Relational Logistic Regression

by

Seyed Mehran Kazemi

B.Sc., Amirkabir University of Technology, 2012

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

August 2014

Abstract

Aggregation is a technique for representing conditional probability distributions as an analytic function of parents. Logistic regression is a commonly used representation for aggregators in Bayesian belief networks when a child has multiple parents. In this thesis, we consider extending logistic regression to directed relational models, where there are objects and relations among them, and we want to model varying populations and interactions among parents. We first examine the representational problems caused by population variation. We show how these problems arise even in simple cases with a single parametrized parent, and propose a linear relational logistic regression which we show can represent arbitrary linear (in population size) decision thresholds, whereas the traditional logistic regression cannot. Then we examine representing interactions among the parents of a child node, and representing non-linear dependency on population size. We propose a multi-parent relational logistic regression which can represent interactions among parents and arbitrary polynomial decision thresholds. We compare our relational logistic regression to Markov logic networks and represent their analogies and differences. Finally, we show how other well-known aggregators can be represented using relational logistic regression.

Preface

This thesis is largely based on two published works: one in Knowledge Representation and Reasoning (KR-2014) conference [22] and one in AAAI-2014 Statistical Relational AI (StarAI) workshop [23]. The former introduces the research problem described in this thesis and points out the solution and the latter summarizes the results and compares the model proposed in [22] with other similar methods in the literature. In both publications, I was the first and the correspondence author and I collaborated with David Buchman, Kristian Kersting, Sriraam Natarajan, and David Poole. In preparation of both papers, we had a great deal of active discussion among all the co-authors and numerous refinements. Below is a description of my contributions to the work.

The preliminary problems with applying logistic regression to relational models and the need to have a model which addresses those problems were realized by *David Poole*, my academic supervisor. He introduced this research topic to me and we started working on it and discussing it in our weekly meetings. Through our discussions, we found out that there are other issues (besides the preliminary problems) that we should take into account when using logistic regression for relational models. We developed a relational version of logistic regression (RLR) and investigated how it could address the issues with standard logistic regression.

Having the RLR model, I proposed the idea of defining canonical forms for that. I realized that both positive conjunctive and positive disjunctive formulae can be used as canonical forms for RLR, and made the proofs for both. We also suggested a third canonical form in terms of *XOR* (without proof), the idea for which was from *David Buchman*.

Considering the open problem mentioned in [48] about representing *k-degree* polynomial decision thresholds, we decided to figure out if our RLR can represent this class of decision thresholds or not. I proved that every polynomial decision threshold can be represented by our RLR, and every decision threshold that can be represented by our RLR is a polynomial decision thresholds.

We also considered representing other well-known aggregators using our RLR. I worked out how *OR*, *AND*, *Noisy-OR*, *Noisy-AND*, *Mean* $> t$, *More-than-t True*s, *More-than-t% True*s, *Max* $> t$ and *Mode* $= t$ can be represented by RLR. *David Buchman* then suggested how the aggregators *Max* and *Mode* can be modeled using

$Max > t$ and $Mode = t$ respectively.

After preparing the aforementioned contents for our papers, I designed the structure and wrote the initial draft for [22] and *David Poole* did this for [23]. All co-authors revised, proofread, and gave comments on the drafts, especially *David Poole* who modified several portions of [22]. Throughout the period working on both papers, *David Poole* and I had many active discussions in our weekly meetings, and *David Buchman*, *Kristian Kersting* and *Sriraam Natarajan*'s knowledge of the domain was very helpful in progression of the ideas.

Table of Contents

List of Tables

List of Figures

Acknowledgements

Many people have helped me make this thesis possible and I owe to all of them a debt I can never repay.

I am foremost indebted to my research supervisor Prof. David Poole. He taught me how to do research, how to think about research problems, how to evaluate ideas, how to express ideas, and many other aspects of the academia. Without his uninterrupted and endless guidance and support, this thesis was not possible.

I thank all my colleagues and friends in the department, my thesis second examiner Dr. Nicholas Harvey, all members of StarAI reading group at UBC, and all people who helped me during these years in my academic and non-academic life.

Dedication

I would like to dedicate my thesis to my beloved family, especially...

- to my Mom for her inexhaustible supports and motivations
- to my Dad for instilling the importance of hard work and higher education
- to my brothers and sisters for always being there for me

I dedicate the following Persian poem to my beloved parents...

Garche dar alam pedar darad maghami arjamn
Likan afzoon az pedar ghadro maghame MADAR ast
MADARe daanaa konad farzande daanaa tarbiat
Har ke bar har jaa resad az ehtemame MADAR ast

Chapter 1

Introduction

Probabilistic graphical models, including Bayesian (belief) networks and Markov networks (also known as Markov random fields) [41] are probabilistic models for representing the dependency among random variables. These networks use a graphical representation to model the interdependence among random variables. Bayesian networks are directed models representing the joint probability distribution (JPD) of a set of random variables in terms of conditional probability distributions (CPD): one CPD for each random variable given its parents in the directed model. This allows for a compact and natural representation. On the other hand, Markov networks are undirected models representing the joint probability of a set of random variables in terms of a set of potential functions, where each potential function is a non-negative real-valued function of a subset of random variables. In this work, we focus on directed models with conditional probability distributions.

CPDs in Bayesian networks are often represented as tables. The advantage of a tabular representation is that it is as general as possible in terms of representing conditional probabilities; however, it also has several disadvantages. One of the disadvantages of a tabular representation is that the number of parameters required to describe a CPD for a random variable grows exponentially with the number of parents it has in the Bayesian network.

Example 1. Consider a medical domain where whether someone has fever or not depends on ten different diseases. Representing this domain as a Bayesian network with tabular CPDs, the CPD for random variable *Fever* will have $2^{10} = 1024$ parameters (assuming all random variables are binary). The Bayesian network and the tabular CPD for this example are represented in Fig. 1.1 and Table 1.1 (D_i represents the i-th disease and $fever \equiv$ "$Fever = True$"). Not only it is computationally expensive to perform operations on this CPD, but also it is quite tiresome to acquire the probabilities from expert knowledge; experts will lose patience if we ask them 1024 questions. Learning such a table from data is also problematic because, given the standard ways for learning a tabular CPD from data, we cannot generalize from similar conditions. (example taken from [26])

Example 1 indicates how using a tabular representation of a CPD for a random variable may cause troubles. This example suggests considering other represen-

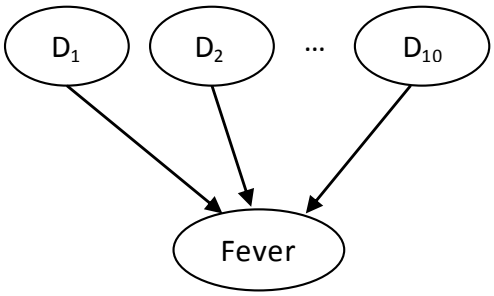

Figure 1.1: A Bayesian network for random variable *Fever* based on ten diseases (represented by $D_1, D_2, \ldots, D_{10}$) affecting the probability of someone having fever.

tations of CPDs in such particular situations. Tree-CPD and rule-CPD are two alternatives to the tabular representation of CPDs, offering a more compact representation by using the contextual independence of random variables [15, 50, 60]. Aggregation is another compact representation for CPDs, defining a CPD in terms of a function. Logistic regression [5, 32] is a form of aggregation which is used for representing the conditional probability of random variables having many parents (e.g., see [35, 53]). We will explain how it works in later chapters.

One of the shortcomings of probabilistic graphical models is that they are not often adequate to represent large and complex domains where there are entities in a variety of configurations. Furthermore, while they enable us to efficiently handle uncertainty, their representational power of a wide variety of knowledge is usually limited. Since first-order logic [54] gives a powerful and compact representation for knowledge, and knowledge representation and uncertainty management are two

Table 1.1: Conditional probability table for random variable *Fever* based on ten diseases (represented by $D_1, D_2, \ldots, D_{10}$) affecting the probability of someone having fever.

D_1	D_2	...	D_{10}	$Pr(fever \mid D_1, \ldots, D_{10})$
True	*True*	...	*True*	α_1
True	*True*	...	*False*	α_2
...	...	...	...	...
False	*False*	...	*False*	α_{1024}

key elements in most applications, combining probability with first-order logic has received a great deal of attention. Early works on this field include combining probability with Horn clauses [33, 44, 58], frame-based systems [11, 39], and database query languages [56]. These works have led to the introduction of what is known as relational probabilistic models which combine first-order logic with probabilistic models [1, 14].

Relational probabilistic models are models where there are probabilities about relations among individuals that can be specified independently of the actual individuals, and where the individuals are exchangeable; before we know anything about the individuals, they are treated identically. These models extend Bayesian networks and Markov networks by adding the concepts of objects, object properties, and relations.

Similar to Bayesian networks, in directed relational probabilistic models the joint probability is defined in terms of conditionals. One of the features of relational probabilistic models is that the conditional probability of a relation may depend on the number of individuals[1] (in relational models, we refer to the number of individuals as *population size*) [45, 48]. In such cases, we cannot represent the conditional probability of the relation in terms of a table.

Example 2. Suppose a group of people are invited to a party. Whether the party is fun or not depends on the number of invited people that attend the party (population size in this example refers to the number of invited people). Therefore, the number of parents that the random variable *FunParty* has is equal to the population size of people that have been invited to that party. This number, however, is not always fixed, because the number of people that are invited to different parties is not the same. Since we cannot bound the number of invited people to different parties, *FunParty* might have an unbounded number of parents, so a tabular representation of the conditional probability for this random variable is no longer a possible option.

Varying population sizes are quite common. They can appear in a number of ways including:

- The actual population may be arbitrary. For example, in considering the probability of someone committing a crime (which depends on how many other people could have committed the crime) [45] we could consider the population to be the population of the neighbourhood, the population of the city, the population of the country, or the population of the whole world. It

[1] Sometimes the dependence of a relation on the the number of individuals in a relational model is desirable; in other cases, model weights may need to change. See [20, 21] for more information.

would be good to have a model that does not depend on this arbitrary decision. We would like to be able to compare models which involve different choices.

- The population can change. For example, the number of people in a neighbourhood or in a school class may change. We would like a model to make reasonable predictions as the population changes. We would also like to be able to apply a model learned at one or a number of population sizes to different population sizes. For example, models from drug studies are acquired from very limited populations but are applied much more generally.

- The relevant populations can be different for each individual. For instance in Example 2, whether a party is fun for a person or not may depend on the number of people at that party who are friends with him or her; however, this number is different for different people at the party. We would like a model that makes reasonable predictions for diverse numbers of friends.

- The train and test populations may differ. For example, the efficiency of a new hospitality management may be first examined in a small hospital with few patients, and later used in large hospitals with many patients.

Variation in population sizes, and the way the predictions of a model change with it, is an important factor which should be taken into account when dealing with probabilistic relational models. Not only it can affect the correctness of a model [48], but also it has a great influence on the performance of different methods used for inference in relational models [24]. It also necessitates the use of representations other than tables in certain cases as described in Example 2. Aggregation is often used in relational models to handle the problem of representing conditional probabilities for relations with an unbounded number of parents.

In this work, we consider extending standard logistic regression as an aggregator for relational models and investigate how varying populations can cause problems for logistic regression. We propose a relational logistic regression model which addresses these problems and works appropriately for relational models.

1.1 Literature Review

Since their introduction, probabilistic relational models have drawn many researchers' attention. They have been used in different fields such as making recommendations [13, 17], clustering [57], security risk analysis [55], and sorting rocks [9].

A great deal of attention in probabilistic relational models has been drawn towards extending standard machine learning models of propositional data to work

for relational models. Neville et al. [37] and Blockeel and De Raedt [2] developed relational probability tree and relational regression tree models by extending standard decision trees and regression trees respectively. Considering the data is heterogeneous and interdependent, Neville et al. [37] and Blockeel and De Raedt [2] use aggregated values of random variables in the data such as mean, mode, max, count, etc. along with the values of random variables in each tuple of the data to split the data in tree nodes. Jaeger [19] extends Bayesian networks to relational domains by defining a language for specifying Bayesian networks whose nodes are extensions of first-order predicates.

Other extensions of propositional machine learning models to relational domains include relational dependency networks [36], relational Bayesian classifiers [37], relational Markov networks (or Markov logic networks) [10, 52], etc.

Some models have been proposed in the literature which can lead to learning of logistic regression models for relational data. For instance, Popescul et al. [51] use inductive logic programming (ILP) [28] to generate first-order rules for a target relation, create features by propositionalizing the rules, and then use logistic regression to learn a classifier based on these features. There are also methods for discriminative learning of Markov logic networks which can be considered as a logistic regression model with relational features. For instance, Huynh and Moony [18] use an ILP technique to generate discriminative clauses for a target relation and then use logistic regression with L1-regularizer to learn the weights with automatic feature selection (automatic structure learning). These methods are all designed for learning purposes and are not used as an aggregator in relational models similar to the way logistic regression is used for aggregation in propositional models. In this work, we propose a relational version of logistic regression which can be used both for learning and aggregation purposes and we discuss what can and cannot be done by our proposed model.

Aggregation in relational models is necessary when a variable has possibly an unbounded number of parents in the grounding. The use of aggregators for defining conditional probabilities in such situations has been investigated for many years and many aggregation methods have been proposed and used in the literature. Horsch and Poole [16] proposed using probabilistic existential and universal quantifiers to define a conditional probability (e.g. $Pr(b|\exists X, a(X)) = 0.7$ and $Pr(b|\neg\exists X, a(X)) = 0.05$). An existential quantifier model is equivalent to a logical OR operation whose value is *True* if a property holds for at least one individual, and a universal quantifier is equivalent to a logical AND operation whose value is *True* if a property holds for all individuals. Noisy-OR [40] and noisy-AND [8] are two of the common aggregators which are extensions of standard OR and AND operations. They define a set of noise parameters for each parent and the probability of the child being *True* increases according to those parameters when the

property holds for more parents. Generalized linear models [29] are a class of popular aggregators that satisfy the independence of causal influence. They consist of a function whose input is the values of the parents of a random variable and whose output is a real number, as well as a threshold on the output of the function determining if the child node is *True* or *False* according to the output. Logistic regression is a generalized linear model with soft threshold.

One of the important issues which should be taken into account when designing a relational model is the variations in the population sizes. These variations can have desirable or undesirable effects on predictions of model. Poole et al. [48] consider the population size effects on three simple models (naive Bayes, logistic regression with one parent, and a simple Markov network) and indicate that in a relational setting, even these simple models make strong assumptions about how the size of a population affects the predictions. This study has been extended later on by considering population size extrapolation [47]. In this work, we also consider the effects of population size and the problems they introduce for using logistic regression in relational domains and address those problems in our proposed relational logistic regression.

1.2 Perspective

In this work, we demonstrate the problems that arise when using standard logistic regression as an aggregator in relational models, where a variable has possibly an unbounded number of parents in the grounding. Considering these problems, we initially propose a *linear, single parent relational logistic regression* which solves the problems with standard logistic regression when there is only one (parametrized) parent and the decision threshold is linear. Then we demonstrate what happens when we have more parents that can interact with each other and when we want to model non-linear decision thresholds. We develop a general relational logistic regression which works with an arbitrary number of parents and models every arbitrary polynomial decision threshold. We define canonical forms and prove what can and cannot be modeled by our relational logistic regression. We also compare our relational logistic regression model with Markov logic networks, which are a class of undirected relational models, and point out the similarities and differences. We conclude the thesis by introducing many other popular aggregators and representing how we can approximate them using our relational logistic regression.

In this work, we focus on binary child variables and categorical parent variables, but explain a possible approach for extending the model to multi-valued child nodes. Extension of the model to continuous parent variables (as done in

[31] for non-relational models) is left as a future work.

The rest of the thesis is organized as follows. Chapter 2 provides sufficient information for readers to read the rest of the thesis and defines terminologies used in the thesis. Chapter 3 demonstrates problems with standard logistic regression when used as aggregator in relational models and defines relational logistic regression to overcome these problems. This chapter also compares relational logistic regression with Markov logic networks, and proves theorems about canonical forms and what can and cannot be represented by relational logistic regression. Chapter 4 represents how other well-known aggregation models can be approximated in terms of relational logistic regression. Finally, chapter 5 summarizes the thesis and points out some future directions.

