Reference Classes and Relational Learning

Michael Chiang, David Poole

{*mchc,poole*}*@cs.ubc.ca.*
201-2366 Main Mall, Vancouver,
British Columbia, Canada, V6T1Z4.
Tel: +1 604 822 6625
Fax: +1 604-822-5485

Abstract

This paper studies the connections between relational probabilistic models and reference classes, with specific focus on the ability of these models to generate the correct answers to probabilistic queries. We distinguish between relational models that represent only observed relations and those which additionally represent latent properties of individuals. We show how both types of relational models can be understood in terms of *reference classes*, and that learning such models correspond to different ways of identifying reference classes. Rather than examining the impact of philosophical issues associated with reference classes on relational learning, we directly assess whether relational models can represent the correct probabilities of a simple generative process for relational data. We show that models with only observed properties and relations can only represent the correct probabilities under restrictive conditions, whilst models that also represent latent properties avoids such restrictions. As such, methods for acquiring latent-property models are an attractive alternatives to traditional ways of identifying reference classes. Our experiments on synthetic as well as real-world domains support the analysis, demonstrating that models with latent relations are significantly more accurate than those without latent relations.

Keywords: Relational learning, prediction, reference class, clustering, latent-variable models

1. Introduction

The *reference class problem* [41] is a long-standing problem in philosophy, where the works of Kyburg [22, 23, 21] have been influential. The goal in solving the reference class problem is a procedure for identifying the *right reference class*; one that predicts the *correct probability* for any probabilistic query. Correctness in prediction in turn validates knowledge upon which the predictions were drawn.

Given some outcome of interest, say F, a reference class is a population sample for measuring the rate – specifically a *proportion* – of F. Relative to a

particular individual x, the *right reference class* is one proportion of F matches the *correct probability* that F is true for x. Using this proportion as the probability that F is true for x is an act of *direct inference* (see [27, 29]). Direct inference with reference classes is commonplace. For example, suppose it is known that 20% of trains due at Central Station between the hours of 2pm to 4.30pm this month have arrived late. We wish to predict the probability that the next train due at Central Station between 2pm and 4.30pm will be late. By direct inference, we conclude that the answer is 0.2. The reference class is a main tool in the theory of intuitive judgement [14] for correcting human bias in decision making.

This goal of this paper is to understand models used in *relational learning* (the inductive acquisition of (statistical) models about individuals and relations amongst them) and assess whether they can represent the correct probabilities to answer probabilistic queries. Our point of departure is that relational models considered can be expressed in terms of reference classes, explained in Section 3 and 4. Given the ties with reference classes, the relational models in question must confront questions concerning reference classes regarding, for example, the justifiability and possibility of finding the right reference class [41, 22, 6, 23, 27, 21, 29].

In this work, we study how relational models can achieve the correct answers to probabilistic queries directly, with respect to the underlying generative process of data. As the true generative process of data is inaccessible, we assume a particularly simple generative process. We use the parameters of the generative process to derive predictions by our models, and compare these predictions to the correct probabilities that are also obtained from the generative process (Section 5).

We consider two widely-studied classes of relational models: those built from *observed properties and relations* only, and those which also include *latent properties* of individuals.

Relational models that represent only observed properties and relations have been well-studied [25, 33, 8, 4]. They are useful for generalising occurrences of relations amongst individuals, as well as modelling how one relation can depend on another. To illustrate, assume a social domain where a number of relations amongst people are recorded [43], e.g. friendship (written as $\mathsf{friends}(X, Y)$), preferences (for example $\mathsf{likes}(X, Y)$), or esteem (written as $\mathsf{esteem}(X, Y)$). The kind of relational models discussed here model dependency structures over such relations. For instance the dependency "$\mathsf{friends}(X, Y)$ depends on $\mathsf{likes}(X, Y)$ and $\mathsf{esteem}(X, Y)$" states that for all pairs of individuals (X, Y), the value of the friendship predicate for (X, Y) depends on whether X likes Y and whether X holds Y in esteem. We show that such relational models represent a set of reference classes, where each reference class is defined by some logical description using observed properties and relations. Inference using such models is akin to direct inference (see [26, 29]).

The second class of relational models we consider additionally introduces *latent properties* of individuals. In such models, latent properties are used to explain relations. For example, latent properties of individual x and y are used

to explain whether $\mathsf{likes}(x, y)$ holds. This approach has been useful in in network analysis [35, 10, 11, 1] as well as collaborative filtering [47, 12, 28, 20], where typically only one observed relation is given. Latent properties are commonly interpreted as *clusters*, where a cluster consists of all individuals that share the same latent property values. We explain in Section 4 that latent-property models define reference classes in a way that allows *disjunctions of equalities* in the language.

In Section 5, we show that relational models that do not represent latent properties can only entail the correct probabilities under restrictive circumstances relating to the underlying generative process. Latent-property models, on the other hand, can model the correct probabilities without such restrictions. Empirically, using synthetic and real-world data, we demonstrate in Section 6 that latent-property models achieve significantly lower empirical loss[1] (i.e. better accuracy) in probability estimation on both training and test data.

2. Preliminaries

In this work we use both predicate logic and probability notations. This section covers the notation necessary for expressing relational probabilistic models and relational data.

2.1. Predicate Language

To begin with, *constants* are expressed in lower-case, e.g. *joe* or *venus*, and are used to represent individuals. A *type* is associated with each individual, e.g. *joe* is a *person*. We use $\mathcal{D}(\tau)$ to represent a domain of type τ, which is the set of individuals of type τ. Types are assumed disjoint – that for any pair $\tau_i \neq \tau_j$, $\mathcal{D}(\tau_i) \cap \mathcal{D}(\tau_j) = \emptyset$. A *logical variable* is written in upper-case (e.g. X or $Person$) and denotes some individual. A logical variable is also typed, e.g. $Person$ denotes some member of $\mathcal{D}(\tau)$.

A *relation* is given by

$$r : \Omega_r \rightarrow V_r$$

where r is the name of the relation, $\Omega_r = \mathcal{D}(\tau_1) \times \ldots, \times \mathcal{D}(\tau_a)$ is the *domain of the relation*, and $T_r = (\tau_1, \ldots, \tau_a)$ is the *type of the relation*. $V_r = \{v_1, \ldots, v_k\}$ is the *range of the relation* – an enumerated set of values not appearing in any domain. Number a and k are positive integers denoting the *arity* and *size* of r; relation r is thus referred to as a k-valued a-ary relation. When $a = 1$, r is a *unary relation*. In this paper, a unary relation is also referred to as a *property*. When $V_r = \{\mathsf{F}, \mathsf{T}\}$, where F, T are Boolean values, r is a Boolean relation. (Note that this description of a relation is more general than defined in standard predicate logic, as we are interested in representing multi-valued relations in addition to Boolean relations.)

