al., 2015).

Our approach seeks to combine different interaction measures and similarities, S_m, from various sources into one coherent kernel. Note that the only requirement on the similarity matrices is that $S_m \in \mathbb{S}^N$, the space of all $N \times N$ *symmetric* matrices. We *do not assume positive semi-definiteness* (psd[3]) of similarity measures; as we show below, it is possible to align a psd kernel with non-psd similarity matrices. Thus, any symmetric scoring function $\sigma_m(\mathbf{d}_i, \mathbf{d}_j)$ can be used to generate a similarity matrix S_m. This allows our approach to be *agnostic to multiple representations* of a drug. For example, σ_1 can be string alignment similarity of the genomic strings of two drugs, while σ_2 can be the bag-of-words co-occurence count of the two drugs in a biomedical corpus. Broadly, any scoring function that measures similarity of a potential for interaction can be considered a candidate similarity measure.

The (i, j)-th element of S_m is denoted s_m^{ij}, and describes the interaction between $\mathbf{d}_i$ and $\mathbf{d}_j$ according to interaction measure S_m. The interaction label $y_{ij} = +1$ if the drugs $\mathbf{d}_i$ and $\mathbf{d}_j$ interact adversely with each other and $y_{ij} = -1$ otherwise. We denote the matrix of all drug-drug interactions as $Y \in \mathbb{S}^N$, the symmetric matrix whose (i, j)-th entry is the interaction label y_{ij}. Generally, we only know the true labels for a small subset of drug pairs, $\mathcal{L} \subset \mathcal{P}$, and our goal is to learn a model on $\mathcal{L}$ in order to discover drug-drug interactions in the remaining pairs $\mathcal{U} = \mathcal{P} \setminus \mathcal{L}$. Our problem can be formulated as follows:

[3]A symmetric matrix is positive semi-definite if its eigenvalues are all non-negative (≥ 0), and positive definite if its eigenvalues are strictly positive (> 0). Positive semi-definiteness allows us to manipulate kernels instead of explicitly transforming the data into a higher dimensional space.

Given: For N drugs, M interaction similarities S_m, a small subset of *known* interactions y_{ij} for pairs $ij \in \mathcal{L} \subset \mathcal{P}$,

Learn: A kernel $Z \succeq 0$, and interaction similarity combination weights $\alpha_m \geq 0$, $\sum_{m=1}^{M} \alpha_m = 1$,

Predict/Discover: Previously unknown pairwise drug-drug interactions $\hat{y}_{ij} = \mathsf{sign}(z_{ij})$, for pairs $ij \in \mathcal{U} = \mathcal{P} \setminus \mathcal{L}$.

Our novel formulation addresses kernel learning at an element-wise, local and global level, enabling us to learn robust models for discovery of new drug-drug interactions.

4.3.3 Incorporating Neighborhood Information

We view each interaction/similarity measure as a *graph* that provides a *different view of the neighborhood* of a drug. That is, each similarity matrix S_m represents a fully-connected graph with s_m^{ij} representing the edge weight between drugs $\mathbf{d}_i$ and $\mathbf{d}_j$. Since each S_m measures similarities differently, the neighborhood of a drug $\mathcal{N}_m(\mathbf{d}_i)$ with respect to different S_m will be different. In order to effectively incorporate this multi-view neighborhood information, we construct graph Laplacians L_m, $m = 1, \ldots, M$, for each similarity. Laplacians are naturally locality-preserving (He and Niyogi, 2003; Belkin and Niyogi, 2003), that is, they preserve the neighborhood structure in the data. This allows us to learn a kernel that fuses neighborhood information $\mathcal{N}_m$ from multiple interaction types. Without loss of generality, we set the diagonal of S_m to zero: $\mathsf{diag}(S_m) = \mathbf{0}$, reflecting that drugs do not interact with themselves. The Laplacian can be constructed as

$$L_m = (1 + \delta)I_N - D^{-\frac{1}{2}} S_m D^{-\frac{1}{2}}, \tag{4.1}$$

where I_N is an $N \times N$ identity matrix and D is a diagonal matrix with entries $d_{ii} = \sum_{j=1}^{N} s_{ij}^m$ (the row sum of the similarity matrix S_m).

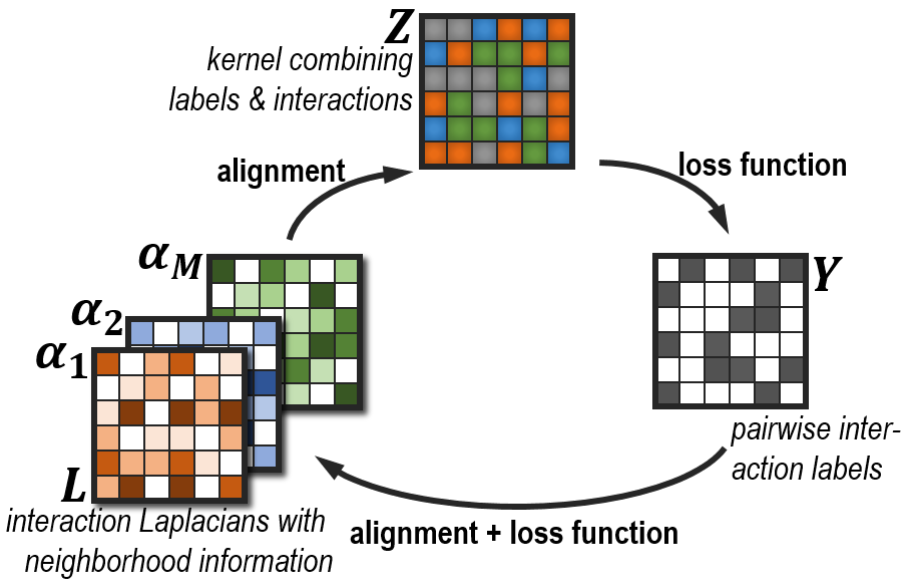


Figure 4.5: Our formulation aims to learn a positive semi-definite kernel Z and combination weights α_m for various similarity measures. These similarities represent interaction scores, which help determine how likely two drugs are to interact. The similarities can be constructed from diverse sources (such as molecular, structural, genomic, text). The similarity measures are expressed through Laplacians, which view the interactions as a *neighborhood graph*. In this manner, we can incorporate local information into the kernel. The loss functions ensure that the learned Z is element-wise consistent with the labels.

We formulate the following kernel learning problem:

$$\begin{aligned} \underset{L,\,Z,\,\boldsymbol{\alpha}}{\text{minimize}} \quad & \overbrace{\langle L,\,Z\rangle \,+\, \langle L,\,Y\rangle}^{\text{alignment}} + \overbrace{\lambda_1 r(\alpha)}^{\text{regularization}} \\ & \underbrace{\lambda_2\left(\ell_1(Z,\,Y) \,+\, \ell_2(L,\,Y)\right)}_{\text{loss functions}} \\ \text{subject to} \quad & L \,=\, \sum_{m=1}^{M} \alpha_m L_m,\; \boldsymbol{\alpha} \geq 0,\; \mathbf{e}'\boldsymbol{\alpha} = 1,\; Z \succeq 0. \end{aligned} \tag{4.2}$$

We highlight the various components of the formulation (4.2):

The variable $L = \sum_{m=1}^{M} \alpha_m L_m$ is a *convex combination* of the Laplacians L_m arising from the various interaction similarities. The matrix variable L is introduced purely for convenience of notation and can easily be eliminated from the objective function of (4.2). We select a convex combination as against a linear or conic combination in order to *improve interpretability* (Gönen and Alpaydın, 2011). That is, positive α_m enable us to intuitively interpret the importance of one

similarity relative to the others. The formulation attempts to identify a combination weights $\boldsymbol{\alpha}$ as well as a kernel $Z \succeq 0$, which ensures that Z is psd.

The *alignment* terms are inspired by the success of alignment-based regularization for kernel learning (Cortes et al., 2012). Hoi et al. (2007) observe that these alignment terms essentially perform manifold regularization (Belkin et al., 2004), which has the effect of incorporating local neighborhood information encoded in the different Laplacians as well as the labels into learning $\boldsymbol{\alpha}$ and Z. Specifically, $\langle L,\, Y \rangle$ encourages the weights on the Laplacians α to be consistent with the labels Y. The impact of labels is also propagated into Z by the $\langle L,\, Z \rangle$ term.

The entries of the learned kernel z_{ij} directly provide a unified interaction score and we predict drug interactions as

$$\hat{y}_{ij} \;=\; \mathsf{sign}(z_{ij}). \tag{4.3}$$

While the unified kernel Z is positive semi-definite, it's entries can still be negative, which is a fact that we exploit here. Enforcing positive semi-definiteness also naturally imposes symmetry on the learned kernel.

In order that the elements of Z capture interactions effectively into a score, we require a loss function that ensures that the *interaction margin* is maximized. We use the *hinge loss* to ensure that $y_{ij} z_{ij} \;\geq\; 1$ holds. Intuitively, these constraints ensure that $z_{ij} \;\geq\; 1$ when $y_{ij} \;=\; +1$ and $z_{ij} \;\leq\; -1$ when $y_{ij} \;=\; -1$. The interaction margin behaves very similarly to the margin in SVMs (Shawe-Taylor and Cristianini, 2004). Thus, we select ℓ_1 to be the hinge loss in (4.2), which is applied to the drug pairs with known labels (indexed by) $ij \in \mathcal{L}$:

$$\ell_1(Z,\, Y) \;=\; \sum_{ij \in \mathcal{L}} \max(1 - y_{ij} z_{ij},\, 0). \tag{4.4}$$

The loss function ℓ_1 ensures element-wise consistency between the learned kernel Z and the labels Y. In a similar vein, the loss function ℓ_2 aims to propagate this consistency into the combination weights $\boldsymbol{\alpha}$. To this end, we measure the element-wise deviation of the weighted Laplacian with the

labels as well, through the Frobenius norm:

$$\ell_2(L,\, Y) \,=\, \frac{1}{2}\,\|L \,-\, Y\|_F^2. \tag{4.5}$$

Finally, we also add a regularization term over $\boldsymbol{\alpha}$, typically to ensure robustness in weight learning. We chose the classical L_2 regularizer, $r(\boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{\alpha}\|_2^2$. Other norms can also be used, depending on what properties of $\boldsymbol{\alpha}$ are desired. For instance, the L_1 regularizer, $r(\boldsymbol{\alpha}) = \|\boldsymbol{\alpha}\|_1$ encourages sparsity, while the L_∞ regularizer, $r(\boldsymbol{\alpha}) = \|\boldsymbol{\alpha}\|_\infty$ encourages the model to select the single best kernel. We use L_2 regularization here, and defer the exploration of the properties of the other regularizers to future work.

We formulate the following kernel learning problem:

$$\begin{aligned}
\underset{L,\, Z,\, \boldsymbol{\alpha}}{\text{minimize}} \quad & \langle L,\, Z\rangle \,+\, \langle L,\, Y\rangle \,+\, \frac{\lambda_1}{2}\,\|\boldsymbol{\alpha}\|_2^2 \\
& \lambda_2 \sum_{ij \in \mathcal{L}} \xi_{ij} \,+\, \frac{\lambda_2}{2}\,\|L \,-\, Y\|_F^2 \\
\text{subject to} \quad & y_{ij} z_{ij} - 1 + \xi_{ij} \,\geq\, 0,\; \xi_{ij} \,\geq\, 0,\; \forall\, ij \in \mathcal{L}, \\
& L \,=\, \sum_{m=1}^{M} \alpha_m L_m,\; \boldsymbol{\alpha} \geq 0,\; \mathbf{e}'\boldsymbol{\alpha} = 1,\; Z \succeq 0.
\end{aligned} \tag{4.6}$$

The *slack variables* $\xi_{ij} \,\geq\, 0$ measure the hinge loss of the pairwise interaction fit between the labels and the entries of Z as shown in equation (4.4). These slack variables function in a manner very similar to the slack variables in SVMs: if the prediction z_{ij} and the label y_{ij} have the same sign, then the model correctly identifies the interaction for drugs $\mathbf{d}_i$ and $\mathbf{d}_j$. In this case, we will have, $y_{ij} z_{ij} > 0$ and consequently, $\xi_{ij} \,=\, 0$. However, for misidentified interactions, $\xi_{ij} \,=\, 1 \,-\, y_{ij} z_{ij} > 0$. Thus, by minimizing ξ_{ij}, we are able to minimize the misclassification of drug-drug interactions. The formulation (4.6) is an instance of a bilinear program, owing to the terms $\langle L,\, Z\rangle \,=\, \sum_{m=1}^{M} \,\alpha_m \,\langle L_m,\, Z_m\rangle$.

We solve (4.6) using *alternating minimization* (Csiszár and Tusnády, 1984). At the t-th iteration, we fix the current estimate of the similarity weights $\hat{\boldsymbol{\alpha}}^t$ (note that when $\boldsymbol{\alpha}$ are fixed, this also fixes L,

owing to the equality constraint in eq. 4.6). This allows us to infer the new interactions scores $\hat{Z}_{ij}^{t+1}$ by solving the following sub-problem, which we denote $\mathsf{SubProbE}(Z \mid \hat{\boldsymbol{\alpha}}^t)$. This can be interpreted as the expectation step of an EM procedure, where we identify the hidden variables, in this case, the drug-drug interactions Z. We can now fix $Z = \hat{Z}^{t+1}$ in (4.6), which gives us a sub-problem we denote $\mathsf{SubProbM}(\boldsymbol{\alpha} \mid \hat{Z}^{t+1})$. Again, this step can be considered equivalent to the maximization step of an EM procedure, where we estimate the parameters (here, $\boldsymbol{\alpha}$, which parameterize the influence of the various similarities on the final kernel). This procedure is summarized in Algorithm 10. Both sub-problems were solved using `SDPT3` (Toh et al., 1999).

Algorithm 2 Alternating Minimization for Learning `SKID`^{3}

$\hat{\boldsymbol{\alpha}}^0 = 1/m$ ▷ Initialize weights uniformly
$\hat{Z}^0 = I_N$ ▷ Initialize kernel to identity matrix
for $t \leq t_{\mathsf{max}}$ **do**
  $\hat{Z}^{t+1} \leftarrow \mathsf{SubProbE}(Z \mid \hat{\boldsymbol{\alpha}}^t)$ ▷ Update Z
  $\hat{\boldsymbol{\alpha}}^{t+1} \leftarrow \mathsf{SubProbM}(\boldsymbol{\alpha} \mid \hat{Z}^{t+1})$ ▷ Update $\boldsymbol{\alpha}$
  if $\frac{1}{N^2} \|\hat{Z}^{t+1} - \hat{Z}^t\|_F^2 + \frac{1}{m} \|\hat{\boldsymbol{\alpha}}^{t+1} - \boldsymbol{\alpha}^t\|_2^2 \leq \tau_{\mathsf{tol}}$ **then**
    break ▷ Converged to tolerance
  end if
  $t \leftarrow t + 1$
end for

4.4 Experiments

In this section, we aim to answer the following questions, which address the effectiveness of our proposed approach:

(**Q**1) How effective are the similarity measures on their own for the task of identifying drug-drug interactions?

(**Q**2) Is kernel learning effective for the DDI task?

(**Q**3) Is combining multiple similarity measures more advantageous than using a single similarity measure? How do the learned weights change with increasing database size?

(**Q**4) Does our work motivate further clinical investigations?

(**Q**5) How scalable is our method?

Our data set consists of 78 drugs obtained from DrugBank[4]. This gives rise to 3003 possible interactions[5]. All our reported results were obtained across five runs with a held-out test set of 600 drug pairs. Different methods were trained with increasing number of drug pairs ranging from 400 to 2400, chosen randomly for each run.

The results of our experiments are shown in Figure 4.6. Figures 4.6(a)–4.6(d) show that a kernels learned from each individual similarity measure (described in Sec. 4.3.1 and 4.3.1) are able to perform reasonably well on the DDI prediction task, thereby answering **Q1** affirmatively.

We also learn a single kernel ($Z \equiv \texttt{SKID}^3$) as well as the weights for the five similarity measures (α_m). It is evident that learning from multiple similarity measures provide a more stable learning curve that performs well. Our initial hypothesis was that the similarities generated from molecular structures (SS, FS, FP and MACCS) and chemical reaction pathways (RW) fused into a single kernel could combine the advantages of both. That is, our hypothesis was that similarity fusion should achieve the high precision of the molecular structures similarity as well as the high recall of the chemical reaction pathways similarity. The results clearly confirm this, thereby answering **Q2** and **Q3** affirmatively. Figure 4.6(e) shows the change of the learned weights as the number of training drug pairs increases. A key observation from Figure 4.6(e) is that the influence of the random walk (RW) similarity decreases, while the weight of the molecular structure similarities increases. This suggests that RW similarities are particularly effective in smaller databases, for targeted identification of interactions.

Q1–**Q3** evaluate the performance of our approaches and *confirm existing interactions* as provided by DrugBank. Our goal with **Q4** was to see if $\texttt{SKID}^3$ is able to *discover new interactions*. In order

[4] `https://www.drugbank.ca/`

[5] Given n drugs, since each drug can interact with every other drug except itself, there will be a total of $\binom{n}{2} = \frac{n(n-1)}{2}$ interactions.

to answer **Q4**, it is necessary that our analysis goes beyond ground truth that we are considering in constructing the model. Thus, we look closely at the false positives and false negatives, under the intuition that DrugBank (or any other database) is never fully complete or accurate.

Table 4.4: Table depicting example drug pairs where the prediction does not match the ground truth. However, we additionally cite sources (last column) that support our prediction.

Drug 1	Drug 2	DrugBank Ground Truth	Predicted Classification	Independent Source
Amitriptyline	Tamsulosin	Not interacting	Interacting	Drugs.com (Multiple, a)
Omeprazole	Metformin	Not interacting	Interacting	Nies et al. (Nies et al., 2011), rxlist.com (Multiple, c)
Salbutamol	Clonidine	Not interacting	Interacting	Thoolen et al. (Thoolen et al., 1984)
Cephalexin	Diclofenac	Not interacting	Interacting	Ali et al. (Ali et al., 2015)
Amoxicillin	Metronidazole	Not interacting	Interacting	Pavicic et al. (Pavicić et al., 1991)
Amphetamine	Salbutamol	Not interacting	Interacting	DrugBank
Cephalexin	Methadone	Interacting	Not interacting	Drugs.com (Multiple, b)

In Table 4.4 we present *a few drug pairs* that are supposedly "incorrectly classified" by our method using the DrugBank ground truth[6]. Table 4.4 shows that the interactions discovered by our approach can be supported by independent sources or research. Specifically, according to the ground truth, 6 interactions were flagged as false positives. On the contrary, according to literature, these are likely true interactions. Thus, we answer **Q4** affirmatively. This is a crucial observation in the task of drug surveillance: many sources of DDIs need to be carefully and continuously curated for updating this ground truth. This result highlights the fact that SKID^3 can indeed not only classify DDIs, but can help in knowledge refinement as well as knowledge discovery. Validating this hypothesis more fully requires large-scale evaluation, which is an interesting direction for future research.

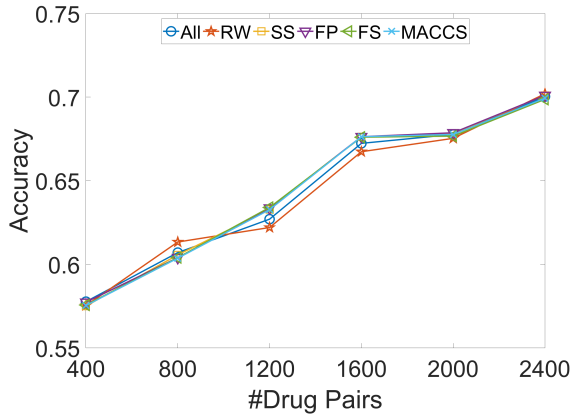
Finally, Figure 4.6(f) shows the time taken by our method. The training time increases linearly with the number of drug pairs, showing the scalability of our method and answering **Q5** affirmatively. This result has practical implications for scalable DDI discovery with full drug databases.

[6]In the previous instance of the Drugbank database download in April 2017, this instance was not present whereas in February 2018, when checked again, this interaction was added. We use the previous instance as ground truth.

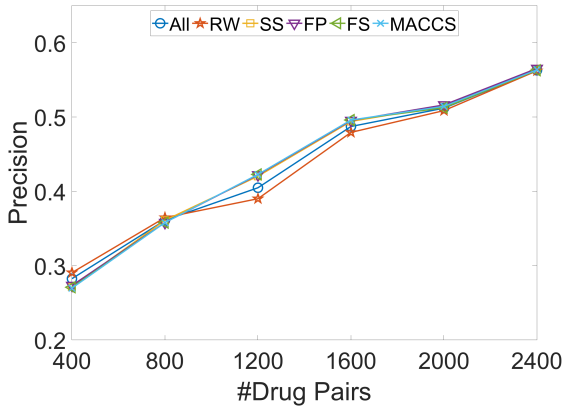
4.5 Conclusion and Future work

We consider the problem of drug-drug interaction discovery, and develop a framework to exploit deeper structures and drug features using kernel learning. Our extensible framework can fuse information from multiple views including chemical reaction pathways and molecular structure, which we have demonstrated here. Furthermore, our formulation can easily admit other types of interactions as similarities including phenotypic, pharmacological, genomic and text, to name a few.

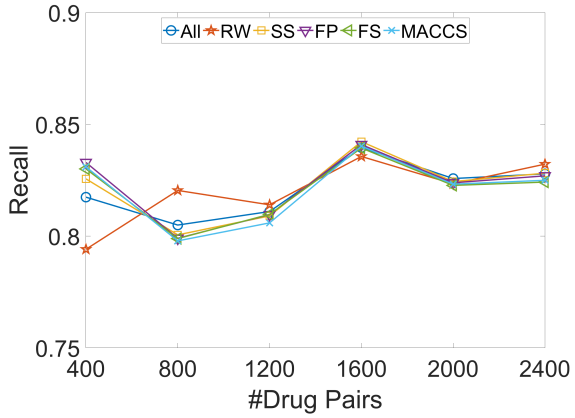
Our evaluations on the DrugBank database established the superiority of our proposed approach, which is distinct from many current approaches that generally ignore drug properties and instead seek interactions through text mining of existing literature. A potential limitation of our approach is that our optimization function is inherently noisy owing to its non-convex nature. Extending this work to include more features including other semantic similarity metrics is an interesting direction. Combining the results of learning from DrugBank with other NLP based extraction techniques is another direction. Finally, using other labeling techniques such as weak supervision or distant supervision can potentially lead to larger training sets and can make the discovery process more effective.



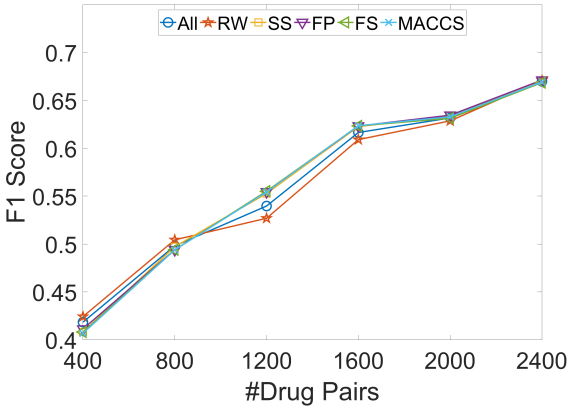
(a) Accuracy, averaged over 5 runs on a hold-out test set



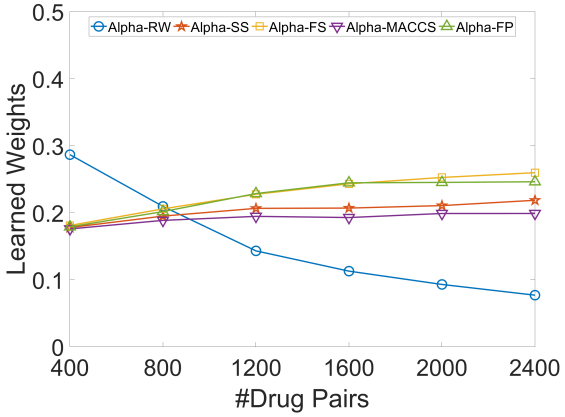
(b) Precision, averaged over 5 runs on a hold-out test set



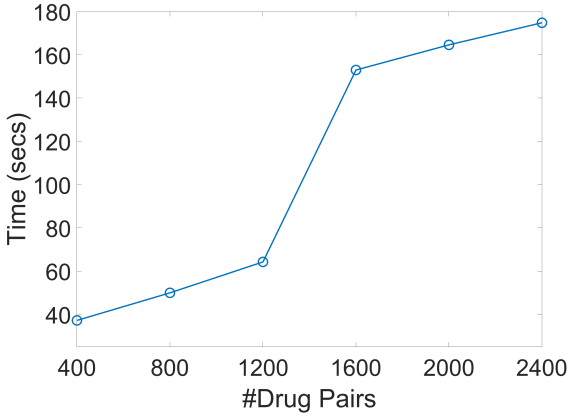
(c) Recall, averaged over 5 runs on a hold-out test set



(d) F_1-score, averaged over 5 runs on a hold-out test set



(e) Similarity matrix weights, $\boldsymbol{\alpha}$



(f) Training Time for learning the *combined* kernel, SKID^3

Figure 4.6: Experimental results, with kernels learned using **All** similarity measures (SKID^3), along with each similarity.

PART II

LEARNING FROM MULTI-RELATIONAL DATA

CHAPTER 5

LEARNING LOCAL NEIGHBORHOOD BASED MODELS

In this chapter we consider, the problem of structure learning for Gaifman models as learning relational features that can be used to derive feature representations from a given knowledge base. These relational features are first-order rules that are then grounded and counted over local neighborhoods of a Gaifman model to obtain the feature representations. We propose a method (Dhami et al., 2020) for learning the relational features for a Gaifman model by using relational tree distances.

5.1 Introduction

Learning embeddings of large knowledge bases has become a necessity due to the importance of reasoning about objects, their attributes and relations in large graphs. Statistical Relational AI/Learning (StaRAI) (Raedt et al., 2016; Getoor and Taskar, 2007), addresses the problem of learning and reasoning with multi-relational data in the presence of uncertainty. While specific models such as Markov Logic (Richardson and Domingos, 2006), ProbLog (De Raedt et al., 2007) and PSL (Bröcheler et al., 2010) (to name a few) exist, a more scalable model (Niepert, 2016) was proposed recently. This work built on Gaifman's locality theorem (Gaifman, 1982; Grohe and Wöhrle, 2004), which states that every first-order sentence is equivalent to a Boolean combination of sentences whose quantifiers range over local neighborhoods of the Gaifman graph. The key idea is that if one could identify effective representations from local neighborhoods (of objects or tuples of objects), one could learn machine learning models that can be used for reasoning in large graphs. This "local representation" approach was inspired by the success and scalability of convolutional neural networks (CNNs, (Goodfellow et al., 2016a)), specifically, the ability of CNNs to engineer complex features from locally-connected image neighborhoods.

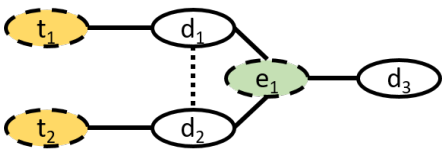
In a similar manner, relational Gaifman models seek to identify locally-connected relational neighborhoods within knowledge bases for effective representation, learning and inference. While

effective, the relational learning model recently proposed by Niepert (Niepert, 2016), called **Discriminative Gaifman Models**, used relational features that were hand-crafted rather than learned. That is to say that, structure learning (to use the terminology from probabilistic graphical models) was not performed.

We address this problem of structure learning: learning relational features for training the Gaifman model. We consider three different approaches. (1) As suggested by Niepert (2016), we employ Inductive Logic Programming (ILP) (Muggleton, 1991) to learn discriminative first-order rules; (2) Inspired by the success of random walks in deep relational models (Lao and Cohen, 2010a; Lao et al., 2011; Kaur et al., 2017a), we employ relational random walks; (3) Finally, as a novel contribution, we propose the use of paths from relational trees learned via relational one-class classification (Khot et al., 2014); specifically, each path from root to leaf of a relational tree can be considered a relational feature. Given these relational features, we apply traditional discriminative learning algorithms such as Gradient Boosting and Logistic Regression.

We make the following key contributions. (1) We present a method for *learning relational embeddings* for reasoning over large graphs. (2) We adapt a recently developed relational learning method for constructing relational features. (3) We adapt well-known relational rule learners for learning local neighborhood representations. (4) We combine these relational features with discriminative classifiers to learn discriminative Gaifman models. (5) We demonstrate that combining the more novel relational trees with a discriminative classifier is more effective in learning on large graphs compared to a standard ILP learner. (6) Our empirical evaluation reveals an important characteristic of our approach: high recall without sacrificing precision in both medical and imbalanced data sets. **This is the first work on structure learning for Gaifman models**. Given the importance of local neighborhoods in graphs, this is an important direction.

Discriminative Gaifman Models: We first introduce some basic notation and concepts in first-order logic. An **atom** is of the form $\mathtt{R}(t_1, \ldots, t_k)$ where $\mathtt{R}$ is a functor and the arguments t_i are **terms**.



TransportSubstr(Pravastatin, BileSaltExportPump)
TransportInhib(Simvastatin, MultidrugResProt1)
EnzymeInhib(Pravastatin, CytochromeP4502C9)
EnzymeSubstr(Acetaminophen, CytochromeP4502C9)
EnzymeInhib(Simvastatin, CytochromeP4502C9)

Figure 5.1: An example Gaifman graph for a drug-drug interaction (DDI) knowledge base. Here d_1, d_2, d_3 = {Pravastatin, Simvastatin, Acetaminophen}, t_1={Bile salt export pump}, t_2={Multidrug resistance protein 1} and e_1={Cytochrome P450 2C9}. Note that the dotted line between d_1 and d_2 is the link we want to predict.

A **substitution** is of the form $\boldsymbol{\theta} = \{\langle x_1, \ldots, x_k\rangle / \langle t_1, \ldots, t_k\rangle\}$ where x_is are logical variables and t_is are terms. A **grounding** of a predicate with logical variables $x_1, \ldots, x_k$ is a substitution $\{\langle x_1, \ldots, x_k\rangle / \langle X_1, \ldots, X_k\rangle\}$ mapping each of its variables to a constant in the population of that variable. A **literal** is an atom or its negation. A **formula** φ is a literal, the conjunction of two formulae $\varphi_1 \wedge \varphi_2$, or a disjunction of two formulae $\varphi_1 \vee \varphi_2$. The application of a **substitution** $\boldsymbol{\theta} = \{\langle x_1, \ldots, x_k\rangle / \langle t_1, \ldots, t_k\rangle\}$ on a formula φ is represented as $\varphi\boldsymbol{\theta}$ and replaces each x_i in φ with t_i. An **instance** of a formula φ is obtained by replacing each logical variable x in φ by one of the objects in its domain. A **conjunctive formula** contains no disjunction. For a predicate, $\mathtt{R}(v_1, \ldots, v_k)$, the predicate **counts** are the number of true instances of that predicate, given the grounding $\boldsymbol{\theta}$ of the its variables; we denote such predicate counts as $\#(\mathtt{R} \mid \boldsymbol{\theta})$. For a clause $\mathtt{C}$, the clause counts are the number of true instances of $\mathtt{C}$ in database $\mathcal{F}$, given the partial groundings $\boldsymbol{\theta}$ of the variables in the clause. We denote the clause counts for a clause $\mathtt{C}$ as $n_{\mathtt{C}} = \#(\mathtt{C} \mid \boldsymbol{\theta})$.

A **knowledge base** $\mathcal{B}$ consists of (1) a finite domain of objects $\mathcal{D}$ (also known as entities), (2) a set of predicates $\mathcal{R}$ that describe the attributes and relationships of the objects, and (3) an interpretation that assigns a truth value to every grounded predicate. The Gaifman graph $\mathcal{G}$, also known as the primal graph, of a knowledge base $\mathcal{B}$ is an **undirected graph**, where the nodes are the entities $e \in \mathcal{D}$. $\mathcal{G}$ contains edges joining two nodes only if the entities a and b corresponding to those nodes are present in a relation together $\mathtt{R}(\ldots, a, \ldots, b, \ldots) \in \mathcal{B}$. $\mathcal{G}$ can be used to easily identify co-occurrences (or lack thereof) among every pair of entities in $\mathcal{B}$. Furthermore, cliques in $\mathcal{G}$ group

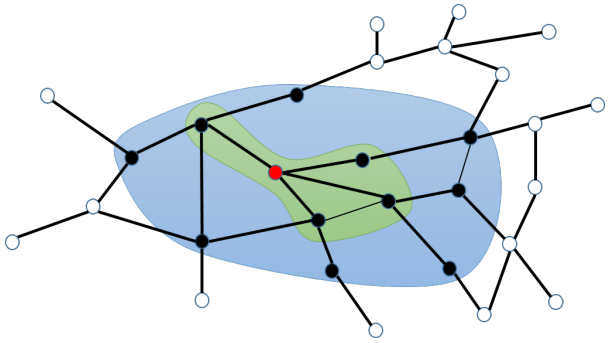


Figure 5.2: Gaifman neighborhoods.

entities that co-occur pairwise through shared relationships, and such cliques capture the local structure of a knowledge base. We illustrate this in Figure 5.1, which shows a knowledge-base fragment and the corresponding Gaifman graph for drug-drug interaction (DDI). Given entities (drugs, enzymes, transporters) and relations between them, the underlying machine-learning task is to predict if two drugs interact. The dotted line represents the target predicate, and identifying it is **link prediction**.