Chapter 2

Background

In this chapter, we provide sufficient information for readers to read the rest of the thesis. We also define terminologies used throughout the thesis.

2.1 Bayesian Networks

Suppose we have a set of random variables $\{X_1, \ldots, X_n\}$. A **Bayesian network** or **belief network** [41] is a directed acyclic graph (DAG) where the random variables are the nodes, and the arcs represent interdependence among the random variables. Each variable is independent of its non-descendants given the values for its parents. Thus, if X_i is not an ancestor of X_j, then $Pr(X_i \mid parents(X_i), X_j) = Pr(X_i \mid parents(X_i))$, where $parents(X_i)$ returns the parents of the random variable X_i in the DAG. The joint probability of the random variables in a Bayesian network can be factorized as:

$$Pr(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} Pr(X_i \mid parents(X_i)) \tag{2.1}$$

Example 3. Fig. 2.1 represents a Bayesian network for *cancer* (network taken from [27]) with five random variables. In this network, *Cancer* (C) has two parents, *Pollution* (P) and *Smoke* (S), and two children, *Xray* (X) and *Dyspnea* (D). We can infer from the network that if we observe whether someone has cancer or not, the probability of them having dyspnea is independent of whether they smoke or not. In order to model the joint probability of the random variables in the network, five conditional probabilities are constructed; one for each random variable given its parents. The joint probability is then as follows:

$$Pr(C, P, S, X, D) = Pr(P) * Pr(S) * Pr(C \mid S, P) * Pr(X \mid C) * Pr(D \mid C)$$

One way to represent a conditional probability distribution $Pr(X_i \mid parents(X_i))$ in a Bayesian network is in terms of a table. Such a tabular representation for a random variable increases exponentially in size with the number of parents. For instance, a Boolean child having 10 Boolean parents requires $2^{10} = 1024$ numbers to specify the conditional probability (as in Fig. 1.1).

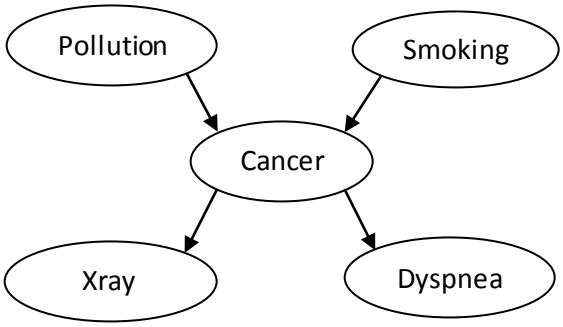

Figure 2.1: A Bayesian network for random variable *Cancer* representing few of its causes and few of the complications caused by it (taken from [27]).

A compact alternative to a table is an **aggregation** operator, or aggregator, that specifies a function of how the distribution of a variable depends on the values of its parents. Examples for common aggregators include OR, AND, as well as "noisy-OR" and "noisy-AND". These can be specified much more compactly than a table.

Example 4. Suppose in Fig. 1.1 we know that all diseases have the same effect on fever and whether someone has fever or not only depends on the number of diseases they have. We can model this conditional dependency of fever on its parents as follows:

$$Pr(fever \mid D_1, D_2, \ldots, D_{10}) = sign(w_0 + w_1 \sum_{i=1}^{10} D_i)$$

where $fever \equiv$ "$Fever = True$" and $sign(x)$ is equal to 1 if $x \geq 0$ and 0 otherwise. This model assumes the probability of *fever* given the diseases is either 0 or 1. Having the above model, if we know that people have fever as soon as they have one of the ten diseases, we can represent this by setting $w_0 = -1$ and $w_1 = 2$ (assuming that *True* is represented by 1 and *False* is represented by 0). This is a compact form requiring only 2 instead of 1024 parameters.

2.2 Logistic Regression

Logistic regression [5, 32] is an aggregator in Bayesian networks. We describe how it works and how it can be used as an agregator.

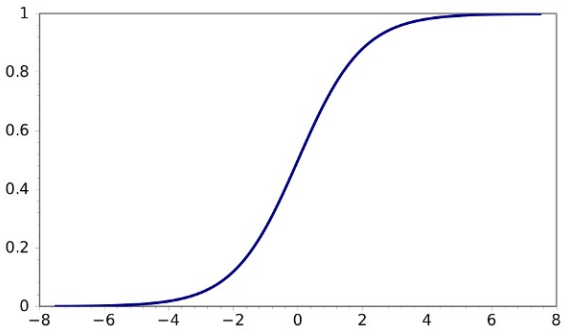

Figure 2.2: The sigmoid function.

Suppose a Boolean random variable Q is a child of the numerical random variables $\{X_1, X_2, \ldots, X_n\}$. Logistic regression is an aggregation operator defined as:

$$Pr(q \mid X_1, \ldots, X_n) = \text{sigmoid}(w_0 + \sum_i w_i X_i) \tag{2.2}$$

where $q \equiv$ "$Q = True$", $\text{sigmoid}(x) = 1/(1+e^{-x})$ (Fig. 2.2 shows a sigmoid function) and $w_0, w_1, \ldots, w_n$ are real-valued weights. It follows from the definition that $Pr(q \mid X_1, \ldots, X_n) > 0.5$ iff $w_0 + \sum_i w_i X_i > 0$. Logistic regression definition in Eq. 2.2 assumes numerical parameters, so Boolean inputs need to be mapped to numerical ones. There are many ways to do this; for now we assume *True* is represented by 1 and *False* is represented by 0.

The space of assignments to the w's so that $w_0 + \sum_i w_i X_i = 0$ is called the **decision threshold**, as it is the boundary of where $Pr(q \mid X_1, \ldots, X_n)$ changes between being closer to 0 and being closer to 1. Logistic regression provides a soft threshold, in that it changes from close to 0 to close to 1 in a continuous manner. How fast it changes can be adjusted by multiplying all weights by a positive constant.

Example 5. Suppose in Fig. 1.1 we know that if people have none of the ten diseases, the probability of them having fever is low; however, if they have at least one of the ten diseases, with a high probability they have fever, and this probability increases with the number of diseases they have. Below is an example of how we can model this conditional probability for fever using logistic regression.

$$Pr(fever \mid D_1, \ldots, D_{10}) = \text{sigmoid}(-3 + 5D_1 + 5D_2 + \cdots + 5D_{10})$$

Given the above conditional probability, the probability of having fever for a person with none of the ten diseases is $\text{sigmoid}(-3) \simeq 0.0474$. Once they have one of

the ten diseases, this probability becomes sigmoid$(-3+5) \simeq 0.8808$. This probability increases as the number of diseases increases. For instance, the probability of having fever for a person having two of the ten diseases is sigmoid$(-3+2*5) \simeq 0.9991$. The decision threshold for this example is $D_1 + D_2 + \cdots + D_{10} = \frac{3}{5}$, meaning that the probability of fever is more than a half if the sum of the indicators of diseases being true is more than $\frac{3}{5}$, less than a half if the sum of the indicators is less than $\frac{3}{5}$, and is exactly a half of the sum equals $\frac{3}{5}$.

2.2.1 The Factorization Perspective

A simple and general formulation of logistic regression can be defined using a multiplicative factorization of the conditional probability. Eq. 2.2 then becomes a special case, which is equivalent to the general case when Q is binary and probabilities are positive (non-zero).

We define a **general logistic regression** for Q with parents $X_1, \ldots, X_n$ (all variables here may be discrete or continuous) to be when $Pr(Q \mid X_1, \ldots, X_n)$ can be factored into a product of non-negative pairwise factors and a non-negative factor for Q:

$$Pr(Q \mid X_1, \ldots, X_n) \propto f_0(Q) \prod_{i=1}^{n} f_i(Q, X_i)$$

where $\propto$ *(proportional-to)* means it is normalized separately for each assignment to the parents. This differs from the normalization for joint distributions (as used in undirected models), where there is a single normalizing constant. Here the constraint that causes the normalization is $\forall X_1, \ldots, X_n : \sum_Q Pr(Q \mid X_1, \ldots, X_n) = 1$, whereas for joint distributions, the normalization is to satisfy the constraint $\sum_{Q, X_1, \ldots, X_n} Pr(Q, X_1, \ldots, X_n) = 1$.

If Q is binary, then:

$$Pr(q \mid X_1, \ldots, X_n) = \frac{f_0(q) \prod_{i=1}^{n} f_i(q, X_i)}{f_0(q) \prod_{i=1}^{n} f_i(q, X_i) + f_0(\neg q) \prod_{i=1}^{n} f_i(\neg q, X_i)}$$

If all factors are positive, we can divide and then use the identity $y = e^{\ln y}$:

$$\begin{aligned} Pr(q \mid X_1, \ldots, X_n) &= \frac{1}{1 + \frac{f_0(\neg q)}{f_0(\ \ q)} \prod_{i=1}^{n} \frac{f_i(\neg q, X_i)}{f_i(\ \ q, X_i)}} \\ &= \frac{1}{1 + exp\left(\ln \frac{f_0(\neg q)}{f_0(\ \ q)} + \sum_{i=1}^{n} \ln \frac{f_i(\neg q, X_i)}{f_i(\ \ q, X_i)}\right)} \\ &= \text{sigmoid}\left(\ln \frac{f_0(\ \ q)}{f_0(\neg q)} + \sum_{i=1}^{n} \ln \frac{f_i(\ \ q, X_i)}{f_i(\neg q, X_i)}\right). \end{aligned}$$

When the $\ln \frac{f_i(\ q,X_i)}{f_i(\neg q,X_i)}$ are linear functions w.r.t. X_i, it is possible to find values for all w's such that this can be represented by Eq. (2.2). This is always possible when the parents are binary.

2.2.2 Multi-valued Child Variables

Suppose a multi-valued categorical random variable Q, where Q can take $k \geq 2$ different values denoted by $\{V_1, \ldots, V_k\}$, is a child of the numerical random variables $\{X_1, X_2, \ldots, X_n\}$. Logistic regression learns $(k-1)(n+1)$ weights denoted by:

$$\begin{pmatrix} w_{10} & w_{11} & \cdots & w_{1n} \\ w_{20} & w_{21} & \cdots & w_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ w_{(k-1)0} & w_{(k-1)1} & \cdots & w_{(k-1)n} \end{pmatrix}$$

and defines the conditional probability of Q given its parents as:

$$\begin{aligned} if(l < k) \rightarrow Pr(Q = V_l | X_1, \ldots, X_n) &= \frac{exp(w_{l0} + \sum_{i=1}^{n} X_i w_{li})}{1 + \sum_{l'=1}^{k-1} exp(w_{l'0} + \sum_{i=1}^{n} X_i w_{l'i})} \\ if(l = k) \rightarrow Pr(Q = V_l | X_1, \ldots, X_n) &= \frac{1}{1 + \sum_{l'=1}^{k-1} exp(w_{l'0} + \sum_{i=1}^{n} X_i w_{l'i})} \end{aligned} \tag{2.3}$$

Note that the definition of logistic regression in Eq. 2.3 reduces to the Eq. 2.2 when $K = 2$.

2.3 Relational Models

Relational models deal with objects and relations among them. Non-relational (or propositional) models have a set of features and make predictions about a target feature based on those features. Relational models have a set of objects (also called individuals or entities), properties of the objects, and relations among these objects, and make predictions about target relations. The following subsections describe what relational models are and how they are used, and motivate why we should use relational models instead of non-relational models in certain domains.

2.3.1 Motivation

Bayesian networks, e.g., as shown in Fig. 1.1 and 2.1, are defined in terms of features (represented by nodes) and the probabilistic dependencies among them (represented by arcs). In some domains, we have a number of individuals, properties of individuals, and relationships among them and we want to make probabilistic

predictions about a random variable having a certain value, an individual having a property, or a group of individuals having a relationship.

Example 6. In social networks, we have a set of individuals (users of the social network), their properties (e.g., gender, age, etc.), and the relationship among them (e.g., being friend or following each other), and we might want to predict if a user is fake or real given their properties and relations (e.g. in [7, 59]).

These domains are best modeled in terms of individuals and relationships rather than in terms of features. The following example (inspired by two-digit addition example in [49]) motivates the use of individuals and relations in the domain of intelligent tutoring systems.

Example 7. Consider an automated tutoring system for diagnosing the arithmetic errors students make in converting a number in binary format to gray code[2]. The system should be able to predict if the students know XOR and the conversion process well enough or have problems in specific tasks.

Fig. 2.3 represents a simple case of two-bit binary to gray code conversion (B2G) and a corresponding Bayesian network for diagnosing whether the student knows the conversion process and how to XOR two bits or not. As we can see in this model, there is one node in the Bayesian network for each digit (X_i) of the binary number (plus one extra node which we observe to be zero), one for each digit (Y_i) of resulting gray code format, one representing whether the student knows XOR (*KnowsXOR*), and one representing whether the student knows the conversion process (*KnowsB2G*). If instead of having a two-bit conversion problem, our intelligent tutoring is teaching 10-bit conversion, we have to use ten nodes for the number in binary format and ten for the resulting gray code. Furthermore, if our system is teaching addition to several students, we need different copies of the *KnowsXOR*, *KnowsB2G* and Y_is for each student. We also need another copy of the X_is and Y_is for each conversion problem that a student solves. Having all these copies, our network will be very huge and it will be very time-consuming to perform operations on it.

The problem of facing with a huge Bayesian network having many copies of its nodes arises in this domain because we have many individuals (students, conversion problems and digits), individual properties (whether the students know how to *XOR* and the *B2G* process or not) and relations among individuals (the digits of the gray code calculated by students for conversion problems). Relational probabilistic models are an appropriate alternative to be used in such situations.

[2]Gray code is a binary numerical system where two successive values differ in only one bit. In order to convert a number in binary format to gray code, the d-th digit of the gray code is calculated by XOR-ing the d-th and (d+1)-th digits in the binary format, assuming there is an extra 0 to the left of the binary number.

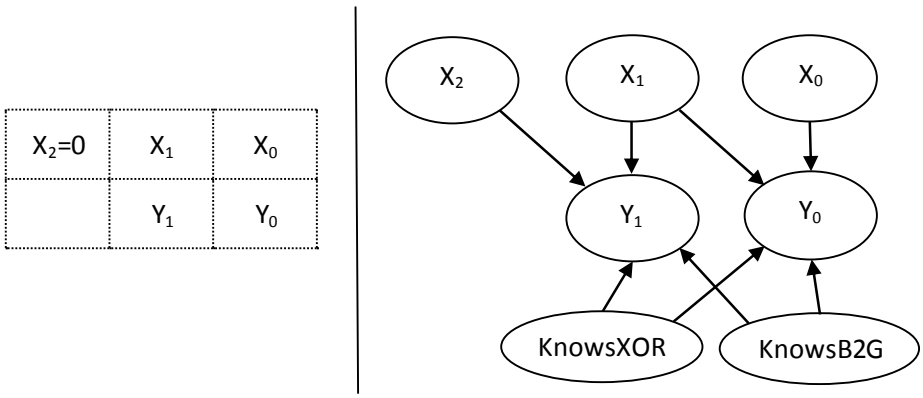

Figure 2.3: Two-digit binary to gray (B2G) conversion task and its corresponding Bayesian network for an intelligent tutoring system.

2.3.2 Relational Probabilistic Models

Relational probabilistic models [14] or template based models [26] extend Bayesian or Markov networks by adding the concepts of individuals (objects, entities, things), relations among individuals (including properties, which are relations of a single individual), and by allowing for probabilistic dependencies among these relations. In these models, individuals about which we have the same information are exchangeable, meaning that, given no evidence to distinguish them, they should be treated identically. We provide some basic definitions and terminologies in these models which are used in the rest of the thesis.

Some Definitions

A **population** is a set of **individuals**. A population corresponds to a domain in logic. The **population size** is the cardinality of the population which can be any non-negative integer. For instance, a population can be the set of movies in a movie rating system where *Titanic* and *Her* are two individuals and the size of the population is equal to the number of movies in the database.