[1] Empirical loss is an asymptotically consistent measure of discrepancies of probability estimates. That is, the minimiser of loss, in the limit of infinite data, is the expected value of the outcome of interest which is, in turn, the true probability of the outcome.

An *atom* is an expression of the form $r(\sigma_1, \ldots, \sigma_a)$ where each σ_i is either a constant or logical variable. The types of $\sigma_1, \ldots, \sigma_a$ must match the type of r. If all of $\sigma_1, \ldots, \sigma_a$ are constants, $r(\sigma_1, \ldots, \sigma_a)$ is a *ground atom*.

A *literal* specifies the value of an atom, e.g. $r(X_1, \ldots, X_a) = v$ where $v \in V_r$. A literal that contains no logical variables, a ground literal, is a proposition. For a Boolean relation r, the literal $r(X_1, \ldots, X_a) = \mathsf{T}$ is written simply as $r(X_1, \ldots, X_a)$, and $r(X_1, \ldots, X_a) = \mathsf{F}$ is written as $\neg r(X_1, \ldots, X_a)$. A literal is also a *formula*.

Formulae with multiple literals are formed using connectives $\wedge$ and/or $\vee$. Connecting literals using only $\wedge$ forms a *conjunctive formula* or *conjunction*, e.g. $\neg\mathsf{pass}(Student) \wedge \mathsf{difficulty}(Course) = high$. A *disjunctive formula* or *disjunction* is formed using only $\vee$, e.g. $\neg\mathsf{pass}(Student) = high \vee \neg\mathsf{difficulty}(Course) = high$.

A substitution is a set $\theta = \{X_1 \backslash x_1, \ldots, X_k \backslash x_k\}$ where X_i are distinct logical variables and x_i are constants. When applied to a formula f, each occurrence of X_i in f is replaced with x_i. We denote the application of substitution of θ to f as $f\theta$. For example, suppose f is $a(X) = u \wedge \neg b(X, Y)$ and $\theta = \{X \backslash x, Y \backslash y\}$, to $f\theta$ is then $a(x) = u \wedge \neg b(x, y)$. If there are no logical variables $f\theta$, θ is called a *grounding substitution*. We also allow substitutions for (sets of) atoms, e.g. for $b(X, Y)\theta$ is $b(x, y)$, and for the set g given by $\{a(X), b(X, Y)\}$, $g\theta$ is $\{a(x), b(x, y)\}$.

Given some formula f containing logical variables $X_1, \ldots, X_n$, where each X_i has type τ_i, let the domain of f be $\Omega_f = \mathcal{D}(\tau_i) \times \ldots, \times \mathcal{D}(\tau_n)$. The *substitution space* of f, Γ_f, is the set of all possible grounding substitutions for f, given by

$$\Gamma_f = \{\{X_1 \backslash x_1, \ldots, X_n \backslash x_n\} : (x_1, \ldots, x_n) \in \Omega_f\}$$

For example, if formula f is $a(X) \wedge b(X, Y)$, where $\Omega_f = \mathcal{D}(\tau_X) \times \mathcal{D}(\tau_Y)$, then Γ_f is $\{\{X \backslash x, Y \backslash y\} : (x, y) \in \Omega_f\}$.

2.2. Relational Data

A *dataset* for relation r is a non-empty set $\mathbb{D}_r = \{d_1, \ldots, d_m\}$. Each d_i is a tuple of the form $\langle x_1, \ldots, x_a, v\rangle$ where $(x_1, \ldots, x_a) \in \Omega_r$ and $v \in V_r$. If $\langle x_1, \ldots, x_a, v\rangle \in \mathbb{D}_r$ and $\langle x_1, \ldots, x_a, v'\rangle \in \mathbb{D}_r$, then $v = v'$. A *database* is a set of datasets, where no more than one dataset for each relation. The following defines what a database entails.

$$\begin{array}{lllll} \text{(i)} & \mathbb{D} & \models \mathsf{T} & & \\ \text{(ii)} & \mathbb{D} & \models (r(x_1, \ldots, x_a) = v) & \text{iff } \langle x_1, \ldots, x_a, v\rangle \in \mathbb{D}_r & \\ \text{(iii)} & \mathbb{D} & \models \alpha \wedge \beta & \text{iff } \mathbb{D} \models \alpha \wedge \mathbb{D} \models \beta & \\ \text{(iv)} & \mathbb{D} & \models \alpha \vee \beta & \text{iff } \mathbb{D} \models \alpha \vee \mathbb{D} \models \beta & \end{array} \tag{1}$$

where α, β are formulae. We say that r is an *observed relation* if $\mathbb{D}_r \neq \emptyset$, and is a *latent relation* otherwise.

Counts can be obtained from a database $\mathbb{D}$ via logical formulae. The *count* of cases satisfying formula f with respect to $\mathbb{D}$ is given by

$$\#_{\mathbb{D}}(f) = \sum_{\theta \in \Gamma_f} \mathbb{I}(\mathbb{D} \models f\theta) \tag{2}$$

where $\mathbb{I}(s)$ is a characteristic function; returns 1 if s holds, and 0 otherwise.

2.3. Probability

Relational probabilistic models are expressed in languages that combine first-order logic (FOL) and probability. Each ground atom in the model is treated as a random variable, and a proposition (ground literal) is an instantiation of the random variable. Free logical variables are assumed to be universally quantified unless stated otherwise. The notation $P(a(x) = v)$ denotes the probability of the proposition $a(x) = v$. Since all relations have discrete ranges, all random variables are therefore discrete in this work.

3. Modelling with Observed Relations

This section gives an account of reference classes, relational models defined with only observed relations, and how the latter can be understood in terms of reference classes.

3.1. Reference Classes

A *reference class* is a set of tuples of individuals (an individual is a 1-tuple). Logical formulae are commonly used to define reference classes [22, 27, 21, 2] where the tuples of individuals in the reference class satisfy the given formula. For example, the description "tall and athletic" is used to define a set of tall and athletic individuals.

Let $\mathcal{X} = \{X_1, \ldots, X_k\}$ be a set of logical variables, and f a formula where all variables in f appear in $\mathcal{X}$. A reference class can be defined as

$$\mathbb{C}^{\mathbb{D}}_{\mathcal{X}}(f) = \{\langle x_1, \ldots, x_k\rangle : \mathbb{D} \models f\{X_1 \backslash x_1, \ldots, X_k \backslash x_k\}\} \tag{3}$$

where $\{X_1 \backslash x_1, \ldots, X_k \backslash x_k\}$ is a substitution. We abbreviate $\mathbb{C}^{\mathbb{D}}_{\mathcal{X}}$ by $\mathbb{C}(f)$ when $\mathcal{X}$ and $\mathbb{D}$ can be understood from context. The following discusses examples of reference classes.