The distance $d(a,\, b)$ between two nodes $(a,\, b) \in \mathcal{G}$ is the minimum number of hops required to reach node b from node a. For example, in Fig. 5.1, $d(d_1,\, t_1) = 1$ and $d(d_1,\, d_2) = 2$. The r-neighborhood of a $a \in \mathcal{G}$ is the set of all nodes that are at most a distance r from a in the Gaifman graph: $N_r^{\mathcal{G}}(a) = \{\bar{a} \in \mathcal{G} \mid d(a, \bar{a}) \leq r\}$. For example, $N_1(d_1) = \{t_1, e_1\}$ and $N_2(d_1) = \{t_1, e_1, d_2, d_3\}$. Figure 5.2 shows the 1 and 2-neighborhood of a node (colored red) in a given Gaifman model. When a first-order rule/clause $\varphi(x)$ is relativized by the neighborhood of the free variable x, the resulting first-order rule $\psi^{N_r(x)}(x)$ is called r-local. A Gaifman neighborhood can be thought of as representing second-order proximity between nodes. The interpretation is that nodes with shared neighbors are more likely to be similar and as a result more likely to have a link between them.

Discriminative Gaifman Models (DGMs, (Niepert, 2016)) are relational models that can exploit structural features of a local neighborhood of a knowledge base. These structural features are aggregated from locally-sampled neighborhoods, and the aggregation is based on the *Gaif-*

man locality theorem (Gaifman, 1982) stated as: *Every first-order sentence is logically equivalent to a Boolean combination of basic r-local sentences*. An r-local sentence is of the form $\exists x_1 \ldots \exists x_k \left(\bigwedge_{1 \leq i < j \leq k} d(x_i, x_j) > 2r \ \wedge \ \bigwedge_{1 \leq i \leq j} \varphi(x_i) \right)$, where $r,\ j \geq 1$ and φ is an r-local first order formula. In simpler terms, the locality theorem states that only a small part of a given structure is relevant for evaluating a query statement and thus a global structure search is not required. For example, if querying about the drug d_1 in Figure 5.1, a search within the 1-neighborhood of e_1 (say), that is $\{t_1,\ e_1\}$ is more relevant than searching through the complete graph which can be greatly computationally inefficient. Another way to look at the theorem is: *a first-order rule is true if it is true in the local r-neighborhoods of a given graph*. The DGM approach uses the Gaifman locality theorem to generate low-level embeddings for a given knowledge graph, which can then be used as propositional features in a standard classifier.

5.2 Learning Discriminative Gaifman models

Given: A knowledge base $\mathcal{B}$, facts F_s, and its corresponding Gaifman graph $\mathcal{G}$;
Output: A DGM $\mathcal{M}$ that is trained for a particular link prediction task $\mathcal{T}$;
To Do: Construct a set of relational features Φ, and train a discriminative learner to predict $\mathcal{T}$.

Our approach, *Learning Gaifman Embeddings* (LGE), (1) constructs rules Φ that form the base set of relational features; (2) instantiates rules and performs counting based on task $\mathcal{T}$ to construct propositional features $\mathcal{F}$; and finally, (3) learns a discriminative classifier with $\mathcal{F}$ (Figure 5.3).

5.2.1 Learning Relational Rules

Given a knowledge base $\mathcal{B}$, the Gaifman graph $\mathcal{G}$ is obtained by instantiating the entities that are connected by an edge type (relation) together in the form $\mathtt{R}(\mathtt{e}_1,\ \mathtt{e}_2)$, that is, $relation(type_1,\ type_2)$. The relation (link) to be predicted, defined by the target predicate, forms the set of positive examples. We make the *closed-world assumption*, that is, unobserved edges in the graphs are negative examples. Each relational example also has *facts* associated with it, which are the ground predicates in $\mathcal{B}$ that describe relational example, its attributes and relationships. All such facts are denoted F_s.

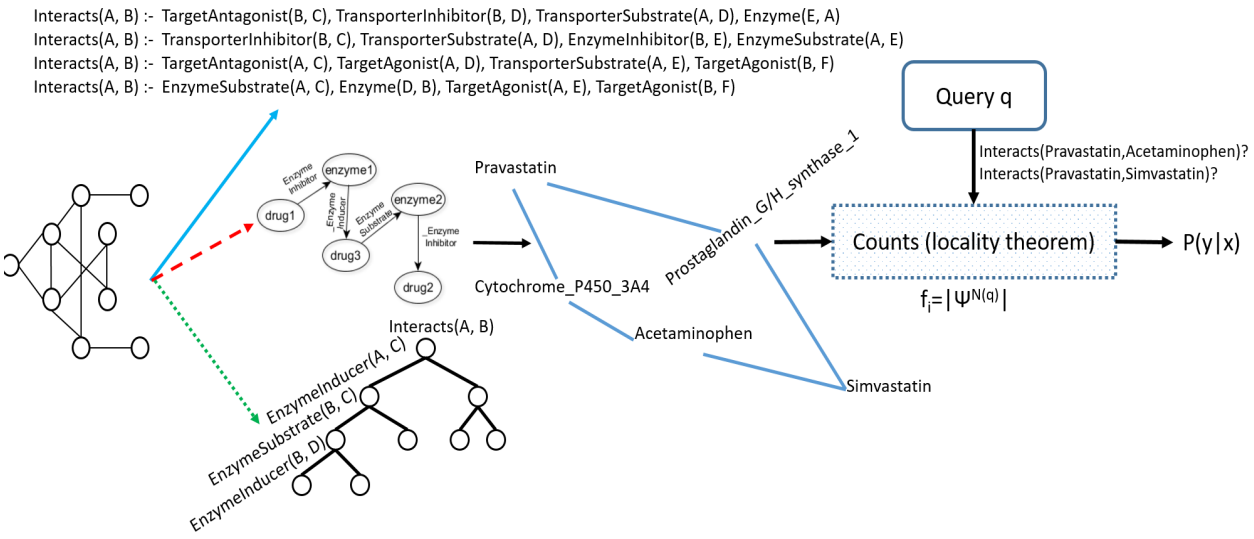


Figure 5.3: A general overview of link prediction using Gaifman models.

Features via Relational Rule Learning. Our first solution is inspired by Niepert (Niepert, 2016), who suggested the use of an Inductive Logic Programming (ILP) style learning method. This method learns a set of discriminative Horn clauses (implications of the form *if ¡condition¿ then ¡consequence¿*). Specifically, we use an ILP system called WILL (Walker T., 2009) to learn the relational features[1]. WILL first selects an example from the set of all examples and then finds a clause (rule) that *best covers* the examples. The *best covering* is the most general clause that maximizes the difference between the number of positive and negative examples covered [2]. Each *best covering* clause becomes a relational rule in our model. The examples covered by the clause are then removed and the process is repeated till a stopping criterion is satisfied; for example, we have extracted a maximum number of rules/clauses. Note that when a stopping criterion requires n rules to be extracted, it is sometimes possible to extract $m < n$ rules that cover the examples adequately.

Features via Relational Random Walks. Relational data is often represented using a graph that defines a domain's schema; in such a representation, a relation $\mathtt{R}(\mathtt{e}_1, \mathtt{e}_2)$ is a predicate edge between two entity type nodes: $\mathtt{e}_1 \xrightarrow{\mathtt{R}} \mathtt{e}_2$. A relational random walk (RW) through a graph is

[1]Any other ILP learner such as (Srinivasan, 2001), (Quinlan, 1990) or (Muggleton, 1995) could be used.

[2]Ideal coverage means *all* positive examples and *no* negative examples which can easily overfit.

a chain of such edges corresponding to a conjunction of predicates. For a random walk to be semantically sound, we should ensure that the input type (domain) of the $i+1$-th predicate is the same as the output type (range) of the i-th predicate. An example relational random walk from the drug-discovery domain is:

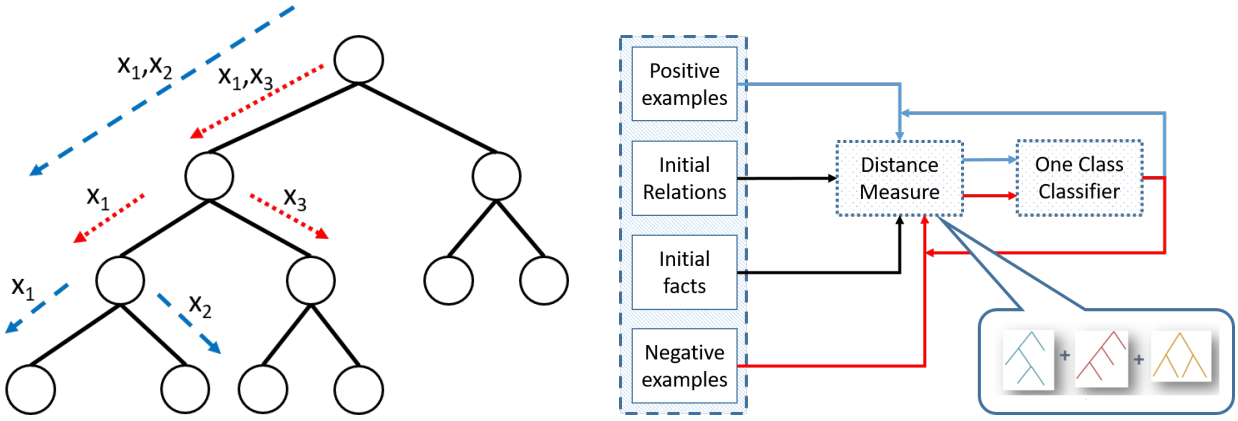
$$\texttt{Interacts}(\texttt{d0}, \texttt{d3}) \Leftarrow \texttt{TargetInhib}(\texttt{d0}, \texttt{t0})$$
$$\wedge \texttt{\_TargetInhib}(\texttt{t0}, \texttt{d1}) \wedge \texttt{TransporterSubstr}(\texttt{d1}, \texttt{t2})$$
$$\wedge \texttt{\_TransporterInhib}(\texttt{t2}, \texttt{d3}).$$

This is a semantically sound random walk as it is possible to chain the second argument of each predicate to the first argument of the succeeding predicate. This random walk also contains *inverse predicates* (prefixed by an underscore, such as _Transporter). Inverse predicates are distinct from their corresponding predicates as their arguments are *reversed*. Thus, this relational random walk chains the first variable d0 in the target predicate Interacts(d0, d3) with the second variable d3. The chain represents a relational feature and constitutes a random local structure of the form:

$$\texttt{d0} \xrightarrow{\texttt{TargetInhibitor}} \texttt{t0} \xrightarrow{\texttt{\_TargetInhibitor}} \cdots$$
$$\texttt{d1} \xrightarrow{\texttt{TransporterSubstrate}} \texttt{t2} \xrightarrow{\texttt{\_TransporterInhib}} \texttt{d3}.$$

Thus, to construct a relational random walk, only the schema describing the knowledge base is required. We adapt path-constrained random walks (PCRW, (Lao and Cohen, 2010a)) to construct relational random walks. The algorithm starts at the first entity in the target relation, and makes a walk over the (parameterized) graph to end at the second entity present in the target relation. One limitation of PCRW is that the random walks are only performed over binary relations. However, since we employ a predicate representation, *we generalize and learn with arbitrary n-ary relations*.

Features via Relational One-Class Classification (relOCC) Features. A common issue in many real-world relational domains, especially knowledge bases, is that only "positive" instances of a relation are annotated, while "negative" instances are not explicitly identified. This is because the number of instances where the relation does not hold is very large, and annotation can be



(a) An illustration of Least common ancestor.

(b) Structure learning of Gaifman models using relational distance. Each left branch of the learned tree represents a relational feature.

Figure 5.4: Learning relational rules with relOCC

prohibitively expensive. Learning with highly imbalanced data sets requires reasoning over just the positive instances, commonly referred to as *one-class classification*. Intuitively, if we can construct a relational one-class classifier describing the positive examples, then rules characterizing this classifier are essentially features that describe positive examples. One-class classification typically requires a *distance measure* to characterize the density of the positive class. While, for standard vector and matrix data, many different distance measures exist, the issue is far more challenging for relational data, and depends on the underlying representation of the classifier.

Suppose we use an off-the-shelf learner to learn relational trees (Blockeel and Raedt, 1998) to describe each class in the data. Such relational trees form a decision-list of relational rules. These trees can then be used to compute the *relational distance* between a pair of examples x_1 and x_2,

$$d(x_1, x_2) = \begin{cases} 0, & \mathsf{LCA}(x_1,\, x_2) \text{ is leaf;} \\ e^{-\lambda \cdot \mathsf{depth}(\mathsf{LCA}(x_1, x_2))}, & \text{otherwise,} \end{cases} \tag{5.1}$$

where LCA refers to the *least common ancestor* of the examples x_1 and x_2. Figure 5.4(a) shows examples $x_1 \equiv$ `advisedBy(Tom, Mary)` and $x_2 \equiv$ `AdvisedBy(Tom, John)`; they both follow the same path down the tree before diverging at a node at depth 2. Now, consider x_1 and

$x_3 \equiv$ `AdvisedBy(Ada, Dan)`. In this case, we have that the least common ancestor is at depth 1. Since the distance measure is inversely related to depth of the least common ancestor, we have that x_1 and x_2 are closer together than x_1 and x_3. Typically, more than one tree is learned (say, via functional gradient boosting), and the one-class classifier is a weighted combination of these trees. Then, the overall distance function is simply the weighted combination of the individual tree-level distances: $D(x_1, x_2) = \sum_i \beta_i \, d_i(x_1, x_2)$ where β_i is the weight of the i^{th} tree and $\sum_i \beta_i = 1, \beta_i \geq 0$. The non-parametric function $D(\cdot, \cdot)$ is a relational distance measure learned on the data. The distance function can then be used to compute the density estimate for a new relational example z as a weighted combination of the distance of z from all training examples x_j, $E(z \notin \texttt{class}) = \sum_j \alpha_j D(x_j, z)$, where α_j is the weight of the labeled example x_j and $\sum \alpha_j = 1, \alpha_j \geq 0$. Note that expectation above is for $z \notin$ `class`, since the likelihood of class membership of z is inversely proportional to its distance from the training examples describing that `class`.

We learn a tree-based distance iteratively (Khot et al., 2014) to introduce new relational features that perform one-class classification. The left-most path in each relational tree is a conjunction of predicates, that is, a clause, which can be used as a relational feature. The splitting criteria is the squared error over the examples and the goal is to minimize squared error in each node as shown in equation 5.2.

$$\begin{aligned} &\min \textstyle\sum_{y \in \mathbf{x}_r} \left[I(z) - E(z \notin \texttt{class}) - \Sigma_{j:x_j \in \mathbf{x}_l} \alpha_j \beta_i d_i \left(x_j, z\right) \right]^2 \\ &+ \textstyle\sum_{y \in \mathbf{x}_l} \left[I(z) - E(z \notin \texttt{class}) - \Sigma_{j:x_j \in \mathbf{x}_r} \alpha_j \beta_i d_i \left(x_j, z\right) \right]^2 \end{aligned} \tag{5.2}$$

$I(z)$ is the indicator function and returns 1 if z is an unlabeled example or 0 otherwise. Also, x_l and x_r are the examples that take the left and right branch respectively. A greedy search approach is employed for tree learning thereby providing a *non-parametric* approach for learning these relational trees. Algorithm 3 shows our structure learning method for DGMs using relOCC.

As mentioned above, in relOCC rule learning, there is a necessity to learn 2 different set of weights – α and β, where α is the weight of the example and β is the weight of the tree since while calculating the $E(z \notin \texttt{class})$ i.e. $P(z \notin \texttt{class})$ we need to combine distances in two levels, tree

Algorithm 3 Structure Learning using relOCC; **Input**: fact base $\mathbf{F_s}$, positive ex. pos, negative ex. neg

function `LearnGaifmanStruct`($\mathbf{F_s}$, pos, neg)
  for every $\mathbf{x_1}, \mathbf{x_2}$ **in** pos **do**
    Calculate $d(x_1, x_2)$ according to equation 5.1
    $D(x_1,\ x_2) = \sum_i\ \beta_i\ d_i(x_1, x_2)$ ▷ Compute weighted distance between x_1, x_2
  end for
  for a given new unlabeled example **z do**
    $E(z \notin \texttt{class}) = \sum_j\ \alpha_j D(x_j,\ z)$ ▷ Calculate the density estimate
  end for
  Learn the tree T iteratively by minimizing equation 5.2
  return `LeftBranch`(T)
end function

level and instance level as shown in figure 5.5. These weights are learnt iteratively by minimizing the squared loss function:

$$\texttt{L} = \sum_{y \in \mathbf{x}} [I(z \notin \texttt{class}) - E(z \notin \texttt{class})]^2 \tag{5.3}$$

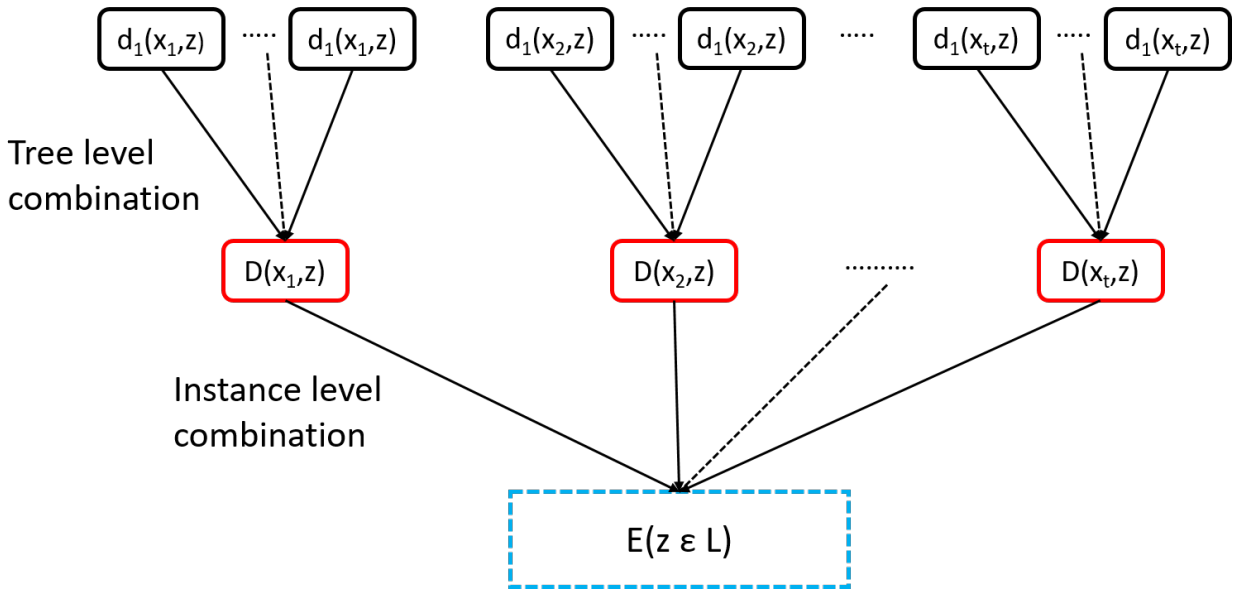


Figure 5.5: 2-level distance combination for learning relOCC rules.

The different gradients with respect to α and β can be calculated as:

$$\begin{aligned}
\frac{\partial \mathtt{L}}{\partial \alpha} &= \frac{\partial \mathtt{L}}{\partial \alpha_j} \sum_z [I(z) - E(z \notin \mathtt{class})] \\
&= \frac{\partial}{\partial \alpha_j} \sum_z \left[I(z) - \Sigma_i \alpha_j \beta_i d_i \left(x_j, z\right)\right]^2 \\
&= 2 \sum_z [I(z) - \Sigma_i \alpha_j \beta_i d_i (x_j, z)] \times -\Sigma_i \beta_i d_i (x_j, z) \\
&\boxed{\frac{\partial \mathtt{L}}{\partial \alpha} = -2 \sum_z \left[I(z) - E(z \notin \mathtt{class})\right] \Sigma_i \beta_i d_i (x_j, z)}
\end{aligned} \tag{5.4}$$

$$\begin{aligned}
\frac{\partial \mathtt{L}}{\partial \beta} &= \frac{\partial \mathtt{L}}{\partial \beta_i} \sum_z [I(z) - E(z \notin \mathtt{class})] \\
&= \frac{\partial}{\partial \beta_i} \sum_z \left[I(z) - \Sigma_j \alpha_j \beta_i d_i \left(x_j, z\right)\right]^2 \\
&= 2 \sum_z [I(z) - \Sigma_i \alpha_j \beta_i d_i (x_j, z)] \times -\Sigma_j \alpha_j d_i (x_j, z) \\
&\boxed{\frac{\partial \mathtt{L}}{\partial \beta} = -2 \sum_z \left[I(z) - E(z \notin \mathtt{class})\right] \Sigma_j \alpha_j d_i (x_j, z)}
\end{aligned} \tag{5.5}$$

In the tree level combination, we calculate the LCA based distance between all the labeled examples $(x_1, x_2 .... x_t)$ and the unlabeled example z ***in every learned relational tree*** which are then combined to yield a combined distance D between each example and the unlabeled example. After calculating the distance between each each example with the unlabeled example, a second level of combination is performed to yield the probability of the unlabeled example to belong to a certain class.

5.2.2 Feature Construction

Once extracted, relational rules are instantiated (grounded) to obtain graphs G_{pos} and G_{neg}. While several *feature aggregations* exist, we employ *counts* since they have been successfully employed in many relational models. For every relational feature $\varphi \in \Phi$, the first and last entity are instantiated

corresponding to the tuples satisfying the query. For example, consider the knowledge base snippet in Fig. 5.1; let the positive example be `Interacts(Pravastatin, Simvastatin)`. For a relational feature, say `EnzymeInhib(d0, t0)` $\wedge$ `_EnzymeInhib(t0, d1)`, and the substitution $\{$`d0`/`Pravastatin`, `d1`/`Simvastatin`$\}$ we obtain the **partially-grounded relational feature** `EnzymeInhib(Pravastatin, t0)` $\wedge$ `_EnzymeInhib(t0, Simvastatin)`. Next, all the entities that completely satisfy this partially grounded feature are obtained. The features for each query variable are then obtained as counts of the number of entities in the satisfied grounded features that are also present in the neighborhood of the query entities in the Gaifman graph $\mathcal{G}$. For example, in the Gaifman graph in Figure 5.1, we check if `EnzymeInhib(Pravastatin, CytochromeP4502C9)` $\wedge$ `_EnzymeInhib(CytochromeP4502C9, Simvastatin)` satisfies the given relational feature i.e. this grounding $\in G$. If the grounding satisfies the relational feature and since `CytochromeP4502C9` is present in the Gaifman neighborhood of `Pravastatin` (as well as `Simvastatin`), the count of the relational feature is increased by 1. Thus, for every query variable q we obtain a propositional feature $f = [f_1, ..., f_{|\Phi|}]$ of length $|\Phi|$:

$$f_i = \begin{cases} |\psi^{N_r(q)}(q)|, & \text{if } q(e_1, e_2) \text{ partially grounds } \Phi_i, \\ 0, & \text{otherwise.} \end{cases} \tag{5.6}$$

Recall that ψ refers to the relativized first-order formula, and consequently $\psi^{N_r(q)}(q)$ is the r-local formula for a neighborhood N of depth r. Thus, we count the number of entities in the satisfied grounded features that are also satisfied in the neighborhood structure of the Gaifman graph.

Table 5.1: Evaluation domains and their properties.

Data set	#Entities	#Relations	#Pos	#Neg	#RW rules	#ILP rules	#relOCC rules
DDI	355	15	2832	3188	68	36	25
PPI	797	7	1915	1915	42	5	15
NELL Sports	4147	6	300	600	36	15	13
Financial NLP	650	7	186	1029	222	6	25
ICML Co-Author	558	5	155	6498	7	15	7

Algorithm 4 presents our method, LGE for extracting embeddings from DGMs. In **[Line 2–3]:** we build the initial Gaifman graph $\mathcal{G}$. In **[Line 4]:** we learn the relational features from one of the

methods defined in 5.2.1, which are then grounded using both the positive and negative examples **[Line 5]**; in addition, tuples of the positive and negative examples are also obtained **[Line 6]**. For both positive ($T_{\mathbf{q}}^{\mathsf{pos}}$) and negative tuples ($T_{\mathbf{q}}^{\mathsf{pos}}$), the neighborhood of each entity in the tuple is obtained, and each relational feature is partially grounded with the entities $\in t$ **[Line 8, 16]**. `GenerateNeighbors` (Niepert, 2016) generates entity neighborhoods for a tuple $\mathbf{t} \in T_{\mathbf{q}}$.

Algorithm 4 Learning Embeddings from Discriminative Gaifman Models; **Input**: target query $\mathbf{q}$, knowledge base $\mathcal{B}$, positives pos, negatives neg; **Params**: depth r, size k and number of Gaifman neighborhoods w

1: **function** LGE($\mathbf{q}$, $\mathcal{B}$, pos, neg)
2: $\mathcal{G} = \texttt{MakeGaifmanGraph}(\mathcal{B})$ ▷ construct Gaifman graph from facts
3: $F_s = \texttt{MakeFactBase}(\mathcal{B})$
4: $\Phi = \texttt{LearnGaifmanStruct}(F_s, \text{pos}, \text{neg})$ ▷ extract relational features (section 5.2.1)
5: $G_{pos}, G_{neg} = \texttt{Ground}(\Phi, F_s, \text{pos})$ ▷ ground positive and negative examples
6: $T_{\mathbf{q}}^{\mathsf{pos}}, T_{\mathbf{q}}^{\mathsf{neg}} = \texttt{GetQueryTuples}(\mathbf{q}, \mathtt{F_s})$ ▷ all tuples satisfying $\mathbf{q} \in F_s$ (pos), $\neg\mathbf{q} \in F_s$, (neg)
7: **for every** $\mathbf{t}$ **in** $T_{\mathbf{q}}^{\mathsf{pos}}$ **do**
8: $\mathcal{N} = \texttt{GenerateNeighborhoods}(\mathbf{t}, r, k, w)$ ▷ generate w neighborhoods of depth r and size k
9: **for every** φ **in** Φ **do**
10: $\boldsymbol{\theta} = \varphi/\mathbf{t}$ ▷ substitute query tuple $\mathbf{t}$ in feature φ
11: $x_{\mathbf{t}}^{\varphi} = \texttt{Count}(\boldsymbol{\theta}, \mathcal{N}, G_{pos})$ ▷ count groundings satisfied in the neighborhoods
12: **end for**
13: $\mathbf{x}_{\mathbf{t}}^{\mathsf{pos}} = [\ldots, x_{\mathbf{t}}^{\varphi}, \ldots, x_{|\Phi|}]$ ▷ embedding for tuple $\mathbf{t}$
14: **end for**
15: **for every** $\mathbf{t}$ **in** $T_{\mathbf{q}}^{\mathsf{neg}}$ **do**
16: $\mathcal{N} = \texttt{GenerateNeighborhoods}(\mathbf{t}, r, k, w)$ ▷ generate w neighborhoods of depth r and size k
17: **for every** φ **in** Φ **do**
18: $\boldsymbol{\theta} = \varphi/\mathbf{t}$ ▷ substitute query tuple $\mathbf{t}$ in feature φ
19: $x_{\mathbf{t}}^{\varphi} = \texttt{Count}(\boldsymbol{\theta}, \mathcal{N}, G_{neg})$ ▷ count groundings satisfied in the neighborhoods
20: **end for**
21: $\mathbf{x}_{\mathbf{t}}^{\mathsf{neg}} = [\ldots, x_{\mathbf{t}}^{\varphi}, \ldots, x_{|\Phi|}]$ ▷ embedding for tuple $\mathbf{t}$
22: **end for**
23: **return** $\mathcal{F} = \{\mathbf{x}_{\mathbf{t}}^{\mathsf{pos}}\}, \{\mathbf{x}_{\mathbf{t}}^{\mathsf{neg}}\}$ ▷ return embeddings
24: **end function**

Neighborhood generation relies on three parameters: (1) r, the depth of neighborhood when counting, (2) k, the number of neighbors to sample, and (3) w, the number of neighborhoods to be generated. For each entity in tuple t, all neighbors at a maximum distance of r form the neighborhood (Fig. 5.2, the outer region). This process is repeated until we obtain w neighborhoods for each training example. For example, if $r = 1, w = 5$ and $k = 10$ and we have 10 relational features

($|\Phi| = 10$), we obtain 50 propositional examples with 10 features by looking at 1-neighbors for each entity. The `Count` function **[Line 11, 19]** counts how many entities in the neighborhood of each query satisfy the partially-grounded relational features. Each such count becomes a propositional feature. In this manner, we can construct a propositionalized data set of $k \times w$ positive examples and $k \times w$ negative examples.

Learning a Discriminative Model: After learning the propositional features, any standard classifier can be used for link prediction. In our experiments, we employ gradient-boosting (Friedman, 2001) and logistic regression. Results using more algorithms are given in the Appendix. The classification algorithm itself is not a key contribution of our work and as we demonstrate empirically next, a standard classifier suffices for learning an effective discriminative model.

5.3 Experiments

We consider 5 ***novel, real-world relational data sets*** (Table 7.1). We aim to answer the following questions: **Q1:** How do different structure learning strategies compare across diverse domains from different applications? **Q2:** Does choice of the discriminative algorithm impact the performance? **Q3:** How do different structure learning strategies impact performance in the presence of high class imbalance? **Q4**: What are effects of Gaifman locality parameters r, w and k? **Q5**: How does our method compare with state-of the art probabilistic ILP systems? **Q6**: How does our method compare with Niepert's original approach (Niepert, 2016) of using hand-crafted rules?

Data sets: Drug-Drug interactions (DDI) (Dhami et al., 2018) consists of 78 drugs obtained from DrugBank. The data set has 15 relations and the target is *Interactions* between drug entities. **Protein-Protein interactions (PPI)** (Kok et al., 2009) has 7 relations and is obtained from Alchemy. The target is *interaction* relation between two protein entities. **NELL Sports** was generated by the Never Ending Language Learner (Mitchell et al., 2018) consisting of information about players and teams. It has 6 relations and the task is to predict whether a team plays a particular sport i.e *teamplayssport*. **Financial NLP** is obtained by extracting information from *SEC Form S-1*

documents, which were scraped and converted into relational format. This data set has 7 relations and the target the relation *sentenceContainsTarget* between sentence and word entities. **ICML Co-Author** is obtained by mining publication data from ICML 2018; the data set has 5 relations and the target is the *CoAuthor* relation between persons.

Results: Table 7.1 also shows the number of relational rules learned by different techniques. Table 5.2 present the results for all the relational domains, after 5-fold cross validation, with logistic regression (LR) and gradient boosting (GB). All experiments were run on a *64-bit Intel(R) Xeon(R) CPU E5-2630 v3* server with parameter values r=1, k=10 and w=5.

Table 5.2: Results (≈ 3 $decimals$) for the relational domains. Note that the first three data sets are relatively balanced and the last two are highly unbalanced. Thus, the accuracy and AUC-ROC values for the last two data sets are reported only for completion.