A **logical variable** is written in lower case. Each logical variable is typed with a population; we use $|x|$ for the size of the population associated with a logical variable x. For instance, u and m may be two logical variables typed with the population of *users* and *movies* in a movie rating system respectively. Constants, denoting individuals, start with an upper case letter. We refer to a set of logical

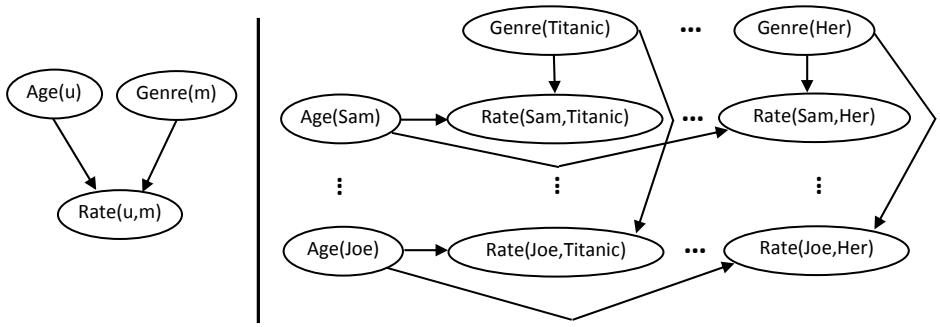

Figure 2.4: On the left is a relational belief network for predicting the rates of users for different movies given the users' ages and movies' genres, and on the right is a grounding of the model.

variables by a lower case letter in bold (e.g., $\mathbf{x}$).

A **parametrized random variable (PRV)** is of the form $F(t_1,\ldots,t_k)$ where F is a k-ary functor (a function symbol or a predicate) and each t_i is a logical variable or a constant. Each functor has a range, which is $\{True, False\}$ for predicate symbols. A PRV represents a set of random variables, one for each assignment of individuals to its logical variables. The range of the functor becomes the range of each random variable. For instance, $Rate(u,m)$, $Rate(Sam,m)$ and $Rate(Sam,Titanic)$ are three different PRVs.A **ground random variable** is a PRV where all t_is are constants (e.g. $Rate(Sam,Titanic)$).

A **relational belief network** is an acyclic directed graph where the nodes are PRVs and arcs represent the conditional independence among them. A **grounding** of a relational belief network with respect to a population for each logical variable is a belief network created by replacing each PRV with the set of random variables it represents, while preserving the structure. Fig. 2.4 represents a relational belief network on the left, where the rate given by a user u to a movie m depends on the age of the user and genre of the movie, and its grounding on the right.

An **atom** is an assignment of a value to a PRV. For instance, $R(x)$ is a PRV and $R(x) = True$ is an atom. For a Boolean PRV $R(x)$, we represent $R(x) = True$ by $R(x)$ and $R(x) = False$ by $\neg R(x)$. We refer to an atom $\neg R(x)$ as a negated atom.

A **formula** is made up of atoms with logical connectives. A Boolean formula is a formula in which conjunction, disjunction and negation are the only logical operators used. A conjunctive formula is a formula which is the conjunction of literals, where a literal is an atom or the negation of an atom. A disjunctive formula is a

formula which is the disjunction of literals. A positive formula is a formula with no negations. For instance $R(x) \wedge S(y)$ is a positive conjunctive Boolean formula. In this work, we only consider Boolean formulae and for simplicity we use the term *formula* to refer to *Boolean formula*.

A **substitution** is a finite set $\theta = \{x_1/t_1, x_2/t_2, \ldots, x_k/t_k\}$ where x_is are distinct logical variables and each t_i is a constant or a logical variable different from x_i. Let F be a formula. $F\theta$ is called an **instance** of F and obtained by replacing simultaneously all occurrences of every x_i, $1 \leq i \leq k$, in F with the corresponding t_i. θ is called a **unifier** for a set $\{F_1, F_2, \ldots, F_m\}$ iff $F_1\theta = F_2\theta = \cdots = F_m\theta$. θ is called the most general unifier for a set $\{F_1, F_2, \ldots, F_m\}$ iff any other unifier ψ of this set can be expressed as $\psi = \theta\theta'$, where θ' is a substitution. The set $\{F_1, F_2, \ldots, F_m\}$ is **unifiable** iff there exists a unifier for it. (Definitions taken from [6].)

A **Boolean interaction** among a series of PRVs is an interaction that can be represented by a Boolean formula of the PRVs.

When using a single population, we write the population as $A_1 \ldots A_n$, where n is the population size, and use $R_1 \ldots R_n$ as short for $R(A_1) \ldots R(A_n)$. We also use n_{val} for the number of individuals x for which $R(x) = val$. When $R(x)$ is binary, we use the shortened $n_T = n_{True}$ and $n_F = n_{False}$.

Intelligent Tutoring Example with Relational Models

Example 7 represented how modeling a domain with multiple individuals and relations among them can be troublesome for a standard Bayesian network. In order to model this problem in a relational setting, we define logical variables s, d and p typed with the population of *students*, *digits* and *problems* respectively. Having these logical variables, we define the digits of X by two PRVs $X(d,p)$ and $X(d+1,p)$, where the ground random variable $X(D,P)$ represents the D-th digit of the number in binary format from the P-th problem. We also define $KnowsXOR(s)$ and $KnowsB2G(s)$ where each ground random variable $KnowsXOR(S)$ represents whether student S knows how to XOR and $KnowsB2G(S)$ represents whether student S knows the binary to gray conversion process or not. Finally, we define the result variables as $Y(d,p,s)$, where the ground random variable $Y(D,P,S)$ represents the D-th digit of the result calculated by student S for the P-th problem. The relational belief network can be then represented by plate notation [4] as in Fig. 2.5. Having this relational model, there are several efficient algorithms for weight learning [11] and inference [30, 46] in these models which can be used to predict the value of each PRV given any of the other PRVs.

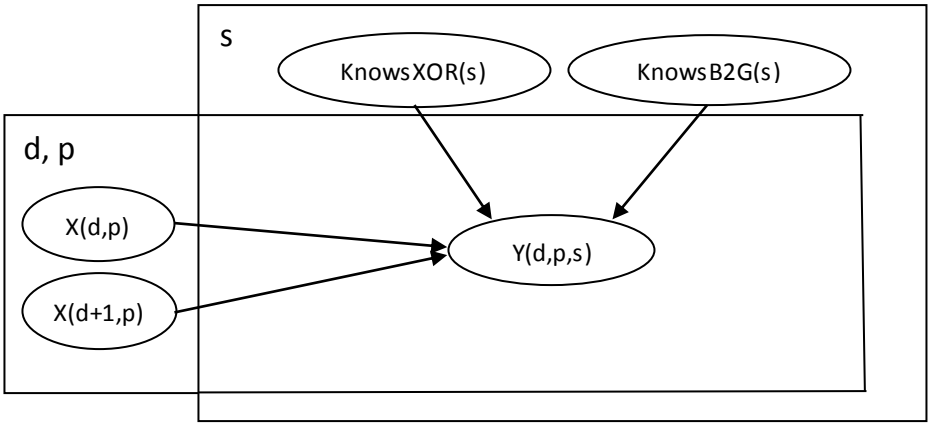

Figure 2.5: Intelligent tutoring system for teaching multi-digit binary to gray (B2G) conversion modeled in a relational setting using plates notation.

A Relational Model Requiring Aggregation

Fig. 2.6 (taken from [25]) represents a relational model and a grounding of the model. According to the model, the probability of a team having a substitute depends on the probability of substitution for any of the players, where the probability of substitution for each player depends on whether the players are injured in the game or not, and the probability of them having an injury depends on whether they are in shape or not. In this model, the conditional probability of the PRVs $Shape(player)$, $Inj(player)$ and $Sub(player)$ can be represented in terms of a table, because their parents do not have an extra logical variable, thus the number of parents they have in the grounding is fixed. However, the PRV $Subs()$ has a parent $Sub(player)$ which has an extra logical variable. Since the number of players of different teams and in different sports can be different, $Subs()$ may have an unbounded number of parents in the grounding. Therefore, the conditional probability for this PRV cannot be represented in terms of a table and we have to use aggregation instead. Logical OR (or equivalently an existential quantifier) is an appropriate aggregation operator for the PRV $Subs()$ because $Subs()$ is *True* if there is at least one individual P for which $Sub(P)$ is *True*.

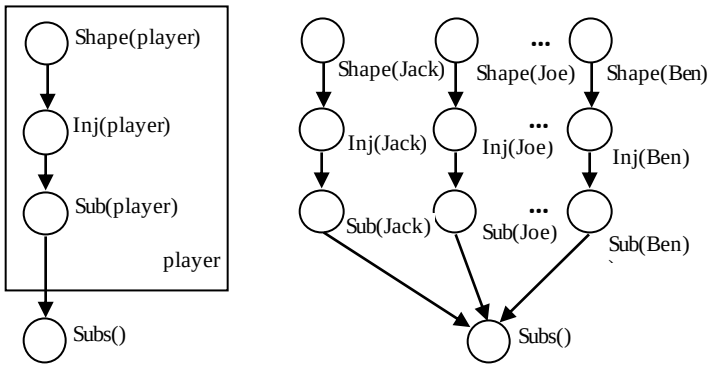

Figure 2.6: A relational model having a PRV ($Subs()$) with an unbounded number of parents in the grounding, whose conditional probability should be represented using an aggregation operator (taken from [25]).

2.3.3 Markov Logic Networks

Markov logic networks (MLNs) [10, 52] are undirected models for representing the joint probability distribution of a set of PRVs. MLNs extend the standard Markov networks to relational domains. They represent the joint probability of PRVs in terms of a first order knowledge-base with soft constraints. Before we explain how MLNs work, we need to give two definitions.

A *world* is an assignment of a value to each ground random variable. The number of worlds is exponential in the number of ground random variables.

A *first-order knowledge-base* [12] is a set of sentences or formulae in first-order logic.

A first-order knowledge-base can be viewed as a set of hard constraints on the possible worlds. A hard constraint means that if a world violates even one of the formulae in first-order knowledge-base, the probability of that world is zero. MLNs soften these hard constraints by making a world less probable (not impossible) when it violates a formula. Softening the constraint is by associating a weight to each of the formulae whose value is proportional to the amount of decrease in probability of a world violating this formula.

More formally, an MLN defines the probability distribution over worlds using a set of weighted formulae of the form $\langle F, w \rangle$, where F is a formula and w is the weight of the formula. The set of all weighted formulae of a model can be viewed as a first-order knowledge-base with soften rules. The probability of a world is proportional to the exponential of the sum of the weights of the instances of the

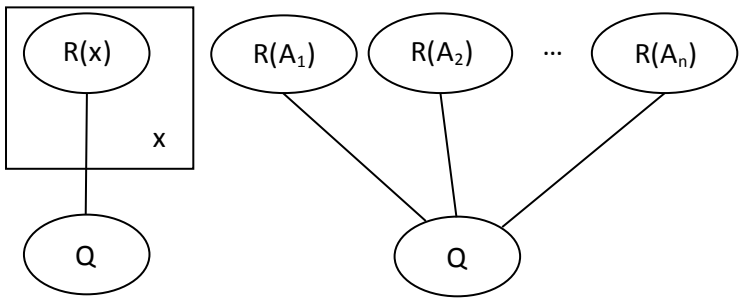

Figure 2.7: An undirected relational model in plate notation on the left, and its grounding for the population $\{A_1, A_2, \ldots, A_n\}$ on the right.

formulae that are *True* in the world. The probability of any formula is obtained by summing over the worlds in which the formula is *True*.

Example 8. Consider the undirected relational model in Fig. 2.7. An MLN for this model may define the probability distribution using the following weighted formulae:

$$\langle Q, \alpha_0 \rangle$$
$$\langle Q \wedge \neg R(x), \alpha_1 \rangle$$
$$\langle Q \wedge R(x), \alpha_2 \rangle$$
$$\langle R(x), \alpha_3 \rangle$$

The probability of any world can be calculated based on the number of instances of formulae that are *True* in the world.

MLNs can also be adapted to define conditional distributions. Below is an example of representing a conditional distribution using MLNs.

Example 9. Consider the undirected relational model in Fig. 2.7 and suppose the joint probability of the PRVs is defined using the weighted formulae in Example 8. Also suppose that the truth value of $R(x)$ for every individual x is observed. The MLN of Fig.2.7 defines the conditional probability of Q given the observed values of the $R(x)$ as follows:

$$Pr(q \mid obs) = sigmoid(\alpha_0 + n_F \alpha_1 + n_T \alpha_2) \tag{2.4}$$

where *obs* has $R(x)$ is true for n_T individuals, and false for n_F individuals out of a population of n individuals (so $n = n_T + n_F$), and $sigmoid(x) = 1/(1+e^{-x})$.

Chapter 3

Relational Logistic Regression

Logistic regression is an aggregator for standard Bayesian networks; however, it cannot be used as an aggregator for relational models without making appropriate changes. In this chapter, we indicate the problems of using logistic regression in relational domains and develop a relational version of logistic regression which we call *relational logistic regression*.

3.1 Aggregation with Logistic Regression in Relational Models

We saw earlier in this thesis that using an aggregator to define a conditional probability in standard Bayesian networks can save a noticeable amount of memory and reduce computations. While aggregation is optional in non-relational models, it is necessary in directed relational models whenever the parent of a PRV contains extra logical variables. For example, suppose Boolean PRV Q is a child of the Boolean PRV $R(x)$, which contains an extra logical variable, x, as in Fig. 3.1. In the grounding, Q is connected to n instances of $R(x)$, where n is the population size of x. For the model to be defined before n is known, it needs to be applicable for all values of n.

Common ways to aggregate the parents in relational domains, e.g. [11, 16, 25, 34, 38, 42], include logical operators such as *OR*, *AND*, *noisy-OR*, *noisy-AND*, as well as ways to combine probabilities.

Logistic regression, as described in previous chapter, may also be used for relational models. For instance for the model in Fig. 3.1, logistic regression defines the conditional probability of child node as:

$$Pr(q \mid R_1, \ldots, R_n) = \text{sigmoid}(w_0 + \sum_i w_i R_i). \tag{3.1}$$

Since the individuals in a relational model are exchangeable, w_i must be identical for all parents R_i (this is known as parameter-sharing or weight-tying), so Eq. 2.2 becomes:

$$Pr(q \mid R_1, \ldots, R_n) = \text{sigmoid}(w_0 + w_1 \sum_i R_i). \tag{3.2}$$

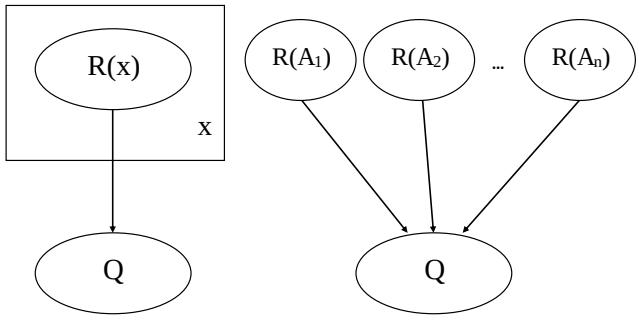

Figure 3.1: Logistic regression (with i.i.d. priors for the $R(x)$). The left side is the relational model in plate notation and the right side is the grounding for the population $\{A_1, A_2, \ldots, A_n\}$.

Consider what happens with a relational model when n is not fixed.

Example 10. Suppose we want to represent "Q is True if and only if R is True for 5 or more individuals", i.e., $q \equiv \left(|\{i : R_i = True\}| \geq 5\right)$ or $q \equiv (n_T \geq 5)$, using a logistic regression model $(Pr(q) \geq 0.5) \equiv (w_0 + w_1 \sum_i R_i \geq 0)$, which we fit for a population of 10. Consider what this model represents when the population size is 20.

If $R = False$ is represented by 0 and $R = True$ by 1, this model will have $Q = True$ when R is true for 5 or more individuals out of the 20. It is easy to see this, as $\sum_i R_i$ only depends on the number of individuals for which R is *True*.

However, if $R = False$ is represented by -1 and $R = True$ by 1, this model will have $Q = True$ when R is *True* for 10 or more individuals out of the 20. The sum $\sum_i R_i$ depends on how many more individuals have R *True* than have R *False*.