Example 1. Assume database $\mathbb{D} = \{\mathbb{D}_{\mathsf{tall}}, \mathbb{D}_{\mathsf{athletic}}\}$, and logical variable set $\mathcal{X} = \{Person\}$. The formula $\mathsf{tall}(Person)$ (where *tall* is Boolean) defines the reference class $\mathbb{C}(\mathsf{tall}(Person))$ that consists of the set of tall individuals. Similarly, the formula $\mathsf{tall}(Person) \wedge \mathsf{athletic}(Person)$ yields a more specific reference class $\mathbb{C}(\mathsf{tall}(Person) \wedge \mathsf{athletic}(Person))$.

Whilst the formula $\mathsf{tall}(Person)$ is logically equivalent to $(\mathsf{tall}(Person) \wedge \mathsf{athletic}(Person)) \vee (\mathsf{tall}(Person) \wedge \neg\mathsf{athletic}(Person))$, the reference class $\mathbb{C}(\mathsf{tall}(Person))$ is not the same as

$$\mathbb{C}(\mathsf{tall}(Person) \wedge \mathsf{athletic}(Person)) \cup \mathbb{C}(\mathsf{tall}(Person) \wedge \neg\mathsf{athletic}(Person))$$

because $\mathbb{C}(\mathsf{tall}(Person) \wedge \mathsf{athletic}(Person))$ and $\mathbb{C}(\mathsf{tall}(Person) \wedge \neg\mathsf{athletic}(Person))$ include only individuals for whom *athletic* is observed in the database, whereas $\mathbb{C}(\mathsf{tall}(Person))$ also includes tall individuals for whom *athletic* is not observed.

A special reference class $\mathbb{C}(\mathsf{T})$ is obtained when the formula T is used. For this example, $\mathbb{C}(\mathsf{T})$ represents the set of all individuals in $\mathcal{D}(person)$, regardless of what properties are observed about them.

Consider a case where reference classes consist of non-singleton tuples of individuals, e.g. for relational domains.

Example 2. Assume database $\mathbb{D} = \{\mathbb{D}_{\mathsf{technical}}, \mathbb{D}_{\mathsf{mathematical}}, \mathbb{D}_{\mathsf{pass}}\}$, logical variables $\mathcal{X} = \{Student, Course\}$, and Boolean relations

$$\mathsf{technical} : \mathcal{D}(student) \to \{\mathsf{F}, \mathsf{T}\}$$
$$\mathsf{mathematical} : \mathcal{D}(course) \to \{\mathsf{F}, \mathsf{T}\} \mathsf{pass} : \mathcal{D}(student) \times \mathcal{D}(course) \to \{\mathsf{F}, \mathsf{T}\}$$

which represent whether a student is technically minded, whether a course is mathematical, and whether a student passes a course, respectively. The following reference classes can be defined

(i) $\mathbb{C}(\mathsf{T})$ – the set of all student-course tuples.

(ii) $\mathbb{C}(\mathsf{technical}(S))$ – the set of all student-course tuples with technically-minded students.

(iii) $\mathbb{C}(\mathsf{mathematical}(C))$ – the set of all student-course tuples with courses that involve mathematics.

(iv) $\mathbb{C}(\mathsf{technical}(S) \wedge \mathsf{mathematical}(C))$ – the set of all student-course tuples with technically-minded students and courses that involve mathematics.

Reference classes can be constructed to predict the probability of particular propositions. The proposition of interest is a *query*.

Example 3. We construct reference classes to predict the probability of the query $\mathsf{pass}(t.smith, math120) = \mathsf{T}$.

Let $\mathsf{eq\_student} : \mathcal{D}(student) \times \mathcal{D}(student) \to \{\mathsf{F}, \mathsf{T}\}$ and $\mathsf{eq\_course} : \mathcal{D}(course) \times \mathcal{D}(course) \to \{\mathsf{F}, \mathsf{T}\}$ be equality relations for student and courses respectively[2], e.g. where $\mathsf{eq\_student}(S, S') = \mathsf{T}$ if and only if S and S' are the same individual, and similarly for $\mathsf{eq\_course}(C, C')$. The following reference classes are possible.

(i) $\mathbb{C}(\mathsf{T})$ – the set of all student-course tuples.

(ii) $\mathbb{C}(\mathsf{eq\_student}(S, t.smith))$ – the set of all student-course tuples where the student is $t.smith$.

(iii) $\mathbb{C}(\mathsf{eq\_course}(C, math120))$ – the set of all student-course tuples where the course is $math120$.

(iv) $\mathbb{C}(\mathsf{eq\_student}(S, t.smith), \mathsf{eq\_course}(C, math120))$ – the set containing the tuple $\langle t.smith, math120 \rangle$.

[2] An equality relation is defined separately for students and courses in the interest of maintaining the type formalism for relations.

Here we assume that relations $\mathsf{eq\_student}(\cdot)$ and $\mathsf{eq\_course}(\cdot)$ are built-in – as often done in programming languages – and not explicitly defined in the input database as an observed relation. They are observed relations as they are well-defined for every member of $\mathcal{D}(student)$ and $\mathcal{D}(course)$.

The reference class $\mathbb{C}\left(\mathsf{eq\_student}(S, t.smith) \wedge \mathsf{eq\_course}(C, math120)\right)$ is interesting, because it consists of a single tuple $\langle t.smith, math120\rangle$, which also appears in the query. If the proposition queried is observed (i.e. entailed by $\mathbb{D}$) then the reference class returns 0 or 1 depending on the observed value. That is, if $\mathsf{pass}(t.smith, math120)$ is observed to be false according to $\mathbb{D}$, then the answer to our query is 0. If the proposition if not observed, then the reference class is empty and cannot define an answer.