Data set	Methods	Accuracy		Recall		F1		AUC-ROC		AUC-PR	
		LR	GB	LR	GB	LR	GB	LR	GB	LR	GB
DDI	RW	0.657	0.669	0.469	0.530	0.564	0.602	0.647	0.662	0.581	0.593
	ILP	0.696	0.774	0.467	0.674	0.592	0.729	0.684	0.767	0.710	0.765
	relOCC	0.860	**0.897**	0.939	**0.991**	0.864	**0.901**	0.864	**0.902**	0.797	**0.853**
	Niepert (Niepert, 2016)	0.534	0.534	0.0	0.0	0.0	0.0	0.5	0.5	0.466	0.466
	MLN-Boost	0.638		0.504		0.618		0.798		0.784	
	RDN-Boost	0.755		0.662		0.718		0.828		0.831	
PPI	RW	0.700	**0.785**	0.586	0.707	0.661	0.767	0.699	**0.785**	0.651	**0.740**
	ILP	0.613	0.661	0.397	0.553	0.506	0.620	0.613	0.661	0.579	0.614
	relOCC	0.727	0.733	0.996	**0.999**	0.785	**0.789**	0.727	0.733	0.647	0.652
	Niepert (Niepert, 2016)	0.608	0.652	0.382	0.524	0.499	0.606	0.613	0.654	0.591	0.619
	MLN-Boost	0.548		0.453		0.571		0.743		0.733	
	RDN-Boost	0.671		0.615		0.652		0.728		**0.740**	
NELL Sports	RW	0.783	0.822	0.414	0.569	0.569	0.689	0.696	0.762	0.565	0.594
	ILP	0.782	0.824	0.431	0.590	0.578	0.699	0.699	0.769	0.530	0.564
	relOCC	0.793	**0.833**	0.431	0.6	0.59	**0.731**	0.708	0.778	0.574	0.643
	Niepert (Niepert, 2016)	0.756	0.780	0.314	0.485	0.465	0.597	0.648	0.707	0.512	0.549
	MLN-Boost	0.605		0.533		0.667		**0.894**		**0.853**	
	RDN-Boost	0.812		**0.756**		0.714		0.884		0.834	
Financial NLP	RW	0.833	0.833	0.0	0.0	0.0	0.0	0.5	0.5	0.168	0.168
	ILP	0.838	0.921	0.068	0.633	0.112	0.727	0.530	0.806	0.200	0.6023
	relOCC	0.965	0.967	0.788	0.800	0.882	0.889	0.867	0.879	0.826	0.833
	Niepert (Niepert, 2016)	0.827	0.914	0.0	0.59	0.0	0.705	0.5	0.787	0.173	0.587
	MLN-Boost	0.928		0.764		0.757		**0.989**		0.807	
	RDN-Boost	**0.975**		**0.963**		**0.929**		**0.989**		**0.901**	
ICML CoAuthor	RW	0.977	0.977	0.0	0.0	0.0	0.0	0.5	0.5	0.023	0.023
	ILP	0.983	0.985	0.272	0.339	0.427	0.506	0.636	0.669	0.289	0.356
	relOCC	0.986	**0.997**	0.346	0.386	0.517	**0.557**	0.653	**0.693**	0.370	**0.40**
	Niepert (Niepert, 2016)	0.981	0.984	0.195	0.327	0.326	0.493	0.597	0.664	0.214	0.343
	MLN-Boost	0.938		0.326		0.214		0.294		0.210	
	RDN-Boost	0.940		**0.434**		0.231		0.153		0.157	

To answer **Q1**, we note that relOCC outperform ILP and relational RWs across a majority of the domains. This is expected since relOCC considers the density of the positive and negative examples separately, allowing the features it generates to discriminate better. To answer **Q2**, we note from Table 5.2 that the choice of classifier does not result in significant differences in performance after learning relational rules, though the performance of GB is almost always higher than LR.

The ICML CoAuthor (neg-to-pos ratio of 42:1) and Financial NLP (neg-to-pos ratio of 6:1) data sets are highly imbalanced; consequently, we report AUC-PR. In both domains, AUC-PR for relOCC outperforms the other structure-learning methods by a large margin. Random walk rules, in particular, cause all the examples to be classified as negative, resulting in recall and F1-scores of 0 in both domains. Thus, we can answer **Q3**: highly-imbalanced domains benefit from density-estimation-based structure learning.

Fig. 5.6 show the effects of varying r (depth of neighborhoods), k (number of neighbors) and w (number of neighborhoods) on the DDI data set. Generally, k does not affect performance significantly, but increasing r causes recall report to drop sharply. This is because, with $r = 1$, entities in the query neighborhood are more tightly coupled with entities in the query variables. This parametric sensitivity analysis addresses **Q4**. Also, another important takeaway is that relOCC rules exhibit *high clinically-relevant recall ($\approx$ 1) on medical data sets*: DDI and PPI. This has considerable implications for bioinformatics domains as recall is the most important metric; this is because a false negative (such as a misdiagnosis) could result in much more serious consequences (Dhami et al., 2018) than a false positive. Finally, from Fig. 5.6 (right), we note that varying r and k does not affect training time, as these parameters do not affect the search space. However, increasing w increases the run time since the size of the neighborhood graph to be searched increases.

To answer **Q5**, we compare our approaches to three probabilistic ILP systems, MLN-Boost (Khot et al., 2011), RDN-Boost (Natarajan et al., 2012a) and Tuffy (Niu et al., 2011). *Our core contribution is end-to-end learning of Gaifman models that requires only data and no domain knowledge and thus we focus on comparison with a full model learning methods of MLN-Boost*

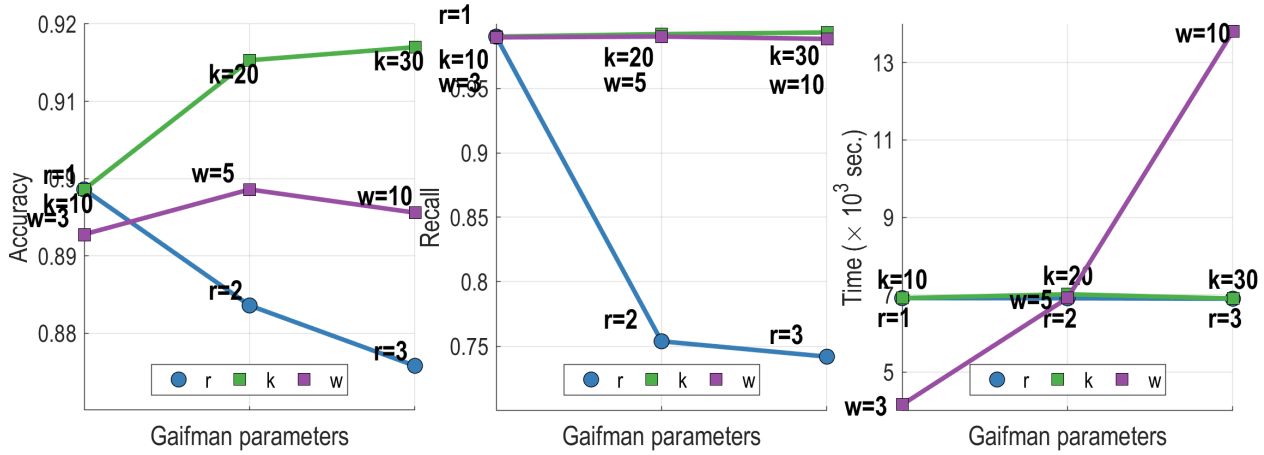
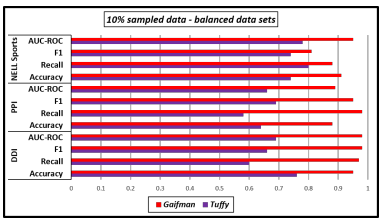
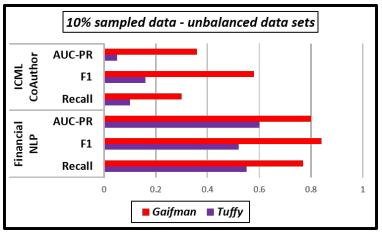


Figure 5.6: (left) Accuracy, (middle) recall and (right) running time for various values of r, k and w for the DDI domain. For varying r: w=5 and k=10, for varying w: r=1 and k=10 and for varying k: w=5 and r=1.

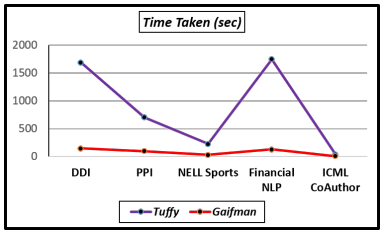
and RDN-Boost. Table 5.2 shows that our method outperforms MLN-Boost by a significant margin and outperforms/is comparable to the performance of RDN-Boost in 4 out of 5 domains. We also note that Tuffy[3] could not effectively scale to the amount of data that we have used in our learning framework, and could not learn the structure. Instead, we tried using the ILP rules that we learned, and learned the weights. In this case as well, Tuffy could not complete training after a few hours. To put this in perspective, we sampled 10% data from all the data sets and the results for the same are presented in Fig. 5.7 which shows that our method is significantly better than Tuffy.



(a) Comparison of our method with Tuffy on balanced data sets.



(b) Comparison of our method with Tuffy on unbalanced data sets.



(c) Comparison of learning + inference time taken by our method with Tuffy.

Figure 5.7: Comparison of our method with Tuffy with 10% sampled data sets.

[3]We also tried other systems: Alchemy, Problog, ProbCog.

Finally, to answer **Q6**, we compared against Niepert (2016). We created generic relational features as suggested by Niepert and of the form: $r(e1, e2)$; $r(e2, e1)$; $\exists x\ r(x, e)$, $\exists x\ r(e, x)$, $\exists x\ r(e1, x) \wedge r(x, e2)$, $\exists x\ r(e2, x) \wedge r(x, e1)$. These relational features are very simple, and do not cover the relational search space sufficiently, resulting in significantly poor perforamance. And hence, we created more domain-specific rules to enhance the score. It is clear from the results, that even after enhancing the hand-crafted rules, learning the structure leads to much better predictive models. Combining the human-guided rules with learning remains an interesting future direction.

5.4 Conclusion

Our proposed approach, LGE, is the **first algorithm for learning the structure of Gaifman models for relational data.** Given the *increasing importance of local neighborhoods in graph data, automatic learning of these neighborhoods is an important direction and contribution*, and we hope to encourage more widespread adoption of this powerful learning approach. Our learned rules, or relational features, are instantiated and aggregated over a Gaifman graph to produce raw features, which are used to train a discriminative classifier. Our work provides an efficient and effective method of constructing local-neighborhood-based relational embeddings.

Beyond structure learning, there are several avenues to explore such as joint learning of Gaifman models, generating explanations for a given prediction and extending Gaifman locality to hyper-graphs. Another direction is employing more graph based embedding methods that can integrate with Gaifman's locality principle.

CHAPTER 6

GRAPH NEURAL MODELS FOR RELATIONAL DATA

In this chapter we consider the problem of knowledge base population (KBP) of a coauthor network. Specifically, we consider the link prediction problem as a relational modeling task. We develop a relational extension of graph convolutional networks (GCNs) that exploits recent success inside relational probabilistic models. As a result of this extension, we present a novel knowledge base consisting of knowledge for four major machine learning conferences extracted from the Microsoft Academic Graph *(Under Review)*.

6.1 Introduction

Statistical Relational Learning (SRL) (Getoor and Taskar, 2007; Raedt et al., 2016) combines the power of probabilistic models to capture uncertainty with logic/relational models to take advantage of the rich domain structure. One of the key successes of these models lie in the task of Knowledge Base Population (KBP), specifically, in link prediction or relation extraction tasks. While successful, most methods make several simplifying assumptions – presence of supervision in the form of labels, closed world assumption, presence of only binary relations and most importantly, the presence of hand-crafted rules.

We go beyond these assumptions and inspired by the recent success of Graph Convolutional Networks (GCNs) (Defferrard et al., 2016; Kipf and Welling, 2017), and develop a new relational extension of GCNs. The proposed method is capable of (1) automatically learning rules from one-class data, i.e., only from the positive annotations of the relation (in our case, coauthor) (2) automatically converting these rules into observed features and (3) training a GCN that is specific to the task of link prediction. Our method employs a one-class density estimation method that uses a tree-based distance metric to learn relational rules iteratively. We call this method as *Relational One-Class GCN* (ROCGCN). Since the two different steps of learning the relational rules and the

GCN training both employ the same set of positive examples, they allow for richer representation of the combination of the attributes, entities and their relations. While previous methods simply used the observed features as the observed layer, our method uses the combinations (rules) as the observed layer. This has the added advantage of the latent layer being richer – it combines the combinations (rules) themselves allowing for a more richer representation. As we show empirically, this is specifically useful when employed on large-scale KBP tasks such as link prediction.

We make a few key contributions: (1) We develop the first relational GCN capable of learning from positive and unlabeled data. (2) Instead of employing hand-crafted rules, our method learns these rules automatically by focusing on the link prediction task and then constructs the GCN for the same task. These two steps that are conditioned on the link prediction task allows for a better classifier. (3) ROCGCN can handle arbitrary relations – we do not make the assumption of binary relations that most methods make. Given that our base learner is a logic-based one, the predicates can include arbitrary number of parameters. (4) Finally, an important contribution of this work is the **release of Microsoft Knowledge Graph in the form of first-order logic factbase**. This will allow for benchmarking several SRL models in the immediate future.

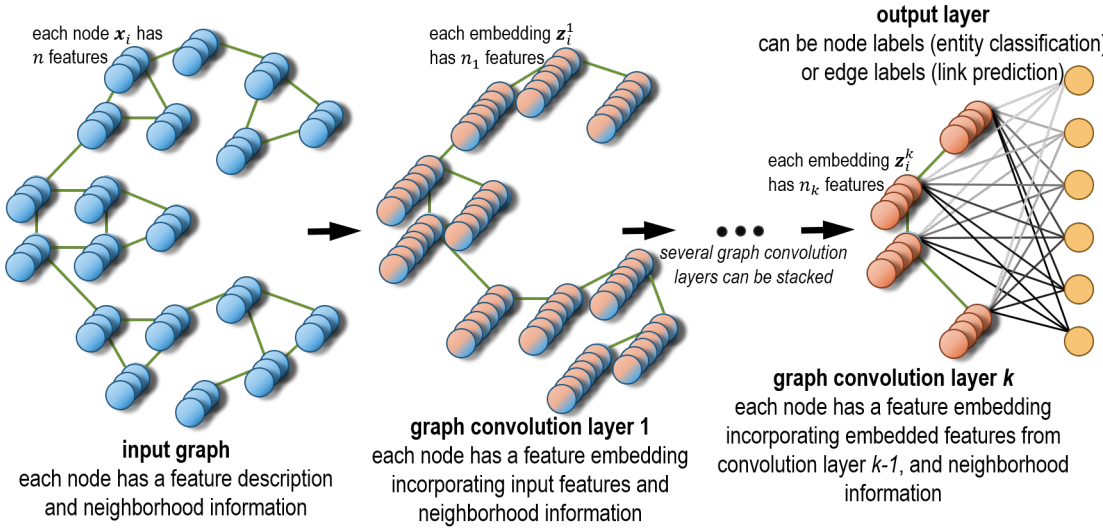


Figure 6.1: Graph Convolutional Networks

6.2 Relational Graph Convolution Networks

The main reason behind the success of GCNs is that they exploit two key types of information: node feature descriptions (x_i) and node neighborhood structure (captured through the adjacency matrix $\mathcal{A}$ of the graph). The basic idea behind GCNs is shown in figure 6.1. As mentioned earlier, GCNs by themselves cannot fully exploit the inherent structures inside a multi-relational graph. Motivated by this, we propose an extension to the graph convolutional network that can handle large multi-relational networks. Although a recent work (Schlichtkrull et al., 2018) that extends the GCN framework to relational domains exists, this approach is still limited to graphs represented as (subject; predicate; object) triples. We propose a novel and a more general approach that is not limited by assumptions regarding the multirelationality of the data and can handle general multi-graphs and hypergraphs implicitly and without loss of information. Figure 6.2 shows a representation of our approach, Relational One Class Graph Convolutional Networks (ROCGCN), to construct the feature matrix $\mathcal{X}$ containing the node feature descriptors x_i and the distance matrix $\mathcal{D}$.

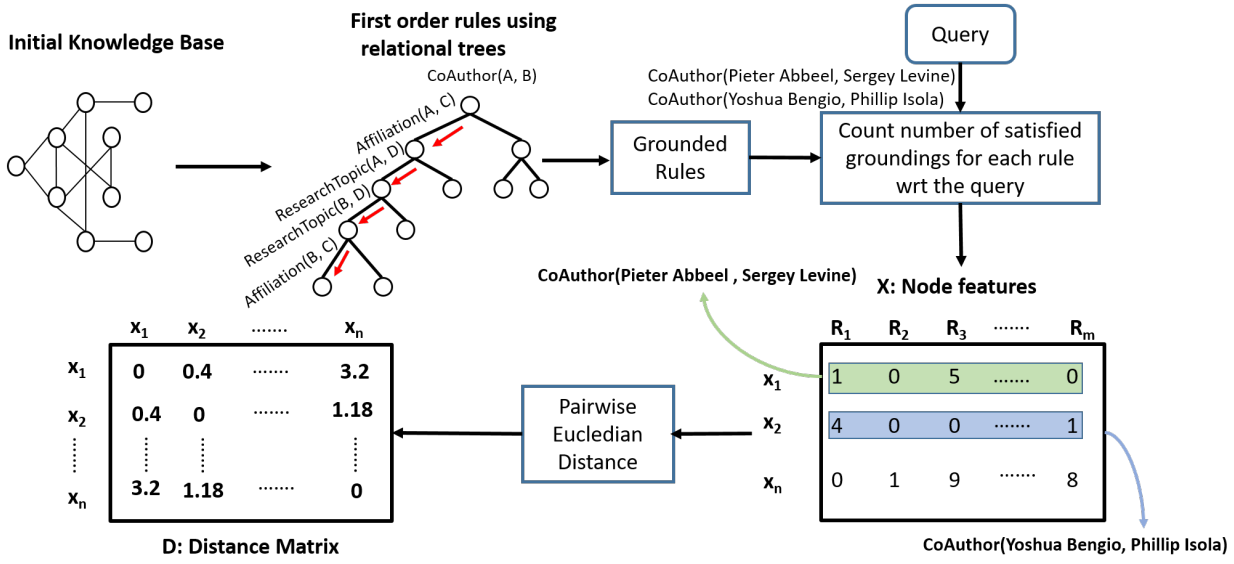


Figure 6.2: Feature and Distance Matrix Construction for Relational One-Class Graph Convolutional Networks

Before we define the process of obtaining $\mathcal{D}$ and $\mathcal{X}$ we present some required background. We first introduce some basic notation and concepts in first-order logic. An **atom** is of the form $\mathcal{R}(t_1, \ldots, t_k)$ where $\mathcal{R}$ is a functor and the arguments t_i are **terms**. A **substitution** is of the form $\boldsymbol{\theta} = \{\langle x_1, \ldots, x_k \rangle / \langle t_1, \ldots, t_k \rangle\}$ where x_is are logical variables and t_is are terms. A **grounding** of an atom with logical variables $x_1, \ldots, x_k$ is a substitution $\{\langle x_1, \ldots, x_k \rangle / \langle X_1, \ldots, X_k \rangle\}$ mapping each of its variables to a constant in the population of that variable. A knowledge base $\mathcal{B}$ consists of (1) entities: a finite domain of objects $\mathcal{D}$, (2) relations: a set of predicates $\mathcal{R}$ describing the attributes and relationships between objects $\in \mathcal{D}$, and (3) an interpretation assigning a truth value to every grounding.

Given a knowledge base $\mathcal{B}$, we first learn a set of first-order rules that represent the relational search space effectively. The intuition is that these relational rules can be viewed as meta-features that connect entities and their attributes. Particularly, when *learned for a specific task*, these features can be both predictive and informative. Given that they are typically conjunctions of relational features (attributes of entities and relationships), they have the added advantage of being interpretable. Our hypothesis, that we verify empirically is that these rules can potentially yield richer latent representations than a relational GCN that simply uses the entity and relationship information.

A common issue in many real-world relational knowledge bases is that only true instances of any relation(s) are present while the false instances are not explicitly identified as they are prohibitively expensive. Consequently ***closed world assumption*** is applied to sample negative instances from the set of unobserved ones. While successful, this is a strong assumption particularly when the number of positively labeled examples are sparse such as our coauthor network. Inspired by the success of learning only from positive examples in relational domains (Khot et al., 2014), we learn first-order rules using relational density estimation. The intuition behind using a density estimation method is as follows: *Learning first order rules for positive and sampled negative examples independently can result in better utilization of the search space thereby learning more discriminative features*. The

density estimation approach uses a tree-based distance measure that iteratively introduces newer features (as short rules) that covers more positive examples. These rules at the end of the density estimation step form the relational features for our model.

The learned first order rules are then grounded to obtain all instantiations of these rules. The counts of each feature, i,e., the count of the number of times a target example (the coauthor relation between the target entities) is satisfied in every first order rule is obtained which forms our feature matrix $\mathcal{X}$. For example, the learned first order rule from true instances

$$\texttt{CoAuthor}(\texttt{person}_1, \texttt{person}_2) \Leftarrow \texttt{Affiliation}(\texttt{person}_1, \texttt{university}_1)$$
$$\wedge\texttt{Affiliation}(\texttt{person}_2, \texttt{university}_1) \wedge \texttt{ResearchTopic}(\texttt{person}_1, \texttt{topic}_1)$$
$$\wedge\texttt{ResearchTopic}(\texttt{person}_2, \texttt{topic}_1).$$

implies that if 2 people have the same affiliation and their research interests lie in similar topics then they are more likely to coauthor. Suppose the given target entities are $person_1$ = "Pieter Abbeel" and $person_2$ = "Sergey Levine". The partially grounded first order rule can then be written as

$$\texttt{CoAuthor}(\texttt{Pieter Abbeel}, \texttt{Sergey Levine}) \Leftarrow \texttt{Affiliation}(\texttt{Pieter Abbeel}, \texttt{university}_1)$$
$$\wedge\texttt{Affiliation}(\texttt{Sergey Levine}, \texttt{university}_1) \wedge \texttt{ResearchTopic}(\texttt{Pieter Abbeel}, \texttt{topic}_1)$$
$$\wedge\texttt{ResearchTopic}(\texttt{Sergey Levine}, \texttt{topic}_1).$$

Then substitutions for all the other entities within the first order rule is performed and checked whether the substituted first order rule is satisfied in the groundings. For example, the substitution $\boldsymbol{\theta} = \{\langle university_1, topic_1\rangle / \langle University\ of\ California\ Berkeley, Artificial\ Intelligence\rangle\}$ is satisfied but the substitution $\boldsymbol{\theta} = \{\langle university_1, topic_1\rangle / \langle University\ of\ California\ Berkeley, Computer\ Networks\rangle\}$ is not satisfied. Since there can be multiple values taken by $topic_1$ that can satisfy the first order rule, the count of all such satisfied groundings becomes a feature value for the target query `CoAuthor(Pieter Abbeel, Sergey Levine)`. Thus we obtain a feature set $\mathcal{X}$ of size $n \times k$ where n is the number of target queries and k is the number of first order rules.

In order to obtain the distance matrix $\mathcal{D}$ a pairwise euclidean distance of all the node feature descriptors i.e. the counts $x_i \in \mathcal{X}$ is computed. Thus, every element $d_{ij} \in \mathcal{D}$,

$$d_{ij} = \|x_i - x_j\| = \sqrt{\|x_i\|^2 + \|x_j\|^2 + 2 \cdot x_i \cdot x_j} \tag{6.1}$$

The original formulation of graph convolutional networks (Kipf and Welling, 2017) require an adjacency matrix $\mathcal{A}$ to perform the layer wise propagation. Instead of building the adjacency matrix from the relation triples, we use the computed distance matrix $\mathcal{D}$ and use it as an approximation to the adjacency matrix for the GCN. To obtain this approximation, we perform the following steps:

1. A threshold, t, is set as the average of all the distances (since the distance matrix is symmetric, the average is calculated from the upper-right part of the distance matrix). This case is considered as far-away case.

2. $\forall d_{ij} \in \mathcal{D}$, new distances are computed as $\hat{d}_{ij} = d_{ij}/t$ and max($\hat{d}_{ij}$) is set as 1.

3. Since the higher values in $\mathcal{D}$ represent nodes that are far way as opposed to the $\mathcal{A}$ where the higher values i.e. 1 represents the nodes adjacent to each other, the distance between nodes is subtracted from 1 i.e. $\hat{d}_{ij} = 1 - \hat{d}_{ij}$. This is similar to $\mathcal{A}$ with d_{ij}= 1 representing that two nodes are connected and d_{ij}= 0 representing that two nodes are not connected with the only difference being the presence of values $0 < d_{ij} < 1$ that denote the closeness of two nodes.

4. Since the above operator sets the diagonals of the distance matrix $\mathcal{D}$ to 1, in order to approximate the adjacency matrix, the values in the diagonals are reverted to 0.

For a GCN with M layers, the layer wise propagation rule for the layer l $\in$ M can now be written as,

$$f(H^{(l)}, \mathcal{D}) = \sigma(\mathcal{D}H^{(l)}W^{(l)}) \tag{6.2}$$

where $H(0)$ is the input layer i.e. the feature matrix $\mathcal{X}$ with $H(1) \dots H(M-1)$ being the hidden layers. Since we replace $\mathcal{A}$ with $\mathcal{D}$ before the symmetric normalization and addition of self loops, these operations are now be performed on $\mathcal{D}$ thereby giving the updated propagation rule as,

$$f(H^{(l)}, \mathcal{D}) = \sigma(\hat{\mathcal{N}}^{\frac{-1}{2}}\hat{\mathcal{D}}\hat{\mathcal{N}}^{\frac{-1}{2}}H^{(l)}W^{(l)}) \tag{6.3}$$

such that $\hat{\mathcal{D}} = \mathcal{D} + \mathcal{I}$ where $\mathcal{I}$ is the identity matrix and $\hat{\mathcal{N}}$ is the diagonal node degree matrix of $\hat{\mathcal{D}}$.

Given that we have outlined the procedure for constructing our ROCGCN, we now proceed to discuss the KBP construction using ROCGCN.

6.3 Knowledge Base Extraction from Microsoft Academic Graph

To create our knowledge base (KB), we extract data from Microsoft Academic Graph (MAG) (Sinha et al., 2015) for 4 major AI/ML conferences, namely International Conference on Machine Learning (ICML), International Conference on Learning Representations (ICLR), International Joint Conference on Artificial Intelligence (IJCAI) and Neural Information Processing Systems (NeurIPS). The data is extracted by querying the academic graph with the keyword 'venue'. For each conference, we extract title, #citations, year, author(s), affiliation of the author(s), every author's field of study i.e. the research topic and the citation of each paper. All the obtained information is then converted into a first order logic representation which are collectively referred to as *facts*. Thus a set of facts $\in \mathcal{B}$ describe relational examples through their attributes and relationships. The citation information is then used as the positive examples in the KB. To generate the negative examples for the baselines, the closed world assumption was used. First, all pairs of citation (relations of the form $citations(citing\ paper\ id, cited\ paper\ id)$) were collected and the cross-product of these combinations were generated. The pairs not in the positive examples constitute the negative examples thereby completing the construction of the KB. The number of facts, positive and negative examples grouped by each conference is shown in the table 6.1.

Table 6.1: Properties of the extracted knowledge base grouped by conference

Conference	# facts in Knowledge Base	#Positive examples	#Negative examples
ICML	71277	77526	142079901
ICLR	2764	4086	424937
IJCAI	115575	98730	383165323
NeurIPS	107368	110926	308649177

Figure 6.3 shows the top 5 fields of study for the papers accepted in the 4 major conferences and figure 6.4 shows the count of the number of authors, unique papers and the coauthor relations

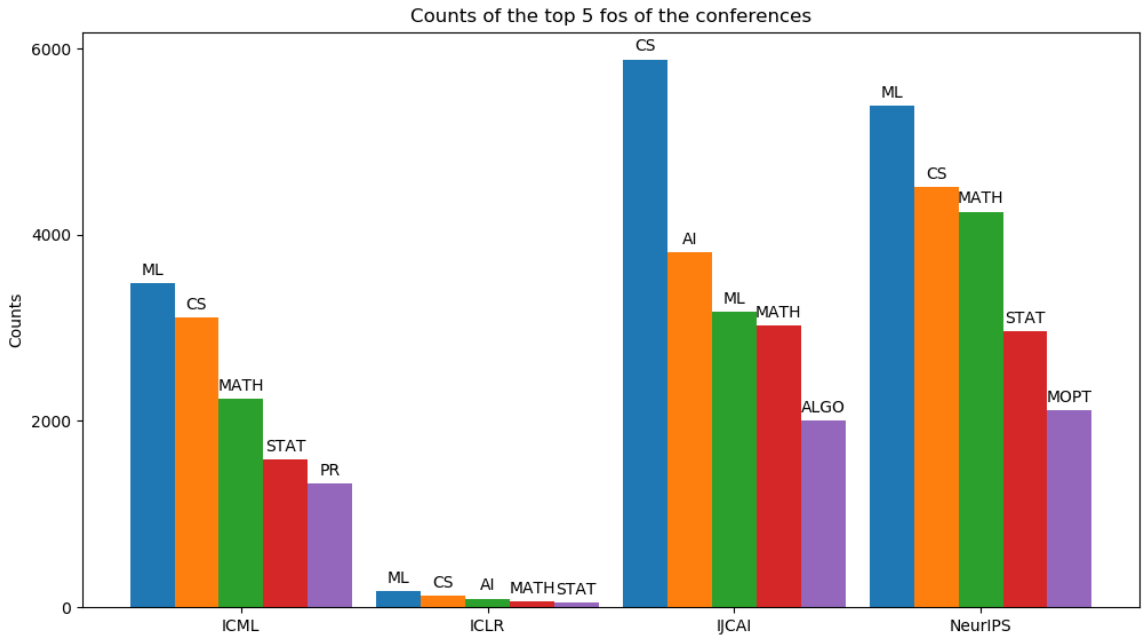


Figure 6.3: Counts of the top 5 fields of study (research topics) in the knowledge base, grouped by the conference name. Here ML="Machine Learning", AI="Artificial Intelligence", CS="Computer Science", MATH="Mathematics", STAT="Statistics", PR="Pattern Recognition", ALGO="Algorithms" and MOPT="Mathematical Optimization".

in the extracted knowledge base respectively. Not surprisingly, "Machine Learning" is the field of study for most of the papers in all the conferences except IJCAI since these are primarily machine learning conferences. In IJCAI, papers with "Computer Science" as the field of study outnumber the other fields showing that papers with a more general focus are accepted in the conference.

Also papers with the field of study as "Mathematics" appear in every conference thereby depicting the connection of the areas of mathematics and ML/AI. Some more interesting patterns are visible such as the presence of a lot of "Mathematical optimization" papers in NeurIPS and "Algorithms" papers in IJCAI thereby showing the difference in focus of these conferences.

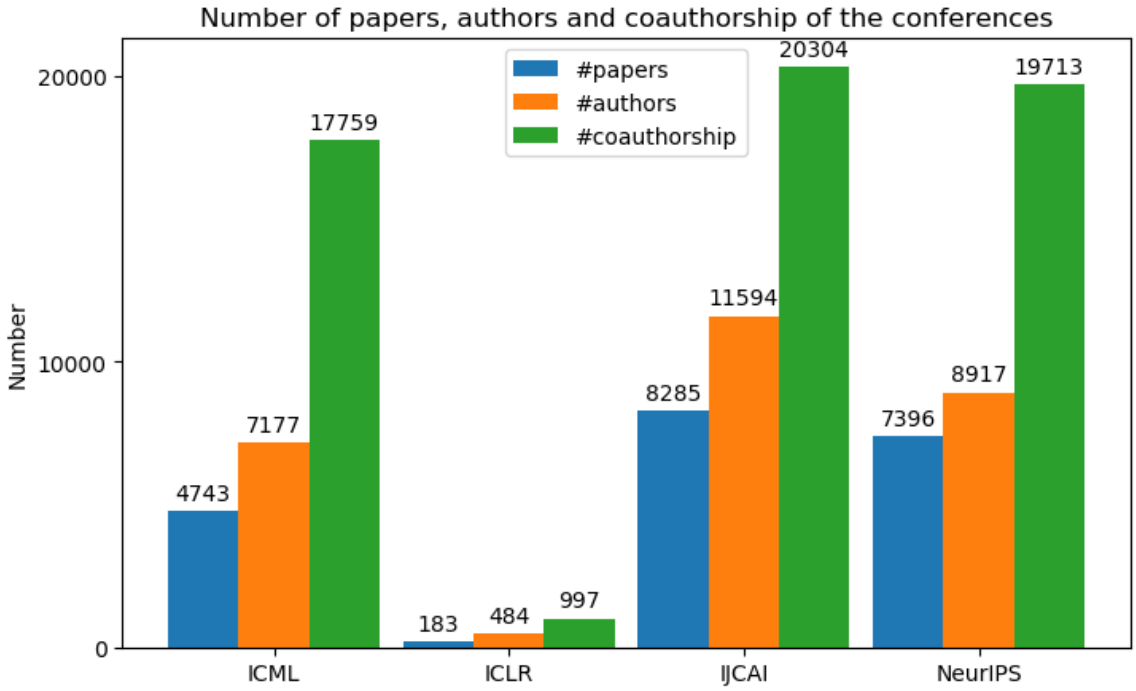


Figure 6.4: Count of unique papers, authors and coauthor relations grouped by conference name.

6.4 Experimental Results

Recall that our original goal is to perform link prediction on the generated KB. To evaluate the algorithm, we use a reduced KB consisting of papers from ICML 2018 and predicted coauthor relation. The properties of this reduced KB is shown in table 6.2.

Table 6.2: Properties of the reduced knowledge base for ICML'18

# facts in Knowledge Base	#Positive examples	#Negative examples
1395	155	6498

The reduced KB is shown in Figure 6.5 with a smaller snippet of 4 authors and their field of study (research topics) is shown in Figure 6.6. Note that some interesting properties about the entities in the KB can be extracted. For example, "Yann LeCun" and "Pieter Abbeel" have not coauthored and do not share the same affiliation but the KB suggests that their fields of study highly intersects with one another.

As the first step in the coauthor link prediction, we learn the first order logic rules using the density estimation method of (Khot et al., 2014) using relational trees from positive examples. The number of rules learned is 7. Some examples of the learned rules are given in table 6.3. The feature matrix $\mathcal{X}$ and the distance matrix $\mathcal{D}$ are then obtained following the method described in section 6.2.