If $R = True$ is represented by 0 and $R = False$ by any other value, this model will have $Q = True$ when R is *True* for 15 or more individuals out of the 20. The sum $\sum_i R_i$ depends on how many individuals have R *False*.

While the choice of representation for *True* and *False* was arbitrary in the non-relational case, in the relational case different parametrizations can result in different decision thresholds as a function of the population. Table 3.1 gives some numerical representations for *False* and *True*, with corresponding parameter settings (w_0 and w_1), such that all regressions represent the same conditional distribution for $n = 10$. However, for $n = 20$, the predictions are different.

The decision thresholds in all situations in Table 3.1 are linear functions of population size. It is straightforward to prove the following proposition:

Table 3.1: Predictions of a logistic regression model in a relational domain with a population size of 20 as a function of numerical representations for *True* and *False*, where the parameters are learned for a population of 10.

False	*True*	w_0	w_1	**Prediction for** $n = 20$
0	1	-4.5	1	$Pr(Q = True) > 0.5$ iff $n_T \geq 5$
-1	1	0.5	0.5	$Pr(Q = True) > 0.5$ iff $n_T \geq 10$
-1	2	$-\frac{7}{6}$	$\frac{1}{3}$	$Pr(Q = True) > 0.5$ iff $n_T \geq 8$
-1	0	5.5	1	$Pr(Q = True) > 0.5$ iff $n_T \geq 15$
-1	100	$-\frac{889}{202}$	$\frac{1}{101}$	$Pr(Q = True) > 0.5$ iff $n_T \geq 5$
1	2	-14.5	1	$Pr(Q = True) > 0.5$ iff $n_T \geq 0$
-100	1	$\frac{1091}{202}$	$\frac{1}{101}$	$Pr(Q = True) > 0.5$ iff $n_T \geq 15$

Proposition 1. *Let $R = False$ be represented by the number α and $R = True$ by $\beta \neq \alpha$. Then, for fixed w_0 and w_1 (e.g., learned for one specific population size), the decision threshold for a population of size n is*

$$\frac{w_0}{w_1(\alpha - \beta)} + \frac{\alpha}{\alpha - \beta} n.$$

Proof. Let n_T represent the number of individuals for which the parent is *True* and n represent the population size. Considering the values for w_0, w_1, α and β, the decision threshold can be determined by solving the following equation for n_T (we assume $w_1 \neq 0$, otherwise the child does not depend on the parent).

$$\begin{aligned} & w_0 + w_1 (\beta n_T + \alpha(n - n_T)) = 0 \\ \Rightarrow & \frac{w_0}{w_1} + n_T(\beta - \alpha) + \alpha n = 0 \\ \Rightarrow & n_T = \frac{w_0}{w_1(\alpha - \beta)} + \frac{\alpha}{\alpha - \beta} n \end{aligned}$$

□

What is important about this proposition is that the way the decision threshold changes with the population size n, i.e., the coefficient $\frac{\alpha}{\alpha - \beta}$, does not depend on data (which affects the weights w_0 and w_1), but only on the arbitrary choice of the numerical representation of R.

Thus, Eq. (3.2) with a specific numeric representation of *True* and *False* is only able to model one of the dependencies of how predictions depend on population size, and so cannot properly fit data that does not adhere to that dependence.

We need an additional degree of freedom to get a relational model that can model any linear dependency on n, regardless of the numerical representation.

3.2 Single-parent, Linear Relational Logistic Regression

We define a single-parent, linear relational logistic regression by adding a degree of freedom to the logistic regression formalism in Eq. 3.2.

Definition 1. Let Q be a Boolean PRV with a single parent $R(\mathbf{x})$, where $\mathbf{x}$ is the set of logical variables in R that are not in Q (so we need to aggregate over $\mathbf{x}$). A **single-parent, linear relational logistic regression (SPL-RLR)** for Q with parents $R(\mathbf{x})$ is of the form:

$$Pr(q \mid R(A_1),\ldots,R(A_n)) = \text{sigmoid}\left(w_0 + w_1 \sum_i R_i + w_2 \sum_i (1-R_i)\right) \tag{3.3}$$

where R_i is short for $R(A_i)$ (A_i is the i-th assignment of individuals to $\mathbf{x}$), and is treated as 1 when it is *True* and 0 when it is *False*. Note that $\sum_i R_i$ is the number of (tuple of) individuals for which R is *True* ($= n_T$) and $\sum_i (1-R_i)$ is the number of (tuple of) individuals for which R is *False* ($= n_F$).

An alternative but equivalent parametrization is:

$$Pr(q \mid R(A_1),\ldots,R(A_n)) = \text{sigmoid}(w_0 + w_2 \sum_i 1 + w_3 \sum_i R_i) \tag{3.4}$$

where 1 is a function that has value 1 for every individual, so $\sum_i 1 = n$. The mapping between these parametrizations is $w_3 = w_1 - w_2$; w_0 and w_2 are the same.

Proposition 2. *Let $R = False$ be represented by α and $R = True$ by $\beta \neq \alpha$. Then, for fixed w_0, w_2 and w_3 in Eq. (3.4), the decision threshold for a population of size n is*

$$\frac{w_0}{w_3(\alpha-\beta)} + \frac{\alpha + w_2/w_3}{\alpha-\beta} n.$$

Proof. Let n_T represent the number of individuals for which the parent is *True* and n represent the population size. Considering the values for w_0, w_2, w_3, α and β, the decision threshold can be determined by solving the following equation for n_T (we assume $w_3 \neq 0$, otherwise the child does not depend on the values of the individuals in the parent).

$$\begin{aligned} & w_0 + w_2 n + w_3 (\beta n_T + \alpha(n - n_T)) = 0 \\ \Rightarrow & \frac{w_0}{w_3} + \frac{w_2}{w_3} n + n_T(\beta - \alpha) + \alpha n = 0 \\ \Rightarrow & n_T = \frac{w_0}{w_3(\alpha-\beta)} + \frac{\alpha + w_2/w_3}{\alpha-\beta} n \end{aligned}$$

□

Proposition 2 implies that the way the decision threshold in an SPL-RLR grows with the population size n, i.e. the coefficient $\frac{\alpha+w_2/w_3}{\alpha-\beta}$, depends on the weights. Moreover, for fixed α and β, with $\alpha \neq \beta$, any linear function of population can be modeled by varying the weights. This was not true for the traditional logistic regression.

For the rest of this thesis, when we embed logical formulae in arithmetic expressions, we take *True* formulae to represent 1, and *False* formulae to represent 0. Thus $\sum_L F$ is the number of assignments to the variables L for which formula F is *True*.

3.3 Multi-parent, Linear Relational Logistic Regression

The SPL-RLR proposed in Definition 1 can be extended to multiple (parametrized) parents by having a different pair of weights ((w_1, w_2) or (w_2, w_3)) for each parent PRV. This is similar to the non-relational logistic regression, where each parent has a (single) different weight. We define a *multi-parent, linear relational logistic regression (MPL-RLR)* as follows:

Definition 2. Let Q be a Boolean PRV with parents $R_1(\mathbf{x_1}), R_2(\mathbf{x_2}), \ldots, R_k(\mathbf{x_k})$, where $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_k}$ are the set of logical variables in $R_1, R_2, \ldots, R_k$ respectively that are not in Q (so we need to aggregate over them). A **multi-parent, linear relational logistic regression (MPL-RLR)** for Q with parents $R_1(\mathbf{x_1}), R_2(\mathbf{x_2}), \ldots, R_k(\mathbf{x_k})$ is of the form:

$$\begin{aligned} Pr(q \mid R_1(\mathbf{x_1}), \ldots, R_k(\mathbf{x_k})) = \text{sigmoid}\big(&w_0 \\ &+ w_{11} \sum_{\mathbf{x_1}} R_1(\mathbf{x_1}) + w_{21} \sum_{\mathbf{x_1}} (1 - R_1(\mathbf{x_1})) \\ &+ w_{12} \sum_{\mathbf{x_2}} R_2(\mathbf{x_2}) + w_{22} \sum_{\mathbf{x_2}} (1 - R_2(\mathbf{x_2})) \\ &+ \ldots \\ &+ w_{1k} \sum_{\mathbf{x_k}} R_k(\mathbf{x_k}) + w_{2k} \sum_{\mathbf{x_k}} (1 - R_k(\mathbf{x_k}))\big) \end{aligned} \tag{3.5}$$

This formulation of MPL-RLR needs $2k+1$ parameters, where k is the number of parents. Similar to SPL-RLR, we can use an alternative representation for MPL-

RLR as follows:

$$\begin{aligned} Pr(q \mid R_1(\mathbf{x_1}),\ldots,R_k(\mathbf{x_k})) = \text{sigmoid}\big(&w_0 \\ &+ w_{21}\sum_{\mathbf{x_1}} 1 + w_{31}\sum_{\mathbf{x_1}} R_1(\mathbf{x_1}) \\ &+ w_{22}\sum_{\mathbf{x_2}} 1 + w_{32}\sum_{\mathbf{x_2}} R_2(\mathbf{x_2}) \\ &+ \ldots \\ &+ w_{2k}\sum_{\mathbf{x_k}} 1 + w_{3k}\sum_{\mathbf{x_k}} R_k(\mathbf{x_k})\big) \end{aligned} \tag{3.6}$$

where 1 is a function that has value 1 for every individual. The mapping between these parametrizations is $\forall i, w_{3i} = w_{1i} - w_{2i}$; w_0 and w_{2i} are the same. Similar to the previous parametrization in Eq. 3.5, this parametrization also needs $2k+1$ parameters. In cases where parents have the same logical variable, however, this parametrization can be more compact.

Suppose $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_r}$ are the sets of logical variables of the parents of Q where $r \leq k$. Eq.3.7 can be re-written with only $k+r+1$ parameters as follows:

$$\begin{aligned} Pr(q \mid R_1(\mathbf{x_1}),\ldots,R_k(\mathbf{x_k})) = \text{sigmoid}\big(&w_0 \\ &+ w_{21}\sum_{\mathbf{x_1}} 1 + \cdots + w_{2r}\sum_{\mathbf{x_r}} 1 \\ &+ w_{31}\sum_{\mathbf{x_{R_1}}} R_1(\mathbf{x_1}) + \cdots + w_{3k}\sum_{\mathbf{x_{R_k}}} R_k(\mathbf{x_k})\big) \end{aligned} \tag{3.7}$$

where $\mathbf{x_{R_j}}$ represents the set of logical variable of the parent R_j.

3.4 Interactions Among Parents

Definition 2 represents how SPL-RLR can be extended to multiple parents with no interactions. However, there are cases where we want to model the interactions among the parents.

Example 11. Suppose we want to model whether someone being happy depends on the number of their friends that are kind. We assume the PRV $Happy(x)$ has as parents $Friend(y,x)$ and $Kind(y)$ as demonstrated in Fig. 3.2. Note that the number of friends for each person can be different.

Consider the following hypotheses:

1. A person is happy as long as they have 5 or more friends who are kind.

$$happy(x) \equiv |\{y : Friend(y,x) \wedge Kind(y)\}| \geq 5$$

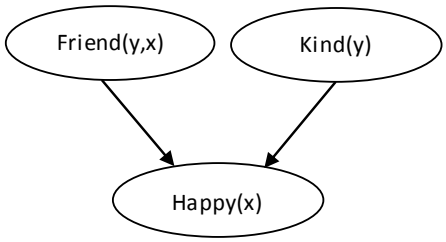

Figure 3.2: A relational model for representing people's happiness based on the number of friends they have and whether their friends are kind or not.

2. A person is happy if half or more of their friends are kind.

$$happy(x) \equiv |\{y : Friend(y,x) \wedge Kind(y)\}| \geq |\{y : Friend(y,x) \wedge \neg Kind(y)\}|$$

3. A person is happy as long as fewer than 5 of their friends are not kind.

$$happy(x) \equiv |\{y : Friend(y,x) \wedge \neg Kind(y)\}| < 5$$

These three hypotheses coincide for people with 10 friends, but make different predictions for people with 20 friends.

All three hypotheses are based on the interaction between the two parents, each requiring the number of individuals for which some formulae of the parents (instead of just a single parent) holds. For instance modeling the first hypothesis needs the number of people that are simultaneously friends with x and are kind. MPL-RLR does not include formulae of the parents and considers each parent separately. We cannot count the number of individuals that are simultaneously friends with x and are kind by having two numbers one indicating the number of people that are friends with x and the other indicating the number of people that are kind. Therefore, MPL-RLR cannot model these interactions without including weighted formuale of the parents in its parametrization. In order to model such aggregators, we need to extend MPL-RLR by adding such weighted formulae.The following extended MPL-RLR models these cases:

$$\begin{aligned} &Pr(happy(x) \mid \Pi) \\ &\quad = \text{sigmoid}\Big(w_0 + w_1 \sum_y Friend(y,x) \wedge \quad Kind(y) \\ &\qquad\qquad + w_2 \sum_y Friend(y,x) \wedge \neg Kind(y) \Big) \end{aligned} \tag{3.8}$$

where Π is a complete assignment of *friend* and *kind* to the individuals, and the right hand side is summing over the propositions in Π for each individual. To model each of the above three cases, we can set w_0, w_1, and w_2 in Eq. (3.8) as follows:

1. Let $w_0 = -4.5$, $w_1 = 1$, $w_2 = \quad 0$
2. Let $w_0 = \quad 0.5$, $w_1 = 1$, $w_2 = -1$
3. Let $w_0 = \quad 5.5$, $w_1 = 0$, $w_2 = -1$

Going from Eq. (3.3) to Eq. (3.4) allowed us to only model the positive cases in SPL-RLR. We can also model this example using only positive formulae by replacing:

$$w_2 \sum_y \mathit{Friend}(y,x) \wedge \neg \mathit{Kind}(y)$$

with:

$$w_2 \sum_y \mathit{Friend}(y,x) - w_2 \sum_y \mathit{Friend}(y,x) \wedge \mathit{Kind}(y)$$

in Eq. 3.8 resulting in:

$$\begin{aligned} &Pr(\mathit{happy}(x) \mid \Pi) \\ &\quad = \text{sigmoid}\Big(w_0 + (w_1 - w_2) \sum_y \mathit{Friend}(y,x) \wedge \mathit{Kind}(y) + w_2 \sum_y \mathit{Friend}(y,x) \Big) \end{aligned} \tag{3.9}$$

Example 11 represented that Boolean formulae of the parents may be required to be considered when parents interact with each other. It also represented a particular case where we can model the domain using only positive formulae. We can use a similar construction for more general cases:

Example 12. Suppose a PRV Q is a child of PRVs $R(x)$ and $S(x)$ and its conditional probability depends on a conjunctive formula of the parents such as $R(x) \wedge S(x)$, $R(x) \wedge \neg S(x)$, $\neg R(x) \wedge S(x)$ or $\neg R(x) \wedge \neg S(x)$. As in Example 11, we need to count the number of instances of a formula that are *True* in an assignment to the parents. It turns out that in this case $R(x) \wedge S(x)$ is the only non-atomic formula required to model the interactions between the two parents, because other conjunctive interactions can be represented using this count as follows:

$$\begin{aligned} \sum_x \quad R(x) \wedge \neg S(x) &= \sum_x R(x) - \sum_x R(x) \wedge S(x) \\ \sum_x \neg R(x) \wedge \quad S(x) &= \sum_x S(x) - \sum_x R(x) \wedge S(x) \\ \sum_x \neg R(x) \wedge \neg S(x) &= |x| - \sum_x R(x) - \sum_x S(x) + \sum_x R(x) \wedge S(x) \end{aligned}$$

with $|x| = \sum_x \mathit{True}$.

Example 12 shows that the positive conjunction of the two interacting parents is the only formula required to compute arbitrary conjunctive formulae consisting of one atom of each parent. In more complicated cases, however, subtle changes to the representation may be required.