The purpose of defining a reference class is to measure the proportion that some outcome of interest holds. Assume atom h denotes our outcome of interest, and we wish to measure the proportion of $h = v$ in some reference class $\mathbb{C}^{\mathbb{D}}_{\mathcal{X}}(f)$, the proportion – which we call the *reference class statistic* – is given by

$$\mathbb{P}(h = v \mid f) = \frac{|\mathbb{C}(f \wedge h = v)|}{\sum_u |\mathbb{C}(f \wedge h = u)|} = \frac{\#_{\mathbb{D}}\left(f \wedge h = v\right)}{\sum_u \#_{\mathbb{D}}\left(f \wedge h = u\right)} \tag{4}$$

Example 4. Continuing from Example 2, suppose we are interested in measuring the pass rate of technically-minded students over all courses, i.e. we seek the proportion for which $\mathsf{pass}(Student, Course) = \mathsf{T}$ holds in the reference class $\mathbb{C}^{\mathbb{D}}_{\mathcal{X}}(\mathsf{technical}(Student))$ ($\mathbb{D}$ and $\mathcal{X}$ are given in Example 2). The proportion sought is

$$\begin{aligned}&\mathbb{P}(\mathsf{pass}(Student, Course) \mid \mathsf{technical}(Student))\\ &= \frac{\#_{\mathbb{D}}\left(\mathsf{technical}(Student) \wedge \mathsf{pass}(Student, Course)\right)}{\begin{array}{r}\#_{\mathbb{D}}\left(\mathsf{technical}(Student) \wedge \mathsf{pass}(Student, Course)\right) +\\ \#_{\mathbb{D}}\left(\mathsf{technical}(Student) \wedge \neg\mathsf{pass}(Student, Course)\right)\end{array}}\end{aligned}$$

3.2. Relational Probabilistic Models

A key goal in relational learning is to learn a model that generalises examples in a given domain [25, 33, 36, 8, 4]. Such models generalise the given examples by abstracting over domain individuals. We illustrate with the following example.

Example 5. In the education domain of Example 2, suppose the database $\mathbb{D}$ contains the following set of examples

$$\left\{\begin{array}{ll}\mathsf{pass}(j.smith, cs100), & \mathsf{technical}(j.smith),\\ \neg\mathsf{pass}(m.jones, bio120), & \mathsf{technical}(m.jones),\\ \mathsf{pass}(s.wang, math120), & \mathsf{technical}(s.wang),\\ \neg\mathsf{pass}(x.ahn, math120), & \neg\mathsf{technical}(x.ahn),\\ \mathsf{pass}(k.stevens, latin120), & \neg\mathsf{technical}(k.stevens),\\ \mathsf{pass}(x.ahn, phil101), & \ldots\end{array}\right\} \tag{5}$$

The relational probabilistic models in question can model the probability that $\mathsf{pass}(Student, Course)$ is true for any student-course pair. Similarly, it can

model the probability of $\mathsf{technical}(Student)$ being true for any student, as well as how $\mathsf{pass}(Student, Course)$ may probabilistically depend on $\mathsf{technical}(Student)$ or vice versa. In other words, the relational probabilistic models discussed here represent generalisations of relations and their dependencies over individuals.

Learning programs in first-order logic (FOL) is one approach to obtain such models, and underpins research in ILP [33, 36]. The limited ability of FOL for handling quantitative uncertainty, however, led to languages that extend the semantics of FOL to include probability semantics [9, 16, 38, 32, 39, 44, 7, 30, 42], with proposals for learning in these languages also emerging [7, 17, 18]. We refer models in these languages as *relational probabilistic models*, which we will focus on in the rest of this paper. We further specialise for relational probabilistic models that combine FOL and Bayesian networks of Pearl [37], e.g. [38, 32, 39, 44, 7, 16, 30], which represents the majority of relational probabilistic models proposed, with the notable exception of Markov logic [42].

A relational probabilistic model consists of tuples of the form $\langle \theta, h \leftarrow b \rangle$ – which we call a *probabilistic clause* – where h is the *head* literal and b is a conjunction called the *body*. $h \leftarrow b$ is a logical clause and parameter θ represents the conditional probability $P(h \mid b)$. The discussion that follows explains how each probabilistic clause relates to reference classes.

Depending on the settings of logical variables in the head and body, a (probabilistic) clause can be *constrained* or *non-constrained* [36]. In the following we show how these relate to reference classes.

3.2.1. The Constrained Case

When all logical variables that appear in b also appear in h, the clause is *constrained*. Let $\mathcal{X} = \{V_1, \ldots, V_n\}$ be logical variables appearing in the clause, and $\{\tau_1, \ldots, \tau_n\}$ the respective types, the domain of the clause is then $\mathcal{D}(\tau_1) \times \ldots \times \mathcal{D}(\tau_n)$. Tuples in the domain that satisfy b – with respect to the database $\mathbb{D}$ – represent the reference class $\mathbb{C}^{\mathbb{D}}_{\mathcal{X}}(b)$.

Example 6. Consider the constrained clause (using the domain illustrated in Example 2)

$$\langle \theta, \mathsf{pass}(Student, Course) \leftarrow \mathsf{technical}(Student) \rangle$$

where relative to database $\mathbb{D}$, the body defines the set of student-course pairs that involve only technically-minded students, and in turn represents the reference class $\mathbb{C}^{\mathbb{D}}_{\{Student, Course\}}(\mathsf{technical}(Student))$.

Parameter θ directly represents the conditional probability $P(h \mid b)$. For discrete models, θ's maximum likelihood value is the proportion of student-course pairs in $\mathbb{C}(\mathsf{technical}(Student))$ such that $\mathsf{pass}(Student, Course)$ is true, as in Equation 4, and represents the reference class statistic $\mathbb{P}(\mathsf{pass}(Student, Course) \mid$

$\mathsf{technical}(Student)$).

$$\theta = \frac{\#_{\mathbb{D}}\left(\begin{array}{l}\mathsf{pass}(Student, Course)\wedge\\ \mathsf{technical}(Student)\end{array}\right)}{\#_{\mathbb{D}}\left(\begin{array}{l}\mathsf{pass}(Student, Course)\wedge\\ \mathsf{technical}(Student)\end{array}\right) + \#_{\mathbb{D}}\left(\begin{array}{l}\neg\mathsf{pass}(Student, Course)\wedge\\ \mathsf{technical}(Student)\end{array}\right)}$$

Figure 1 illustrates how different reference classes and statistics can be obtained, using the data shown in Equation 5. In particular, each conditioning step (from top to bottom) specialises the body of a clause and yields a narrower reference class. Reference classes at the leaves of the tree are the most specific.