Table 6.3: Example rules learned by the density estimation method for +ve and -ve examples. Here "MO" = Mathematical Optimization and "PR" = Pattern Recognition.

ResearchTopic(B, "MO") ∧ ResearchTopic(A, "PR") ∧ ResearchTopic(A,C) ∧ ResearchTopic(B,C)
Affiliation(A, C) ∧ Affiliation(B, C)
ResearchTopic(A, C) ∧ ResearchTopic(B, C)

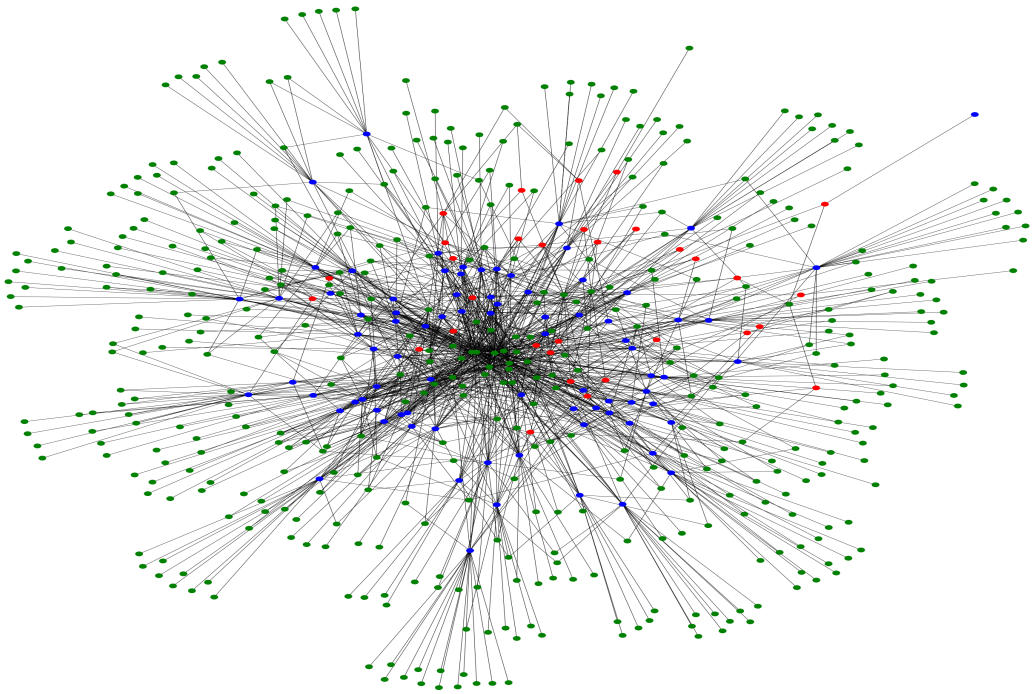


Figure 6.5: Snippet of the original Knowledge Base for the coauthor network. The blue nodes denote the authors, the red nodes denote the institutes and green nodes denote the institute type, location of institute and the research topic of each author (Best viewed in color).

6.4.1 Baselines

We compare our method, ROCGCN, to 3 relational embedding baselines.

1. Gaifman models (Niepert, 2016): This model makes use of the Gaifman locality principal (Gaifman, 1982) to simply enumerate all **hand-written first order rules** (relational features) of the specific kind within the neighborhood of the target/query variables. For our experiments handwritten rules for the reduced knowledge base are constructed. After obtaining the counts for the satisfied grounded handwritten rules, which serve as propositional features, logistic regression is used for prediction.

2. Relational GCN (Schlichtkrull et al., 2018): This model extends the GCN to the relational setting. This method can handle different weighted edge types i.e. relations and uses a two step message passing techniques to learn new node representations. For the task of link prediction these node representations are fed to a factorization method, DistMult (Yang et al., 2015), to predict the possible link between the nodes. We use the tenserflow implementation[1] for our baseline.

3. ComplEx (Trouillon et al., 2016): This model proposes a latent factorization approach for the problem of link prediction in multi-relational graphs and uses vectors with complex values for entities and relations. The scoring function for each example (a triple) is a combination of the real and imaginary parts of the vectors of both the entities and the relations. We use the ComplEx implementation in the AmpliGraph python library[2] for our baseline.

A comparison of our method, ROCGCN, with the baselines is shown in table 8.1.

For ROCGCN, the examples for training, validation and testing are randomly sampled without replacement from the KB while for R-GCN and ComplEx, since they are trained on true relations, the positive examples are randomly split with 80, 20 and 55 examples in training, validation and

[1] https://github.com/MichSchli/RelationPrediction

[2] https://github.com/Accenture/AmpliGraph

Table 6.4: Comparison of our method with state of the art baselines

Methods	Recall	F1	AUC-PR
Gaifman (Niepert, 2016)	0.10	0.186	0.127
R-GCN (Schlichtkrull et al., 2018)	0.636	0.13	0.13
ComplEx (Trouillon et al., 2016)	**0.85**	0.03	0.04
ROCGCN (Our method)	0.389	**0.561**	**0.556**

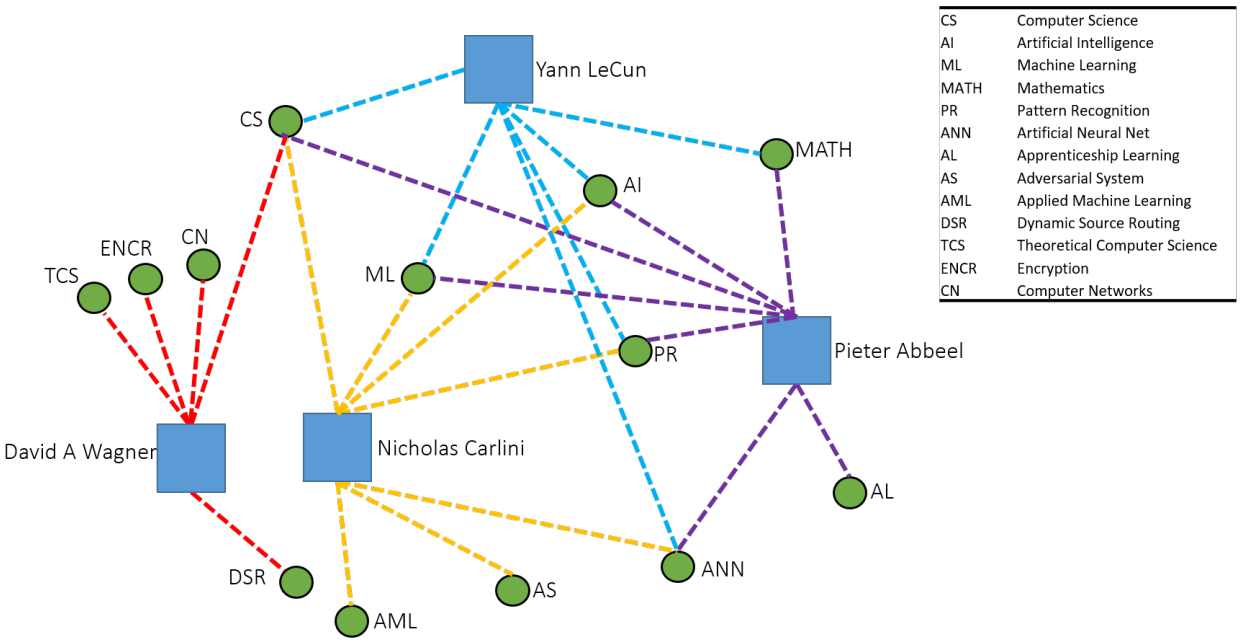


Figure 6.6: Snippet of the original Knowledge Base for the coauthor network focusing on 4 authors and their research topics.

testing respectively. To obtain the recall, F1 and AUC-PR values for the baselines (R-GCN and ComplEx), the scores for each pair of nodes in the test examples were thresholded by the average of the obtained scores. If the score obtained between a pair of nodes is greater than average score we predict the link else we predict no link. As can be seen from the results, our method outperforms the baselines significantly in the F1 and AUC-PR metrics. Note that although the recall is high for the neural embedding baselines, the corresponding F1 score and AUC-PR is very low which implies that these methods have a high rate of false positives. This clearly demonstrates that ROCGCN is significantly better than the strong baselines for the link prediction task.

6.5 Conclusion

We presented the first GCN method that can learn from positive only multi-relational data. Our ROCGCN does not make assumptions on the supervision or the arity of predicates and automatically constructs rules that allow for a rich latent representation. ROCGCN significantly outperforms the recently successful methods on link prediction task in the knowledge base extracted from the Microsoft Academic Graph. Extensive evaluations on other data sets including drug-drug interactions is an interesting direction of research. Allowing for joint learning and inference over multiple types of relations is another future direction. Finally, learning with the presence of hidden/latent data is essential for deploying SRL methods in real tasks.

PART III

LEARNING SYNTHETIC HEALTH CARE DATA

CHAPTER 7

HUMAN-GUIDED DEEP GENERATIVE MODELS

A key major challenge when learning from health care data is the difficulty in obtaining health care related data due to privacy and cost issues. This has, to an extent, limited the use of powerful machine learning models that can help reveal various underlying patterns in the data that the human experts could potentially miss. Thus, the use of generative machine learning models to generate synthetic data becomes of utmost importance. Also, since machine learning models benefit by using domain knowledge incorporating such knowledge in generative models can help generate better quality data. To this extent, we propose knowledge based deep generative models that can be used to create synthetic health care data which can mitigate the above issues and encourage high level machine learning research in health care.

7.1 Introduction

Although a huge amount of medical data is generated around the world (Dinov, 2016; Groves et al., 2016) a number of issues exist. These issues can primarily categorized into the following:

1. Obtaining medical data sets may be difficult and coslty.

2. Privacy issues (Barrows Jr and Clayton, 1996): Utmost care needs to be taken such that the identification of the patient from the corresponding medical records cannot be done.

3. Access issues (Doshi et al., 2016): Even if a medical data set is affordable, there are a lot of permissions needed to access it. HIPAA guidelines must be closely followed, thereby making the use of real medical data sets for research purposes difficult.

Due to all this existing issues, generation of synthetic data sets has been studied extensively (Jeske et al., 2006; Buczak et al., 2010; Shamsuddin et al., 2018) and this area of research has received a significant push (Guibas et al., 2017; Mahmood et al., 2018; Frid-Adar et al., 2018) after the

introduction of generative adversarial networks (GANs) (Goodfellow et al., 2014). If we generate synthetic medical data addressing all the above mentioned issues, the application of technologies such as machine learning can become more prevalent. Real world medical data is highly multimodal i.e. the probability distribution which describes the data has multiple "peaks" where different sub-groups of samples are concentrated (Beam et al., 2018). Creation of a synthetic data set will remove the cost and privacy issues associated with real data thereby making medical data sets more accessible to researchers in different communities. Though it has to be made sure that the synthetic data distribution is as close to the real data distribution as possible without giving the patient information away.

Most clinical models are built on confidential patient records, so data sharing is rare. Synthetic medical data could enable researchers to design a new generation of reproducible clinical decision support models, along with standardized performance benchmarks for new methods. Thus, the problem of generating synthetic data sets from real medical data sets keeping the intrinsic relationships between the features intact becomes important. In medical domain, such relationships are important and can be provided as additional knowledge or advice to the model in order to faithfully generate data. Incorporation of extra knowledge into systems has shown to create better performing and more explainable models (Towell and Shavlik, 1994; Fung et al., 2002; Kunapuli et al., 2010) and thus incorporating advice providing extra knowledge to the model regarding these relationships can help in generating better quality data.

Inspired by these, we consider the hugely successful generative adversarial networks (GAN) and propose the **H**uman-**A**llied **G**enerative **A**dversarial **N**etworks (HA-GAN) model where correlations between different features are captured and used to create a new generated data using the Iman-Conover method (Iman and Conover, 1982) which is then passed back to the GAN for training. We make the following contributions: (1) We present the first work on incorporating human advice to generate better quality data, (2) We present the first work using GANs for creation of synthetic data from small clinical data sets, (3) Finally, our experiments show that including extra knowledge

about the feature relationships produce higher quality synthetic data as can be seen by the higher performance of prediction models on synthetic data with advice when compared to synthetic data without advice.

7.2 Human-Allied GANs

The key principle behind Generative Adversarial Networks (Goodfellow et al., 2014) is a zero-sum game (Kuhn and Tucker, 1953), from game theory, which is a mathematical representation of a situation where each participant's gain or loss is exactly balanced by the losses or gains of the other participants and is generally solved by a minimax algorithm. GANs leverage this concept to estimate generative distributions by emulating such a zero-sum game between 2 models, namely, the generator and the discriminator (can also be considered as an actor-critic model). The generator tries to generate examples as close to the real distribution as possible starting from a random distribution to fool the discriminator and the discriminator tries to correctly distinguish real examples from the generated examples. The generator tries to maximize the probability that the discriminator makes a mistake and the discriminator tries to minimize its mistakes thereby resulting in a min-max optimization problem which can be solved as any mini-max algorithm. Following the notations in (Goodfellow et al., 2014), let $\mathcal{G}$ and $\mathcal{D}$ represent the generator and the discriminator respectively. To learn the generator distribution $p_{data}(\boldsymbol{x})$ over the given data $\boldsymbol{x}$, samples $\boldsymbol{z}$ are obtained from a random distribution $p_{\boldsymbol{z}}(\boldsymbol{z})$. This random distribution was initially proposed to be a uniform distribution but research has shown that a Gaussian distribution works better (Arjovsky and Bottou, 2017). The zero-sum game played between the discriminator $\mathcal{D}$ and the generator $\mathcal{G}$ can then be formulated as the following optimization problem:

$$\begin{aligned}\min_{\mathcal{G}} \max_{\mathcal{D}} V(\mathcal{D}, \mathcal{G}) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\mathbf{x})}[\log \mathcal{D}(\boldsymbol{x})] + \\ \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}}(\boldsymbol{z})[\log(1 - \mathcal{D}(\mathcal{G}(\boldsymbol{z})))]\end{aligned} \tag{7.1}$$

One of the many disadvantages of an adversarial training formulation is that the training is slow and very unstable, leading to the problem of mode collapse (Arjovsky and Bottou, 2017) where

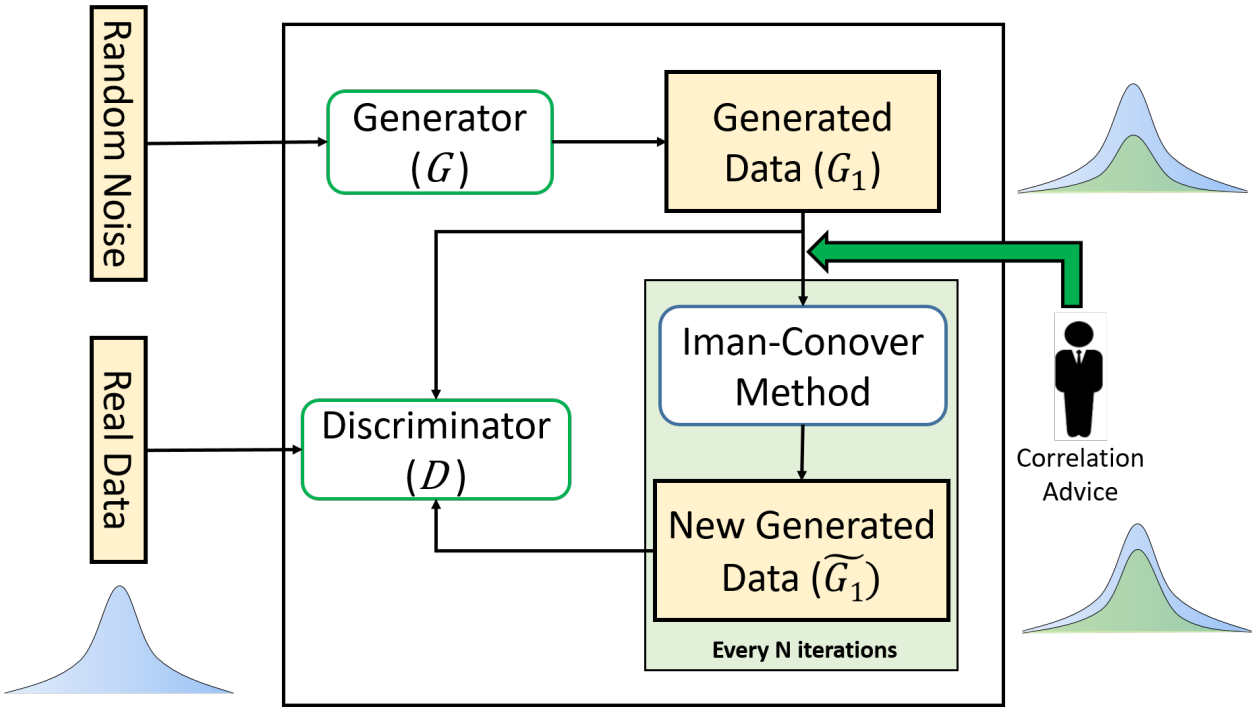


Figure 7.1: Proposed Human-Allied Generative Adversarial Network architecture.

the generator starts generating data of only a single modality. As a result, generative adversarial networks have not been exploited to their full potential in generating synthetic non-image medical data sets. Since human advice can force a model to concentrate on different areas of the feature space and helps learn more stable models (Odom and Natarajan, 2018), we propose a human-allied generative adversarial network (HA-GAN) architecture (figure 7.1). The architecture consists of human advice in form of feature correlations since such intrinsic relationships between the features in medical data sets are important and thus become a natural candidate to be provided as additional knowledge or advice to the model in order to faithfully generate data.

Our proposed approach starts with a simple generative adversarial network (GAN) architecture (Goodfellow et al., 2014) where a random noise vector is provided to the generator which tries to generate examples as close to the real distribution as possible. The discriminator then tries to identify examples from the real data from examples generated by the generator. The generator tries to maximize the probability that the discriminator makes a mistake and the discriminator

tries to minimize its mistakes thereby resulting in a min-max optimization problem which can be solved as a mini-max algorithm. In our work we use the Wasserstein generative adversarial network (WGAN) architecture (Arjovsky et al., 2017; Gulrajani et al., 2017) that focuses on defining a distance/divergence (Wasserstein or earth movers distance) to measure the closeness between the real distribution and the model distribution i.e. the distribution of the data and the examples generated by the generator.

7.2.1 Human input as inductive bias

There are two major ways of using human input to induce bias in a machine learning model. One way is to provide advice on the labels, as hard constraints or preferences) to constrain the search space. For example, while predicting heart attack some example advice rules on the labels might include:

1. $(200 \leq \text{cholesterol level} \leq 300) \Rightarrow \text{label} = 1$
2. $(80 \leq \text{cholesterol level} \leq 100) \wedge (80 \leq \text{diabetes} \leq 100) \Rightarrow \text{label} = 0$

These types of advice make more sense in an discriminative setting since the goal is to not change the real data in the GAN setting. Since GANs are shown to be sensitive to the training data and the labels are getting generated, it is desirable to not change these during the training process. Another way of providing advice is to use correlations between features as preferences.

After a fixed number of iterations, `N`, we calculate the correlation matrix of the generated data and provide a set of advice ψ on the correlation's between different features. We first motivate the use of correlations as a form of advice with an example.

Example: *Consider a heath care data set to predict heart attack, with 3 features say, cholesterol level, blood pressure, income level. The values of the given features can vary (sometimes widely) between different patients due to several latent factors that might not be captured in the data set (say smoking habits), it is difficult to come up with a distributional assumption for the given features.*

In other words, it is difficult to deduce whether the values for the features come from the same distribution (even though the feature values in the data set are similar).

We propose to increase the co-variance coefficient between the features if the advice suggests that two features are highly correlated. Conversely, we decrease the co-variance coefficient if the advice says otherwise.

Example: *Continuing the above example, since raise in the cholesterol level can cause a rise in blood pressure and vice versa, an expert advice here can be that cholesterol level and blood pressure should be highly correlated. Also, since income is (generally) not a factor in one's blood pressure and cholesterol levels, another advice here can be to decrease the correlation between both cholesterol level and blood pressure with the income level* (note that these are for illustrative purposes only).

Thus some example advice rules $\in \psi$ are:

1. Correlation("cholesterol level","blood pressure")$\uparrow$
2. Correlation("cholesterol level","income level")$\downarrow$
3. Correlation("blood pressure","income level")$\downarrow$

Based on the 1st advice we need to increase the correlation coefficient between *cholesterol level* and *blood pressure*. Then

$$\mathcal{C} = \begin{bmatrix} 1 & 0.2 & 0.3 \\ 0.2 & 1 & 0.07 \\ 0.3 & 0.07 & 1 \end{bmatrix} \mathcal{A} = \begin{bmatrix} 1 & \lambda & 1 \\ \lambda & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{7.2}$$

$$\hat{\mathcal{C}} = \mathcal{C} \odot \mathcal{A} = \begin{bmatrix} 1 & 0.2 & 0.3 \\ 0.2 & 1 & 0.07 \\ 0.3 & 0.07 & 1 \end{bmatrix} \odot \begin{bmatrix} 1 & \lambda & 1 \\ \lambda & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{7.3}$$

Here $\mathcal{C}$ is the correlation matrix, $\mathcal{A}$ is the advice matrix, $\hat{\mathcal{C}}$ is the new correlation matrix and λ is the factor by which the correlation value has to be increased or decreased. In case of our example, we need to increase the value of the correlation coefficient, and thus λ should be > 1. We keep the value of $\lambda = \frac{1}{max(|\mathcal{C}|)}$. Since -1.0 $\leq \forall c \in \mathcal{C} \leq$ 1.0, in this case, the value of $\lambda \geq$ 1.0 and thus multiplying by λ will increase the correlation value. Thus,

$$\begin{aligned}\hat{\mathcal{C}} = \mathcal{C} \odot \mathcal{A} &= \begin{bmatrix} 1 & 0.2 & 0.3 \\ 0.2 & 1 & 0.07 \\ 0.3 & 0.07 & 1 \end{bmatrix} \odot \begin{bmatrix} 1 & \frac{1}{0.3} & 1 \\ \frac{1}{0.3} & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0.667 & 0.3 \\ 0.667 & 1 & 0.07 \\ 0.3 & 0.07 & 1 \end{bmatrix}\end{aligned} \tag{7.4}$$

In case of negative correlations, the process remains the same. If the advice says that features have low correlations (the 2nd advice in our example), then we need to decrease the correlation coefficient and in that case we need λ to be < 1 and we choose $\lambda = max(|\mathcal{C}|)$. Since -1 $\leq \forall c \in \mathcal{C} \leq$ 1.0, in this case, the value of $\lambda \leq$ 1.0 and thus multiplying by λ will decrease the correlation value, and the new correlation matrix becomes,

$$\begin{aligned}\hat{\mathcal{C}}_1 = \hat{\mathcal{C}} \odot \mathcal{A} &= \begin{bmatrix} 1 & 0.667 & 0.3 \\ 0.667 & 1 & 0.07 \\ 0.3 & 0.07 & 1 \end{bmatrix} \odot \begin{bmatrix} 1 & 1 & 0.3 \\ 1 & 1 & 0.3 \\ 0.3 & 0.3 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0.667 & 0.09 \\ 0.667 & 1 & 0.021 \\ 0.09 & 0.021 & 1 \end{bmatrix}\end{aligned} \tag{7.5}$$

which can the be used to create the new generated data $\tilde{\mathcal{G}}_1$ as described next. As with the earlier case, the process of generating data with negative correlations remains the same.

7.2.2 Post-advice data generation

After we obtain the new correlation matrix, $\hat{\mathcal{C}}_1$, the next step is to generate data that is faithful to the modified correlation matrix based on the expert advice. To achieve this we use the Iman-Conover method (Iman and Conover, 1982) which is a distribution free method, to define dependencies between distributional variables based on rank correlations such as Spearman or Kendell Tau correlations. Since we deal with linear relationships between the features and since Pearson coefficient has shown to perform better with the Iman-Conover method (Naveršnik and Rojnik, 2012), we use the Pearson correlations between features. Also note that we assume that the features in our data sets follow a Gaussian distribution. This might seem a strong assumption at first, but since we are making use of only the clinical data (mostly lab test values), the data can safely be assumed to be following the Gaussian distribution. The Iman-Conover method consists of the following steps:

1. Create a random standardized matrix $\mathcal{M}$ with values $x \in \mathcal{M} \sim$ Gaussian distribution. This is obtained by the process of inverse transform sampling described as follows. Let $\mathcal{V}_1$ be a uniformly distributed random variable and $\mathcal{CDF}$ be the cumulative distribution function of the distribution to be sampled from which takes a value v and defines the probability of obtaining a value less than the given value v.

$$\mathcal{CDF}(v) = \mathcal{P}(V \leq v) \tag{7.6}$$

   Thus, to generate samples from a desired distribution, the values $v \sim \mathcal{V}$ are passed through $\mathcal{CDF}^{-1}$ to obtain the desired values x.

$$\mathcal{CDF}^{-1}(v) = inf\{x|\mathcal{CDF}(x) \leq v, \quad v \in [0,1]\} \tag{7.7}$$

where $\inf$ is the inverse function. For Gaussian distribution,

$$\begin{aligned} \mathcal{CDF}(x) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp^{\frac{-x^2}{2}} dx \\ &= \frac{1}{\sqrt{2\pi}} \int_{0}^{x} \exp^{\frac{-x^2}{2}} dx \\ &= [-\exp(\frac{-x^2}{2})]_0^x \\ \mathcal{CDF}^{-1}(v) &= 1 - \exp(\frac{-x^2}{2}) \leq v \end{aligned} \tag{7.8}$$

Thus, the desired values $x \in \mathcal{M}$ can be obtained as,

$$x = \sqrt{2ln(1-v)} \tag{7.9}$$

2. Calculate the correlation matrix $\mathcal{E}$ of $\mathcal{M}$.

3. Calculate the Cholesky decomposition $\mathcal{F}$ of the correlation matrix $\mathcal{E}$. Cholesky decomposition (Scheuer and Stoller, 1962) of a *positive-definite* matrix is given as the product of a lower triangular matrix and its conjugate transpose. Note that for Cholesky decomposition to be unique, the matrix being decomposed should be positive definite, such as the co-variance matrix whereas the correlation matrix, used in our algorithm, is only positive semi-definite. We convert the correlation matrix into a positive-definite matrix by adding very small values to the diagonal of the correlation matrix. Specifically, we keep adding 0.01 to the diagnol until the matrix is positive definite. Given a symmetric and positive definite matrix. $\mathcal{E}$ in our case, its Cholesky decomposition $\mathcal{F}$ is such that $\mathcal{E} = \mathcal{F} \cdot \mathcal{F}^\top$.

4. Calculate the Cholesky decomposition $\mathcal{Q}$ of the correlation matrix obtained after modifyications based on human advice, $\hat{\mathcal{C}}$. As above, the Cholesky decomposition is such that $\hat{\mathcal{C}} = \mathcal{Q} \cdot \mathcal{Q}^\top$.

5. Calculate the reference matrix $\mathcal{T}$ by transforming the sampled matrix $\mathcal{M}$ from step 1 to have the desired correlations of $\hat{\mathcal{C}}$, by using their Cholesky decompositions.

6. Rearrange values in columns of the generated data $\mathcal{G}_1$ to have the same ordering as corrresponding column in the reference matrix $\mathcal{T}$.

7.2.3 Cholesky decomposition captures correlations

Given an randomly generated data set with no correlations $\mathcal{P}$, a correlation matrix $\mathcal{C}$ and its Cholesky decomposition $\mathcal{Q}$, data that faithfully follows the given correlations $\in \mathcal{C}$ can be generated by multiplication of the obtained lower triangular matrix with the original uncorrelated data i.e. $\hat{\mathcal{P}}$=$\mathcal{QP}$. The correlation of the newly obtained data, $\hat{\mathcal{P}}$ is,

$$\begin{aligned} Corr(\hat{\mathcal{P}}) &= \frac{Cov(\mathcal{P})}{\sigma_{\hat{\mathcal{P}}}} \\ &= \frac{\mathbf{E}[\hat{\mathcal{P}}\hat{\mathcal{P}}^\top] - \mathbf{E}[\hat{\mathcal{P}}]\mathbf{E}[\hat{\mathcal{P}}]^\top}{\sigma_{\hat{\mathcal{P}}}} \end{aligned} \tag{7.10}$$

Since we consider uncorrelated data $\mathcal{P}$ with zero mean and unit variance,

$$\begin{aligned} Corr(\hat{\mathcal{P}}) &= \frac{\mathbf{E}[\hat{\mathcal{P}}\hat{\mathcal{P}}^\top] - \mathbf{E}[\hat{\mathcal{P}}]\mathbf{E}[\hat{\mathcal{P}}]^\top}{\sigma_{\hat{\mathcal{P}}}} \\ &= \mathbf{E}[\hat{\mathcal{P}}\hat{\mathcal{P}}^\top] = \mathbf{E}[(\mathcal{QP})(\mathcal{QP})^\top] \\ &= \mathbf{E}[\mathcal{QP}\mathcal{Q}^\top\mathcal{P}^\top] = \mathcal{Q}\mathbf{E}[\mathcal{P}\mathcal{P}^\top]\mathcal{Q}^\top \\ &= \mathcal{Q}\mathcal{Q}^\top = \mathcal{C} \end{aligned} \tag{7.11}$$

Thus Cholesky decomposition can capture the desired correlations faithfully and can be used for generating correlated data. Since we already have a normal sampled matrix $\mathcal{M}$ and a calculated correlation $\mathcal{E}$ of $\mathcal{M}$, we need to calculate a reference matrix (step 5).

7.2.4 Human-Allied GAN training

Since the human expert advice is provided independent of the GAN architecture, our method is agnostic of the underlying GAN architecture. As mentioned earlier, we make use of Wassertein GAN (WGAN) architecture since its shown to be more stable while training and can handle mode collapse (Arjovsky et al., 2017). Only the error backpropagation values differ when we are using the data generated by the underlying GAN or the data generated by the Iman-Conover method. Algorithm 5 shows the overall process of Human-Allied GANs.

Algorithm 5 HA-GAN: **H**uman-**A**llied **G**enerative **A**dversarial **N**etworks

procedure HA-GAN(Data $\mathcal{G}$, Generator $\mathcal{G}$, Discriminator $\mathcal{D}$, Correlation Advice ψ)
  for epochs k=1 to K **do**
    Sample $\mathbb{Z}_{\mathfrak{m}}$, $\mathfrak{m}$ examples from noise vector $\mathbb{Z}$
    Sample $\mathbb{X}_{\mathfrak{m}}$, $\mathfrak{m}$ examples from real data $\mathbb{X}$
    $\mathcal{G}_1$=$\mathcal{G}(\mathbb{Z}_{\mathfrak{m}})$ ▷ generate data from noise using generator
    if k $\bmod$ N is 0 **then** ▷ number of epochs $1 \leq$ N $\leq$ K
      $\tilde{\mathcal{G}}_1$ = IMAN-CONOVER($\mathcal{G}$)
      $\hat{y} = \mathcal{D}(\mathbb{X}_{\mathfrak{m}}, \tilde{\mathcal{G}}_1)$ ▷ pass generated data after advice and real data to discriminator; backpropagate error
    else
      $\hat{y} = \mathcal{D}(\mathbb{X}_{\mathfrak{m}}, \mathcal{G}_1)$ ▷ backpropagate error; no advice
    end if
  end for
end procedure

---

procedure IMAN-CONOVER(Generated data $\mathcal{G}_1$)
  $\mathcal{C}$ = CALCCORR($\mathcal{G}_1$) ▷ calculate correlation matrix
  λ_{inc}= $\frac{1}{max(\mathcal{C})}$ ▷ calculate correlation increasing factor
  λ_{dec}= $max(\mathcal{C})$ ▷ calculate correlation decreasing factor
  $\mathcal{A}$=CREATEADVICEMAT(ψ) ▷ See section 7.2.1
  $\hat{\mathcal{C}} = \mathcal{C} \odot \mathcal{A}$ ▷ $\lambda \in \mathcal{A}$. See section 7.2.1
  for t epochs **do**
    $\mathcal{M}$ = INVSAMP($\mathfrak{m}, \mathfrak{d}$) ▷ $\mathfrak{m}$: #examples, $\mathfrak{d}$:#features
    $\mathcal{E} = \frac{\mathcal{M}^\top \cdot \mathcal{M}}{\mathfrak{m}}$ ▷ correlation matrix of $\mathcal{M}$
    calculate $\mathcal{F}$ s.t. $\mathcal{E}$=$\mathcal{F} \cdot \mathcal{F}^\top$ ▷ Cholesky decomp of $\mathcal{E}$
    calculate $\mathcal{Q}$ s.t. $\hat{\mathcal{C}}$=$\mathcal{Q} \cdot \mathcal{Q}^\top$ ▷ Cholesky decomp of $\hat{\mathcal{C}}$
    calculate $\mathcal{T} = \mathcal{M}\mathcal{F}^{-1}\mathcal{Q}$ ▷ reference matrix with desired correlations
    $\mathcal{R}$ = RANK($\mathcal{T}$)
    $\tilde{\mathcal{G}}_1$ = REORDER($\mathcal{R}$,$\mathcal{G}_1$)
  end for
  return $\tilde{\mathcal{G}}_1$
end procedure

---

procedure INVSAMP(number of sampled examples $\mathfrak{m}$, number of features in data $\mathfrak{d}$)
  $\mathfrak{a}$=generate $\mathfrak{m}$ random numbers
  $\mathfrak{p}$ = QUALTILEFUNCTION($\mathfrak{a}$) ▷ induce Gaussian dist
  $\hat{\mathfrak{p}}$ = NORMALISE($\mathfrak{p}$)
  $\mathcal{M}$ = PERMUTE($\hat{\mathfrak{p}}$) ▷ shuffle the sampled numbers
end procedure

Our algorithm starts with the general process of training a GAN where the generator takes random noise as an input and generates data which is then passed, along with the real data, to

the discriminator. The discriminator tries to identify the real and generated data and the error is back propagated to the generator. After every specified number of iterations the correlations between features $\mathcal{C}$ in the generated data is obtained and a new correlation matrix $\hat{\mathcal{C}}$, is obtained with respect to the expert advice (section 7.2.1). A new data set is generated with respect to $\hat{\mathcal{C}}$ using the Iman-Conover method (Section 7.2.2) which is then passed to the discriminator along with the real data set. We discuss our results next.