Example 13. Suppose a PRV Q is a child of PRVs $R(x,y)$ and $S(x,z)$. Suppose we want to represent "Q is *True* if and only if $R(x,y) \wedge \neg S(x,z)$ is *True* for more than t triples $\langle x,y,z \rangle$". If we follow what we did in Example 12 to represent $R(x,y) \wedge \neg S(x,z)$ in terms of positive formulae, we will get that $\sum_{x,y,z} R(x,y) \wedge \neg S(x,z) = \sum_{x,y} R(x,y) - \sum_{x,y,z} R(x,y) \wedge S(x,z)$. However, we need the number of triples $\langle x,y,z \rangle$, instead of the number of pairs $\langle x,y \rangle$, for which $R(x,y)$ is *True*. We thus need to use $\sum_{x,y,z} R(x,y)$ as the number of assignments to x, y and z for which $R(x,y)$ is *True* as follows:

$$\sum_{x,y,z} R(x,y) \wedge \neg S(x,z) = \sum_{x,y,z} R(x,y) - \sum_{x,y,z} R(x,y) \wedge S(x,z)$$

So as part of the representation, we need to include the set of logical variables and not just a weighted formula.

3.5 Non-linear Decision Thresholds

Examples 12 and 13 suggest how to model interactions among the parents. Now consider the case where the decision threshold for the child PRV is a non-linear function of its parents' population sizes. For instance, if the individuals are the nodes in a dense graph, some properties of arcs grow with the square of the population of nodes. We describe how MLNs represent a class of non-linear decision thresholds for undirected models and use an analogous idea in our relational logistic regression.

Markov logic networks (MLNs) as described earlier are undirected models for representing the joint probability of a set of PRVs, which can be also adapted to define a conditional distribution. One of the characteristics of MLNs is that they can also model a class of non-linear dependencies on the population sizes. The following example shows a case where a non-linear conditional distribution is modeled by MLNs.

Example 14. Consider the MLN for PRVs Q and $R(x)$, consisting of a single formula $Q \wedge R(x) \wedge R(y)$ with weight w, where y represents the same population as x. The probability of q given observations of $R(A_i)$ for all A_i has a quadratic decision threshold:

$$Pr(q \mid R(A_1), \ldots, R(A_n)) = \text{sigmoid}(w\, {n_T}^2).$$

More formally, MLNs use Boolean formulae with more than one instance of a PRV (each having a different logical variable typed with the same population) to model a non-linear decision thresholds on the population size of the PRV. This allows for modeling a class of non-linear dependencies using only Boolean formulae. This idea can be also used by relational logistic regression. Consider the following example:

Example 15. Suppose a PRV Q is a child of the PRV $R(x)$, and we want to represent "Q is *True* if and only if ${n_T}^2 > n_F$". This dependency can be represented by introducing a new logical variable x' with the same population as x and treating $R(x')$ as if it were a separate parent of Q. Then we can use the interaction between $R(x)$ and $R(x')$ to represent the model in this example as:

$$\sum_{x,x'} R(x) \wedge R(x') - \sum_{x} True + \sum_{x} R(x).$$

3.6 Using Weighted Formulae for Relational Logistic Regression

The previous section represented how the idea of representing non-linear dependencies in MLNs can be also used by relational logistic regression. MLNs define the joint probability of PRVs in terms of a set of weighted formulae, which makes the model more intuitive. We can also define our relational logistic regression in terms of weighted formulae, but for representing a conditional probability distribution instead of the joint distribution. Each of the sigmas in our definitions of SPL-RLR and MPL-RLR can be represented by a weighted formula, and then we take the sigmoid of the sum of these sigmas.

Example 16. The three sigmas used in Example 15 for representing a non-linear decision threshold with relational logistic regression can be represented by the following weighted formulae:

- $\sum_{x,x'} R(x) \wedge R(x') \Rightarrow \langle R(x) \wedge R(x'), 1 \rangle$
- $\sum_{x} True \Rightarrow \langle True, -1 \rangle$
- $\sum_{x} R(x) \Rightarrow \langle R(x), 1 \rangle$

Using these weighted formulae to represent the conditional probability, our relational logistic regression will be the directed analog of MLNs. We will discuss this in more detail after we define our general relational logistic regression.

3.7 General Relational Logistic Regression

Previous examples show the potential for using relational version of logistic regression as an aggregator for relational models. We need a language for representing aggregation in relational models in which we can address the problems mentioned. We propose a generalized form of relational logistic regression as a directed analog of MLNs, which works for multi-parent cases and can model a same class of non-linear decision thresholds. We first give a formal definition of weighted formulae used by relational logistic regression and then define relational logistic regression based on these weighted formulae.

Definition 3. A **weighted formula (WF)** for a Boolean PRV $Q(\mathbf{x})$, where $\mathbf{x}$ is a tuple of logical variables, is a triple $\langle L, Q(\mathbf{x}') \wedge F', w \rangle$ where L is a set of logical variables such that $L \cap \mathbf{x}' = \{\}$, $Q(\mathbf{x}')$ is an instance of $Q(\mathbf{x})$, F' is a formula of parent PRVs of Q such that each logical variable in F' either appears in $\mathbf{x}'$ or is in L , and w is a weight.

A child PRV can have a set of WFs. We represent the set of WFs for $Q(\mathbf{x})$ by WFs_Q.

Example 17. Suppose $Q(x,y)$ is a child of PRVs $R(x,z)$ and $S(y)$. The following are all valid WFs:

- $\langle \{\}, Q(x,y), 1 \rangle$
- $\langle \{y'\}, Q(x,y) \wedge S(y) \vee S(y'), 5 \rangle$
- $\langle \{z\}, Q(x,x) \wedge R(x,z), -2 \rangle$
- $\left\langle \{x,y,z\}, Q(X_i,Y_j) \wedge S(y) \wedge R(X_i,z), \frac{3}{5} \right\rangle$

where X_i and Y_j are two individuals from the population assigned to x and y respectively. The following WFs, however, are not valid WFs because the first one has logical variable z in its formula which does not appear in the set of logical variables, the second one does not have an instance of the child PRV Q, and the third one has two instances of Q.

- $\langle \{\}, Q(x,y) \wedge R(x,z), 1 \rangle$
- $\langle \{x,y\}, S(y), 5 \rangle$
- $\langle \{x,y\}, Q(x,y) \wedge Q(X_i,Y_j) \wedge S(y), 5 \rangle$

Definition 4. A weighted formula $\langle L, Q' \wedge F', w \rangle$ for PRV $Q(\mathbf{x})$, where Q' represents an instance of Q and F' represents a formula of the parents of Q, is **compatible** with a ground random variable $Q(\mathbf{X})$, where $\mathbf{X}$ is a tuple of individuals,if Q' and $Q(\mathbf{X})$ are unifiable. We represent the set of WFs for Q that are compatible with $Q(\mathbf{X})$ by $comp(WFs_Q, X)$.

Example 18. Consider a PRV $Q(x,y)$ and a ground random variable $Q(X_i, Y_j)$. WFs with the following formulae are compatible with this ground random variable (F' shows any Boolean formula of the parents):

- $Q(x,y) \wedge F'$
- $Q(x, Y_j) \wedge F'$
- $Q(X_i, Y_j) \wedge F'$

and the following are not compatible:

- $Q(X_{i'}, y) \wedge F'$
- $Q(X_i, Y_{j'})$

Definition 5. Let $Q(\mathbf{x})$ be a Boolean PRV with parents $R_i(\mathbf{x_i})$, where $\mathbf{x_i}$ is the tuple of logical variables in R_i. A (general) **relational logistic regression (RLR)** for Q with parents $R_i(\mathbf{x_i})$ is defined using a set of WFs WFs_Q for Q as:

$$Pr(q(\mathbf{X}) \mid \Pi) = \text{sigmoid}\Big(\sum_{\langle L, Q' \wedge F', w \rangle \in comp(WFs_Q, \mathbf{X})} w \sum_{L} F'_{\Pi} \theta(Q', Q(\mathbf{X})) \Big)$$

where Π represents the assigned values to parents of $Q(\mathbf{x})$, $\mathbf{X}$ represents a tuple of individuals, $\langle L, Q' \wedge F', w \rangle$ represents a WF for Q where Q' is an instance of Q and F' is a formula of the parents of Q, $comp(WFs_Q, \mathbf{X})$ represents all weighted formulae for Q compatible with $\mathbf{X}$, $\theta(Q', Q(\mathbf{X}))$ represents the most general unifier of Q' and $Q(\mathbf{X})$, and $F'_{\Pi}\theta(Q', Q(\mathbf{X}))$ is formula F' with substitution $\theta(Q', Q(\mathbf{X}))$ applied to it, and evaluated in Π. (The first summation is over the set of WFs; the second summation is over the tuples of L. Note that $\sum_{\{\}}$ sums over a single instance.)

The SPL-RLR (Definition 1) is a subset of Definition 5, because the terms of Eq. (3.4) can be modeled as follows:

- w_0 can be represented by $\langle \{\}, Q, w_0 \rangle$
- $w_2 \sum_i 1$ can be represented by $\langle \{\mathbf{x}\}, Q, w_2 \rangle$

- $w_3 \sum_i R_i$ can be represented by $\langle \{\mathbf{x}\}, Q \wedge R(\mathbf{x}), w_3 \rangle$

RLR then sums these WFs, resulting in:

$$\begin{aligned} Pr(q \mid \Pi) &= \text{sigmoid}\left(w_0 \sum_{\{\}} \mathit{True} + w_2 \sum_{\{\mathbf{x}\}} \mathit{True} + w_3 \sum_{\{\mathbf{x}\}} R(\mathbf{x})\right) \\ &= \text{sigmoid}\left(w_0 + w_2 n + w_3 \sum_i R_i\right). \end{aligned}$$

It is straight forward to see that MPL-RLR (Definition 2) is also a subset of Definition 5.

Example 19. Consider the problem introduced in Example 13. Using general RLR (Definition 5), we can model the conditional probability of Q using the following WFs:

$$\langle \{\}, Q, w_0 \rangle$$
$$\langle \{x, y, z\}, Q \wedge R(x, y) \wedge \neg S(y, z), w_1 \rangle$$

Or alternatively:

$$\langle \{\}, Q, w_0 \rangle$$
$$\langle \{x, y, z\}, Q \wedge R(x, y), w_1 \rangle$$
$$\langle \{x, y, z\}, Q \wedge R(x, y) \wedge S(y, z), -w_1 \rangle$$

3.8 RLR vs MLNs

The definition of WFs in Definition 3 is similar to the WFs used by MLNs. The major difference is that we allow exactly one instance of the child node in the formula and it should be conjoined with the formula of the parents, whereas MLNs allow arbitrary numbers of each PRV in the formulae. The other minor difference is that we represent the set of logical variables L to be summed over explicitly, whereas MLNs have it implicitly. Instead of summing over the logical variables in a set L, MLNs sum over the logical variables appearing in the formula. We allow extra logical variables that are not in the formula to appear L and the formula sums over these extra logical variables to calculate the value of WFs in Definition 5. In order to sum over an extra logical variable z in MLNs, one could conjoin a $\mathit{True}(z)$ to the formula, where *True* is a property which holds for all individuals. In this section, we use explicit set of logical variables in weighted formulae of MLNs.

We demonstrate that RLR is directed analog of MLNs using the following example:

Example 20. Consider the model in Example 8 where we had the following weighted formulae:

$$\langle\{\}, Q, \alpha_0\rangle$$
$$\langle\{x\}, Q \wedge \neg R(x), \alpha_1\rangle$$
$$\langle\{x\}, Q \wedge R(x), \alpha_2\rangle$$
$$\langle\{x\}, R(x), \alpha_3\rangle$$

Treating this as an MLN (as in Fig.2.7), if the truth value of $R(x)$ for every individual x is observed:

$$Pr(q \mid obs) = sigmoid(\alpha_0 + n_F\alpha_1 + n_T\alpha_2) \tag{3.10}$$

where *obs* has $R(x)$ is true for n_T individuals, and false for n_F individuals out of a population of n individuals (so $n = n_T + n_F$).

Note that in the MLN, α_3 is not required for representing the conditional probability (because it cancels out), but can be used to affect $Pr(R(A_i))$, where A_i is an individual of x.

In RLR, the sigmoid, as in Equation (3.10), is used as the definition of RLR. RLR only defines the conditional probability of Q being *True* given each combination of assignments to the $R(x)$ (using Equation (3.10)); when not all $R(x)$ are observed, separate models of the probability of $R(x)$ are needed.

MLNs and RLR agree for the supervised learning case when all variables except a query leaf variable are observed (such as in Example 20). However, they are quite different in representing distributions.

Note that in MLNs, there is a single normalizing constant, guaranteeing the probabilities of the worlds sum to 1. In RLR, normalization is done separately for each possible assignment to the parents.

In summary: RLR uses the weighted formulae to define the conditional probabilities, and MLNs use the weighted formulae to define the joint probability distribution.

3.9 Canonical Forms of RLR

While in Definition 3 we allow for any Boolean formula of parents, we can prove that a positive conjunctive form is sufficient to model all the Boolean interactions among parents. A Boolean interaction is defined in the background section. Since all formulae in the WFs for a child PRV Q are conjoined with an instance of Q, we only consider the formulae of parents of Q in our proofs.

Proposition 3. *Let Q be a Boolean PRV with parents $R_i(\mathbf{x_i})$, where $\mathbf{x_i}$ is a set of logical variables in R_i which are not in Q. Using only positive conjunctive formulae*

of the parents in the WFs for Q, all Boolean interactions among the parents can be modeled by RLR.

Proof. Every Boolean formula F can be represented as a disjunction of mutually exclusive conjunctive formulae as $F_1 \vee F_2 \vee \cdots \vee F_m$ for some m, where all F_is are conjunctive formulae and are mutually exclusive (i.e., $\forall i,j \neq i : F_i \wedge F_j$ is *False*) [43]. Therefore, a WF $\langle L, F, w \rangle$ can be replaced by $\langle L, F_1 \vee F_2 \vee \cdots \vee F_m, w \rangle$. Since the conjunctive formulae are mutually exclusive, this new WF can be replaced my m WFs $\langle L, F_1, w \rangle, \langle L, F_2, w \rangle, \ldots, \langle L, F_m, w \rangle$. So we prove that any WF $\langle L, F_j, w \rangle$ where F_j is a conjunctive formula can be represented using only positive conjunctive formuale.We prove this by induction on the number of negations denoted by n_{neg}.

For $n_{neg} = 0$, the formula F_j is in a positive conjunction form and the proposition holds. Assume the proposition holds for n_{neg}. For $n_{neg} + 1$, let $\neg R_i(\mathbf{x_i})$ be one of the negated atoms of the formula F_j. We write F_j as $F'_j \wedge \neg R_i(\mathbf{x_i})$. Note that F'_j has n_{neg} negations. The WF $\left\langle L, F'_j \wedge \neg R_i(\mathbf{x_i}), w \right\rangle$ can be replaced by $\left\langle L \cup \mathbf{x_i}, F'_j, w \right\rangle$ and $\left\langle L, F'_j \wedge R_i(\mathbf{x_i}), -w \right\rangle$. Each of the formulae in these WFs has only n_{neg} negations for which the proposition holds according to our assumption.

□

Example 21. Suppose we want to represent a WF $\langle x, F, w \rangle$, where $F = A(x) \wedge (B \vee C(x))$ conjoined with the child PRV, in terms of WFs with positive conjunctive formulae. First we write F in sum of products form as $(A(x) \wedge B \wedge \neg C(x)) \vee (A(x) \wedge C(x))$. Note that no pair of the product form formulae can be simultaneously *True*. The second product form formula is in positive conjunctive form and can be represented by a single WF:

$$\langle \{x\}, A(x) \wedge C(x), w_1 \rangle$$

and the first one can be represented using the following WFs:

$$\langle \{x\}, A(x) \wedge B, w_2 \rangle$$
$$\langle \{x\}, A(x) \wedge B \wedge C(x), -w_2 \rangle$$

Proposition 3 suggests using only positive conjunctive formuale pf parents in WFs. We refer to an RLR conditional probability using WFs with only positive conjunctive formulae of parents as a *positive conjunctive RLR* and an RLR conditional probability using WFs with only positive disjunctive formulae of parents as a *positive disjunctive RLR*. Proposition 4 proves that positive disjunctive RLR has the same representational power as positive conjunctive RLR. Therefore, all

propositions proved for positive conjunctive RLR in the rest of the thesis also hold for positive disjunctive RLR.