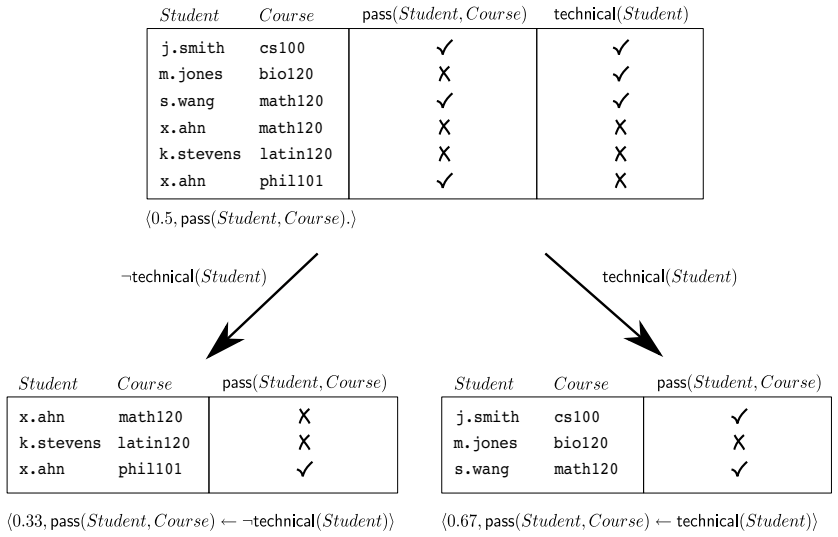


Figure 1: The simple student-course domain, where $\mathsf{pass}(Student, Course)$ is a Boolean relation denoting whether *Student* passes *Course*. The property $\mathsf{technical}(Student)$ is also Boolean, denoting whether *Student* is technically-minded. Observed cases of Equation 5 in Example 5 are split depending on the value of $\mathsf{technical}(Student)$. Each table represents a reference class and associated data, and corresponding probabilistic clauses are show below each table.

3.2.2. The Non-constrained Case

When there are logical variables in b that are not in h, the probabilistic clause is *non-constrained*, and θ is calculated depending on how the additional body variables are quantified.

Example 7. Consider the probabilistic clause

$$\langle \theta, \mathsf{pass}(Student, Course) \leftarrow \mathsf{passed\_mod}(Student, Course, Mod)\rangle \qquad (6)$$

where $\mathsf{passed\_mod}(Student, Course, Mod)$ is an observed Boolean relation indicating whether a student has passed a course module indicated by logical variable Mod.

The probability that a given student passes a given course depends on which modules of that course the student has passed. Specifically, assuming there are n modules per course, θ parameterises the conditional probability table

$$P\left(\mathsf{passed}(Student, Course) \mid \left\{ \begin{array}{l} \mathsf{passed\_mod}(Student, Course, mod_i) = v_i \\ : i = 1, \ldots, n \end{array} \right\}\right)$$

where $v_i \in \{\mathsf{F}, \mathsf{T}\}$. The immediate problem is that the size of this conditional probability table is exponential in n, which quickly becomes representationally infeasible. A common approach is to assume that each condition is independent of another[3], leading to a more feasible representation, i.e.

$$\left\{ \begin{array}{l} \phi\left(\mathsf{passed}(Student, Course) \mid \mathsf{passed\_mod}(Student, Course, mod_1) = v_1\right) \\ \phi\left(\mathsf{passed}(Student, Course) \mid \mathsf{passed\_mod}(Student, Course, mod_2) = v_2\right) \\ \ldots \\ \phi\left(\mathsf{passed}(Student, Course) \mid \mathsf{passed\_mod}(Student, Course, mod_n) = v_n\right) \end{array} \right\} \tag{7}$$

where each member of the set expresses a conditional probability, and has a size that is independent of n. Suppose the conditional probabilities in Equation 7 have parameters $\theta^{(1)}, \ldots, \theta^{(n)}$ respectively, θ for Equation 6 can be obtained by combining $\theta^{(1)}, \ldots, \theta^{(n)}$, which is commonly done combination functions such as the noisy-OR (see [13, 16]).

At this point, observe that each member of the conditional probability set in Equation 7 can be modelled by a constrained probabilistic clause. Since each constrained probabilistic clause represents a reference class and its statistic, a range-restricted probabilistic clause represents *a set of reference classes* and a *combined statistic* computed from the individual reference class statistics.

3.2.3. Learning

The above shows that relational probabilistic models represent a set of reference classes and associated statistics, and that inference using such models amount to direct inference. For identifying the *right reference class*, the *specificity principle* has been at the centre of philosophical discussions [41]. Here we comment on how existing methods for learning relational probabilistic models relate to the specificity principle for reference classes.

Learning relational probabilistic models boil down to learning the clausal structure and parameter of each probabilistic clause. For a probabilistic clause $\langle \theta, h \leftarrow b \rangle$, *parameter learning* pertains to learning θ when given $h \leftarrow b$. *Structure learning* refers to learning $h \leftarrow b$ – namely, find the best conjunction b – such that the model best fits the data. Two approaches are possible for structure learning: top-down [40, 25] or bottom up [31].

The top-down approach can be visualised in Figure 1, where the initial dataset is split successively by conditioning on a new relation at each step.

[3]Assuming independent conditional influences also referred to as *causal independence* [51], where any joint influence of multiple conditions are neglected.

The relation chosen at each step is done in a greedy manner. The top-down process is a general-to-specific search. The bottom-up approach reverses this process: starting with the most specific clauses, successively compute the best generalising clause, thus performing a specific-to-general search. In top-down and bottom-up approaches, the goal is to find the best clause that fits the data whilst adhering to some regularisation constraint to avoid over-fitting (see [33, 25, 4]).

The relational learning procedures described can be seen as implementations of the specificity principle, in the sense that they seek the *most specific* reference class [41, 22, 21, 27]. The difference is that these procedures adhere to heuristic regularisation constraints to avoid over-fitting. The regularisation principle also ensures that even when a large set of relations are available to construct very specific clauses (specific reference classes), such specific clauses will be avoided if they violate the regularisation constraint. Also, narrow reference classes with insufficient statistics due to small sample sizes can also be avoided.

4. Modelling with Latent Properties of Individuals

It is often the case in the analysis of user preferences (e.g. in collaborative filtering [12, 28, 20]) and social systems [15, 49, 10, 1, 48] that latent properties of individuals are introduced to explain the observed relations amongst individuals. Modelling latent properties for individuals is better known as *clustering*; i.e. setting the value of a latent property for an particular individual is tantamount to assigning that individual to a cluster, where possible values of the latent property represent distinct clusters. Clustering domain individuals in turn induces clusters of observed data for properties and relations.

We distinguish between two types of clustering models for relational data – *hard-clustering* and *soft-clustering* models.

4.1. Hard-clustering

Consider latent property $\alpha : \mathcal{D}(\tau) \rightarrow \{\mathsf{F}, \mathsf{T}\}$. Hard-clustering with α hard-assigns each $x \in \mathcal{D}(\tau)$ to one of two clusters denoted respectively by $\alpha(x) = \mathsf{F}$ and $\alpha(x) = \mathsf{T}$. Namely, it induces two clusters $\{x : x \in \mathcal{D}(\tau), \alpha(x) = \mathsf{T}\}$ and $\{x : x \in \mathcal{D}(\tau), \alpha(x) = \mathsf{F}\}$. Latent properties of individuals are often used to explain observed relations. Relational clustering models in the machine learning literature are often hard-clustering models [12, 15, 49, 10, 1, 11, 20].