7.3 Experiments

We aim to answer the following questions:

Q1: Does providing advice to generative adversarial networks help in generating better quality data?

Q2: Are GANs with advice effective for data sets that are not exhaustive i.e. have less examples?

Q3: How does bad advice effect the quality of generated data?

Q4: How well does human advice handle class imbalance?

Data sets: We consider 3 novel clinical data sets to test our method. Table 7.1 contains the properties of these domains.

Table 7.1: Evaluation domains and their properties.

Data set	#Features	#Positive examples	#Negative examples
Nephrotic Syndrome	19	44	6
Parkinson's	35	554	919
Alzheimer's	68	76	260

1. **Nephrotic Syndrome:** Nephrotic syndrome is a collection of symptoms that indicate kidney damage. Some of the symptoms include: albuminuria (large amounts of protein in the urine), hyperlipidemia (higher than normal fat and cholesterol levels in the blood) and hypoalbuminia (low levels of albumin in the blood). This novel data set consists of 50 kidney biopsy images

along with the clinical reports sourced from Dr Lal PathLabs, India [1]. In this work we make use of the clinical reports that consists of the values for kidney tissue diagnosis which can confirm the clinical diagnosis. The analysis will help to identify high-risk patients and influence treatment decisions and will help medical practitioners to plan and prognosticate treatments.

2. **Parkinson's:** data is defined in chapter 3 and consists of aggregated clinical data for prediction of Parkinson's disease and is obtained from an observational study with the main aim of identifying features or biomarkers that impact Parkinson's disease progression.

3. **Alzheimer's:** data is obtained from the Alzheimer's Disease Neuroimaging Initiative(ADNI) study aimed to find progression markers for Alzheimer's disease and develop treatments that can slow the progression. The ADNI database consists of a variety of data ranging from imaging data to genetic and bio-specimen markers. For this work we consider data obtained by processing the MRI images of 336 patients.

Table 7.2: Train on synthetic, test on real (TSTR) Results ($\approx 3\ decimals$) for the medical domains.

Data set	Methods	Accuracy	Recall	F1	AUC-ROC	AUC-PR
Nephrotic Syndrome	GAN (no advice)	0.566	0.584	0.666	0.509	0.911
	GAN (bad advice)	0.44	0.42	0.511	0.518	0.886
	GAN (with advice)	**0.894**	**0.998**	**0.943**	**0.566**	**0.947**
Parkinson's	GAN (no advice)	0.684	0.316	0.429	0.611	0.622
	GAN (bad advice)	0.593	0.046	0.079	0.484	0.258
	GAN (with advice)	**0.737**	**0.576**	**0.622**	**0.705**	**0.706**
Alzheimer's	GAN (no advice)	0.319	0.855	0.362	0.509	0.559
	GAN (bad advice)	**0.755**	0.011	0.017	0.491	0.197
	GAN (with advice)	0.363	**0.879**	**0.388**	**0.545**	**0.58**

To measure the quality of the generated data we make use of the train on synthetic, test on real (TSTR) method as proposed in (Esteban et al., 2017). We use gradient boosting with 100 estimators

[1] https://www.lalpathlabs.com/

(decision stumps) as the underlying machine learning model. Table 7.2 shows the results of the TSTR method with data generated with and without advice and the data generated with advice has higher TSTR performance than the data generated without advice. Thus, to answer **Q1**, providing advice to generative adversarial networks does help in generating better quality data.

As results show in table 7.2 the TSTR results for GANs with advice are especially impressive in the nephrotic syndrome domain which consists of only 50 examples with a big jump across all metrics. In fact, all the considered data sets are small in size when compared to the large number of samples required to train a GAN model. Thus **Q2** can be answered affirmatively, GANs with advice are effective for data sets with less samples.

7.3.1 Providing incorrect advice

Table 7.2 also shows the results for data generated with incorrect advice. To incorporate the bad advice in the correlations we follow a simple process: if the advice says that the correlation between features should be high, we set the correlations in $\hat{\mathcal{C}}$ to 0 and if the advice says that the correlation should be low, we set the correlations in $\hat{\mathcal{C}}$ to be either 1 or -1. Continuing our example from section 7.2.1, the advice says that we need to increase the correlation coefficient between cholesterol level and blood pressure. Then the new correlation matrix after incorrect advice can be calculated as:

$$\mathcal{C} = \begin{bmatrix} 1 & 0.2 & 0.3 \\ 0.2 & 1 & 0.07 \\ 0.3 & 0.07 & 1 \end{bmatrix} \mathcal{A} = \begin{bmatrix} 1 & \lambda & 1 \\ \lambda & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{7.12}$$

$$\hat{\mathcal{C}} = \mathcal{C} \odot \mathcal{A} = \begin{bmatrix} 1 & 0.2 & 0.3 \\ 0.2 & 1 & 0.07 \\ 0.3 & 0.07 & 1 \end{bmatrix} \odot \begin{bmatrix} 1 & \lambda & 1 \\ \lambda & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{7.13}$$

Since we need to set the correlations to 0, the value of λ is set to 0.

$$\hat{\mathcal{C}} = \begin{bmatrix} 1 & 0.2 & 0.3 \\ 0.2 & 1 & 0.07 \\ 0.3 & 0.07 & 1 \end{bmatrix} \odot \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0.3 \\ 0 & 1 & 0.07 \\ 0.3 & 0.07 & 1 \end{bmatrix} \tag{7.14}$$

As results show in 7.2, giving bad advice adversely affects the TSTR performance of the machine learning algorithm thereby answering **Q3**.

The nephrotic syndrome and Alzheimer's data sets are relatively unbalanced with a pos to neg ratio of $\approx$ 8:1 and 1:3.5 respectively. Most of the medical data sets, except highly curated data sets, are expected to be unbalanced and thus a data generator model should be able to handle this imbalance. Since our method explicitly focuses on the correlations between features and generates better quality data based on such relationships between features, our method is more robust to the imbalance in the underlying data set. This can be seen in the results in table 7.2 where advice based data generation outperforms the non-advice and bad advice based data generation. Thus, we can answer **Q4**, human advice does handle class imbalance robustly.

7.4 Conclusion

We present a new GAN formulation by incorporating correlation information between features as advice to generate a new correlated data and train the underlying GAN model. We test our model on real clinical data sets and show that incorporating advice helps generate good quality synthetic medical data. We also consider a specific method to test the quality of synthetic data generated, namely, train on synthetic, test on real (TSTR) and show that the generated data with advice is more aligned with the real data than the generated data without any advice component.

There can be multiple future directions for our work. First, there can be multiple advice options that can be used, that can capture feature relationships explicitly. Second, posterior regularization (Ganchev et al., 2010) can be used on the generator as advice or an advice on the generator regarding the label can be used. Third, although in this work we do not have identifiers in the data we use, thereby eliminating the need of differential privacy (Dwork, 2008), a general framework that can uphold the privacy of patient data can be a natural next step.

CHAPTER 8

ADDITIONAL EXPLORATION

This chapter outlines additional research work that I was involved in and made significant contributions. Section 8.1 presents our graph-based approach (Das et al., 2019) for approximating counts of satisfied instances of generalized patterns which can be used for obtaining the counts of the satisfied first order features by extending many of our approaches such as Gaifman models in chapter 5 and ROCGCN in chapter 6.

Section 8.2 presents our work (Das et al., 2019) on augmenting a relational deep learning model with human advice inspired by the success of human-knowledge guided learning in AI, especially in data-scarce domain. In this work, we propose Knowledge-augmented Column Networks that leverage human advice/knowledge for better learning with noisy/sparse samples. This work can be used for the prediction task in our multi-relational domains such as drug-drug interactions (chapter 4) and can be used for comparison with our ROCGCN approach (chapter 6). This also aligns well with our approach in chapter 7 where we also augment a deep learning model (generative in this case) with expert human knowledge.

Predicting and discovering drug-drug interactions (DDIs) is an important problem and has been studied extensively both from medical and machine learning point of view. Almost all of the machine learning approaches have focused on text data or textual representation of the structural data of drugs. In section 8.3, we present the first work (Dhami et al., 2019) that uses drug structure images as the input and utilizes a Siamese convolutional network architecture to predict DDIs and is an important step in the path of using heterogeneous data for important healthcare problems.

8.1 Motif Based Approximate Counting via Hypergraphs

Significant advancements in research on Statistical Relational Learning (SRL) and AI (Raedt et al., 2016) and in lifted inference (Poole, 2003; Kersting et al., 2009) have allowed for exploiting

the symmetries of the data and model during learning and inference. The advantage of these algorithms is that they can succinctly represent and reason with dependencies among the attributes and relations of related objects. One of the key operations inside most, if not all, algorithms is *counting* the satisfied groundings (instances) of a partially instantiated relational rule (a first-order clause). For instance, when learning the parameters or structure of a Markov Logic Network (MLN) (Domingos and Lowd, 2009; Khot et al., 2011), or when performing lifted inference (Poole, 2003) one has to compute the expected/true counts in the model/data and inside a given cluster of objects. Counting is a hard combinatorial search problem ($\#P$-complete). Consequently, algorithms for fast, approximate counting have been developed (Das et al., 2016; Sarkhel et al., 2016). *The key observation is that computing exact counts is not essential, particularly when the number of groundings for an object/relation is high*. For instance, knowing whether a Professor published 300 papers or 319 papers does not significantly change the belief over the popularity of the Professor. While reasonably successful, they make a few assumptions – including MLN-specific formulation and lack of support for partial groundings (Sarkhel et al., 2016) or restricted arity of relations (Das et al., 2016). We relax these assumptions and present a *general approximate counting* technique that transforms the problem of counting partially instantiated clauses to motif-matching in equivalent hypergraph.

Provably, counting in the original data corresponds to **computing *the expected counts* of the motif occurrences** in the transformed hypergraph: When this expectation is computed exactly, one can retrieve the true counts. In large data sets, this motivates the approximation of the expectation using summary statistics on the hypergraph. Our *experimental results across several domains on both learning and inference tasks* demonstrate clearly that this approximation indeed relaxes the assumptions of the previous methods and results in more efficient counting while exhibiting on-par performance to exact counting.

Our goal is to compute the counts of a potentially partially instantiated clause given a database of ground assertions. We proceed by transforming a SRL model into a directed graph notation.

Trivial conversion from a logic statement (essentially a *conjunctive rule*) to a simple directed graph has an important limitation of assumimng binary relations. Such graphs, however, cannot represent n-ary relations, which are very common. We employ hypergraphs (Berge and Minieka, 1973), generalization of graphs in which a hyperedge joins an arbitrary number of nodes/vertices, in contrast to a graph in which an edge joins two vertices. Formally, a hypergraph is a pair of sets of vertices and hyperedges. Since a hypergraph has hyperedges that connect an arbitrary number of nodes, a hyperedge itself is a set of nodes. Our problem is.

Given: A set of grounded assertions (facts) $\mathcal{F}$, a conjunctive rule/clause $\mathtt{C}$ from a SRL model and a (possibly partial) substitution (instantiations) $\boldsymbol{\theta}$ of variables $\mathtt{C}$,

To Do: Return the counts of true groundings $\#(\mathtt{C} \mid \boldsymbol{\theta})$ of the clause $\mathtt{C}$,

Construct: A partially grounded structural hypergraph motif, $\mathcal{M} \equiv (\mathcal{V}_{\mathcal{M}}, \mathcal{E}_{\mathcal{M}})$ representing clause $\mathtt{C}$ and fully grounded hypergraph, $\mathcal{G} \equiv (\mathcal{V}_{\mathcal{G}}, \mathcal{E}_{\mathcal{G}})$ representing grounded assertions ($\mathcal{F}$) - count instances of $\mathcal{M}$ in $\mathcal{G}$ yields an approximation to $\#(\mathtt{C} \mid \boldsymbol{\theta})$.

Example 1. *A university domain has entity types (variables)* `Professor (p)`, `Student (s)`, `Course (c)`, `Term (t)`, `ResearchProject (r)` *and* `Year (y)`. *We consider two conjunctive rules in this domain:*

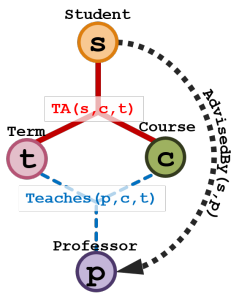
$$\mathtt{AdvisedBy}(\mathtt{s}, \mathtt{p}) \wedge \mathtt{Teaches}(\mathtt{p}, \mathtt{c}, \mathtt{t}) \wedge \mathtt{TA}(\mathtt{s}, \mathtt{c}, \mathtt{t}) \qquad (\mathtt{C}_1)$$

$$\mathtt{AdvisedBy}(\mathtt{s}, \mathtt{p}) \wedge \mathtt{WorksIn}(\mathtt{p}, \mathtt{r}, \mathtt{y}) \wedge \mathtt{WorksIn}(\mathtt{s}, \mathtt{r}, \mathtt{y}) \qquad (\mathtt{C}_2)$$

The first clause states that $\mathtt{s}$ *is advised by* $\mathtt{p}$ *and is a TA for* $\mathtt{c}$ *that* $=\mathtt{p}$ *teaches in* $\mathtt{t}$. *The second states that* $\mathtt{s}$ *advised by* $\mathtt{p}$ *works on* $\mathtt{r}$ *in a* $\mathtt{y}$. *In SRL they are* soft *statements.*

Definition 1 (Counts). *For* $\mathtt{Relation}(\mathtt{v}_1, \ldots, \mathtt{v}_t)$*, the predicate counts are the number of true instances of that predicate due to the assertions/facts* $\mathcal{F}$*, given the (partial) grounding* $\boldsymbol{\theta}$ *of the its variables. We denote the predicate counts of* `Relation (R)` *as* $n_{\mathtt{R}} = \#(\mathtt{R} \mid \boldsymbol{\theta})$.

For a clause $\mathtt{C}$*, the clause counts are the number of true instances of* $\mathtt{C}$ *in database* $\mathcal{F}$*, given the partial groundings* $\boldsymbol{\theta}$ *of the variables in the clause. We denote the clause counts for a clause* $\mathtt{C}$ *as* $n_{\mathtt{C}} = \#(\mathtt{C} \mid \boldsymbol{\theta})$.



Teaches(Amy, AI, Fa17),
Teaches(Amy, ML, Fa17)
Teaches(Amy, AI, Sp18),
Teaches(Amy, Opt, Sp18)
TA(Ben, AI, Fa17),
TA(Ena, ML, Fa17)
TA(Cam, AI, Sp18),
TA(Deb, Opt, Sp18)
AdvisedBy(Ben, Amy),
AdvisedBy(Deb, Amy),
AdvisedBy(Fei, Amy)

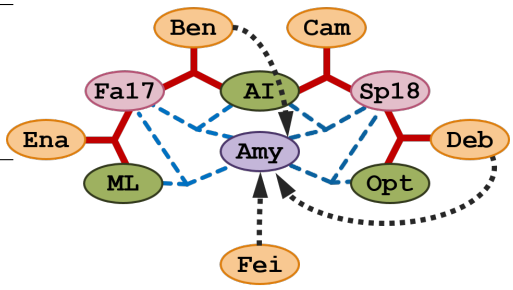
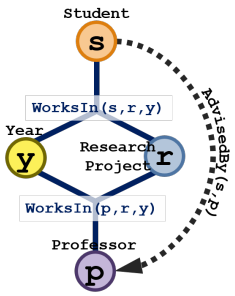


Figure 8.1: (**left**) Motif $\mathcal{M}_1$ for $\mathtt{C}_1$; (**center**) Facts used to ground $\mathcal{M}_1$; (**right**) Ground graph, $\mathcal{G}_1$. Ternary predicates `Teaches` and `TA` are represented as hyperedges in both $\mathcal{M}_1$ and $\mathcal{G}_1$. The edges `AdvisedBy(Deb, Amy)` and `AdvisedBy(Fei, Amy)` also appear in the grounding of $\mathtt{C}_2$ (Fig. 8.2).



WorksIn(Amy, MLNs, 2017),
WorksIn(Amy, PRMs, 2017)
WorksIn(Hal, MLNs, 2018)
WorksIn(Ben, PRMs, 2017),
WorksIn(Fei, PRMs, 2017)
WorksIn(Fei, MLNs, 2018),
AdvisedBy(Ben, Amy),
AdvisedBy(Deb, Amy),
AdvisedBy(Fei, Amy),
AdvisedBy(Fei, Hal),

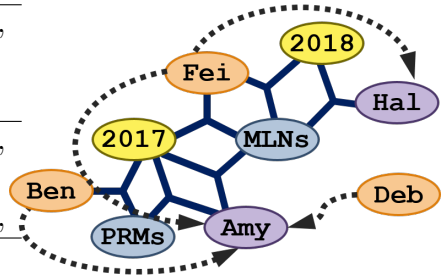


Figure 8.2: (**left**) Motif $\mathcal{M}_2$ for $\mathtt{C}_2$; (**center**) Facts used to ground $\mathcal{M}_2$; (**right**) Ground graph, $\mathcal{G}$. The ternary predicate `WorksIn` are hyperedges in $\mathcal{M}_2$ and $\mathcal{G}_2$. The edges `AdvisedBy(Deb, Amy)` (`Deb` $\rightarrow$ `Amy`) and `AdvisedBy(Fei, Amy)` (`Fei` $\rightarrow$ `Amy`) also appear in the grounding of $\mathtt{C}_1$ (see Fig. 8.1).

Example 2 (continued). *Consider the facts in Figure 8.1 (center), where* `Amy` *teaches* 3 *courses* {`AI`, `ML`, `Opt`}, *teaching* `AI` *twice in the* `Fa17` *and* `Sp18` *terms.* `Ben`, *is a* `TA` *for* `AI`, *then the count for clause* $\mathtt{C}_1$ *given a partial grounding* $\boldsymbol{\theta}_1 = \{\mathtt{p/Amy}, \mathtt{s/Ben}\}$ *is* $\#(C_1 \mid \boldsymbol{\theta}_1) = 1$, *since* `Ben` *is a* `TA` *for only one class. The count for* C_1 *given a partial grounding* $\boldsymbol{\theta}_2 = \{\mathtt{P/Amy}, \mathtt{T/Fa17}\}$ *is* $\#(C_1 \mid \boldsymbol{\theta}_2) = 1$.

Our approach has 3 steps – (i) convert the ground assertions ($\mathcal{F}$) to a hypergraph $\mathcal{G}$, (ii) convert a partially-grounded clause to a partially-grounded structural motif ($\mathcal{M}$), and (iii) count the number of subgraphs matching $\mathcal{M}$ in $\mathcal{G}$.

Definition 2 (Partially Grounded Structural Motif). *A motif $\mathcal{M}$ is a Partially Grounded Structural Motif (PGSM) if it is a hypergraph representation of a clause* C*, where some of the nodes are parameterized, while others are instantiated to their respective values. That is, a PGSM is a structural motif arising from a partial grounding.*

8.1.1 Conversion to Hypergraphs

Given a clause C, we construct a hypergraph motif $\mathcal{M}$ as follows. Each variable in every predicate of C is added as a vertex to $\mathcal{V}_{\mathcal{M}}$, the vertex set of $\mathcal{M}$. Next, all the arguments of a predicate are connected by a hyperedge, which is added to $\mathcal{E}_{\mathcal{M}}$, the edge set of $\mathcal{M}$. Directed edges connect variables appearing in binary predicates in the order in which they appear, while an n-hyperedge connects the n variables appearing in a n-ary predicate. The nodes of $\mathcal{M}$ correspond to the variables in C (which can be partially grounded) and the edges correspond to the predicates that contain the respective variables. Given facts ($\mathcal{F}$), a fully-grounded hypergraph $\mathcal{G}$ is similarly constructed. Each constant in $\mathcal{F}$ is added as a vertex to $\mathcal{V}_{\mathcal{G}}$, vertex set of $\mathcal{G}$. Then, all constants appearing in a fact in $\mathcal{F}$ are connected by a hyperedge, and added to $\mathcal{E}_{\mathcal{G}}$, the edge set of $\mathcal{G}$. The construction of $\mathcal{G}$ essentially amounts to parsing assertions (in predicate logic), indexing and insertion into a hypergraph database (Iordanov et al., 2010).

Example 3 (continued). *Figs. 8.1 and 8.2 (left) show motifs $\mathcal{M}_1$ and $\mathcal{M}_2$ for* $\mathtt{C}_1$ *and* $\mathtt{C}_2$ *respectively.* Amy *advises three students:* Deb*, who is a teaching assistant,* Fei*, who is a research assistant and* Ben *who is both. Thus, $\mathcal{G}_1$, the ground graph for the given assertions (Fig. 8.1, middle & right) Similarly, $\mathcal{G}_2$ (Fig. 8.2) based on the given assertions. Note that,* Ben *is both a teaching and a research assistant and appears in both $\mathcal{G}_1$ and $\mathcal{G}_2$. Also, a student may have more than one advisors*

(Ex: `Fei` *– 2 advisors:* `Amy` *and* `Hal`*). This highlights various complex interactions among entities and attributes; counting these is critical in inference and learning.*

8.1.2 Approximate Counting via Partially Grounded Structural Motif(s)

We consider the case of counts of conjunctive clauses. Partial grounding in a clause $\mathtt{C}$ has 2 scenarios based on number of substitutions. Let number of variables in $\mathtt{C}$ be $\ell_{\mathtt{C}}$.

(**Case 1**) When $|\boldsymbol{\theta}| = \ell_{\mathtt{C}}$, that is, when the clause is fully grounded, $\#(\mathtt{C} \mid \boldsymbol{\theta}) = 1$ if $\mathcal{M} \in \mathcal{G}$ else 0. Thus counting, here, is equivalent to checking that the grounded motif $\mathcal{M}$ is a subgraph of $\mathcal{G}$.

(**Case 2**) When $0 \leq |\boldsymbol{\theta}| < \ell_{\mathtt{C}}$, that is, when the clause is either fully lifted ($\ell_{\mathtt{C}} = 0$) or is partially grounded ($|\boldsymbol{\theta}| < \ell_{\mathtt{C}}$), counting is considerably harder. This is the case we address in the rest of this chapter.

Now, for clause $\mathtt{C}$ with $\ell_{\mathtt{C}}$ variables, we assume that $m_{\mathtt{C}}$ of these variables are *not grounded*, and $\ell_{\mathtt{C}} - m_{\mathtt{C}}$ variables are grounded. Then, the task is to count the number of groundings of $m_{\mathtt{C}}$, termed as *query* variables, given assignments ($\boldsymbol{\theta}$) to the $\ell_{\mathtt{C}} - m_{\mathtt{C}}$ ground variables. *Ex: For clause* $\mathtt{C}_1$*,* $\ell_{\mathtt{C}_1} = 4$ *and given a partial grounding* $\boldsymbol{\theta} = \{\, \mathtt{p}/\mathtt{Amy},\, \mathtt{s}/\mathtt{Ben} \,\}$*, we have* $m_{\mathtt{C}_1} = 2$ *lifted variables (courses* `c` *and terms* `t`*) which we want to count over.*

Without loss of generality, let the first $m_{\mathtt{C}}$ variables in $\mathtt{C}$ be the the *ungrounded* or *query* variables; that is, $v_i,\ i = 1, \ldots, m_{\mathtt{C}}$. Given a motif $\mathcal{M}$ constructed from $\mathtt{C}$, the maximum number of possible subgraphs that match $\mathcal{M}$ in $\mathcal{G}$ is $\prod_{i=1}^{m_{\mathtt{C}}} n_i$, where n_i is the number of possible groundings for *query* variable v_i. Let $P(e \mid \mathcal{G})$ denote the probability of a hyperedge being present in $\mathcal{G}$, then the count of the number of matches of $\mathcal{M}$ in $\mathcal{G}$ is also the clause count, $\#(\mathtt{C} \mid \boldsymbol{\theta})$:

$$P(\mathcal{M}) \prod_{v \,\in\, \mathcal{V}_{\mathcal{M}}} n_v \;=\; \left[\prod_{e \,\in\, \mathcal{E}_{\mathcal{M}}} P(e \mid \mathcal{G}) \right] \cdot \left[\prod_{v \,\in\, \mathcal{V}_{\mathcal{M}}} n_v \right] \tag{1}$$

Intuitively, $P(e \mid \mathcal{G})$ is the *fractional predicate count* that is the ratio of the predicate count given the partial substitutions, to number of groundings of non-substituted variables.

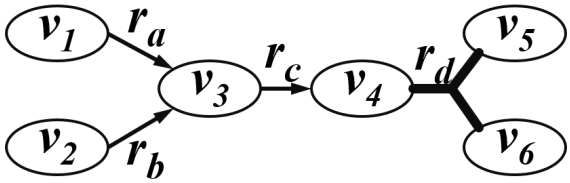


Figure 8.3: The motif $\mathcal{M} \equiv r_a(v_1, v_3) \wedge r_b(v_2, v_3) \wedge r_c(v_3, v_4) \wedge r_d(v_4, v_5, v_6)$. Note that r_d is a hyperedge for the ternary relation $\mathtt{r_d}(\mathtt{v_4}, \mathtt{v_5}, \mathtt{v_6})$.

Example 4. *The probability* $P(\mathtt{Teaches}(\mathtt{Amy}, \mathtt{c}, \mathtt{t}) \mid \mathcal{G})$ *can be computed as the number of courses that* `Amy` *has taught across all terms divided by the cross-product of the free variables (*`c` *and* `t`*) (total number of possible courses times the number of possible terms). Using the partial substitution* $\boldsymbol{\theta} = \{\, \mathtt{p}/\mathtt{Amy} \,\}$ *we have,* $P(\mathtt{Teaches}(\mathtt{Amy}, \mathtt{c}, \mathtt{t}) \mid \mathcal{G}) = \frac{\#(\mathtt{Teaches}(\mathtt{p},\mathtt{c},\mathtt{t})|\boldsymbol{\theta})}{\#(\mathtt{c})\cdot\#(\mathtt{t})}$. *In Fig. 8.1, this is* $4/(3 \cdot 2) = 2/3$.

Expression (1) presents the expected count. If $P(\mathcal{M})$ is computed exactly, we retrieve the true counts. Since that is intractable we find an approximation.

To understand the intuition behind approximating P, consider the motif in Figure 8.3. The maximum number of times this motif can be present in the ground graph is $\prod_{v \in \mathcal{V}_\mathcal{M}} n_v$. The presence of each hyperedge $e \in \mathcal{G}$ is a Boolean concept (i.e., present or absent in $\mathcal{G}$). Without loss of generality, the joint distribution $P(r_a, r_b, r_c, r_d)$ for Fig 8.3 is,

$$\prod_{e \in \mathcal{M}} P(e \mid \mathcal{G}) = P(r_a) \cdot P(r_b) \cdot P(r_c \mid r_a, r_b) \cdot P(r_d \mid r_c).$$

Thus, we now view identifying a motif in a graph as a *search in a directed model with Boolean variables*, where finding an edge depends on the previous edge being found. Note that when any of the edges is absent, the motif will not be present in the grounded graph, and this joint probability is automatically driven to 0. The above expression resembles estimating local models, which is commonly done in standard graphical model estimation. In our case, we further *approximate each of these conditional distributions using summary statistics*.

Reverting to (1), the first term is the product of the individual edge distributions conditioned on the incoming edge, and the second term is the cross-product of the total number of groundings of

the query (ungrounded) variables in the motif. The second term is computed a single pre-processing step, followed by caching. For the first term summary statistics are employed for approximation.

8.1.3 Approximation of clause probability (P)

Summary Computation

In order to approximate the joint distribution P, we employ *graph summaries* similar to the ones used in relational database query engines for cardinality estimation (Schiefer et al., 1998; Neumann and Moerkotte, 2011; Seputis, 2000). Note that these summaries must be selected at the appropriate level of granularity. For instance, finely-grained summaries will correspond to searching the entire database, while highly-aggregated summaries, on the other hand, such as average in-degree and out-degree can lead to poor approximations. Thus, we compute three summaries: (1) node and edge frequencies, (2) node in- and out-degrees, and (3) dependency summaries from hypergraph:

TYPESUM: The **type summary** captures frequencies of node and edge types.

DEGREESUM: **Degree summaries** are the frequencies of incoming and outgoing edges of every node grouped by edge labels.

DEPENDENCYSUM: Most broadly, we seek to summarize all the **possible dependencies** for a relation $\mathcal{R}_j$: $P(\mathcal{R}_j \mid \mathcal{R} \setminus \mathcal{R}_j)$, that is, the dependency of a R_j on all other relations $\mathcal{R} \setminus \mathcal{R}_j$, which is computationally expensive. Given z relation types in $\mathcal{R}$, we instead construct a $z \times z$ *pairwise dependency matrix*, Δ, whose $(i,\ j)$-th element is $\delta_{ij} = P(\mathcal{R}_i \mid \mathcal{R}_j), \forall\, i,\ j = 1, \ldots, z$. For a pair of relations $\mathcal{R}_i$ and $\mathcal{R}_j$, $P(\mathcal{R}_i \mid \mathcal{R}_j)$ is estimated by sampling paths of length 2 from $\mathcal{R}_j \rightarrow \mathcal{R}_k$.

8.1.4 Experiments

We used three standard SRL data sets: *UW-CSE, Citeseer and WebKB*, a biomedical data set *Carcinogenesis* (Srinivasan et al., 1997), and an NLP/Information-Extraction(IE) data set *NELL-Sports* for evaluation.

Table 8.1: Results: Performance vs. Efficiency (running time for **L**earning and **I**nference in seconds). ** indicates n-ary predicates.

		Performance				Efficiency	
Data Sets	**Methods**	**AUC–ROC**	**AUC–PR**	**CLL**	**F1**	**L-Time [s]**	**I-Time [s]**
UWCSE**	MACH	0.981	0.337	-0.133	0.217	**13.2**	5.1
	FACT	0.500	0.0068	-0.061	NaN	7.48	2.8
	MLN-Boost	0.998	0.361	-0.134	0.227	27.5	9.4
Citeseer**	MACH	0.998	0.989	-0.173	0.973	**10837**	12.52
	FACT	0.97	0.92	-0.256	0.934	11042	12.45
	MLN-Boost	0.999	0.998	-0.059	0.977	42499	27.42
Carcinogenesis**	MACH	0.525	0.568	-0.811	0.328	108.48	1.8
	FACT	0.500	0.550	-0.704	NaN	102.5	1.5
	MLN-Boost	0.587	0.572	-0.902	0.489	153.84	2.37
WebKB	MACH	1.0	1.0	-0.049	1.0	**5.8**	0.757
	FACT	1.0	1.0	-0.076	1.0	5.97	0.797
	MLN-Boost	1.0	1.0	-0.075	1.0	8.13	0.896
NELL-Sports	MACH	0.78	0.65	-1.43	0.65	253.92	1.03
	FACT	0.76	0.64	-1.38	0.65	238.07	2.01
	MLN-Boost	0.78	0.66	-0.55	0.68	396.24	2.20

Table 8.1 summarizes the performance and efficiency results of MACH against the baselines for structure and parameter learning of MLNs. It is clearly evident that there is no real deterioration in predictive performance due to count approximation in MACH (across all data sets) compared to MLN-Boost. However, MACH substantially beats the baselines on efficiency (learning time), especially on larger data sets such as *Citeseer*, where it is about 4 times faster. Even on smaller data sets such as *UW-CSE* and *WebKB*, the difference in learning times is sizable. Several other data sets can be used to test the effectiveness of this method such as the drug-drug interactions (chapter 4) and co-author network (chapter 6).