Proposition 4. *A conditional distribution $Pr(Q \mid R_i(x_i))$ can be expressed by a positive disjunctive RLR if and only if it can be expressed by a positive conjunctive RLR.*

Proof. First, suppose $Pr(Q \mid R_i(x_i))$ can be expressed by a positive disjunctive RLR. The corollary of Proposition 3 gives that $Pr(Q \mid R_i(x_i))$ can be expressed by positive conjunctive RLR. We can also prove this without using Proposition 3 as follows.

We can write a disjunctive formula F as $\neg F'$ where F' is a conjunctive formula. So we can change all the disjunctive formulae F_j in the WFs for $Pr(Q \mid R_i(x_i))$ to $\neg F'_j$ where F'_j is a conjunctive formula. A WF $\left\langle L, \neg F'_j, w \right\rangle$ can be modeled by two WFs $\langle L, Q, w \rangle$ and $\left\langle L, F'_j, -w \right\rangle$ having conjunctive formulae, because the former counts all assignments to L, and the latter counts all assignments to L for which F'_j is *True*, thus the subtraction of these WFs gives the number of assignments to L for which F'_j is *False*. The latter WF may consist of negated atoms but we know from Proposition 3 that we can model it by a set of positive conjunctive WFs. Consequently, $Pr(Q \mid R_i(\mathbf{x_i}))$ can be also expressed by a positive conjunctive RLR.

Now, suppose the conditional distribution can be expressed by a positive conjunctive RLR definition of $Pr(Q \mid R_i(\mathbf{x_i}))$. While Proposition 3 is written for positive conjunctive RLR, it is straight forward to see that it also holds for conjunctive formulae of negated atoms, by having the induction on the number of positive atoms and removing them one by one. This means that we can express Q by WFs $\langle L, F_k, w \rangle$ where F_k is a conjunction of negated atoms. We can represent each of these formulae F_k as $\neg F'_k$ where F'_k is a positive disjunctive formula. We also mentioned that a WF $\left\langle L, \neg F'_k, w \right\rangle$ can be expressed by two WFs $\langle L, Q, w \rangle$ and $\left\langle L, F'_k, -w \right\rangle$. Both the former and the latter formulae are in positive disjunctive form. Consequently, $Pr(Q \mid R_i(\mathbf{x_i}))$ can be also expressed by a positive disjunctive RLR. □

Buchman et al. [3] looked at canonical representations for probability distributions with binary variables in the non-relational case. Our positive conjunctive canonical form corresponds to their "canonical parametrization" with a "reference state" *True* (i.e., in which all variables are assigned *True*), and our positive disjunctive canonical form has a connection to using a "reference state" *False*. Their "spectral representation" would correspond to a third positive canonical form for RLR, in terms of **XORs** (i.e., parity functions).

3.10 Non-linear Decision Thresholds

We can also model a class of non-linear decision thresholds using RLR. The following example is a case where the child PRV depends on the square of the population size of its parent.

Example 22. Suppose Q is a Boolean PRV with a parent $R(x)$. By having a WF $\langle\{x,x'\}, Q \wedge R(x) \wedge R(x'), w\rangle$ for Q where x' is typed with the same population as x, the conditional probability of Q depends on the square of the number of assignments to x for which $R(x)$ is *True*. This is similar to the WF used for an MLN in Example 14.

Example 22 represents a case where the conditional probability of a child PRV depends on the square of its parent's population size. We can prove that by using only positive conjunctive formulae in the WFs of a child PRV, we can model any polynomial decision threshold. First we prove this for the single-parent case and then for the general case of multi-parents. We assume in the following propositions that Q is a Boolean PRV and $R_1(\mathbf{x_1}), R_2(\mathbf{x_2}), \ldots$ are its parents where $\mathbf{x_i}$ is the set of logical variables in R_i which are not in Q. We use $\mathbf{x_i'}$ to refer to a new set of logical variable typed with the same population as those in $\mathbf{x_i}$.

Proposition 5. *A positive conjunctive RLR definition of $Pr(Q \mid R(\mathbf{x}))$ (single-parent case) can represent any decision threshold that is a polynomial function of the sizes of logical variables in $\mathbf{x}$ and the number of (tuples of) individuals for which $R(\mathbf{x})$ is True or False.*

Proof. Based on Proposition 3 we know that a WF having any Boolean formula can be written as a set of WFs each having a positive conjunctive formula. Therefore, in this proof we do not commit to using only positive conjunctive formulae. The final set of WFs can be represented by a set of positive conjunctive WFs using Proposition 3.

Each term of the polynomial in the single-parent case is of the form $w(\prod_i |x_i|^{d_i}){n_T}^{\alpha}{n_F}^{\beta}$[3], where n_T and n_F denote the number of individuals for which $R(\mathbf{x})$ is *True* or *False* respectively, $x_i \in \mathbf{x}$ represents the i-th logical variable in $\mathbf{x}$, α, β and d_is are non-negative integers, and w is the weight of the term. First we prove by induction that for any j, there is a WF that can build the term ${n_T}^{\alpha}{n_F}^{\beta}$ for any α and β where $\alpha+\beta=j$, $\alpha \geq 0$ and $\beta \geq 0$.

For $j=0$, ${n_T}^{\alpha}{n_F}^{\beta}=1$. We can trivially build this by WF $\langle\{\}, True, w\rangle$. Assuming it is correct for j, we prove it for $j+1$. For $j+1$, either $\alpha > 0$ or $\alpha = 0$

[3] It is important to consider the population sizes of logical variables in the polynomial terms since some properties may depend on these population sizes. For instance, Poole et al. [47] give a real world example of predicting the age of people using the number of movies they have rated.

and $\beta > 0$. If $\alpha > 0$, using our assumption for j, we can have a WF $\langle L, F, w\rangle$ which builds the term ${n_T}^{\alpha-1}{n_F}^{\beta}$. So the WF $\langle L \cup \{x'\}, F \wedge R(\mathbf{x}'), w\rangle$ builds the term ${n_T}^{\alpha}{n_F}^{\beta}$ because the first WF was *True* ${n_T}^{\alpha-1}{n_F}^{\beta}$ times and now we count it n_T more times because $R(\mathbf{x}')$ is *True* n_T times.

If $\alpha = 0$ and $\beta > 0$, we can have a WF $\langle L, F, w\rangle$ which builds the term ${n_T}^{\alpha}{n_F}^{\beta-1}$. By the same reasoning as in previous case, we can see that the WF $\langle L \cup \{\mathbf{x}'\}, F \wedge \neg R(\mathbf{x}'), w\rangle$ produces the term ${n_T}^{\alpha}{n_F}^{\beta}$.

In order to include the population size of logical variables x_i, where $x_i \in \mathbf{x}$, and generate the term $(\prod_i |x_i|^{d_i}){n_T}^{\alpha}{n_F}^{\beta}$, we only add d_i extra logical variables x_i' to the set of logical variables of the WF that generates ${n_T}^{\alpha}{n_F}^{\beta}$. Then we set the weight of this WF to w to generate the desired term.

Until now we proved that we can generate every term of the polynomial. Since RLR sums over all these terms, we can generate every decision threshold which is a polynomial function of the sizes of logical variables in $\mathbf{x}$ and the number of (tuples of) individuals for which $R(\mathbf{x})$ is *True* or *False*.. □

Conclusion. One of the conclusions of this proposition is that a term $w(\prod_i |x_i|^{d_i}){n_T}^{\alpha}{n_F}^{\beta}$ can be generated by having a WF with its formula consisting of n_T instances of $R(\mathbf{x}')$ and n_F instances of $\neg R(\mathbf{x}')$, adding d_i of each logical variable x_i to the set of logical variables, and setting the weight of WF to w. We will use this conclusion for proving the proposition in multi-parent case.

Example 23. Suppose we want to model the case where Q is *True* if the square of number of *True* individuals in $R(x)$ is at least 5 more than twice the number of *False* individuals in it (i.e., ${n_T}^2 \geq 2n_F + 5$). In this case, we can model the sigmoid of the polynomial ${n_T}^2 - 2n_F - 4.5$. The reason for using 4.5 instead of 5 is to make the polynomial positive when ${n_T}^2 = 2n_F + 5$. The following WFs are used by RLR to model this polynomial. The first WF generates the term ${n_T}^2$, the second one generate $-2n_F$ and the third one generates -4.5.

$$\langle \{x, x'\}, Q \wedge R(x) \wedge R(x'), 1\rangle$$
$$\langle \{x\}, Q \wedge \neg R(x), -2\rangle$$
$$\langle \{\}, Q, -4.5\rangle$$

Note that the second WF above can be written in positive form by using the following two WFs:

$$\langle \{x\}, Q, -2\rangle$$
$$\langle \{x\}, Q \wedge R(x), 2\rangle$$

Using the conclusion following Proposition 5, we now extend Proposition 5 to the multi-parent case:

Proposition 6. *A positive conjunctive RLR definition of $Pr(Q \mid R_1(\mathbf{x_1}), \ldots, R_k(\mathbf{x_k}))$ (multi-parent case) can represent any decision threshold that is a polynomial function of the sizes of logical variables in the parents and the number of (tuples of) individuals for which a Boolean function of parents hold.*

Proof. Let $G_1, G_2, \ldots, G_t$ represent Boolean interactions of parents for our model. Also let $n_{(i)}$ denote the number of individuals for which G_i is *True*. Each term of the polynomial in the multi-parent case is then of the form:
*w * (any polynomial of population sizes) ** $n_{(1)}^{\alpha_1} n_{(2)}^{\alpha_2} \ldots n_{(t)}^{\alpha_t}$.
We demonstrate how we can generate any term $n_{(1)}^{\alpha_1} n_{(2)}^{\alpha_2} \ldots n_{(t)}^{\alpha_t}$. The inclusion of population size of logical variables and the weight w for each term is the same as in Proposition 5.

The conclusion of Proposition 5 can be generalized to work for any Boolean formula G_i instead of a single parent R. We only need to include a conjunction of α_i instances of G_i with different logical variables typed with the same population in each instance. Let F_i represent this conjunction of α_i instances of G_i. We use this generalization in our proof.

For each $n_{(i)}^{\alpha_i}$, we can use the generalization of the conclusion of Proposition 5 to obtain a WF $\langle L_i, F_i, 1 \rangle$ which generates this term. Similar to the reasoning for single-parent case, we can see that the WF $\langle \{\cup_{i=1}^{t} L_i, F_1 \wedge F_2 \wedge \cdots \wedge F_t, w\rangle$ generates the term $n_{(1)}^{\alpha_1} n_{(2)}^{\alpha_2} \ldots n_{(t)}^{\alpha_t}$. We can then use Proposition 3 to write this WF using only positive conjunctive WFs.

Until now we proved that we can generate every term of the polynomial. Since RLR sums over all these terms, we can generate any decision threshold that is a polynomial function of the sizes of logical variables in the parents and the number of (tuples of) individuals for which a Boolean function of parents hold. □

Example 24. Suppose we want to model the case where Q is *True* if the square of number of individuals for which $R_1(x_1) = True$ multiplied by the number of individuals for which $R_2(x_2) = True$ is less than five times the number of *False* individuals in $R_1(x_1)$.In this case, we define $G_1 = \neg R_1(x_1)$ and $G_2 = R_1(x_1) \wedge R_1(x_1') \wedge R_2(x_2)$. We can model the sigmoid of the polynomial $5n_{(1)} - n_{(2)} - 0.5$. The reason why we use -0.5 in the polynomial is that we want the polynomial to be negative when $5n_{(1)} = n_{(2)}$. The following WFs are used by RLR to model this polynomial where the first formula generates the term $5n_{(1)}$, the second generates $-n_{(2)}$ and the third generates -0.5.

$$\langle \{x_1\}, Q \wedge \neg R_1(x_1), 5 \rangle$$
$$\langle \{x_1, x_1', x_2\}, Q \wedge R_1(x_1) \wedge R_1(x_1') \wedge R_2(x_2), -1 \rangle$$
$$\langle \{\}, Q, -0.5 \rangle$$

Note that the first WF above can be written in positive form in the same way as in Example 23.

Proposition 6 proved that RLR can model any polynomial decision threshold. Proposition 7 proves the converse of Proposition 6:

Proposition 7. *Any decision threshold that can be represented by a positive conjunctive RLR definition of $Pr(Q \mid R_1(\mathbf{x_1}),\ldots,R_k(\mathbf{x_k}))$ is a polynomial function of the number of (tuples of) individuals for which a Boolean function of parents hold.*

Proof. We prove that every WF for Q can only generate a term of the polynomial. Since RLR sums over these terms, it will always represent a polynomial decision threshold.

Similar to Proposition 6, let $G_1, G_2, \ldots, G_t$ represent the Boolean functions of parents and let $n_{(i)}$ denote the number of individuals for which G_i is *True*. A positive conjunctive formula in a WF can consist of α_1 instances of G_1, α_2 instances of G_2, ..., α_t instances of G_t. Based on Proposition 6, we know that this formula is *True* $n_{(1)}^{\alpha_1} n_{(2)}^{\alpha_2} \ldots n_{(t)}^{\alpha_t}$ times. The WF can contain more logical variables in its set of logical variables than the ones in its formula. This, however, will only cause the above term to be multiplied by the population size of the logical variable generating a term of the polynomial described in Proposition 6. Therefore, each of the WFs can only generate a term of the polynomial which means that positive conjunctive RLR can only represent the sigmoid of this polynomial. □

3.11 Beyond Polynomial Decision Thresholds

Proposition 7 showed that any conditional probability that can be expressed using a positive conjunctive RLR definition of $Pr(Q \mid R_1(\mathbf{x_1}),\ldots,R_k(\mathbf{x_k}))$ is the sigmoid of a polynomial of the number of *True* and *False* individuals in each parent $R_i(\mathbf{x_i})$. However, given that the decision thresholds are only defined for integral counts, some of the apparently non-polynomial decision thresholds are equivalent to a polynomial and so can be modeled using RLR.

Example 25. Suppose we want to model $Q \equiv \left(\left\lceil\sqrt{n_T}\right\rceil < n_F\right)$. This is a non-polynomial decision threshold, but since n_T and n_F are integers, it is equivalent to the polynomial decision threshold $n_T - (n_F - 1)^2 \leq 0$ which can be formulated using RLR by the following WFs:

$$\langle\{x\}, Q \wedge R(x), -1\rangle$$
$$\langle\{x\}, Q \wedge \neg R(x) \wedge \neg R(x), 1\rangle$$
$$\langle\{x\}, Q \wedge \neg R(x), -2\rangle$$
$$\langle\{\}, Q, 1.5\rangle$$

Example 26. Suppose we want to model $Q \equiv (2^{n_T} > 3^{n_F})$. This is, however, equivalent to the polynomial form $Q \equiv (n_T \log 2 - n_F \log 3 > 0)$ and can be formulated in positive conjunctive RLR using the WFs:

$$\langle \{x\}, Q, -\log 3 \rangle$$
$$\langle \{x\}, Q \wedge R(x), \quad \log 3 + \log 2 \rangle$$

There are, however, non-polynomial decision thresholds that cannot be converted into a polynomial one and RLR is not able to formulate them.

Example 27. Suppose we want to model $Q \equiv (2^{n_T} > n_F)$. This cannot be converted to a polynomial form and RLR cannot formulate it.

Finding a parametrization that allows to model any non-polynomial decision threshold remains an open problem.

3.12 RLR with Multi-valued Child Variables

Definitions 3 and 5 consider Boolean child variables. We can extend these definitions to multi-valued child variables similar to the way logistic regression is extended. Suppose a multi-valued categorical PRV $Q(\mathbf{x})$, where $Q(x)$ can take $k \geq 2$ different values $\{V_1, V_2, \ldots, V_k\}$, is a child of PRVs $\{R_1(\mathbf{x_1}), R_2(\mathbf{x_2}), \ldots, R_m(\mathbf{x_m})\}$. We define $k-1$ sets of WFs $\{wf_1, wf_2, \ldots, wf_{k-1}\}$ each containing a (possibly) different number of WFs, where each formula of WFs in wf_i is conjoined with the atom $Q = V_i$.