Revisiting the student-course example (Example 2), we consider a model involving the observed relation $\mathsf{pass}(Student, Course)$ as being probabilistically dependent on some latent property $\alpha(Student)$ of students and some latent property $\beta(Course)$ of courses. Treating the latent properties as if they are observed – i.e. that latent property values are observed for every individual – observed examples for $\mathsf{pass}(Student, Course)$ are split by conditioning on $\alpha(Student)$ and $\beta(Course)$ (shown in Figure 2 below) in a similar way to that shown in Figure 1.

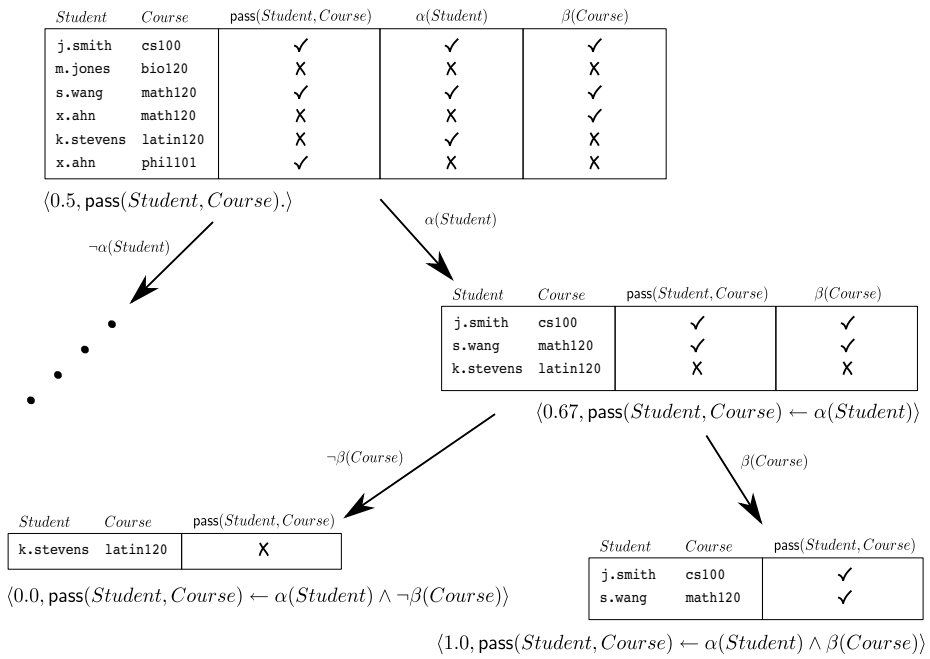


Figure 2: A simple student-course domain, where $\mathsf{pass}(Student, Course)$ is a Boolean relation denoting whether $Student$ passes $Course$. $\alpha(Student)$ and $\beta(Course)$ represent latent (Boolean) properties of students and courses respectively. The illustration splits observed examples of $\mathsf{pass}(Student, Course)$ given by Equation 5 by conditioning on values of the latent properties of individuals. The subtree for $\neg\alpha(Student)$ is not shown. Each table represents a reference class and associated data. Corresponding probabilistic clauses are show below each table.

As $\alpha(Student)$ and $\beta(Course)$ are not observed, their values must be inferred from the available data, using algorithms such as EM (expectation maximisation)[4] [5]. In Figure 2, it is assumed that latent property values for each individual have already been inferred.

Conditioning on latent properties produces reference classes and their statistics in the same way as conditioning on observed properties and relations (compare Figure 1 and 2). The key difference is that, in a latent-property model, an additional inference step is required to compute values of latent properties prior to conditioning. An interesting aspect of latent-property models is that the latent property values can be generated to optimise the model's fit to data.

Another important attribute of latent properties is that they introduce *disjunctions of equalities* to the modelling language, thereby eliciting a richer space of reference classes than that from observed properties and relations only. To

[4]For non-trivial models where there are many correlated latent random variables, approximate inference techniques based on Monte Carlo sampling and variational Bayes are common (see [19] for a general overview).

illustrate, consider the probabilistic clause

$$\langle \theta, r(X,Y) \leftarrow \alpha(X) \wedge \beta(Y) \rangle \tag{8}$$

where r has type (τ_1, τ_2). Suppose that an inference procedure is used to obtain the estimated value of $\alpha(x)$ for all $x \in \mathcal{D}(\tau_1)$ and the value of $\beta(y)$ for all $y \in \mathcal{D}(\tau_2)$. Let $S_\alpha = \{x : x \in \mathcal{D}(\tau_1), \alpha(x) = \mathsf{T}\}$ denote the set containing all $\alpha(x)$ whose inferred value is T, and $S_\beta = \{y : y \in \mathcal{D}(\tau_2), \beta(y) = \mathsf{T}\}$ the set of all $\beta(y)$ whose inferred value is T. Then, $r(X,Y) \leftarrow \alpha(X) \wedge \beta(Y)$ can be expressed directly in terms of disjunctions of equalities as follows

$$r(X,Y) \leftarrow \bigvee_{x \in S_\alpha} \mathsf{eq\_1}(X,x) \wedge \bigvee_{y \in S_\beta} \mathsf{eq\_2}(Y,y) \tag{9}$$

where $\mathsf{eq\_1}(\cdot)$ and $\mathsf{eq\_2}(\cdot)$ are equality relations (first introduced in Example 3) for elements of type τ_1 and τ_2 respectively. Note that Equation 9 is logically equivalent to

$$r(X,Y) \leftarrow \bigvee_{x \in S_\alpha, y \in S_\beta} \mathsf{eq\_1}(X,x) \wedge \mathsf{eq\_2}(Y,y) \tag{10}$$

Referring again to the student-course example (Example 2), the probabilistic clause

$$\langle 1.0, \mathsf{pass}(Student, Course) \leftarrow \alpha(Student) \wedge \beta(Course) \rangle$$

is equivalent to

$$\left\langle \begin{array}{ll} 1.0, & \mathsf{pass}(Student, Course) \leftarrow \\ & (\mathsf{eq\_student}(Student, j.smith) \vee \mathsf{eq\_student}(Student, s.wang)) \wedge \\ & (\mathsf{eq\_course}(Course, cs100) \vee \mathsf{eq\_course}(Course, math120)) \end{array} \right\rangle$$

and also

$$\left\langle \begin{array}{ll} 1.0, & \mathsf{pass}(Student, Course) \leftarrow \\ & (\mathsf{eq\_student}(Student, j.smith) \wedge \mathsf{eq\_course}(Course, cs100)) \vee \\ & (\mathsf{eq\_student}(Student, j.smith) \wedge \mathsf{eq\_course}(Course, math120)) \vee \\ & (\mathsf{eq\_student}(Student, s.wang) \wedge \mathsf{eq\_course}(Course, cs100)) \vee \\ & (\mathsf{eq\_student}(Student, s.wang) \wedge \mathsf{eq\_course}(Course, math120)) \end{array} \right\rangle$$

where members of disjunctions are taken from Figure 2.