8.2 Knowledge-augmented Column Networks

The re-emergence of Deep Learning (Goodfellow et al., 2016b) has found significant and successful applications in difficult real-world domains such as image (Krizhevsky et al., 2012), audio (Lee et al., 2009) and video processing. However, the combinatorial complexity of reasoning in relational domains over a large number of relations and objects has remained a significant bottleneck to

overcome. Recent work in relational deep learning has sought to address this particular issue(Kazemi and Poole, 2018b; Šourek et al., 2015; Kaur et al., 2017b; França et al., 2014). We consider **Column Networks** (CLNs) (Pham et al., 2017) that are composed of several (feedforward) mini-columns each of which represents an entity in the domain. CLNs are attractive for a few reasons (1) hidden layers of a CLN share parameters, which means that making the network deeper does not introduce more parameters, (2) as the depth increases, the CLN can begin to model feature interactions of considerable complexity, which is especially attractive for relational learning, and (3) learning and inference are linear in the size of the network and the number of relations, which makes CLNs highly efficient. However, like other deep learning approaches, CLNs rely on vast amounts of data and incorporate little to no knowledge about the problem domain.

It is well known that biasing learners is necessary in order to allow them to inductively leap from training instances to true generalization over new instances (Mitchell, 1980). While deep learning does incorporate one such bias in the form of domain knowledge (for example, through parameter tying or convolution, which exploits neighborhood information), we are motivated to develop systems that can incorporate richer and more general forms of domain knowledge. One way in which a human can guide learning is by providing *rules over training examples and features*. The earliest such approaches combined explanation-based learning (EBL-NN, (Shavlik and Towell, 1989)) or symbolic domain rules with ANNs (KBANN, (Towell and Shavlik, 1994)). Another natural way a human could guide learning is by expressing *preferences* and has been studied extensively within the preference-elicitation framework due to Boutilier et al. (2006). We are inspired by this form of advice as they have been successful within the context of inverse reinforcement learning (Kunapuli et al., 2013a) and planning (Das et al., 2018).

These approaches span diverse machine learning formalisms, and they all exhibit the same remarkable behavior: **better generalization with fewer training examples** because they effectively exploit and incorporate domain knowledge as an inductive bias. This is the prevailing motivation for our approach: to develop a framework that **allows a human to guide deep learning** by incorporating rules and constraints that define the domain and its aspects. Incorporation of prior knowledge

into deep learning has begun to receive interest recently (Ding et al., 2018), however, in many such approaches, the guidance is not through a human. Our framework is much more general in that a human provides guidance during learning. Furthermore, the human providing the domain advice is not an AI/ML expert but rather a domain expert who provides rules naturally. We exploit the rich representation power of relational methods to capture, represent and incorporate such rules into relational deep learning models. In our work we use first-order logic as a representation language for human advice and employ it in the context of CLNs.

Given: A sparse multi-relational graph $\mathcal{G}$, attributes x_i of each entity (sparse or noisy) in $\mathcal{G}$, equivalent Column-Network $\mathcal{C}$ and access to a Human-expert

To Do: More effective and efficient collective classification by knowledge augmented training of $\mathcal{C}(\theta)$, where $\theta = \langle \{W^t\}_1^T, \{V_r^t\}_{r\in R; t=1}^{t=T}, \{W_\ell\}_{\ell \in L} \rangle$ is the set of all the network parameters of $\mathcal{C}$.

We develop ***K***nowledge-augmented ***C***o***L***umn ***N***etworks (K-CLN), that incorporates human-knowledge, for more effective and efficient learning from relational data (Figure 8.4 illustrates the overall architecture). While knowledge-based connectionist models are not entirely new, our formulation provides - (1) a principled approach for incorporating advice specified in an intuitive logic-based encoding/language (2) a deep model for collective classification in relational data.

8.2.1 Knowledge Representation

Any model specific encoding of domain knowledge, such as numeric constraints or modified loss functions etc., has limitations, namely (1) counter-intuitive to the humans since they are domain expert (2) the resulting framework is brittle and not generalizable. Consequently, we employ preferences (akin to IF-THEN statements) to capture human knowledge.

Definition 3. *A preference is a modified Horn clause,* $\wedge_{\mathtt{k},\mathtt{x}} \mathtt{Attr}_\mathtt{k}(\mathtt{E}_\mathtt{x}) \wedge \ldots \wedge_{\mathtt{r}\in\mathtt{R},\mathtt{x},\mathtt{y}} \mathtt{r}(\mathtt{E}_\mathtt{x}, \mathtt{E}_\mathtt{y}) \Rightarrow [\mathtt{label}(\mathtt{E}_\mathtt{z}, \ell_1) \uparrow; \mathtt{label}(\mathtt{E}_\mathtt{k}, \ell_2) \downarrow]$ *where* $\ell_1, \ell_2 \in L$ *and the* $\mathtt{E}_\mathtt{x}$ *are variables over entities,* $\mathtt{Attr}_\mathtt{k}(\mathtt{E}_\mathtt{x})$ *are attributes of* E_x *and* $\mathtt{r}$ *is a relation.* $\uparrow$ *and* $\downarrow$ *indicate the preferred non-preferred labels respectively. Quantification is implicitly* $\forall$ *and hence dropped. We denote a set of preference rules as* $\mathfrak{P}$.

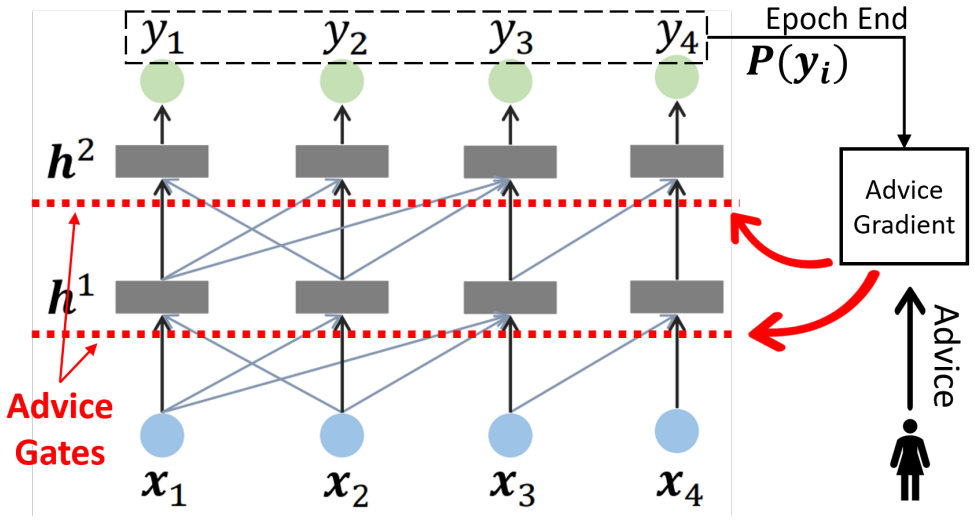


Figure 8.4: Proposed K-CLN architecture. Here x_i denotes the feature of each entity e_i with y_i being the label of each entity. We provide advice in each layer using advice gates and the prediction informs the advice gradient.

Note that we can always, either have just the preferred label in head of the clause and assume all others as non-preferred, or assume the entire expression as a single literal. Intuitively a rule can be interpreted as conditional rule, **IF [conditions hold] THEN label ℓ is preferred**. A preference rule can be partially instantiated as well, *i.e.*, or more of the variables may be substituted with constants.

8.2.2 Knowledge Injection

Given that knowledge is provided as *partially-instantiated* preference rules $\mathfrak{P}$, more than one entity may satisfy a preference rule. Also, more than one preference rules may be applicable for a single entity. The main intuition is that we aim to consider the error of the trained model w.r.t. both the data and the advice. Consequently, in addition to the *"data gradient"* as with original CLNs, there is a *"advice gradient"*. This gradient acts a feedback to augment the learned weight parameters (both column and context weights) towards the direction of the *advice gradient*. It must be mentioned that not all parameters will be augmented. Only the parameters w.r.t. the entities and relations (contexts) that satisfy $\mathfrak{P}$ should be affected. Let $\mathcal{P}$ be the set of entities and relations that satisfy the set of

preference rules $\mathfrak{P}$. The hidden nodes can now be expressed as,

$$h_i^t = g\left(b^t + W^t h_i^{t-1}\Gamma_i^{(W)} + \frac{1}{z}\sum_{r\in R} V_r^t c_{ir}^t \Gamma_{ir}^{(c)}\right)$$

$$\text{s.t. } \Gamma_i, \Gamma_{i,r} = \begin{cases} 1 & \text{if } i, r \notin \mathcal{P} \\ \mathcal{F}(\alpha\nabla_i^{\mathfrak{P}}) & \text{if } i, r \in \mathcal{P} \end{cases} \tag{2}$$

where $i \in \mathcal{P}$ and $\Gamma_i^{(W)}$ and $\Gamma_{ir}^{(c)}$ are advice-based soft gates with respect to a hidden node and its context respectively. $\mathcal{F}()$ is some gating function, $\nabla_i^{\mathfrak{P}}$ is the *"advice gradient"* and α is the trade-off parameter explained later. The key aspect of soft gates is that they attempt to enhance or decrease the contribution of particular edges in the column network aligned with the direction of the *"advice gradient"*. We choose the gating function $\mathcal{F}()$ as an exponential $[\mathcal{F}(\alpha\nabla_i^{\mathfrak{P}}) = \exp(\alpha\nabla_i^{\mathfrak{P}})]$. The intuition is that soft gates are natural, as they are multiplicative and a positive gradient will result in $\exp(\alpha\nabla_i^{\mathfrak{P}}) > 1$ increasing the value/contribution of the respective term, while a negative gradient results in $\exp(\alpha\nabla_i^{\mathfrak{P}}) < 1$ pushing them down. We now present the *"advice gradient"* (the gradient with respect to preferred labels).

Proposition 1. *Under the assumption that the loss function with respect to advice / preferred labels is a log-likelihood, of the form* $\mathcal{L}^{\mathfrak{P}} = \log P(y_i^{(\mathfrak{P})}|h_i^T)$*, then the advice gradient is,* $\nabla_i^{\mathfrak{P}} = I(y_i^{(\mathfrak{P})}) - P(y_i)$*, where* $y_i^{(\mathfrak{P})}$ *is the preferred label of entity and* $i \in \mathcal{P}$ *and* I *is an indicator function over the preferred label. For binary classification, the indicator is inconsequential but for multi-class scenarios it is essential (*$I = 1$ *for preferred label* ℓ *and* $I = 0$ *for* $L \setminus \ell$*).*

Since an entity can satisfy multiple advice rules we take the *most* preferred label, *i.e.*, we take the label $y_i^{(\mathcal{P})} = \ell$ to the preferred label if ℓ is given by most of the advice rules that e_j satisfies. In case of conflicting advice (i.e. different labels are equally advised), we simply set the advice label to be the label given by the data, $y_i^{(\mathfrak{P})} = y_i$.

As illustrated in the K-CLN architecture (Figure 8.4), at the end of every epoch of training the *advice gradients* are computed and soft gates are used to augment the value of the hidden units as shown in Equation 2.

Proposition 2. *Given that the loss function $\mathcal{H}_i$ of original CLN is cross-entropy (binary or sparse-categorical for the binary and multi-class prediction cases respectively) and the objective w.r.t. advice is log-likelihood, the functional gradient of the modified objective is,*

$$\begin{aligned}\nabla(\mathcal{H}_i') &= (1-\alpha)\left(y_i I - P(y_i|h^T)\right) + \alpha\left(I_i^{\mathfrak{P}} - P(y_i^{\mathfrak{P}}|h^T)\right) \\ &= (1-\alpha)\nabla_i + \alpha\nabla_i^{\mathfrak{P}} \end{aligned} \tag{3}$$

where $0 \leq \alpha \leq 1$ is the trade-off parameter between the effect of data and effect of advice, I_i and $I_i^{\mathfrak{P}}$ are the indicator functions on the label w.r.t. the data and the advice respectively and ∇_i and $\nabla_i^{\mathfrak{P}}$ are the gradients, similarly, w.r.t. data and advice respectively (Proof in appendix).

Hence, it follows that the data and the advice balances the training of the K-CLN network parameters $\theta^{\mathfrak{P}}$ via the trade-off hyperparameter α. When data is noisy (or sparse with negligible examples for a region of the parameter space) the advice (if correct) induces a bias on the output distribution towards the correct label. Even if the advice is incorrect, the network still tries to learn the correct distribution to some extent from the data (if not noisy). The contribution of the effect of data versus the effect of advice will primarily depend on α. If both data and human advice are sub-optimal, correct label distribution is not learnable.

8.2.3 Experiments

Experiments are conducted on **four relational** domains – *Pubmed Diabetes* (multi-class), *Corporate Messages* (multi-class), *Internet Social Debates* (binary) and *Social Network Disaster Relevance* (binary). Following (Pham et al., 2017), macro-F1 and micro-F1 scores for the multi-class problems, and F1 scores and AUC-PR for the binary ones are reported. Macro-F1 computes the F1 score independently for each class and takes the average whereas a micro-F1 aggregates the contributions of all classes to compute the average F1 score. Here, we show only the Micro-F1 and the AUC-PR results with all results averaged over 5 runs.

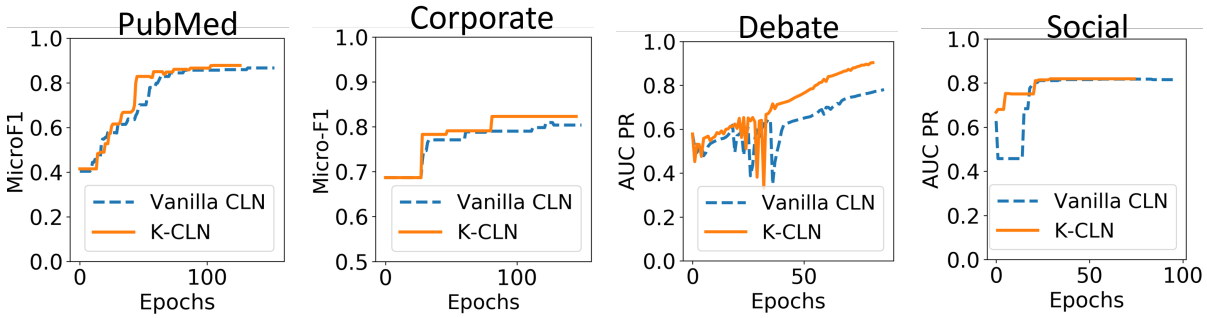


Figure 8.5: Performance w.r.t. **epochs**. *Left to Right* - Pubmed, Corporate Messages, Debates and Social Disaster. *Leftmost 2* show Micro-F1, (multi-class) & *Rightmost 2* show AUC-PR (binary)

We present the aforementioned metrics with **varying sample size** and with **increasing epochs** and compare our model against *Vanilla CLN*. We split the data sets into a training set and a hold-out test set with 60%-40% ratio. For varying epochs we only learn on 40% of our pre-split training set (*i.e.*, 24% of the complete data) to train the model and test on the hold-out test set. Figure 8.5 shows that, although both K-CLN and Vanilla CLN converge to the same predictive performance (Micro-F1 for PubMed & Corporate and AUC-PR for the rest), K-CLN converges **significantly faster** (less epochs). Also. for the *corporate* and the *debate*, K-CLN not only **converges faster but also has a better predictive performance** than Vanilla CLN (Figure 8.5- 2^{nd}, 3^{rd}).

8.3 Drug-Drug Interaction Prediction from Molecular Structure Images

Adverse drug events (ADEs) are "injuries resulting from medical intervention related to a drug" (Nebeker et al., 2004), and are distinct from medication errors (inappropriate prescription, dispensing, usage etc.) as they are caused by drugs at normal dosages. According to the National Center for Health Statistics (NCHS, 2014), 48.9% of Americans took at least one prescription drug in the last 30 days, 23.1% took at least three, and 11.9% took at least five. These numbers rise sharply to 90.6%, 66.8% and 40.7% respectively, among older adults (65 years or older). This means that the potential for ADEs is very high in a variety of health care settings including inpatient, outpatient and long-term care settings. For example, in inpatient settings, ADEs can account for as many

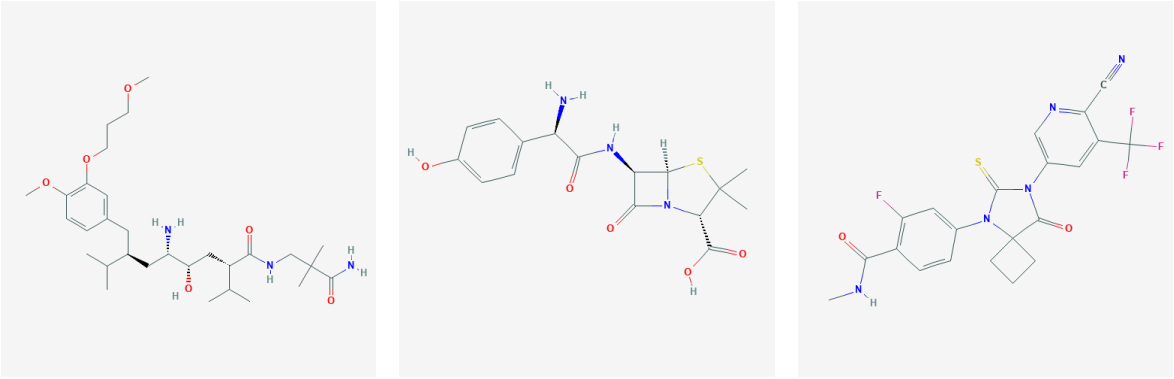
as one-third of hospital-related complications, affect up to 2 million hospital stays annually, and prolong hospital stays by 2–5 days (DHHS, 2010).

The economic impact of these issues is as widespread as the various healthcare settings and can be staggering. Estimates suggest that ADEs contributed to $3.6 billion in excess healthcare costs in the US alone (Aspden et al., 2007). Unsurprisingly, older adults are at the highest risk of being affected by an ADE, and are seven times more likely than younger persons to require hospital admission (Budnitz et al., 2006). In the US, as a large number of older adults are Medicare beneficiaries, this economic impact is borne by an already overburdened Medicare system and ultimately passed on to taxpayers and society at large. Beyond older adults, there are several other patient populations that are also vulnerable to ADEs including children, those with lower socio-economic means, those with limited access to healthcare services, and certain minorities.

Recent research has identified, somewhat surprisingly, that many of these ADEs can be attributed to **very common medications** (Budnitz et al., 2011) and **many of them are preventable** (Gurwitz et al., 2003) or **ameliorable** (Forster et al., 2005). This issue motivates our long-term goal of developing accessible and robust means of identifying ADEs in a disease/drug-agnostic manner and across a variety of healthcare settings. Here, we focus on the problem of **drug-drug interactions** (DDIs), which are a type of ADE. An ADE is characterized as a DDI when multiple medications are co-administered and cause an adverse effect on the patient. DDIs, often caused by inadequate understanding of various drug-drug contraindications, are a major cause of hospital admissions, rehospitalizations, emergency room visits, and even death (Becker et al., 2007).

Identifying DDIs is an important task during drug design and testing, and regulatory agencies such as the U. S. Food and Drug Administration require large controlled clinical trials before approval. Beyond their expense and time-consuming nature, it is impossible to **discover** all possible interactions during such clinical trials. This necessitates the need for computational methods for DDI prediction. A substantial amount of work in DDI focuses on text-mining (Liu and Chen, 2013; Chee et al., 2011) to extract DDIs from large text corpora; however, this type of information extraction

does not discover new interactions, and only serves to extract *in vivo* or *in vitro* discoveries from publications.



(a) Aliskiren (b) Amoxicillin (c) Apalutamide

Figure 8.6: Some example molecular images of different drugs extracted from the PubChem database.

Our goal is to **discover** DDIs in large drug databases by exploiting various properties of the drugs and identifying patters in drug interaction behaviors. Recent approaches consider phenotypic, therapeutic, structural, genomic and reactive properties of drugs (Cheng and Zhao, 2014) or their combinations (Dhami et al., 2018) to characterize drug interactivity. We take a fresh and completely new perspective on DDI prediction through the lens of molecular images, a few examples shown in figure 8.6, via deep learning. Our work is novel in the following significant ways:

- we formulate DDI discovery as a **link prediction problem**;
- we aim to perform DDI discovery directly on **molecular structure images** of the drugs directly, rather than on lossy, string-based representations such as SMILES strings and molecular fingerprints; and
- we utilize deep learning, specifically **Siamese networks** (Chopra et al., 2005) in a contrastive manner to build a **DDI discovery engine** that can be integrated into a drug database seamlessly.

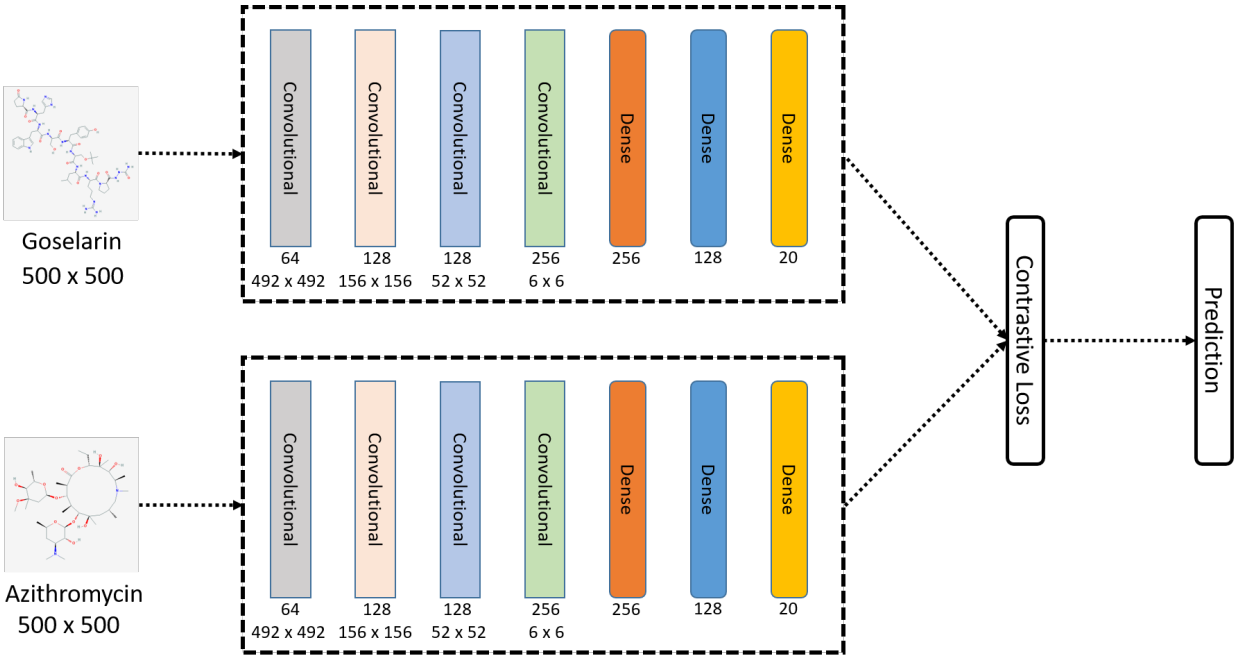


Figure 8.7: An overview of our model for predicting drug-drug interactions

8.3.1 Siamese Convolutional Network for Drug-Drug Interactions

A discriminative approach for learning a similarity metric using a Siamese architecture was introduced in (Chopra et al., 2005) which maps the input (pair of inputs) into a target space such that the distance between the mappings is minimized in the target space for similar pair of examples and maximized in case of dissimilar examples. We adapt the Siamese architecture for the task of link prediction where the link is whether two drugs interact or not. Since the Siamese architecture results in a measure of similarity between the pair of given inputs it can be thresholded in order to obtain a classification. We use contrastive loss (Hadsell et al., 2006), based on a distance metric (Eucledian distance in our case), to learn a parameterized function F to obtain the mapping from the input space to the target space whose minimization can result in pushing the semantically similar examples together. An important property of the loss function is that it calculated on a pair of examples. The loss function is formulated as as follows: Let X_1 and X_2 are a pair of drug images and Y is the label assigned to each of the pairs. The label $Y = 0$ if the pair of drug images do not interact and $Y = 1$ if the pair of drug images interact. Also, let D be the Eucledian distance

between the vector of the image pairs after being processed by the underlying Siamese network and P are the parameters of the function F. The contrastive loss function can then be given as

$$\mathcal{L}(P, X_1, X_2, Y) = \frac{(1-Y)}{2} {D_P}^2 + \frac{Y}{2}\{\max(0, m - D_P)\}^2 \tag{4}$$

where $D_P = \|F_P(X_1) - F_P(X_2)\|_2^2$ is the Eucledian distance between the obtained outputs after the input pairs are processed by the sub-networks. Also *m* is a margin such that $m \geq 0$ that signifies that dissimilar pairs beyond this margin will not contribute to the loss.

Figure 8.7 shows our complete architecture. It consists of two identical sub-networks i.e. networks having same configuration with the same parameters and weights. Each sub-network takes a gray-scale image of size $500 \times 500 \times 1$ as input (we initially have color images that we convert to gray-scale before feeding to sub-networks as input) and consists of 4 convolutional layers with number of filters as 64, 128, 128 and 256 respectively. The kernel size for each convolutional layer is (9×9) and the activation function is *relu*. The *relu* is a non-linear activation function is given as $f(x) = max(0, x)$. Each convolutional layer is followed by a max-pooling layer with pool size of (3×3) and a batch normalization layer. After the convolutional layers, the sub-network has 3 fully connected layers with 256, 128 and 20 neurons respectively. Thus after an image pair is processed by the Siamese sub-networks two vectors of dimension 20×1 are obtained. Contrastive loss is then applied to the obtained pair of vectors to obtain a distance between the input pair which can then be thresholded to obtain a prediction.

Initially, we keep the threshold at 0.5 and then use precision recall curve to identify the best threshold $= 0.65$. Note that the convolutions in the convolutional sub-network provide translational in-variance property but rotational in-variance is also important in our problem domain. This is because isomers (one of the chiral forms) of drugs are expected to react differently when interacting with a certain drug (Nguyen et al., 2006; Chhabra et al., 2013). For example, Fenfluramine and Dexfenfluramine are isomers of each other and where Fenfluramine interacts with Acebutolol but Dexfenfluramine does not (Figure 8.8). Another example is that the L-isomer of methorphan,

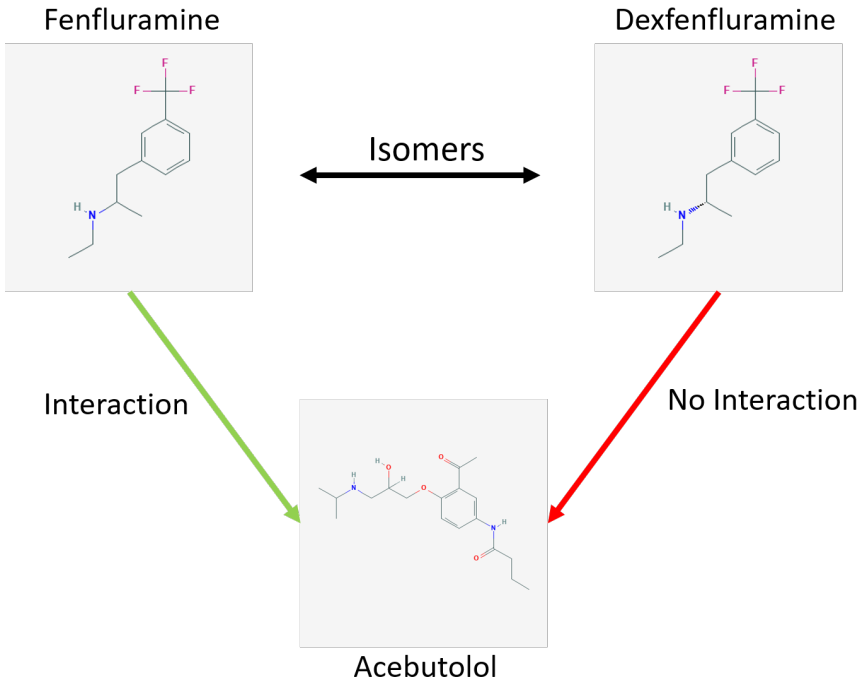


Figure 8.8: An example of how two isomers interact differently with a single drug.

Levomethorphan, is an opioid analgesic, while the D-isomer, Dextromethorphan, is a dissociative cough suppressant[1]. We discuss our results next.

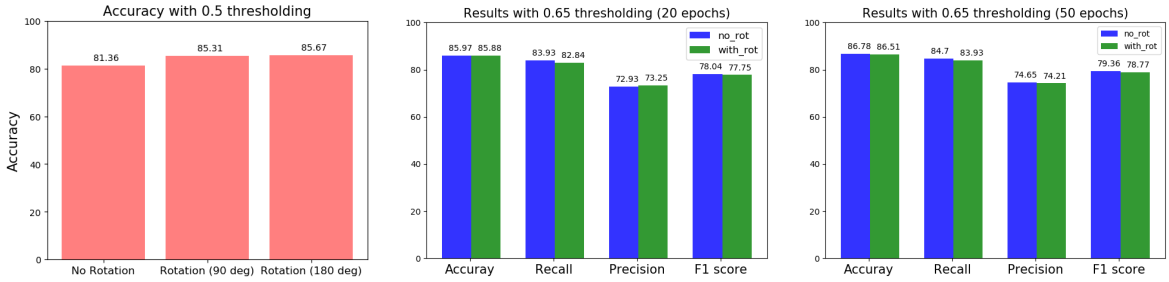
8.3.2 Experiments

Figure 8.9 shows the results for using Siamese network for predicting drug-drug interactions using drug molecular structure images. Our data set consists of images of 373 drugs downloaded from the PubChem database [2] and generate a data set of 19936 drug pairs that interact with each other (Y = 1) and 47424 drug pairs that do not interact with each other (Y = 0). We optimize our Siamese network using the Adam optimization algorithm (Kingma and Ba, 2014) with a learning rate of 5×10^{-5}. The best learning rate was obtained using line search. We also tried the Root Mean

[1]https://en.wikipedia.org/wiki/Enantiopure_drug

[2]https://pubchem.ncbi.nlm.nih.gov/

Square Propagation (RMSprop) optimizer (Hinton, 2012) that is an adaptive learning rate method but the results were not encouraging and thus we make use of the Adam optimizer. As mentioned before, we also keep an initial threshold of 0.5 to obtain the predictions after obtaining a distance between pair of drug images using the Siamese convolutional network. We divide the data set into 44457 training and 22903 testing examples. To introduce rotational in-variance we rotate the drug images by $90°$ and $180°$ and use them to train and test the Siamese network. After including the rotated images the data set size increases to 88914 training and 45806 testing examples. Figure 8.9(a) shows the accuracy after thresholding the obtained distance at 0.5 and training the network for 20 epochs. The results show that at 0.5 threshold, if the data set includes rotated drug images the network shows better results than if the data set does not include the rotated drug images. We also tried Earth movers or Wassertein distance [3] as the distance metric (Yu and Herman, 2005) for the Siamese architecture but it did not make much difference in the final results.



(a) Accuracy of the network with threshold = 0.5 and epochs = 20

(b) Results with threshold = 0.65 and epochs = 20

(c) Results with threshold = 0.65 and epochs = 50

Figure 8.9: Results using Siamese network for predicting drug-drug interactions.

We then pick a best threshold using a precision-recall curve such that the best intuitive trade-off between Precision and Recall can be represented and obtain a value of 0.65 as the best threshold. Figures 8.9(b) and 8.9(c) present different metrics after training the network for 20 and 50 epochs respectively. The results show that even when the number of epochs are increased the results do

[3] https://en.wikipedia.org/wiki/Wasserstein_metric

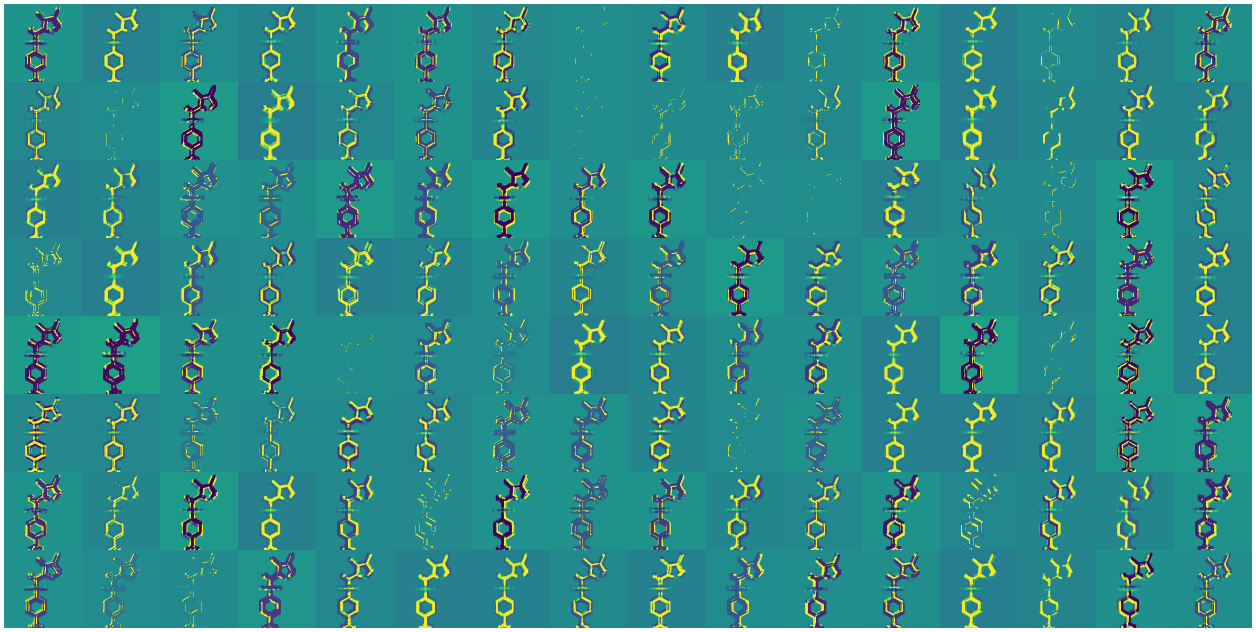


Figure 8.10: An example of abstract features learned by a convolutional layer for Sulfisoxazole.

not vary much. We present the results with the data set including the images rotated by 180° since it gives the best accuracy result as seen in figure 8.9(a). Another important thing to note here is that in our problem formulation recall is the most important factor that should be considered. The simple reason is that we do not want to miss any interaction i.e. a false negative results in much more serious consequences (fatalities in patients) than false positives (monetary losses such as new clinical trials) (Dhami et al., 2018). Our network achieves a recall of $\approx 85\%$ thereby showing the effectiveness of using a Siamese architecture for predicting DDIs. Along with the high recall we also obtain a high precision of $\approx 75\%$ thereby showing that the Siamese architecture can also extract relevant DDIs.