Having the above $k-1$ sets of WFs, the probability of Q taking different values in its domain given a grounding assignment $ga_{\mathbf{x}}$ can be defined as follows:

$$if\,(l < k) \rightarrow Pr(Q(\mathbf{x}) = V_l|\Pi) = \frac{exp\big(\sum_{\langle L, Q' \wedge F', w\rangle \in comp(wf_l, \mathbf{X})} w \ \sum_L F_\Pi \theta(Q', Q(\mathbf{X}))\big)}{1 + \sum_{l'=1}^{k-1} exp\big(\sum_{\langle L, F, w\rangle \in comp(wf_{l'}, \mathbf{X})} w \ \sum_L F_\Pi \theta(Q', Q(\mathbf{x}))\big)}$$
$$if\,(l = k) \rightarrow Pr(Q(ga_{\mathbf{x}}) = V_l|\Pi) = \frac{1}{1 + \sum_{l'=1}^{k-1} exp\big(\sum_{\langle L, F, w\rangle \in comp(wf_{l'}, \mathbf{X})} w \ \sum_L F_\Pi \theta(Q', Q(\mathbf{x}))\big)} \tag{3.11}$$

Note that Equation 3.11 reduces to the Equation 5 in Definition 5 when $k = 2$. The extension of RLR to continuous child PRVs and continuous parents is left as a future work.

Chapter 4

Approximating Other Aggregators Using RLR

We can model other well-known aggregators using positive conjunctive RLR. In most cases, however, this is only an approximation because many aggregators are deterministic taking only values 0 and 1, but the sigmoid function reaches 0 or 1 only in the limit. In order for a sigmoid to produce a 0 or 1 output, we need an infinitely large number, but we cannot choose infinitely large numbers. We can, however, get arbitrarily close to 0 or 1 by choosing arbitrarily large weights. In the rest of this chapter, we use M to refer to a number which can be set sufficiently large to receive the desired level of approximation. n_{val} is the number of individuals x for which $R(x) = val$, when R is not Boolean.

4.1 OR

OR is one of the popular aggregators which is equivalent to the logical existential quantifier ($\exists$). Using OR, the child node is *True* if there exists at least one assignment of individuals to the logical variables in the parents, for which a desired formula holds. In order to model OR in RLR for a PRV Q with parent PRV $R(x)$, we use the WFs:

$$\langle \{\}, Q, -M \rangle$$
$$\langle \{x\}, Q \wedge R(x), 2M \rangle$$

for which $Pr(q \mid R(x)) = \text{sigmoid}(-M + 2Mn_T)$. We can see that if none of the individuals are *True* (i.e. $n_T = 0$), the value inside the sigmoid is $-M$ which is a negative number and the probability is close to 0. If even one individual is *True* (i.e. $n_T \geq 1$), the value inside the sigmoid becomes positive and the probability becomes closer to 1. In both cases, the value inside the sigmoid is a linear function of M. Increasing M pushes the probability closer to 0 or to 1 and the approximation becomes more accurate.

Example 28. Suppose a group of people live in an apartment and they can set off a fire alarm if they smell smoke. We have a random variable *AlarmSounds* which

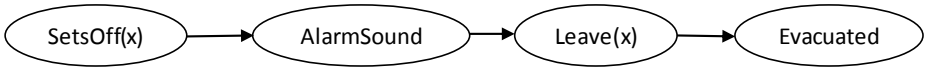

Figure 4.1: A relational model representing the evacuation scenario of a building when a member of the building sets off an smoke alarm. The conditional probability of the PRV *AlarmSounds* in this model should be represented using the aggregation operator *OR* and the conditional probability of the PRV *Evacuated* should be represented using the aggregation operator *AND*.

has a parent $SetsOff(x)$ (as in Fig. 4.1) and whose conditional probability can be approximated by the following WFs:

$$\langle \{\}, AlarmSounds, -10 \rangle$$
$$\langle \{x\}, AlarmSounds \wedge SetsOff(x), 20 \rangle$$

where we chose $M = 10$. We can see in Table 4.1 how close the approximation is for this value of M.

4.2 AND

AND is equivalent to the logical universal quantifier ($\forall$) and can be modeled similarly to OR. In order to model AND in RLR for a PRV Q with parent PRV $R(x)$, we use the WFs:

$$\langle \{\}, Q, M \rangle$$
$$\langle \{x\}, Q \wedge \neg R(x), 2M \rangle$$

or equivalently with the following WFs having only positive conjunctive formulae:

$$\langle \{\}, Q, M \rangle$$
$$\langle \{x\}, Q, -2M \rangle$$
$$\langle \{x\}, Q \wedge R(x), 2M \rangle$$

Table 4.1: The probability of random variable *AlarmSounds* as a function of the number of people *num* who set off the alarm for the weighted formulae in Example 28 with an accuracy of six digits after the decimal point.

num	0	1	2	3	4	>4
Probability	0.000045	0.999955	1	1	1	1

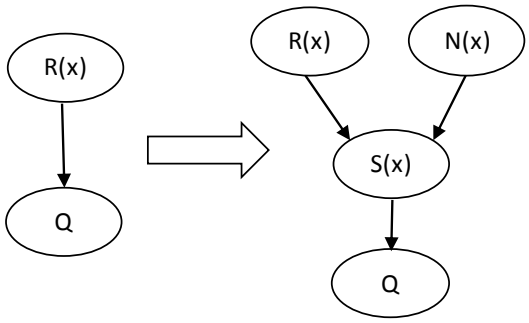

Figure 4.2: On the left is a relational model for a child node with a single parent having an extra logical variable. On the right is a relational model representing the changes to be made to the model on the left for defining a noisy-OR or noisy-AND conditional probability for the child node using RLR.

for which $Pr(q \mid R(x)) = \text{sigmoid}(M - 2Mn_F)$. When $n_F = 0$, the value inside the sigmoid is $M > 0$, so the probability is closer to 1. When $n_F \geq 1$, the value inside the sigmoid becomes negative and the probability becomes closer to 0. Like OR, accuracy increases with M.

Example 29. In Example 28, after hearing the alarm sound, people start to leave the building and the building is evacuated if all people have left (see Fig. 4.1). The conditional probability of the PRV *Evacuated* can be then approximated using the following WFs:

$$\langle \{\}, Evacuated, M \rangle$$
$$\langle \{x\}, Evacuated, -2M \rangle$$
$$\langle \{x\}, Evacuated \wedge Leave(x), 2M \rangle$$

Having *AND* and *OR* aggregators (i.e. universal and existential quantifiers) we can model any other Boolean formuale of the individuals.

4.3 Noisy-OR and Noisy-AND

The previous two sections represented how we can use RLR to approximate the deterministic OR and AND aggregators using a probabilistic model. Now we consider modeling the noisy-OR and noisy-AND using RLR.

Figure 4.2 represents how noisy-OR and noisy-AND can be modeled for the network in Figure 3.1. In this figure, $R(x)$ represents the values of the individuals

being combined and $N(x)$ represents the noise probability. For noisy-OR, $S(x) \equiv R(x) \wedge N(x)$, and Q is the OR aggregator of $S(x)$. For noisy-AND, $S(x) \equiv R(x) \vee N(x)$, and Q is the AND aggregator of $S(x)$. Note that the noise probability can be different for each of the individuals and we cannot model noisy-OR and noisy-AND without adding extra PRVs to the model and by just using the models for deterministic OR and AND, where each individual has the same effect.

4.4 Mean

We represent how we can model **mean** $>$ **t**. We can model "Q is *True* if mean$(R(x)) > t$" using the following WFs (*val* and t are numeric constants. The second WF is repeated for each $val \in \text{range}(R)$, so we have a total of $1+r$ WFs where r is the size of range(R)):

$$\langle \{\}, Q, -M \rangle$$
$$\langle \{x\}, Q \wedge R(x) = val, M^2(val - t) \rangle$$

for which

$$\begin{aligned} Pr(q \mid R(x)) &= \text{sigmoid}(-M + M^2 \sum_{val \in \text{range}(R)} n_{val}(val - t)) \\ &= \text{sigmoid}(-M + M^2 (\sum_{val \in \text{range}(R)} n_{val} val - t \sum_{val \in \text{range}(R)} n_{val})) \\ &= \text{sigmoid}(-M + M^2(sum - nt)) \end{aligned}$$

where $n = |x|$ and *sum* represents the sum of the values of the individuals. When $mean = \frac{sum}{n} > t$, the value inside the sigmoid is positive and the probability is close to 1. Otherwise, the value inside the sigmoid is negative and the probability is close to 0. For this case, M should be greater than the minimum number that $|\frac{1}{sum - nt}|$ can take to generate a number greater than 1 when multiplied by $(sum - nt)$. Otherwise, it may occur that $sum - nt > 0$ but $M^2(sum - nt) \leq M$ which makes the sigmoid produce a number close to 0. Note that this minimum bound for M depends on the accuracy of computations, not on the population size. Another option is to use the following WFs (the second WF is again repeated for each $val \in \text{range}(R)$):

$$\langle \{\}, Q, -\varepsilon \rangle$$
$$\langle \{x\}, Q \wedge R(x) = val, M(val - t) \rangle$$

where ε is a tiny number ensuring the value inside the sigmoid is negative when $mean = t$. It (ε) can be set according to the accuracy of the system, or can be set to zero if a probability of 0.5 is acceptable when $mean = t$. Note that the number of required WFs in both casesgrows with the number of values that the parent can take.

Example 30. Suppose we have a set of movies and the ratings people gave to these movies in a star-rating system. We define a movie to be *popular* if the average of its ratings is more than 3.5. In this case, we have a parametrized random variable $Popular(m)$ which is a child of a parametrized random variable $Rate(p,m)$. The following weighted formulae can approximate the conditional dependence of $Popular(m)$ on its parent (the second WF is repeated 5 times for different values of $i \in \{1,2,3,4,5\}$):

$$\langle \{\}, Popular(m), -\varepsilon \rangle$$
$$\langle \{p\}, Popular(m) \wedge Rate(p,m) = i, i - 3.5 \rangle$$

RLR sums over the above weighted formulae and takes the sigmoid resulting in:

$$Pr(Popular(m) = True | \Pi) = sigmoid(-\varepsilon + sum - 3.5n)$$

where *sum* denotes the sum of the ratings and n represents the number of ratings for this movie. The value inside the sigmoid is positive if $mean = \frac{sum}{n} > 3.5$ and is negative otherwise (ε is used to ensure the the value inside the sigmoid is negative when $mean = 3.5$).

4.5 More-than-t Trues

More-than-t *True*s can be considered as an extended version of OR. When $t = 0$, these two aggregators are equivalent. More-than-t *True*s, corresponding to"Q is *True* if R is *True* for more than t individuals", can be modeled using the WFs:

$$\langle \{\}, Q, -2Mt - M \rangle$$
$$\langle \{x\}, Q \wedge R(x), 2M \rangle$$

giving $Pr(q \mid R(x)) = \text{sigmoid}(-2Mt - M + 2Mn_T)$ and the value inside the sigmoid is positive if $n_T > t$. The number of WFs required is fixed.

Example 31. In Example 30, suppose instead of rating movies, users can only like the movies they enjoyed watching, and a *popular* movie is defined as one having at least 50 likes. The following WFs can be used to approximate this dependency:

$$\langle \{\}, popular(m), -99 \rangle$$
$$\langle \{p\}, popular(m) \wedge liked(p,m), 2 \rangle$$

4.6 More-than-t% Trues

More-than-t% *True*s, corresponding to "Q is *True* if R is *True* for more than t percent of the individuals", is a special case of the aggregator "mean $> \frac{t}{100}$" when we treat *False* values as 0 and *True* values as 1. This directly provides the WFs:

$$\langle \{\}, Q, -M \rangle$$
$$\left\langle \{x\}, Q \wedge \neg R(x), M^2(0 - \tfrac{t}{100}) \right\rangle$$
$$\left\langle \{x\}, Q \wedge R(x), M^2(1 - \tfrac{t}{100}) \right\rangle$$

while requiring $M > |\frac{1}{n_T - nt/100}|$, where n is the populations size of x. Note that we can use Proposition 3 to replace the second WF with two WFs having positive conjunctive formulae. Unlike the aggregator "mean $> \ldots$", here the number of WFs is fixed, because the parent can take only two different values.

Example 32. Consider Example 31 and suppose people like or dislike a movie after they watch it. Also suppose a movie is defined to be *popular* if more than 70% of people liked it. This can be approximated by More-than-t% *True*s aggregator using similar WFs as used in Example 30 with $t = \frac{70}{100}$.

4.7 Max

Firstly, we represent how we can model the case where Q is true if the max of $R(x)$ is greater than t. Then we propose a way of having a random variable whose value is the maximum value of the individuals in R.

The following WFs are used for modeling $max > t$, corresponding to "Q is *True* if $\max(R(x)) > t$", in RLR (the second WF is repeated for each $val > t \in \text{range}(R(x))$):

$$\langle \{\}, Q, -M \rangle$$
$$\langle \{x\}, Q \wedge R(x) = val, 2M \rangle$$

thus $Pr(q \mid R(x)) = \text{sigmoid}(-M + 2M \sum_{val > t \in \text{range}(R)} n_{val})$. The value inside the sigmoid is positive if there is an individual having a value greater than t (i.e. $\exists val > t \in \text{range}(R) : n_{val} > 0$). Note that the number of WFs required grows with the number of values greater than t that the parents can take.

Now suppose we want Q to be the maximum value of individuals in $R(x)$. For binary parents, the "max" aggregator is identical to "OR". Otherwise, $\text{range}(Q) = \text{range}(R(x))$. Let r represent the size of $\text{range}(R(x))$ and $t_1, t_2, \ldots, t_r$ represent the values in $\text{range}(R(x))$. This "max" aggregator can be modeled using a 2-level structure as in Figure 4.3. First, for every $t_i \in \text{range}(R(x))$, we create a separate "max $\geq t$" aggregator using RLR (as described earlier), with $R(x)$ as its parents.

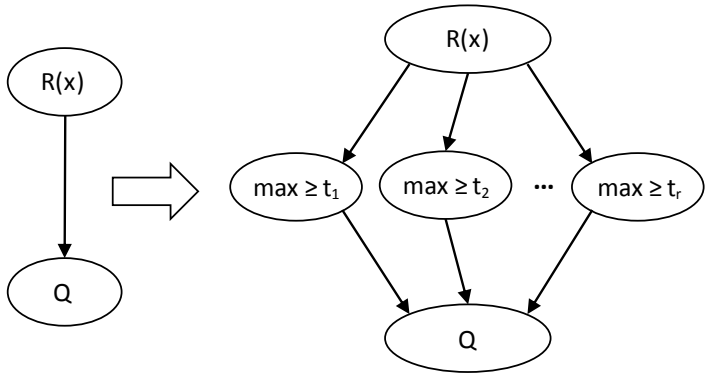

Figure 4.3: On the left is a relational model for a child node with a single parent having an extra logical variable. On the right is a relational model representing the changes to be made to the model on the left for defining a conditional probability representing the aggregator *max* for the child node using RLR.

This can be viewed in the middle level of Figure 4.3 where there are r intermediate random variables each representing if the *max* is greater than or equal to a $t_i \in \text{range}(R(x))$. Then, we define the child Q, with all the "max $\geq t$" aggregators as its parents. Q can compute $\max(R(x))$ given its parents. Note that while r may be arbitrarily large, it has a fixed size and does not change with population size, hence it is possible to use non-relational constructs (e.g., a table) for its implementation.

Example 33. As in Example 30, suppose we have a set of users denoted by the logical variable p, a set movies denoted by the logical variable m, and the rates of the users for the movies. Suppose we want to have a PRV $MaxRate(m)$ representing the maximum rate of each movie.

First, we define 5 PRVs $MGE1(m), MGE2(m), \ldots, MGE5(m)$, where $MGEi(m)$ represents whether the maximum rate of the movie is greater than or equal to i or not. The conditional probability of each of these PRVs can be represented by RLR. As an example, we define the WFs for $MGE3(m)$ as follows (the WFs for the other PRVs are similar):

$$\langle \{\}, MGE3(m), -10 \rangle$$
$$\langle \{p\}, MGE3(m) \wedge Rate(p,m) = 3, 20 \rangle$$
$$\langle \{p\}, MGE3(m) \wedge Rate(p,m) = 4, 20 \rangle$$
$$\langle \{p\}, MGE3(m) \wedge Rate(p,m) = 5, 20 \rangle$$

Then the conditional probability of $MaxRate(m)$ given its parents can be defined

Table 4.2: Conditional probability table for PRV *MaxRate(m)* representing the maximum rate of different movies. For simplicity, we just represent the desired value of the child node (the one having a probability of 1) instead of probabilities of each value it can take.