Note that models using only observed properties and relations can also be written in terms of disjunctions of equalities. However, observed properties only represent particular disjunctions that are entailed by the database. For instance, the probabilistic clause $\mathsf{pass}(Student, Course) \leftarrow \mathsf{technical}(Student)$ can be expressed in terms of disjunctions of equalities, i.e.

$$\mathsf{pass}(Student, Course) \leftarrow \bigvee_{s \in S_{\mathsf{technical}}} \mathsf{eq\_student}(Student, s)$$

where the set $S_{\mathsf{technical}} = \{s : s \in \mathcal{D}(student), \mathbb{D} \models \mathsf{technical}(s)\}$ is determined by $\mathbb{D}$. Latent properties, on the other hand, are more general as they can represent arbitrary disjunctions of domain individuals.

The use of latent properties in the hard-clustering context, after values for all ground instances of latent properties are inferred, produces reference classes in the same manner as models with only observed relations. The inclusion of latent properties augments the modelling language with disjunctions of equalities and permits a richer set of reference classes.

4.2. Soft-clustering

Where hard-clustering places an individual to one cluster, soft-clustering specifies the probability that an individual belongs to each cluster. In terms of latent properties, a hard-clustering model specifies that $P(\alpha(x) = v)$ for some individual x is either 0 or 1, whereas a soft-clustering model allows non-extreme probabilities. A soft-clustering relational model corresponds to a weighted ensemble of hard-clustering relational models.

Consider the probabilistic clause shown in Equation 8 and assume that $r(X, Y)$, $\alpha(X)$, and $\beta(Y)$ are Boolean. Latent property $\alpha(X)$ represents two clusters, and individuals of $\mathcal{D}(\tau_1)$ are assigned to one cluster or the other in a hard-clustering model, and similarly for $\beta(Y)$. There are $2^{|\mathcal{D}(\tau_1)|}$ possible ways to cluster all members of $\mathcal{D}(\tau_1)$ to the two given clusters, and similarly there are $2^{|\mathcal{D}(\tau_2)|}$ ways to cluster members of $\mathcal{D}(\tau_2)$. A *joint assignment* specifies the clustering of each member of $\mathcal{D}(\tau_1)$ and $\mathcal{D}(\tau_2)$, where there are $2^{|\mathcal{D}(\tau_1)|+|\mathcal{D}(\tau_2)|}$ possible joint assignments. Each possible assignment yields a hard clustering, there are therefore $2^{|\mathcal{D}(\tau_1)|+|\mathcal{D}(\tau_2)|}$ possible hard clusterings.

To explain soft-clustering, consider the probabilistic clause

$$\langle \theta, \mathsf{pass}(Student, Course) \leftarrow \alpha(Student) \wedge \beta(Course) \rangle \tag{11}$$

where we assume that pass, α and β are Boolean. Let Λ be the space of all joint assignments of individuals in $\mathcal{D}(student)$ and $\mathcal{D}(course)$ to clusters, i.e. each $w \in \Lambda$ is a unique specification of the value of $\alpha(s)$ for each $s \in \mathcal{D}(student)$ and $\beta(c)$ for each $c \in \mathcal{D}(course)$. A soft-clustering model specifies a probability function Q over Λ, such that $\sum_{w \in \Lambda} Q(w) = 1$. The weight of any joint assignment w is represented by probability $Q(w)$.

In practise, where there are many domain individuals, representing Q is infeasible. For the student-course domain, Q must represent $2^{|\mathcal{D}(student)|+|\mathcal{D}(course)|}$ probabilities. A common simplification is to assume probabilistic independence[5], e.g. that $\alpha(j.smith)$ is probabilistically independent from other latent properties for other individuals. Under this assumption, Q represents the probability of $\alpha(s)$ for each $s \in \mathcal{D}(student)$ and $\beta(c)$ for each $c \in \mathcal{D}(course)$ separately. The representation size is now $(|\mathcal{D}(student)| + |\mathcal{D}(course)|)$. The probability of any joint assignment is then a product of individual probabilities. For example, if the joint assignment w specifies that $\alpha(s) = \beta(c) = \mathsf{T}$ for all

[5]There is a large body of work on approximations for large probabilistic models with many correlated latent variables. Notable examples include [50], and see [19] for a good overview.

$s \in \mathcal{D}(student)$ and $c \in \mathcal{D}(course)$, then

$$Q(w) = \left(\prod_{s \in \mathcal{D}(student)} Q\left(\alpha(s) = \mathsf{T}\right) \right) \left(\prod_{c \in \mathcal{D}(course)} Q\left(\beta(c) = \mathsf{T}\right) \right)$$

Note that if all probabilities returned by Q are extreme (i.e. 0 or 1), then we obtain a hard-clustering model. For instance, the student-course example as illustrated in Figure 2 begins with a hard-clustering corresponding to the joint assignment

$$\begin{array}{llll} \alpha(x.ahn) = \mathsf{F} & \wedge & \beta(bio120) = \mathsf{F} & \wedge \\ \alpha(m.jones) = \mathsf{F} & \wedge & \beta(cs100) = \mathsf{T} & \wedge \\ \alpha(j.smith) = \mathsf{T} & \wedge & \beta(latin120) = \mathsf{F} & \wedge \\ \alpha(k.stevens) = \mathsf{T} & \wedge & \beta(math120) = \mathsf{T} & \wedge \\ \alpha(s.wang) = \mathsf{T} & \wedge & \beta(phil101) = \mathsf{F} & \end{array}$$

which can be achieved by setting Q as follows

$$\begin{array}{llll} Q(\alpha(x.ahn) = \mathsf{T}) & = 0, & Q(\beta(bio120) = \mathsf{T}) & = 0, \\ Q(\alpha(m.jones) = \mathsf{T}) & = 0, & Q(\beta(cs100) = \mathsf{T}) & = 1, \\ Q(\alpha(j.smith) = \mathsf{T}) & = 1, & Q(\beta(latin120) = \mathsf{T}) & = 0, \\ Q(\alpha(k.stevens) = \mathsf{T}) & = 1, & Q(\beta(math120) = \mathsf{T}) & = 1, \\ Q(\alpha(s.wang) = \mathsf{T}) & = 1, & Q(\beta(phil101) = \mathsf{T}) & = 0 \end{array}$$

The probability function Q is used to define soft counts that are required for computing parameter θ. We describe soft counts as follows.