An important factor also to consider here is the effect of the threshold on the obtained results. With the threshold of 0.5 the effect of rotational in-variance becomes evident as the network when trained with rotated images performs better than when trained without the rotated drug images. After we find the best threshold the effect of rotational in-variance becomes negligible.

Examples of features learned by the convolutional layer for 2 drugs, namely, Sulfisoxazole and Venlafaxine are shown in figure 8.10 and 8.11 respectively.

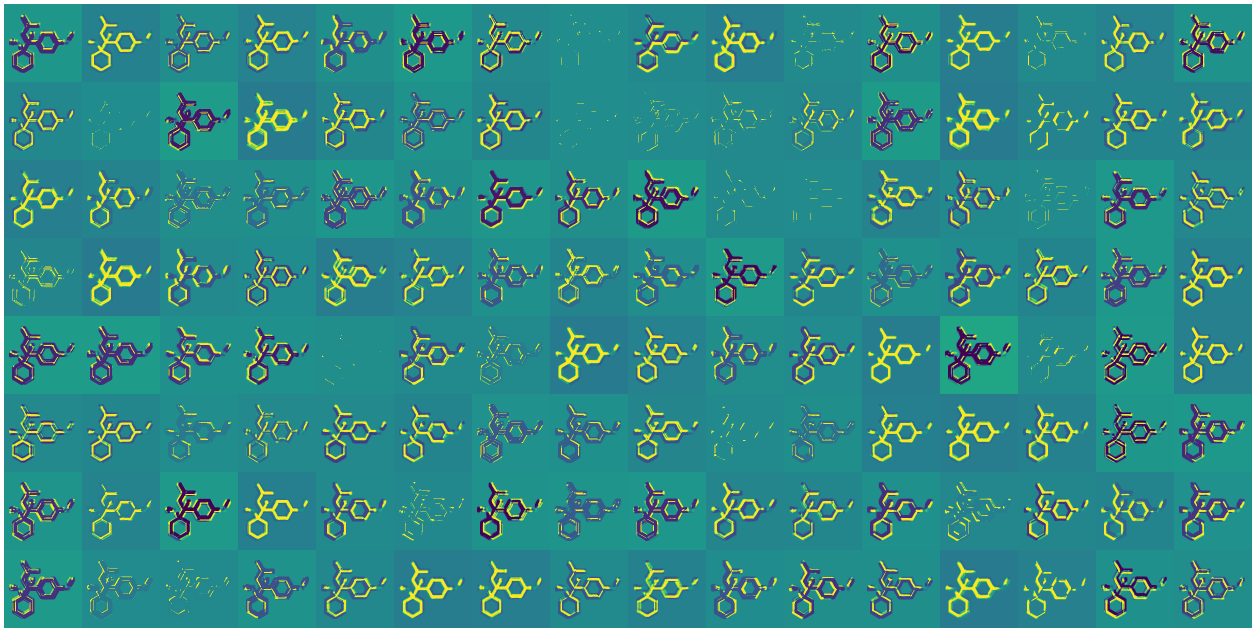


Figure 8.11: An example of abstract features learned by a convolutional layer for Venlafaxine.

8.4 Conclusion

We present several methods that directly relate to and/or extend the different algorithms presented in this dissertation. We present methods to incorporate human knowledge in deep models, extending and developing more efficient statistical relational learning counting methods using hypergraphs and presenting the first method using drug structure image data to predict drug-drug interactions. We show that the methods presented int his chapter are closely connected to the work explained in previous chapters and can result in improving the proposed models in this thesis and learning of new and more efficient machine learning models for healthcare.

CHAPTER 9

CONCLUSION

Developing models which are effective while dealing with thye inherent noisy structure and multi-modality of data, especially in a sensitive domain such as healthcare, is a challenging and an important task to be solved. We have approached this problem in three different yet interconnected ways. In chapter 3 we deal with data from a clinical study and as expected has lot of noise in the data including several uninformative features. To overcome these problems, a human expert advice is utilised which constraints our search space to the most informative feature set. In chapter 4 we consider data that is heterogeneous in nature. Specifically, we deal with data in different forms of first order factbase and strings representing underlying structure of the examples in the data. Our solution consisted of converting all the different modalities into the corresponding similarity matrices and then combining these matrices into a single kernel for the sake of prediction. One key advantage of our approach is that we are able to not just predict but also discover new knowledge. Since most of the real world data is relational in nature, methods that deal with such relational data directly, instead of first constructing a lossy propositional representation from the relational data, are highly desirable. To achieve this, chapters 5 and 6 present different methods combined by the underlying principle of taking advantage of local neighborhood information. Although neighborhood methods have been widely adapted in deep learning (convolution operation captures neighborhood information) and in classical machine learning (using Laplacians), generalizing any of these to a graphical data is still in its nascent stage. To overcome this we propose a novel method of learning structure for the underlying relational data and using the Gaifman locality theorem to incorporate relational neighborhood information in the underlying machine learning model. We also propose to extend graph convolutional networks to a relational realm by proposing the ROCGCN method. Since capturing local neighborhood information can result in construction of richer abstractions and capturing richer latent representations, we envision a wider adaptation of such methods within graph machine learning in the coming future.

The final part of our thesis deals with generate synthetic medical data while faithfully capturing relationships between different features with the help of a human expert. This problem is very crucial considering the high cost and several restrictions that accompany with acquiring medical data for research purposes. We propose the first human-allied generative adversarial network architecture that uses human expert to get advice on feature correlations and generates data capturing these correlations.

Although the above methods make a significant amount of progress in dealing with the various challenges posed by noisy, heterogeneous and multi-relational healthcare data, several challenges still remain and are discussed next as being the immediate future directions along with providing a more long term view and requirements of these challenges.

9.1 Future Directions

9.1.1 Multi-View Learning

In our work we have learned multi-relational models and have combined them with deep learning models to seamlessly take advantage of the rich underlying structure of the relational data as well as learn more richer abstract features using a graph convolutional network. Multi-view learning is area of machine learning from data concerned with learning with data represented by multiple distinct feature sets and each such feature set forms a *view* of the data. The *distinct* feature set signifies that the feature sets are conditionally independent given the class. Another required property is that each view is sufficient for predicting the class of an instance. We propose to extend our ROCGCN model to a multi-view learning setting by using different modalities of data to construct the feature and the distance matrix.

Example: *As we have already shown in our work, we can represent a drug using a graphical structure such as a first order factbase or by its molecular image. In the classical multi-view setting, we learn separate classifiers for both views. Suppose we use a convolutional neural network for the*

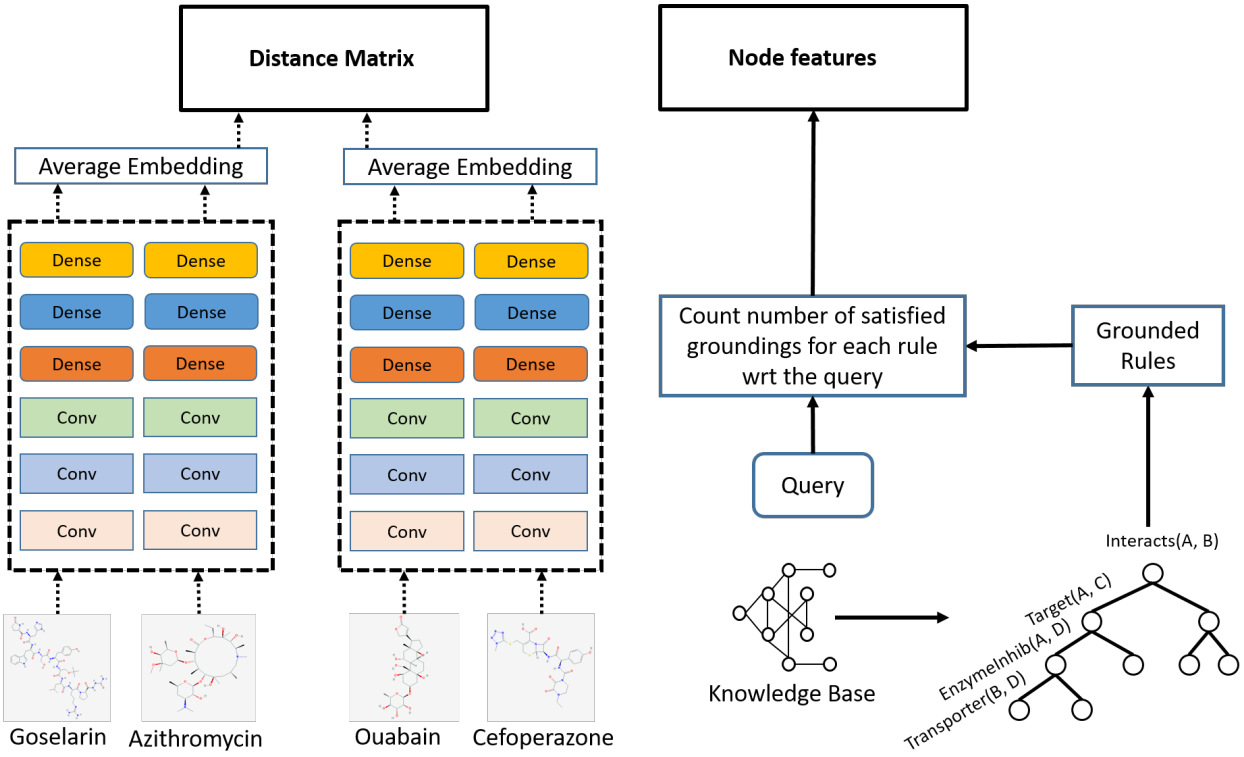


Figure 9.1: Our proposed approach for multi-view learning using graph convolutional networks

image and any statistical relational learning method for the factbase representation. In both the case the network, in each view, will learn from ***a single example at a time*** *with other examples having no direct effect on the current example. This leads to some vital information about the relationship of the current example with other examples, in each view, that can help learn a better model not being used.*

Continuing our example of drug-drug interactions, as can be seen in Figure 9.1, data can be represented by either in the form of graphs (as used in the Gaifman model discussion) or in the form of images (as used in the Siamese architecture for drug-drug interaction discussion). We propose to use a twin Siamese network architecture to create the distance/adjacency matrix and use the same method as discussed in 6 to obtain the feature matrix for inputting to the graph convolutional network. We differ from the classical multi-view learning setting since we use examples from the same view together whereas in the classical multi-view setting, in both the co-training and

co-regularization type techniques, a single data point from each view is considered at a time. Thus, although there is an information flow between the examples from different views (because of the matrix factorization style optimization function), the examples from the same view are not used together. We present a rough sketch of our approach next.

1. Use the method described in chapter 6 to obtain the node feature matrix. To use the drug-drug interactions example, every row in the feature matrix represents the feature set for the $Interacts$ relation between a pair of drugs.

2. To obtain the distance matrix between a pair of $Interacts$ relation we make use of a twin Siamese neural network. Every Siamese neural network provides a pair of embedding for the pair of input drugs and to obtain the embedding capturing the pair of drugs, the obtained embeddings are averaged since the average embeddings have shown promising results (Coates and Bollegala, 2018).

3. The features and distance matrix can then be used in the graph convolutional network architecture.

Multi-view learning has also been used in various sub-fields of machine learning such as transfer learning (Yang and Gao, 2013; Tan et al., 2013), dimensionality reduction (Ding and Fu, 2014; Guo, 2013), clustering (Li et al., 2015, 2014; Xia et al., 2014) and multi-task learning (Zhang et al., 2015; Jin et al., 2013). One of the common theme in majority of these methods is the use of non-negative matrix factorization type technique for ensuring the information flow between different views. Since heath care data has multiple views of data applying multi-view learning on the same seems to be a natural fit.

9.1.2 Extension of Gaifman Models

Joint Learning of Gaifman Models In our previous work in chapter 5 we presented a structure learning method for learning of discriminative Gaifman models, a class of relational models that

take advantage of the local neighborhood information present in the underlying graph data. A major limitation of our previous work is that the prediction is done only a single relation i.e. we have only a single query variable and thus perform a single conditional probability operation. As a natural extension we propose to perform joint learning of Gaifman models which can handle multiple query variables. In theory joint learning can be achieved by multiplying the set of conditionals using the chain rule of probability and has been used extensively in joint learning of several probabilistic graphical models (Koller and Friedman, 2009). This simple method can be extended to obtain the joint learning of Gaifman models as well. Given the set of all relations $\mathcal{R}$ in the underlying relational data set and a set of query relations $\mathcal{Q} \in \mathcal{R}$, the joint probability of the query relations $\mathcal{R}_1, \mathcal{R}_2 .... \mathcal{R}_n \in \mathcal{Q}$ can be written as:

$$\begin{aligned} P(\mathcal{R}_1, \mathcal{R}_2 .... \mathcal{R}_n) &\approx P(\mathcal{R}_1 | \mathcal{R}/\mathcal{R}_1) \times P(\mathcal{R}_2 | \mathcal{R}/\mathcal{R}_2) \times \cdots \times P(\mathcal{R}_n | \mathcal{R}/\mathcal{R}_n) \\ &\approx \prod_{i=1}^{n} P(\mathcal{R}_i / \mathcal{R}/\mathcal{R}_i) \end{aligned} \tag{1}$$

where all the conditional probabilities can be interpreted as the pseudo likelihoods and thus provide a principled approximation to the joint probability.

Hypergraph Embeddings using Gaifman Models A natural extension of Gaifman models can be in obtaining hypergraph embeddings. Although a lot of relational embedding techniques exist (Xiao et al., 2016; Bordes et al., 2013) these are limited to binary graphs i.e. graphs with edges connecting only 2 vertices and cannot handle the family of hypergraphs.

Some of the recent work such as m-TransH (Wen et al., 2016) extends the binary relational embedding model TransH (Wang et al., 2014) to the hypergraph space. Developing embedding models that can directly handle hypergraphs is one way to approach the problem. Another way to tackle this problem, one with more theoretical underpinnigs, is to convert the hypergraph into a binary graph and then apply one of the various binary relational embedding techniques to perform the required task, such as node classification or link prediction.

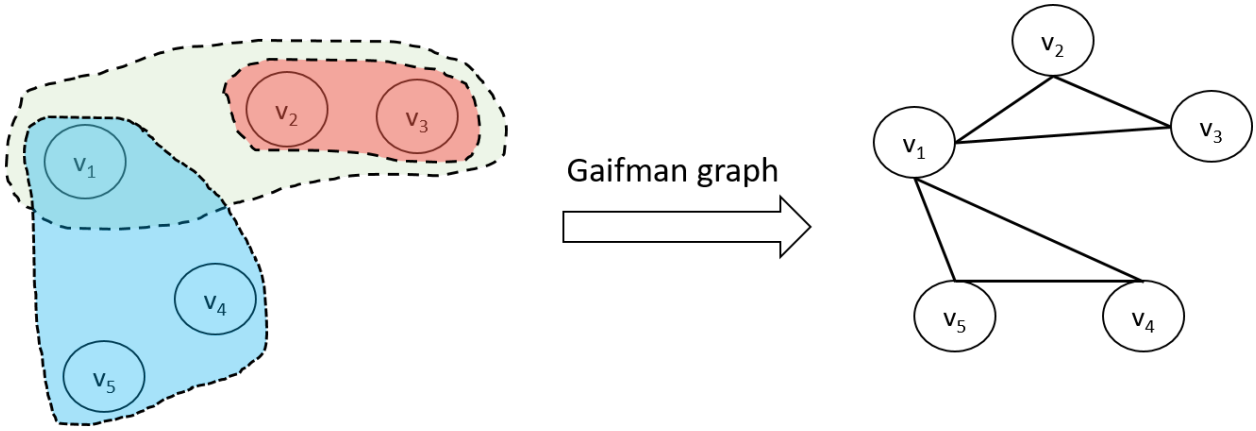


Figure 9.2: Conversion of a hypergraph to its corresponding Gaifman graph.

Although these relational embeddings can be used after converting the hypergraph to a binary graph, this conversion can lead to loss of information which can adversely effect the quality of the embeddings generated. We propose to use the Gaifman locality theorem to capture the sensitive neighborhood information and using the Gaifman graph of the hypergraph as a conversion to a binary graph as shown in figure 9.2.

We can use any of the techniques defined in chapter 5 to learn the structure of the Gaifman models since all the methods are capable of handling n-ary predicates. To obtain the features i.e. the counts over the neighborhoods of the query variables satisfying the first order rules, the Gaifman graph of the hypergraph can be used. The rest of the process remains the same as described in chapter 5.

Considering the work we have already done previously in the space of Gaifman models, the joint learning and hypergraph embeddings are the two most natural extensions.

REFERENCES

Al Hasan, M. and M. J. Zaki (2011). A survey of link prediction in social networks. In *Social network data analytics*.

Ali, S. M., N. S. Hadad, and A. M. Jawad (2015). Effect of amoxicillin and cefalexin on the pharmacokinetics of diclofenac sodium in healthy volunteers. *The Medical J Basrah University*.

Anderson, E., G. D. Veith, and D. Weininger (1987). *SMILES, a line notation and computerized interpreter for chemical structures*. US Environmental Protection Agency, Environmental Research Laboratory.

Arimoto, R., M.-A. Prasad, and E. M. Gifford (2005). Development of cyp3a4 inhibition models: comparisons of machine-learning techniques and molecular descriptors. *Journal of biomolecular screening*.

Arjovsky, M. and L. Bottou (2017). Towards principled methods for training generative adversarial networks. In *ICLR*.

Arjovsky, M., S. Chintala, and L. Bottou (2017). Wasserstein gan. *ICML*.

Aspden, P., J. Wolcott, J. L. Bootman, and L. R. Cronenwett (2007). Committee on identifying and preventing medication errors. *Preventing medication errors: quality chasm series*, 1269–1272.

August, J. T., F. Murad, M. Anders, J. T. Coyle, and A. P. Li (1997). *Drug-Drug interactions: scientific and regulatory perspectives*, Volume 43. Academic Press.

Bach, F. R., G. R. Lanckriet, and M. I. Jordan (2004). Multiple kernel learning, conic duality, and the smo algorithm. In *ICML*.

Bakir, G. H., T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan (2007). *Predicting Structured Data*.

Balazevic, I., C. Allen, and T. Hospedales (2019). Multi-relational poincaré graph embeddings. In *NIPS*.

Baltrušaitis, T., C. Ahuja, and L.-P. Morency (2018). Multimodal machine learning: A survey and taxonomy. *TPAMI*.

Barbosa, L. and J. Feng (2010). Robust sentiment detection on twitter from biased and noisy data. In *COLING*.

Barrows Jr, R. C. and P. D. Clayton (1996). Privacy, confidentiality, and electronic medical records. *JAMIA*.

Beam, A. L., B. Kompa, I. Fried, N. P. Palmer, X. Shi, T. Cai, and I. S. Kohane (2018). Clinical concept embeddings learned from massive sources of multimodal medical data. *arXiv preprint arXiv:1804.01486*.

Beaulieu-Jones, B. K., Z. S. Wu, C. Williams, R. Lee, S. P. Bhavnani, J. B. Byrd, and C. S. Greene (2019). Privacy-preserving generative deep neural networks support clinical data sharing. *Circulation: Cardiovascular Quality and Outcomes*.

Becker, M. L. et al. (2007). Hospitalisations and emergency department visits due to drug–drug interactions: a literature review. *Pharmacoepidemiology and Drug Safety*.

Becker, M. L., M. Kallewaard, P. W. Caspers, L. E. Visser, H. G. Leufkens, and B. H. Stricker (2007). Hospitalisations and emergency department visits due to drug–drug interactions: a literature review. *Pharmacoepidemiology and drug safety*.

Belkin, M., I. Matveeva, and P. Niyogi (2004). Regularization and semi-supervised learning on large graphs. In *Learning Theory*.

Belkin, M. and P. Niyogi (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*.

Berge, C. and E. Minieka (1973). *Graphs and hypergraphs*. North-Holland Publishing Company Amsterdam.

Blockeel, H. and L. De Raedt (1998). Top-down induction of first-order logical decision trees. *Artificial intelligence*.

Blockeel, H. and L. D. Raedt (1998). Top-down induction of first-order logical decision trees. *Artificial Intelligence*.

Bordes, A., N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko (2013). Translating embeddings for modeling multi-relational data. In *NIPS*.

Bordes, A., J. Weston, R. Collobert, Y. Bengio, et al. (2011). Learning structured embeddings of knowledge bases. In *AAAI*.

Braziunas, D. and C. Boutilier (2006). Preference elicitation and generalized additive utility. In *AAAI*.

Bröcheler, M., L. Mihalkova, and L. Getoor (2010). Probabilistic similarity logic. In *UAI*.

Buczak, A. L., S. Babin, and L. Moniz (2010). Data-driven approach for creating synthetic electronic medical records. *BMC medical informatics and decision making*.

Budnitz, D. S., M. C. Lovegrove, N. Shehab, and C. L. Richards (2011). Emergency hospitalizations for adverse drug events in older americans. *New England Journal of Medicine 365*(21), 2002–2012. PMID: 22111719.

Budnitz, D. S., D. A. Pollock, K. N. Weidenbach, A. B. Mendelsohn, T. J. Schroeder, and J. L. Annest (2006, 10). National Surveillance of Emergency Department Visits for Outpatient Adverse Drug Events. *JAMA 296*(15), 1858–1866.

Cai, H., V. W. Zheng, and K. C. Chang (2018). A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE TKDE*.

Cao, D.-S., J.-C. Zhao, et al. (2012). In silico toxicity prediction by support vector machine and smiles representation-based string kernel. *SAR and QSAR in Environmental Research 23*.

Chee, B. W., R. Berlin, and B. Schatz (2011). Predicting adverse drug events from personal health messages. In *AMIA Annual Symposium Proceedings*.

Chen, Y., H. Jiang, C. Li, X. Jia, and P. Ghamisi (2016). Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*.

Cheng, F. and Z. Zhao (2014). Machine learning-based prediction of drug–drug interactions by integrating drug phenotypic, therapeutic, chemical, and genomic properties. *Journal of the American Medical Informatics Association*.

Chhabra, N., M. L. Aseri, and D. Padmanabhan (2013). A review of drug isomerism and its significance. *International journal of applied and basic medical research*.

Choi, E., M. T. Bahadori, L. Song, W. F. Stewart, and J. Sun (2017). Gram: graph-based attention model for healthcare representation learning. In *KDD*.

Choi, E., M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart (2016). Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *NIPS*.

Choi, E., S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun (2017). Generating multi-label discrete patient records using generative adversarial networks. *MLHC*.

Chopra, S., R. Hadsell, Y. LeCun, et al. (2005). Learning a similarity metric discriminatively, with application to face verification. In *CVPR (1)*.

Chowdhury, F. M. and A. Lavelli (2011). Drug-drug interaction extraction using composite kernels. pp. 27–33.

Chowdhury, M. F. M. and A. Lavelli (2013). Fbk-irst: A multi-phase kernel based approach for drug-drug interaction detection and classification that exploits linguistic information. In *SEM*.

Christoudias, C., R. Urtasun, and T. Darrell (2012). Multi-view learning in the presence of view disagreement. *arXiv preprint arXiv:1206.3242*.

Chuan, P. M., M. Ali, et al. (2018). Link prediction in co-authorship networks based on hybrid content similarity metric. *Applied Intelligence*.

Coates, J. and D. Bollegala (2018). Frustratingly easy meta-embedding–computing meta-embeddings by averaging source word embeddings. *NAACL*.

Cortes, C., M. Mohri, and A. Rostamizadeh (2012). Algorithms for learning kernels based on centered alignment. *JMLR*.

Cortes, C. and V. Vapnik (1995). Support-vector networks. *Machine Learning*.

Craven, M. and J. W. Shavlik (1996). Extracting tree-structured representations of trained networks. In *NIPS*.

Csiszár, I. and G. Tusnády (1984). Information Geometry and Alternating minimization procedures. *Statistics and Decisions*.

Cui, Z., H. Chang, S. Shan, and X. Chen (2014). Generalized unsupervised manifold alignment. In *NIPS*.

Dähne, S., F. Biessmann, W. Samek, S. Haufe, D. Goltz, C. Gundlach, A. Villringer, S. Fazli, and K.-R. Müller (2015). Multivariate machine learning methods for fusing multimodal functional neuroimaging data. *Proceedings of the IEEE*.

Das, M., D. S. Dhami, G. Kunapuli, K. Kersting, and S. Natarajan (2019). Fast relational probabilistic inference and learning: Approximate counting via hypergraphs. In *AAAI*.

Das, M., D. S. Dhami, Y. Yu, G. Kunapuli, and S. Natarajan (2019). Knowledge-augmented Column Networks: Guiding Deep Learning with Advice. In *ICML HILL Workshop*.

Das, M., P. Odom, M. R. Islam, J. R. Doppa, D. Roth, and S. Natarajan (2018). Preference-Guided Planning: An Active Elicitation Approach. In *AAMAS*.

Das, M., Y. Wu, T. Khot, K. Kersting, and S. Natarajan (2016). Scaling Lifted Probabilistic Inference and Learning Via Graph Databases. In *SDM*.

Das, S., B. L. Matthews, A. N. Srivastava, and N. C. Oza (2010). Multiple kernel learning for heterogeneous anomaly detection: algorithm and aviation safety case study. In *KDD*.

De Campos, C. P., Z. Zeng, and Q. Ji (2009). Structure learning of bayesian networks using constraints. In *ICML*.

De Raedt, L., A. Kimmig, and H. Toivonen (2007). Problog: A probabilistic prolog and its application in link discovery.

Defferrard, M., X. Bresson, and P. Vandergheynst (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*.

Deprez, P., P. V. Shevchenko, and M. V. Wüthrich (2017). Machine learning techniques for mortality modeling. *European Actuarial Journal*.

Dettmers, T., P. Minervini, P. Stenetorp, and S. Riedel (2018). Convolutional 2d knowledge graph embeddings. In *AAAI*.

Dhami, D. S., G. Kunapuli, M. Das, D. Page, and S. Natarajan (2018). Drug-drug interaction discovery: Kernel learning from heterogeneous similarities. *Smart Health*.

Dhami, D. S., G. Kunapuli, D. Page, and S. Natarajan (2019). Predicting drug-drug interactions from molecular structure images. *AAAI Fall Symposium on AI for Social Good*.

Dhami, D. S., A. Soni, D. Page, and S. Natarajan (2017). Identifying parkinson's patients: A functional gradient boosting approach. In *AIME*.

Dhami, D. S., S. Yen, G. Kunapuli, and S. Natarajan (2020). Non-parametric learning of gaifman models. *STARAI workshop @ AAAI*.

DHHS (2010). U.S. Department of Health and Human Services, Office of Inspector General (OIG). Adverse Events in Hospitals: National Incidence Among Medicare Beneficiaries, Report No.: OEI-06-09-00090. `https://oig.hhs.gov/oei/reports/oei-06-09-00090.pdf`. [Online; accessed 21-April-2019].

Ding, G., Y. Guo, and J. Zhou (2014). Collective matrix factorization hashing for multimodal data. In *CVPR*.

Ding, H., I. Takigawa, H. Mamitsuka, and S. Zhu (2013). Similarity-based machine learning methods for predicting drug–target interactions: a brief review. *Briefings in Bioinformatics*.

Ding, X., Y. Luo, Q. Li, Y. Cheng, G. Cai, R. Munnoch, D. Xue, Q. Yu, X. Zheng, and B. Wang (2018). Prior knowledge-based deep learning method for indoor object recognition and application. *Systems Science & Control Engineering*.

Ding, Z. and Y. Fu (2014). Low-rank common subspace for multi-view learning. In *ICDM*.

Dinov, I. D. (2016). Volume and value of big healthcare data. *Journal of medical statistics and informatics*.

Domingos, P. and D. Lowd (2009). *Markov Logic: An Interface Layer for AI*. San Rafael, CA: Morgan & Claypool.

Donini, M., J. M. Monteiro, M. Pontil, J. Shawe-Taylor, and J. Mourao-Miranda (2016). A multimodal multiple kernel learning approach to alzheimer's disease detection. In *MLSP*.

Doshi, J. A., F. B. Hendrick, J. S. Graff, and B. C. Stuart (2016). Data, data everywhere, but access remains a big issue for researchers: a review of access policies for publicly-funded patient-level health care data in the united states. *eGEMs*.

Dwork, C. (2008). Differential privacy: A survey of results. In *TAMS*.

Esteban, C., S. L. Hyland, and G. Rätsch (2017). Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*.

Feng, J., M. Huang, L. Zhao, Y. Yang, and X. Zhu (2018). Reinforcement learning for relation classification from noisy data. In *AAAI*.

Forster, A. J., H. J. Murff, J. F. Peterson, T. K. Gandhi, and D. W. Bates (2005, Apr). Adverse drug events occurring following hospital discharge. *J Gen Intern Med 20*(4), 317–323.

Förstner, W. (1994). A framework for low level feature extraction. In *ECCV*.

França, M. V. M., G. Zaverucha, and A. S. d'Avila Garcez (2014). Fast relational learning using bottom clause propositionalization with artificial neural networks. *MLJ*.

Frid-Adar, M., E. Klang, M. Amitai, J. Goldberger, and H. Greenspan (2018). Synthetic data augmentation using gan for improved liver lesion classification. In *ISBI*.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*.

Fung, G., O. Mangasarian, and J. Shavlik (2002). Knowledge-based support vector machine classifiers. In *In Advances in Neural Information Processing Systems 14*.

Gaifman, H. (1982). On local and non-local properties. In *Studies in Logic and the Foundations of Mathematics*.

Gamberger, D., N. Lavrac, and S. Dzeroski (2000). Noise detection and elimination in data preprocessing: experiments in medical domains. *Applied Artificial Intelligence*.

Gamon, M. (2004). Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *ACL*.

Ganchev, K., J. Gillenwater, B. Taskar, et al. (2010). Posterior regularization for structured latent variable models. *JMLR*.

Getoor, L. and B. Taskar (2007). *Introduction to statistical relational learning*. MIT press.

Goetz, C. G., B. C. Tilley, S. R. Shaftman, G. T. Stebbins, S. Fahn, P. Martinez-Martin, W. Poewe, C. Sampaio, M. B. Stern, R. Dodel, et al. (2008). Movement disorder society-sponsored revision of the unified parkinson's disease rating scale (mds-updrs): scale presentation and clinimetric testing results. *Movement disorders: official journal of the Movement Disorder Society*.

Gönen, M. and E. Alpaydin (2008). Localized multiple kernel learning. In *ICML*.

Gönen, M. and E. Alpaydın (2011). Multiple kernel learning algorithms. *JMLR*.

Gönen, M. and E. Alpaydın (2011). Multiple kernel learning algorithms. *JMLR*.

Goodfellow, I., Y. Bengio, and A. Courville (2016a). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Goodfellow, I., Y. Bengio, and A. Courville (2016b). *Deep Learning*. The MIT Press.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). Generative adversarial nets. In *NIPS*.

Goyal, P. and E. Ferrara (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*.

Grobe, M. (2009). Rdf, jena, sparql and the 'semantic web'. In *SIGUCCS*.

Grohe, M. and S. Wöhrle (2004). An existential locality theorem. *Annals of Pure and Applied Logic*.

Groves, P., B. Kayyali, D. Knott, and S. V. Kuiken (2016). The'big data'revolution in healthcare: Accelerating value and innovation.

Guengerich, F. P. (1997). Role of cytochrome p450 enzymes in drug-drug interactions. In *Advances in pharmacology*.

Guibas, J. T., T. S. Virdi, and P. S. Li (2017). Synthetic medical images from dual generative adversarial networks. *arXiv preprint arXiv:1709.01872*.

Gulrajani, I., F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville (2017). Improved training of wasserstein gans. In *NIPS*.

Guo, Y. (2013). Convex subspace representation learning from multi-view data. In *AAAI*.