$MGE1(m)$	$MGE2(m)$	$MGE3(m)$	$MGE4(m)$	MGE5(m)	Value
True	*True*	*True*	*True*	*True*	5
True	*True*	*True*	*True*	*False*	4
...	...	...	...	...	...
True	*False*	*False*	*False*	*False*	1
...	...	...	...	...	...
False	*False*	*False*	*False*	*False*	0

as in Table 4.2. We assume the maximum rate of a movie for which we don't have any ratings is zero.

4.8 Mode

First, we represent how we can model the case where Q is true if the mode of $R(x)$ is equal to t. Then we propose a way of having a random variable whose value is the mode value of the individuals in R.

To model $mode = t$, corresponding to"Q is *True* if mode$(R(x)) = t$", we first add another PRV $S(y)$ to the network as in Fig. 34 where the population of y is the range of $R(x)$. Then for each individual Y of y, we use the following WFs for which $Pr(s(Y) \mid R(x)) = \text{sigmoid}(M - 2M(n_Y - n_t))$ and the value inside the sigmoid is positive if value t has occurred more than or equal to value Y in the population of R (i.e. $n_t \geq n_Y$). Note that the number of WFs required grows with the number of values that the parent can take.

$$\langle \{\}, S(Y), M \rangle$$
$$\langle \{x\}, S(Y) \wedge R(x) = C, -2M \rangle$$
$$\langle \{x\}, S(Y) \wedge R(x) = t, 2M \rangle$$

Then Q must be *True* if for all individuals in y, $S(y)$ is *True*. This is because a *False* value for an individual of S means that this individual has occurred more than t and t is not the mode. Therefore, we can use WFs similar to the ones we

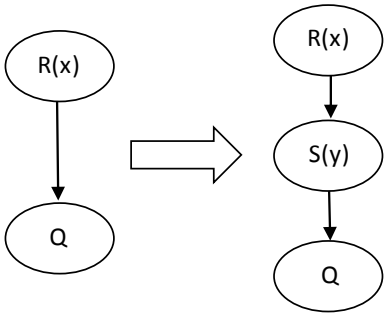

Figure 4.4: On the left is a relational model for a child node with a single parent having an extra logical variable. On the right is a relational model representing the changes to be made to the model on the left for defining a conditional probability representing the aggregator $mode = t$ for the child node using RLR.

used for AND:

$$\langle \{\}, Q, M \rangle$$
$$\langle \{y\}, Q, -2M \rangle$$
$$\langle \{y\}, Q \wedge S(y), 2M \rangle$$

Now suppose we want the value of Q to be the mode of the values of the individuals in $R(x)$. For binary parents, the "mode" aggregator is also called "majority", and can be modeled with the "more-than-t% Trues" aggregator, with $t = 50$. Otherwise, range(Q) = range$(R(x))$, and we can use the same approach as for "max", by having a middle layer with r separate "mode $= t$" aggregators, where r represents the size of range$(R(x))$, and with Q as their child.

Example 34. Suppose in Example 30 a movie is popular if the mode of the rates is 5. In order to model this in RLR, we create a network where $Rate(p,m)$ is the parent of $S(r,m)$, where r represents the rates and belongs to $\{1,2,3,4,5\}$, and $S(r,m)$ is the parent of $Popular(m)$. The following WFs should be used to approximate the conditional probability of $S(R,m)$ (where R is an instance of r):

$$\langle \{\}, S(R,m), 10 \rangle$$
$$\langle \{p\}, S(R,m) \wedge Rate(p,m) = R, -20 \rangle$$
$$\langle \{p\}, S(R,m) \wedge Rate(p,m) = 5, 20 \rangle$$

and the following WFs should be used to approximate the conditional probability

of $Popular(m)$:

$$\langle \{\}, Popular(m), 10 \rangle$$
$$\langle \{r\}, Popular(m), -20 \rangle$$
$$\langle \{r\}, Popular(m) \wedge S(r,m), 20 \rangle$$

4.9 Aggregators Not Represented by RLR

In previous sections of this chapter, we discussed how a series of well-known aggregators can be represented using RLR. There are, however, other well-known aggregators such as *median* $> t$ that we could not model them using our RLR. There are also other aggregators whose outputs are a continuous value. *Mean* and *median* are two such aggregators. Since we only considered child nodes having Boolean or multi-valued ranges, our RLR cannot model such aggregators. Furthermore, we did not consider parents with continuous values in which case the *Max* aggregator has also a continuous output. Future work includes extending RLR to continuous child and parent nodes, and discussing whether such aggregators can be represented using the extended RLR or not.

Chapter 5

Conclusion

Today's data and models are complex, composed of objects and relations, and noisy. Hence it is not surprising that relational probabilistic knowledge representation currently receives a lot of attention. However, relational probabilistic modeling is not an easy task and raises several novel issues when it comes to knowledge representation:

- What assumptions are we making? Why should we choose one representation over another?
- We may learn a model for some population size(s), and want to apply it to other population sizes. We want to make assumptions explicit and know the consequences of these assumptions.

In this work, we provided answers to these questions for the case of the logistic regression model. The introduction of the relational logistic regression (RLR) family from first principle is already a major contribution. Based on it, we have investigated the dependence on population size for different variants and have demonstrated that already for simple and well-understood (at the non-relational level) models, there are complex interactions of the parameters with population size.

The major contributions of this work can be summarized as follows:

- Introducing a relation version of logistic regression (RLR).
- Brief comparison of the proposed RLR with MLNs.
- Defining canonical forms of representation for RLR.
- Proving the class of decision thresholds that can and cannot be modeled using RLR.
- Approximating other well-known aggregators using RLR.

Future work includes:

- extending the current version of RLR to continuous child and parent PRVs and discussing the decision thresholds and aggregators that can and cannot be represented using the extended RLR,
- developing a lifted inference algorithm, or extending existing algorithms, for relational Bayesian networks to allow for RLR conditional probabilities,
- learning the structure and the parameters of an RLR conditional probability for a PRV from data,
- and understanding the relationship to other models such as undirected models like MLNs. Exploring this direction is important since determining which models to use is more than fitting the models to data; we need to understand what we are representing.

Bibliography

[1] Hendrik Blockeel. Statistical relational learning. *Handbook on Neural Information Processing*, pages 241–281, 2013.

[2] Hendrik Blockeel and Luc De Raedt. Top-down induction of first-order logical decision trees. *Artificial intelligence*, 101(1):285–297, 1998.

[3] David Buchman, Mark Schmidt, Shakir Mohamed, David Poole, and Nando De Freitas. On sparse, spectral and other parameterizations of binary probabilistic models. In *AISTATS 2012-15th International Conference on Artificial Intelligence and Statistics*, 2012.

[4] Wray L. Buntine. Operations for learning with graphical models. *arXiv preprint cs/9412102*, 1994.

[5] Saskia Le Cessie and J.C. van Houwelingen. Ridge estimators in logistic regression. *Applied Statistics*, 41(1):191–201, 1992.

[6] Chin-Liang Chang and Richard Char-Tung Lee. *Symbolic logic and mechanical theorem proving*. Academic Press, New York, 1973.

[7] George Danezis and Prateek Mittal. Sybilinfer: Detecting sybil nodes using social networks. In *NDSS*, 2009.

[8] Francisco J. Diez. Parameter adjustment in Bayes networks. the generalized noisy or-gate. In *Proc. ninth UAI*, pages 99–105, 1993.

[9] Matthew Dirks, Andrew Csinger, Andrew Bamber, and David Poole. Reasoning and inference for a relational open-world influence diagram applied to a real-time geological domain. In *Proc. AAAI-2014 Statistical Relational AI Workshop*, 2014.

[10] Pedro Domingos, Stanley Kok, Daniel Lowd, Hoifung Poon, Matthew Richardson, and Parag Singla. Markov logic. In L. De Raedt, P. Frasconi, K. Kersting, and S. Muggleton, editors, *Probabilistic Inductive Logic Programming*, pages 92–117. Springer, New York, 2008.

[11] Nir Friedman, Lisa Getoor, Daphne Koller, and Avi Pfeffer. Learning probabilistic relational models. In *Proc. of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1300–1307. Sweden: Morgan Kaufmann, 1999.

[12] Michael R. Genesereth and Nils J. Nilsson. *Logical foundations of artificial intelligence*. Vol. 9. Los Altos, CA: Morgan Kaufmann, 1987.

[13] Lisa Getoor and Mehran Sahami. Using probabilistic relational models for collaborative filtering. In *Workshop on Web Usage Analysis and User Profiling (WEBKDD'99)*, 1999.

[14] Lisa Getoor and Ben Taskar. *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA, 2007.

[15] Carlos Guestrin, Shobha Venkataraman, and Daphne Koller. Context-specific multiagent coordination and planning with factored mdps. In *AAAI/IAAI*, 2002.

[16] Michael Horsch and David Poole. A dynamic approach to probability inference using bayesian networks. In *Proc. sixth Conference on Uncertainty in AI*, pages 155–161, 1990.

[17] Zan Huang, D. Zeng, and Hsinchun Chen. A unified recommendation framework based on probabilistic relational models. In *Fourteenth Annual Workshop on Information Technologies and Systems (WITS)*, 2004.

[18] Tuyen N. Huynh and Raymond J. Mooney. Discriminative structure and parameter learning for markov logic networks. In *Proc. of the international conference on machine learning*, 2008.

[19] Manfred Jaeger. Relational Bayesian networks. In *Proc. of the Thirteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 1997.

[20] Dominik Jian, Andreas Barthels, and Michael Beetz. Adaptive Markov logic networks: Learning statistical relational models with dynamic parameters. In *9th European Conference on Artificial Intelligence (ECAI)*, pages 937–942, 2009.

[21] Dominik Jian, Kirchlechner Bernhard, and Michael Beetz. Extending Markov logic to model probability distributions in relational domains. In *KI*, pages 129–143, 2007.

[22] Seyed Mehran Kazemi, David Buchman, Kristian Kersting, Sriraam Natarajan, and David Poole. Relational logistic regression. In *Proc. 14th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 2014.

[23] Seyed Mehran Kazemi, David Buchman, Kristian Kersting, Sriraam Natarajan, and David Poole. Relational logistic regression: the directed analog of Markov logic networks. In *Proc. AAAI-2014 Statistical Relational AI Workshop*, 2014.

[24] Seyed Mehran Kazemi and David Poole. Elimination ordeting in first-order probabilistic inference. In *Proc. of Association for the Advancements of Artificial Intelligence (AAAI)*, 2014.

[25] Jacek Kisynski and David Poole. Lifted aggregation in directed first-order probabilistic models. In *Twenty-first International Joint Conference on Artificial Intelligence*, pages 1922–1929, 2009.

[26] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, Cambridge, MA, 2009.

[27] Kevin B. Krob and Ann E. Nicholson. *Bayesian artificial intelligence*. cRc Press, 2003.

[28] Nada Lavrac and Saso Dzeroski. Inductive logic programming. *WLP*, pages 146–160, 1994.

[29] Peter McCillagh. Generalized linear models. *European Journal of Operational Research*, 16(3):285–292, 1984.

[30] Brian Milch, Luke S. Zettlemoyer, Kristian Kersting, Michael Haimes, and Leslie Pack Kaelbling. Lifted probabilistic inference with counting formulae. In *Proceedings of the Twenty Third Conference on Advances in Artificial Intelligence (AAAI)*, pages 1062–1068, 2008.

[31] Tim Mitchell. Generative and discriminative classifiers: naive Bayes and logistic regression. http://www.cs.cmu.edu/ tom/mlbook/NBayesLogReg.pdf, 2010.

[32] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997.

[33] Stephen Muggleton. Stochastic logic programs. *Advances in inductive logic programming*, 32:254–264, 1999.

[34] Sriraam Natarajan, Tushar Khot, Daniel Lowd, Prasad Tadepalli, and Kristian Kersting. Exploiting causal independence in Markov logic networks: Combining undirected and directed models. In *European Conference on Machine Learning (ECML)*, 2010.

[35] Radford M. Neal. Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113, 1992.

[36] Jennifer Neville and David Jensen. Relational dependency networks. *The Journal of Machine Learning Research*, 8:653–692, 2007.

[37] Jennifer Neville, David Jensen, Lisa Friedland, and Michael Hay. Learning relational probability trees. In *Proc. of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003.

[38] Jennifer Neville, Ozgur Simsek, David Jensen, John Komoroske, Kelly Palmer, and Henry Goldberg. Using relational knowledge discovery to prevent securities fraud. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. MIT Press, 2005.

[39] Hanna Pasula and Stuart Russell. Approximate inference for first-order probabilistic languages. In *Proc. of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 741–748. Seattle: Morgan Kaufmann, 2001.

[40] Judea Pearl. Fusion, propagation and structuring in belief networks. *Artificial Intelligence*, 29(3):241–288, 1986.

[41] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaumann, San Mateo, CA, 1988.

[42] Claudia Perlish and Foster Provost. Distribution-based aggregation for relational learning with identifier attributes. *Machine Learning*, 62:65–105, 2006.

[43] David Poole. The independent choice logic and beyond. *Probabilistic inductive logic programming*.

[44] David Poole. Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence*, 64:81–129, 1993.

[45] David Poole. First-order probabilistic inference. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 985–991, Acapulco, 2003.

[46] David Poole, Fahiem Bacchus, and Jacek Kisynski. Towards completely lifted search-based probabilistic inference. *arXiv:1107.4035 [cs.AI]*, 2011.

[47] David Poole, David Buchman, Seyed Mehran Kazemi, Kristian Kersting, and Sriraam Natarajan. Population size extrapolation in relational probabilistic modelling. In *Proc. of the Eighth International Conference on Scalable Uncertainty Management*, 2014.

[48] David Poole, David Buchman, Sriraam Natarajan, and Kristian Kersting. Aggregation and population growth: The relational logistic regression and Markov logic cases. In *Proc. UAI-2012 Workshop on Statistical Relational AI*, 2012.

[49] David Poole and Alan K. Mackworth. *Artificial Intelligence: foundations of computational agents*. Cambridge University Press, 2010.

[50] David Poole and Nevin L. Zhang. Exploiting contextual independence in probabilistic inference. *Journal of Artificial Intelligence Research*, 18:263–313, 2003.

[51] Alexandrin Popescul, Lyle H. Ungar, Steve Lawrence, and David M. Pennock. Towards structural logistic regression: Combining relational and statistical learning. In *KDD Workshop on Multi-Relational Data Mining*, 2002.

[52] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.

[53] Lawrence K. Saul, Tommi Jaakkola, and Michael I. Jordan. Mean field theory for sigmoid belief networks. *arXiv preprint cs/9603102*, 1996.

[54] Raymond M. Smullyan. *First-order logic*. Vol. 6. Berlin: Springer-Verlag, 1968.

[55] Teodor Sommestad, Mathias Ekstedt, and Pontus Johnson. A probabilistic relational model for security risk analysis. *Computers and Security*, 29(6):659–679, 2010.

[56] Ben Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Proc. of the Eighteenth conference on Uncertainty in artificial intelligence*, 2002.

[57] Benjamin Taskar, Eran Segal, and Daphne Koller. Probabilistic classification and clustering in relational data. In *International Joint Conference on Artificial Intelligence*, volume 17, 2001.

[58] Michael P. Wellman, John S. Breese, and Robert P. Goldman. From knowledge bases to decision models. *The Knowledge Engineering Review*, 7(01):35–53, 1992.

[59] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbson, and Abraham Flaxman. Sybilguard: defending against sybil attacks via social networks. *ACM SIGCOMM Computer Communication Review*, 36(4):267–278, 2006.

[60] Nevin L. Zhang and David Poole. On the role of context-specific independence in probabilistic reasoning. pages 1288–1293, 1999.