Gurulingappa, H., A. Mateen-Rajpu, and L. Toldo (2012). Extraction of potential adverse drug events from medical case reports. *Journal of biomedical semantics*.

Gurwitz, J. H. et al. (2003). Incidence and preventability of adverse drug events among older persons in the ambulatory setting. *JAMA*.

Guzella, T. S. and W. M. Caminhas (2009). A review of machine learning approaches to spam filtering. *Expert Systems with Applications*.

Hadsell, R., S. Chopra, and Y. LeCun (2006). Dimensionality reduction by learning an invariant mapping. In *CVPR*.

Hamilton, W., Z. Ying, and J. Leskovec (2017). Inductive representation learning on large graphs. In *NIPS*.

Harada, S., H. Akita, et al. (2018). Dual convolutional neural network for graph of graphs link prediction. *arXiv preprint arXiv:1810.02080*.

Havaei, M., A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin, and H. Larochelle (2017). Brain tumor segmentation with deep neural networks. *Medical image analysis*.

He, S., K. Liu, G. Ji, and J. Zhao (2015). Learning to represent knowledge graphs with gaussian embedding. In *CIKM*.

He, X. and P. Niyogi (2003). Locality preserving projections. In *NIPS*.

Heckerman, D., D. Chickering, C. Meek, R. Rounthwaite, and C. Kadie (2001). Dependency networks for inference, collaborative filtering, and data visualization. *JMLR 1*, 49–75.

Helma, C., T. Cramer, S. Kramer, and L. De Raedt (2004). Data mining and machine learning techniques for the identification of mutagenicity inducing substructures and structure activity relationships of noncongeneric compounds. *J Chem Inform Comput Sci*.

Hinton, G. (2012). Lecture 6d: a separate, adaptive learning rate for each connection. slides of lecture neural networks for machine learning. Technical report, Technical report, Slides of Lecture Neural Networks for Machine Learning.

Hoi, S. C. H., R. Jin, and M. R. Lyu (2007). Learning nonparametric kernel matrices from pairwise constraints. In *ICML*.

Hong, W. S., A. D. Haimovich, and R. A. Taylor (2018). Predicting hospital admission at emergency department triage using machine learning. *PloS one*.

Howe, B., J. Stoyanovich, H. Ping, B. Herman, and M. Gee (2017). Synthetic data for social good. *arXiv preprint arXiv:1710.08874*.

Hu, S. and G. Zheng (2009). Driver drowsiness detection with eyelid related parameters by support vector machine. *Expert Systems with Applications*.

Hu, Z., Z. Yang, R. R. Salakhutdinov, L. Qin, X. Liang, H. Dong, and E. P. Xing (2018). Deep generative models with learnable knowledge constraints. In *NeurIPS*.

Iman, R. L. and W.-J. Conover (1982). A distribution-free approach to inducing rank correlation among input variables. *Communications in Statistics-Simulation and Computation*.

Iordanov, B., K. Vandev, C. Costa, M. Marinov, M. Saraiva de Queiroz, I. Holsman, A. Picard, and I. Bogdahn (2010). HyperGraphDB 1.3.

Jaakkola, T., D. Sontag, A. Globerson, and M. Meila (2010). Learning bayesian network structure using lp relaxations. In *AISTATS*.

Jensen, D. and J. Neville (2002). Linkage and autocorrelation cause feature selection bias in relational learning. In *ICML*.

Jeske, D., D. Gokhale, and L. Ye (2006). Generating synthetic data from marginal fitting for testing the efficacy of data-mining tools. *IJPR*.

Jin, X., F. Zhuang, S. Wang, Q. He, and Z. Shi (2013). Shared structure learning for multiple tasks with multiple views. In *ECML-PKDD*.

Jin, Z., Y. Sun, and A. C. Cheng (2009). Predicting cardiovascular disease from real-time electrocardiographic monitoring: An adaptive machine learning approach on a cell phone. In *EMBC*.

Jordon, J., J. Yoon, and M. van der Schaar (2019). Pate-gan: Generating synthetic data with differential privacy guarantees. *ICLR*.

Kang, N., B. Singh, C. Bui, Z. Afzal, E. M. van Mulligen, and J. A. Kors (2014). Knowledge-based extraction of adverse drug events from biomedical text. *BMC bioinformatics*.

Karlsson, I., J. Zhao, L. Asker, and H. Boström (2013). Predicting adverse drug events by analyzing electronic patient records. In *AIME*.

Kaur, N., G. Kunapuli, S. Joshi, K. Kersting, and S. Natarajan (2019). Neural networks for relational data. *ILP*.

Kaur, N., G. Kunapuli, T. Khot, K. Kersting, W. Cohen, and S. Natarajan (2017a). Relational restricted boltzmann machines: A probabilistic logic learning approach. In *ILP*.

Kaur, N., G. Kunapuli, T. Khot, K. Kersting, W. Cohen, and S. Natarajan (2017b). Relational restricted boltzmann machines: A probabilistic logic learning approach. In *ILP*.

Kazemi, S. M. and D. Poole (2018a). Relnn: A deep neural model for relational learning. In *AAAI*.

Kazemi, S. M. and D. Poole (2018b). RelNN: A deep neural model for relational learning. In *AAAI*.

Kersting, K., B. Ahmadi, and S. Natarajan (2009). Counting Belief Propagation. In *UAI*.

Khot, T., S. Natarajan, K. Kersting, and J. Shavlik (2011). Learning markov logic networks via functional gradient boosting. In *ICDM*.

Khot, T., S. Natarajan, and J. Shavlik (2013). Relational one-class classification: A non-parametric approach. In *AAAI*.

Khot, T., S. Natarajan, and J. W. Shavlik (2014). Relational one-class classification: A non-parametric approach. In *AAAI*.

Kingma, D. P. and J. Ba (2014). Adam: A method for stochastic optimization. *arXiv preprint*.

Kipf, T. N. and M. Welling (2017). Semi-supervised classification with graph convolutional networks. *ICLR*.

Kok, S. and P. Domingos (2005). Learning the structure of markov logic networks. In *ICML*.

Kok, S. and P. Domingos (2009). Learning markov logic network structure via hypergraph lifting. In *ICML*.

Kok, S., M. Sumner, M. Richardson, P. Singla, H. Poon, D. Lowd, J. Wang, and P. Domingos (2009). The alchemy system for statistical relational {AI}.

Koller, D. and N. Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press.

Kolyvakis, P., A. Kalousis, and D. Kiritsis (2019). Hyperkg: Hyperbolic knowledge graph embeddings for knowledge base completion. *arXiv preprint arXiv:1908.04895*.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). Imagenet classification with deep convolutional neural networks. In *NIPS*.

Kubat, M., R. C. Holte, and S. Matwin (1998). Machine learning for the detection of oil spills in satellite radar images. *Machine learning*.

Kuhn, H. W. and A. W. Tucker (1953). *Contributions to the Theory of Games*.

Kunapuli, G., K. Bennett, M. Shabbeer, and J. Shavlik (2010). Online knowledge-based support vector machines. In *ECML PKDD*.

Kunapuli, G., P. Odom, J. Shavlik, and S. Natarajan (2013a). Guiding autonomous agents to better behaviors through human advice. In *ICDM*.

Kunapuli, G., P. Odom, J. W. Shavlik, and S. Natarajan (2013b). Guiding autonomous agents to better behaviors through human advice. In *ICDM*.

Lanckriet, G. R., N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan (2004). Learning the kernel matrix with semidefinite programming. *JMLR*.

Lao, N. and W. W. Cohen (2010a). Relational retrieval using a combination of path-constrained random walks. *Machine learning*.

Lao, N. and W. W. Cohen (2010b). Relational retrieval using a combination of path-constrained random walks. *Machine learning 81*(1), 53–67.

Lao, N., T. Mitchell, and W. W. Cohen (2011). Random walk inference and learning in a large scale knowledge base. In *EMNLP*.

Larrañaga, P. et al. (1996). Structure learning of bayesian networks by genetic algorithms: A performance analysis of control parameters. *TPAMI*.

Le, Q. V., A. J. Smola, and T. Gärtner (2006). Simpler knowledge-based support vector machines. In *ICML*.

Lee, H., P. Pham, Y. Largman, and A. Y. Ng (2009). Unsupervised feature learning for audio classification using convolutional deep belief networks. In *NIPS*.

Lee, S.-I., V. Ganapathi, and D. Koller (2007). Efficient structure learning of markov networks using l_1-regularization. In *NIPS*.

Li, Q., C. Liu, J. Oster, and G. D. Clifford (2016). Signal processing and feature selection preprocessing for classification in noisy healthcare data. *Machine Learning for Healthcare Technologies*.

Li, S.-Y., Y. Jiang, and Z.-H. Zhou (2014). Partial multi-view clustering. In *AAAI*.

Li, Y., F. Nie, H. Huang, and J. Huang (2015). Large-scale multi-view spectral clustering via bipartite graph. In *AAAI*.

Li, Y., K. Swersky, and R. Zemel (2015). Generative moment matching networks. In *ICML*.

Liu, G., T.-M. H. Hsu, M. McDermott, W. Boag, W.-H. Weng, P. Szolovits, and M. Ghassemi (2019). Clinically accurate chest x-ray report generation. *MLHC*.

Liu, X. and H. Chen (2013). Azdrugminer: an information extraction system for mining patient-reported adverse drug events in online patient forums. In *ICSH*.

Lowd, D. and J. Davis (2010). Learning markov network structure with decision trees. In *ICDM*.

Lowd, D. and P. Domingos (2007). Efficient weight learning for markov logic networks. In *ECML-PKDD*.

Lü, L. and T. Zhou (2011). Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*.

Mahmood, F., R. Chen, and N. J. Durr (2018). Unsupervised reverse domain adaptation for synthetic medical images via adversarial training. *IEEE transactions on medical imaging*.

Mansinghka, V. K., C. Kemp, J. B. Tenenbaum, and T. L. Griffiths (2006). Structured priors for structure learning. In *UAI*.

Marek, K., D. Jennings, S. Lasch, A. Siderowf, C. Tanner, T. Simuni, C. Coffey, K. Kieburtz, E. Flagg, S. Chowdhury, et al. (2011). The parkinson progression marker initiative (ppmi). *Progress in neurobiology*.

Mariette, J. and N. Villa-Vialaneix (2018). Unsupervised multiple kernel learning for heterogeneous data integration. *Bioinformatics*.

Martínez, V., F. Berzal, and J.-C. Cubero (2017). A survey of link prediction in complex networks. *ACM Computing Surveys (CSUR)*.

Mitchell, T. (1997). *Machine Learning* (1 ed.). New York, NY, USA: McGraw-Hill, Inc.

Mitchell, T., W. Cohen, et al. (2018). Never-ending learning. *ACM Communications*.

Mitchell, T. M. (1980). *The need for biases in learning generalizations*. Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ. New Jersey.

Miyato, T., T. Kataoka, M. Koyama, and Y. Yoshida (2018). Spectral normalization for generative adversarial networks. *ICLR*.

Mnih, V. and G. E. Hinton (2012). Learning to label aerial images from noisy data. In *ICML*.

Motwani, M., D. Dey, D. S. Berman, G. Germano, S. Achenbach, M. H. Al-Mallah, D. Andreini, M. J. Budoff, F. Cademartiri, T. Q. Callister, et al. (2016). Machine learning for prediction of all-cause mortality in patients with suspected coronary artery disease: a 5-year multicentre prospective registry analysis. *European heart journal*.

Muggleton, S. (1991). Inductive logic programming. *New generation computing*.

Muggleton, S. (1995). Inverse entailment and progol. *New generation computing*.

Multiple. Drugs.com. (`https://www.drugs.com/drug-interactions/flomax-with-limbitrol-2146-1397-169-8640.html`).

Multiple. drugs.com. (`https://www.drugbank.ca/drugs/DB00333`).

Multiple. rxlist.com. (https://www.rxlist.com/drug-interactions/glyburide-metformin-oral-and-omeprazole-oral-interaction.htm).

Natarajan, S., V. Bangera, T. Khot, J. Picado, A. Wazalwar, V. S. Costa, D. Page, and M. Caldwell (2017). Markov logic networks for adverse drug event extraction from text. *KIS*.

Natarajan, S., S. Joshi, B. N. Saha, A. Edwards, T. Khot, E. Moody, K. Kersting, C. T. Whitlow, and J. A. Maldjian (2012). A machine learning pipeline for three-way classification of alzheimer patients from structural magnetic resonance images of the brain. In *ICMLA*.

Natarajan, S., K. Kersting, E. Ip, D. R. Jacobs, and J. Carr (2013). Early prediction of coronary artery calcification levels using machine learning. In *IAAI*.

Natarajan, S., K. Kersting, T. Khot, and J. Shavlik (2015). *Boosted Statistical Relational Learners: From Benchmarks to Data-Driven Medicine*. Springer.

Natarajan, S., T. Khot, K. Kersting, B. Gutmann, and J. Shavlik (2012a). Gradient-based boosting for statistical relational learning: The relational dependency network case. *Machine Learning*.

Natarajan, S., T. Khot, K. Kersting, B. Gutmann, and J. Shavlik (2012b). Gradient-based boosting for statistical relational learning: The Relational Dependency Network case. *MLJ*.

Natarajan, S., B. Saha, S. Joshi, A. Edwards, T. Khot, E. M. Davenport, K. Kersting, C. T. Whitlow, and J. A. Maldjian (2014). Relational learning helps in three-way classification of alzheimer patients from structural magnetic resonance images of the brain. *International Journal of Machine Learning and Cybernetics*.

Naveršnik, K. and K. Rojnik (2012). Handling input correlations in pharmacoeconomic models. *Value in Health*.

NCHS (2014). National Center for Health Statistics, Prescription drug use in the past 30 days, by sex, race and Hispanic origin, and age: United States, selected years 1988–1994 through 2011–2014. https://www.cdc.gov/nchs/data/hus/2017/079.pdf. [Online; accessed 21-April-2019].

Nebeker, J. R., P. Barach, and M. H. Samore (2004, 05). Clarifying Adverse Drug Events: A Clinician's Guide to Terminology, Documentation, and Reporting. *Annals of Internal Medicine 140*(10), 795–801.

Neumann, T. and G. Moerkotte (2011). Characteristic sets: Accurate cardinality estimation for rdf queries with multiple joins. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on*, pp. 984–994. IEEE.

Neville, J. and D. Jensen (2007a). Relational dependency networks. *Introduction to Statistical Relational Learning*.

Neville, J. and D. Jensen (2007b). Relational dependency networks. *JMLR*.

Nguyen, L. A., H. He, and C. Pham-Huy (2006). Chiral drugs: an overview. *International journal of biomedical science: IJBS*.

Nickel, M. and D. Kiela (2017). Poincaré embeddings for learning hierarchical representations. In *NIPS*.

Nickel, M., K. Murphy, V. Tresp, and E. Gabrilovich (2016). A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*.

Nickel, M., L. Rosasco, T. A. Poggio, et al. (2016). Holographic embeddings of knowledge graphs. In *AAAI*.

Nickel, M., V. Tresp, and H.-P. Kriegel (2011). A three-way model for collective learning on multi-relational data. In *ICML*.

Niepert, M. (2016). Discriminative gaifman models. In *NIPS*.

Nies, A. T., U. Hofmann, C. Resch, E. Schaeffeler, M. Rius, and M. Schwab (2011). Proton pump inhibitors inhibit metformin uptake by organic cation transporters (octs). *PLoS One*.

Nigam, S. K. (2015). What do drug transporters really do? *Nature reviews. Drug discovery 14*(1), 29.

Niu, F., C. Ré, A. Doan, and J. Shavlik (2011). Tuffy: Scaling up statistical inference in markov logic networks using an rdbms. *VLDB*.

Odom, P., V. Bangera, T. Khot, D. Page, and S. Natarajan (2015). Extracting adverse drug events from text using human advice. In J. H. Holmes, R. Bellazzi, L. Sacchi, and N. Peek (Eds.), *AIME*.

Odom, P., T. Khot, R. Porter, and S. Natarajan (2015). Knowledge-based probabilistic logic learning. In *AAAI*.

Odom, P. and S. Natarajan (2018). Human-guided learning for probabilistic logic models. *Frontiers in Robotics and AI*.

Ogu, C. C. and J. L. Maxa (2000). Drug interactions due to cytochrome p450. In *Baylor University Medical Center Proceedings*.

Page, D., S. Natarajan, V. Santos Costa, P. Peissig, A. Barnard, and M. Caldwell (2012). Identifying adverse drug events from multi-relational healthcare data. In *AAAI*.

Pavicić, M., A. Van Winkelhoff, and J. De Graaff (1991). Synergistic effects between amoxicillin, metronidazole, and the hydroxymetabolite of metronidazole against actinobacillus actinomycetemcomitans. *Antimicrobial agents and chemotherapy*.

Percha, B. and R. B. Altman (2013). Informatics confronts drug-drug interactions. *Trends Pharmacol Sci 34*, 178—84.

Percha, B., Y. Garten, and R. B. Altman (2012). Discovery and explanation of drug-drug interactions via text mining. In *PSB*.

Perlich, C. and F. Provost (2006). Distribution-based aggregation for relational learning with identifier attributes. *Machine Learning*.

Pham, T., T. Tran, D. Q. Phung, and S. Venkatesh (2017). Column networks for collective classification. In *AAAI*.

Poole, D. (2003). First-Order Probabilistic Inference. In *IJCAI*, pp. 985–991.

Quinlan, J. R. (1990). Learning logical definitions from relations. *Machine learning*.

Raedt, L. D., K. Kersting, S. Natarajan, and D. Poole (2016). Statistical relational artificial intelligence: Logic, probability, and computation. *Synthesis Lectures on Artificial Intelligence and Machine Learning 10*(2), 1–189.

Rakotomamonjy, A., F. Bach, S. Canu, and Y. Grandvalet (2007). More efficiency in multiple kernel learning. In *Proceedings of the 24th international conference on Machine learning*, pp. 775–782. ACM.

Ramanan, N., G. Kunapuli, T. Khot, B. Fatemi, S. M. Kazemi, D. Poole, K. Kersting, and S. Natarajan (2018). Structure learning for relational logistic regression: An ensemble approach. In *KR*.

Richardson, M. and P. Domingos (2006). Markov logic networks. *Machine learning 62*(1-2), 107–136.

Rojas, J. C., K. A. Carey, D. P. Edelson, L. R. Venable, M. D. Howell, and M. M. Churpek (2018). Predicting intensive care unit readmission with machine learning using electronic health record data. *Annals of the American Thoracic Society*.

Salihovic, I., H. Serdarevic, and J. Kevric (2018). The role of feature selection in machine learning for detection of spam and phishing attacks. In *International Symposium on Innovative and Interdisciplinary Applications of Advanced Technologies*.

Sarkhel, S., D. Venugopal, T. Pham, P. Singla, and V. Gogate (2016). Scalable training of Markov logic networks using approximate counting. In *AAAI*.

Scheuer, E. M. and D. S. Stoller (1962). On the generation of normal random vectors. *Technometrics*.

Schiefer, B., L. G. Strain, and W. P. Yan (1998, June 2). Method for estimating cardinalities for query processing in a relational database management system. US Patent 5,761,653.

Schlichtkrull, M., T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling (2018). Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*.

Schölkopf, B., P. Simard, A. J. Smola, and V. Vapnik (1998). Prior knowledge in support vector kernels. In *Advances in neural information processing systems*, pp. 640–646.

Sebban, M. and J.-C. Janodet (2003). On state merging in grammatical inference: A statistical approach for dealing with noisy data. In *ICML*.

Segura-Bedmar, I., P. Martinez, and C. de Pablo-Sánchez (2011). Using a shallow linguistic kernel for drug–drug interaction extraction. *Journal of biomedical informatics*.

Segura Bedmar, I., P. Martinez, and D. Sánchez Cisneros (2011). The 1st ddiextraction-2011 challenge task: Extraction of drug-drug interactions from biomedical texts.

Senechal, T., V. Rapp, H. Salam, R. Seguier, K. Bailly, and L. Prevost (2012). Facial action recognition combining heterogeneous features via multikernel learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*.

Seputis, E. A. (2000, January 4). Database system with methods for performing cost-based estimates using spline histograms. US Patent 6,012,054.

Shamsuddin, R., B. M. Maweu, M. Li, and B. Prabhakaran (2018). Virtual patient model: an approach for generating synthetic healthcare time series data. In *ICHI*.

Shavlik, J. W. and G. G. Towell (1989). Combining explanation-based learning and artificial neural networks. In *Proceedings of the sixth international workshop on Machine learning*. Elsevier.

Shawe-Taylor, J. and N. Cristianini (2004). *Kernel Methods for Pattern Analysis*. Cambridge Univ. Press.

Singla, P. and P. Domingos (2005). Discriminative training of markov logic networks. In *AAAI*.

Sinha, A., Z. Shen, Y. Song, H. Ma, D. Eide, B.-J. Hsu, and K. Wang (2015). An overview of microsoft academic service (mas) and applications. In *WWW*.

Sokolov, A., C. Funk, K. Graim, K. Verspoor, and A. Ben-Hur (2013). Combining heterogeneous data sources for accurate functional annotation of proteins. In *BMC bioinformatics*.

Solberg, A. S., G. Storvik, R. Solberg, and E. Volden (1999). Automatic detection of oil spills in ers sar images. *IEEE Transactions on geoscience and remote sensing*.

Sonnenburg, S., G. Rätsch, C. Schäfer, and B. Schölkopf (2006). Large scale multiple kernel learning. *JMLR*.

Sourek, G., V. Aschenbrenner, F. Zelezny, and O. Kuzelka (2015). Lifted relational neural networks. *arXiv preprint arXiv:1508.05128*.

Srinivasan, A. (2001). The aleph manual.

Srinivasan, A., R. D. King, S. H. Muggleton, and M. Sternberg (1997). Carcinogenesis predictions using ILP. *Inductive Logic Programming*.

Srivastava, N. and R. R. Salakhutdinov (2012). Multimodal learning with deep boltzmann machines. In *NIPS*.

Sun, S. (2013). A survey of multi-view machine learning. *Neural computing and applications*.

Sun, Z., Z.-H. Deng, J.-Y. Nie, and J. Tang (2019). Rotate: Knowledge graph embedding by relational rotation in complex space. *ICLR*.

Tan, B., E. Zhong, E. W. Xiang, and Q. Yang (2013). Multi-transfer: Transfer learning with multiple views and multiple sources. In *SDM*.

Tango, F. and M. Botta (2013). Real-time detection system of driver distraction using machine learning. *IEEE Transactions on Intelligent Transportation Systems*.

Taskar, B., M.-F. Wong, P. Abbeel, and D. Koller (2004). Link prediction in relational data. In *NIPS*.

Tatonetti, N. P., G. H. Fernald, and R. B. Altman (2011). A novel signal detection algorithm for identifying hidden drug-drug interactions in adverse event reports. *JAMIA*.

Taylor, R. A., J. R. Pare, A. K. Venkatesh, H. Mowafi, E. R. Melnick, W. Fleischman, and M. K. Hall (2016). Prediction of in-hospital mortality in emergency department patients with sepsis: a local big data–driven, machine learning approach. *Academic emergency medicine*.

Thirumalai, C., A. Duba, and R. Reddy (2017). Decision making system using machine learning and pearson for heart attack. In *ICECA*.

Thomas, P., M. Neves, I. Solt, D. Tikk, and U. Leser (2011). Relation extraction for drug-drug interactions using ensemble learning. *Training*.

Thoolen, M., B. Wilfert, A. Jonge, P. Timmermans, P. Zwieten, et al. (1984). Effect of salbutamol and the pde-inhibitor ra 642 on the clonidine withdrawal syndrome in rats. *Auton Autacoid Pharmacol*.

Toh, K. C., M. J. Todd, and R. H. Tütüncü (1999). SDPT3 – A Matlab software package for semidefinite programming, v. 1.3. *OMS*.

Towell, G. and J. Shavlik (1994). Knowledge-based artificial neural networks. *Artificial intelligence 70*(1-2), 119–165.

Trouillon, T., J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard (2016). Complex embeddings for simple link prediction. ICML.

Troyanskaya, O. G., K. Dolinski, A. B. Owen, R. B. Altman, and D. Botstein (2003). A bayesian framework for combining heterogeneous data sources for gene function prediction (in saccharomyces cerevisiae). *Proceedings of the National Academy of Sciences*.

US Food and Drug Administration (2012). Drug interaction studies-study design, data analysis, implications for dosing, and labeling recommendations. *Center for Drug Evaluation and Research, Bethesda, MD*.

Van Hulse, J. and T. Khoshgoftaar (2009). Knowledge discovery from imbalanced and noisy data. *Data & Knowledge Engineering*.

Varma, M. and B. R. Babu (2009). More generality in efficient multiple kernel learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 1065–1072. ACM.

Veith, N. and R. Steele (2018). Machine learning-based prediction of icu patient mortality at time of admission. In *ICISDM*.

Vilar, S., R. Harpaz, E. Uriarte, L. Santana, R. Rabadan, and C. Friedman (2012). Drug—drug interaction through molecular structure similarity analysis. *JAMIA*.

Vilar, S., E. Uriarte, L. Santana, T. Lorberbaum, G. Hripcsak, C. Friedman, and N. P. Tatonetti (2014). Similarity-based modeling in large-scale prediction of drug-drug interactions. *Nature protocols*.

Šourek, G., V. Aschenbrenner, F. Železny, and O. Kuželka (2015). Lifted relational neural networks. In *NIPS Workshop on Cognitive Comput.: Integr. Neural & Symbolic Approaches*.

Walker T., e. a. (2009). Ilp for bootstrapped learning: A layered approach to automating the ilp setup problem. In *ILP*.

Wang, C. and S. Mahadevan (2009). Manifold alignment without correspondence. In *IJCAI*.

Wang, Q., Z. Mao, B. Wang, and L. Guo (2017a). Knowledge graph embedding: A survey of approaches and applications. *IEEE TKDE*.

Wang, Q., Z. Mao, B. Wang, and L. Guo (2017b). Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*.

Wang, Z., J. Zhang, J. Feng, and Z. Chen (2014). Knowledge graph embedding by translating on hyperplanes. In *AAAI*.

Weiss, J., S. Natarajan, and D. Page (2012). Multiplicative forests for continuous time processes. In *NIPS*.

Weiss, J. C., S. Natarajan, P. L. Peissig, C. A. McCarty, and D. Page (2012). Machine learning for personalized medicine: Predicting primary myocardial infarction from electronic health records. *AI Magazine*.

Wen, J., J. Li, Y. Mao, S. Chen, and R. Zhang (2016). On the representation and embedding of knowledge bases beyond binary relations. *IJCAI*.

Xia, R., Y. Pan, L. Du, and J. Yin (2014). Robust multi-view spectral clustering via low-rank and sparse decomposition. In *AAAI*.

Xiao, H., M. Huang, and X. Zhu (2016). Transg: A generative model for knowledge graph embedding. In *ACL*.

Yang, B., W.-t. Yih, X. He, J. Gao, and L. Deng (2015). Embedding entities and relations for learning and inference in knowledge bases. *ICLR*.

Yang, P. and W. Gao (2013). Multi-view discriminant transfer learning. In *IJCAI*.

Yang, S. and S. Natarajan (2013). Knowledge intensive learning: Combining qualitative constraints with causal independence for parameter learning in probabilistic models. In *ECML-PKDD*.

Yang, Y., X. Wu, and X. Zhu (2004). Dealing with predictive-but-unpredictable attributes in noisy data sources. In *ECML-PKDD*.

Ying, R., R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec (2018). Graph convolutional neural networks for web-scale recommender systems. In *KDD*.

Yu, Z. and G. Herman (2005). On the earth mover's distance as a histogram similarity metric for image retrieval. In *2005 IEEE International Conference on Multimedia and Expo*.

Zhang, S., Y. Tay, L. Yao, and Q. Liu (2019). Quaternion knowledge graph embeddings. In *NIPS*.

Zhang, X., X. Zhang, and H. Liu (2015). Multi-task multi-view clustering for non-negative data. In *IJCAI*.

Zhao, J., X. Xie, X. Xu, and S. Sun (2017). Multi-view learning overview: Recent progress and new challenges. *Information Fusion*.

Zhou, Y. (2011). Structure learning of probabilistic graphical models: a comprehensive survey. *arXiv preprint arXiv:1111.6925*.

Zien, A. and C. S. Ong (2007). Multiclass multiple kernel learning. In *ICML*.

BIOGRAPHICAL SKETCH

Devendra Singh Dhami is a PhD candidate in the Department of Computer Science at The University of Texas at Dallas, advised by Professor Sriraam Natarajan. His research interests include development and application of machine learning algorithms for healthcare data. He also works on relational machine learning and learning with human guidance specifically for healthcare data. He completed his Master of Science (MS) degree in Computer Science from Indiana University, Bloomington. He obtained his Bachelor of Engineering (B.E) in Information Science & Engineering from Sir M. Visvesvaraya Institute of Technology (Bangalore, Karnataka, India).

Before pursuing graduate studies, Devendra used to work as a technology consultant developer for several years with one of of the leading information technology (IT) companies in India, Hewlett-Packard (HP). During his career with the software industry his expertise was in developing and maintaining HP reporting tools and large-scale data warehouse systems.

CURRICULUM VITAE

Devendra Singh Dhami

Contact Information:

Department of Computer Science
The University of Texas at Dallas
800 W. Campbell Rd., ECSS 4.613
Richardson, TX 75080-3021, U.S.A.

Email: devendra.dhami@utdallas.edu

Educational History:

B.Tech., Information Science & Engineering, Sir M. Visvesvaraya Institute of Technology, 2010
M.S., Computer Science, Indiana University, Bloomington, 2015
PhD, Computer Science, The University of Texas at Dallas, 2020

Effective Learning with Heterogeneous, Noisy, Multi-Relational Healthcare Data
PhD Dissertation
Computer Science Department, The University of Texas at Dallas
Advisors: Dr. Sriraam Natarajan

Employment History:

Research Assistant, The University of Texas at Dallas, June 2017 – present
Research Assistant, Indiana University Bloomington, August 2015 – June 2017
Teaching Assistant, Indiana University Bloomington, August 2013 – August 2015
Technology Consultant, Hewlett-Packard, August 2010 – July 2013

Professional Services:

Proceedings Chair: SDM 2020
PC member: Pacific Symposium on Biocomputing 2019,2020, IJCAI 2020
Reviewer: CoDS-COMADS 2020, SDM 2020
REU Mentor: Mentored 2 undergraduate researchers for Indiana University's Proactive Health Informatics "Research experiences for Undergraduates (REU) program"

Publications:

Book Chapters:
1. J.M. Billings, M. Eder, W.C. Flood, D.S. Dhami, S. Natarajan and C.T. Whitlow, " Machine Learning Applications to Resting-State Functional MR Imaging Analysis", in *Neuroimaging Clinics of North America*, 2017.

Conference Papers:

2. D.S. Dhami, S. Yan and S. Natarajan, "Knowledge Base Population via Relational One-Class Graph Convolutional Networks", in *Automatic Knowledge Base Construction (AKBC)*, 2020 (under review).
3. D.S. Dhami, G. Kunapuli and S. Natarajan, "Non-Parametric Structure Learning for Discriminative Gaifman Models", in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2020 (under review).
4. D.S. Dhami, G. Kunapuli, D. Page and S. Natarajan, "Predicting Drug-Drug Interactions from Molecular Structure Images", in *AI for Social Good - AAAI Fall Symposium*, 2019.
5. M. Das, D.S. Dhami, G. Kunapuli, K. Kersting and S. Natarajan, "Fast Relational Probabilistic Inference and Learning: Approximate Counting via Hypergraphs", in *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.
6. M. Das, D.S. Dhami, G. Kunapuli, K. Kersting and S. Natarajan,
"Approximate Counting for Fast Inference and Learning in Probabilistic Programming", in *The International Conference on Probabilistic Programming (ProbProg)*, 2018.
7. D.S. Dhami, G. Kunapuli, M. Das, D. Page and S. Natarajan,
"Drug-Drug Interaction Discovery:Kernel Learning from Heterogeneous Similarities", in *The IEEE/ACM Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, 2018.
8. D.S. Dhami, A. Soni, D. Page and S. Natarajan, "Identifying Parkinson's Patients : A Functional Gradient Boosting Approach", in *Artificial Intelligence in Medicine (AIME)*, 2017.

Workshop Papers:
10. D.S Dhami, S. Yan, G. Kunapuli and S. Natarajan, "Non-Parametric Learning of Gaifman Models", in *STARAI @ AAAI '20*.
11. D.S Dhami, G. Kunapuli and S. Natarajan, "Efficient Learning of Relational Gaifman Models using Probabilistic Logic", in *PLP @ ICLP'19*.
12. M. Das, D.S. Dhami, Y. Yu, G. Kunapuli and S. Natarajan, "Knowledge-augmented Column Networks: Guiding Deep Learning with Advice", in *HILL @ ICML'19*.
13. D.S. Dhami, D. Leake, and S. Natarajan, "Knowledge-based Morphological Classification of Galaxies from Vision Features", in *KnowPros @ AAAI'17*.