Definition 1. *Assume probability function Q, and a ground literal $r(x_1, \ldots, x_n) = v$ such that $\mathbb{D} \not\models (r(x_1, \ldots, x_n) = v)$. A* soft characteristic function *is*

$$\hat{\mathbb{I}}(r(x_1, \ldots, x_n) = v, Q) = \begin{cases} 1 \text{ if } r \text{ observed and } \mathbb{D} \models (r(x_1, \ldots, x_n) = v) \\ 0 \text{ if } r \text{ observed and } \mathbb{D} \models (r(x_1, \ldots, x_n) = v') \wedge v \neq v' \\ Q(r_i(x_1, \ldots, x_n) = v) \text{ otherwise} \end{cases} \tag{12}$$

Assume a formula $f = f_1 \wedge \ldots \wedge f_n$, a substitution $\theta \in \Gamma_f$ and probability function Q defined for all $f\theta'$ where $\theta' \in \Gamma_f$, then a soft characteristic function for $f\theta$ is

$$\tilde{\mathbb{I}}(f\theta, Q) = \hat{\mathbb{I}}(f_1\theta, Q) \cdot \hat{\mathbb{I}}(f_2\theta, Q) \cdots \hat{\mathbb{I}}(f_n\theta, Q)$$

Finally, a soft count *is given by*

$$\tilde{\#}_{\mathbb{D}}(f, Q) = \sum_{\theta \in \Gamma_f} \tilde{\mathbb{I}}(f\theta, Q) \tag{13}$$

(Note that if $\mathbb{D} \models f_i$, $i = 1, \ldots, n$, then Equation 13 is equivalent to the normal count (Equation 2).)

Using soft counts (Equation 13), the probability parameter θ of Equation 11 is given by

$$\theta_{\mathsf{pass}|\alpha,\beta} = \frac{\tilde{\#}_{\mathbb{D}}\left(\mathsf{pass}(Student, Course) \wedge \alpha(Student) \wedge \beta(Course), Q\right)}{\sum_{v} \tilde{\#}_{\mathbb{D}}\left(\mathsf{pass}(Student, Course) = v \wedge \alpha(Student) \wedge \beta(Course), Q\right)} \tag{14}$$

If Q has only extreme probability values, then Equation 14 is equivalent to Equation 4.

Equation 14 is the proportion that $\mathsf{pass}(Student, Course)$ is true in the set of student-course pairs where $\alpha(Student) \wedge \beta(Course)$ is true, with respect Q. Given Q, it can be seen that the reference class in question is specified by $\alpha(Student) \wedge \beta(Course)$, and the reference class statistic (Equation 14) is obtained by counting weighted cases of $\mathsf{pass}(Student, Course)$.

Answers to probabilistic queries using a soft-clustering model are weighted sums of reference class statistics. To illustrate, suppose we seek the probability that $\mathsf{pass}(j.smith, cs100)$ holds. Whilst in the preceding discussion we have only covered the reference class specified by $\alpha(Student) \wedge \beta(Course)$, we require the remaining reference classes, namely those given by $\alpha(Student) \wedge \neg\beta(Course)$, $\neg\alpha(Student) \wedge \beta(Course)$, and $\neg\alpha(Student) \wedge \neg\beta(Course)$. With these reference classes are associated statistics $\theta_{\mathsf{pass}|\alpha,\beta}$, $\theta_{\mathsf{pass}|\alpha,\neg\beta}$, $\theta_{\mathsf{pass}|\neg\alpha,\beta}$, and $\theta_{\mathsf{pass}|\neg\alpha,\neg\beta}$, all of which can be computed like in Equation 14. Given the probability function Q, the answer to the query $\mathsf{pass}(j.smith, cs100)$ is then

$$\begin{aligned}\hat{P}\left(\mathsf{pass}(j.smith, cs100)\right) = \quad & \theta_{\mathsf{pass}|\alpha,\beta} Q(\alpha(j.smith)) Q(\beta(cs100)) \quad + \\ & \theta_{\mathsf{pass}|\alpha,\neg\beta} Q(\alpha(j.smith)) Q(\neg\beta(cs100)) \quad + \\ & \theta_{\mathsf{pass}|\neg\alpha,\beta} Q(\neg\alpha(j.smith)) Q(\beta(cs100)) \quad + \\ & \theta_{\mathsf{pass}|\neg\alpha,\neg\beta} Q(\neg\alpha(j.smith)) Q(\neg\beta(cs100))\end{aligned}$$

where $\hat{P}(\cdot)$ denote an estimate of the true probability. This shows that inference using our soft-clustering model produces a weighted-sum of reference class statistics.

To summarise, we showed here that a soft-clustering model represents the set of hard-clustering models corresponding unique joint assignments of individuals to clusters defined by the latent properties, and specifies the weight (a probability) for each model. Since each hard-clustering model is shown to correspond to a reference class (and reference class statistic), soft-clustering models therefore represent a weighted-sum of reference classes. Reference class statistics obtained for the soft-clustering model uses soft counts, whereas deterministic counts are used in hard-clustering models. Inference with a soft-clustering model performs a weighted combination of reference class statistics defined by latent properties, whereas a hard-clustering model predicts by *selecting* a reference class statistic according to which clusters the queried individuals are assigned.

In the next section, we compare inferences made by relational models with only observed relations and those with additional latent properties, in the context of recovering the true underlying probabilities of propositions.

5. Understanding Relational Inference

As mentioned in the introduction, we seek to directly evaluate a given relational model's ability to produce the correct probability for given queries. Our approach involves assuming that the generative process of data is known, which enables us to express inferences from our models in terms of the correct probabilities, i.e. those represented by the generative process.

For our analysis we deliberately focus on a simple relational domain consisting of one observed relation whose examples are generated by two latent properties. We first describe the generative process that underlies this simple domain, then analyse in Section 5.3 relational models that contain only observed relations. In Section 5.4 we examine relational models that also model latent properties.

5.1. Generative Model

Our generative process reflects the common intuition that relations amongst individuals are attributed to (hidden) properties of participating individuals. The generative process of interest is defined as follows.

Definition 2. *We assume a Boolean relation* $r : \mathcal{D}(\tau_1) \times \mathcal{D}(\tau_2) \to \{\mathsf{F}, \mathsf{T}\}$*, where domains* $\mathcal{D}(\tau_1), \mathcal{D}(\tau_2)$ *are known, and two unary Boolean relations* $a : \mathcal{D}(\tau_1) \to \{\mathsf{F}, \mathsf{T}\}$ *and* $b : \mathcal{D}(\tau_2) \to \{\mathsf{F}, \mathsf{T}\}$*. We define a probabilistic model* $\mathcal{G}$ *over the set of all atoms of each relation.* $\mathcal{G}$ *is a Bayesian network with representing the joint distribution*

$$J = \prod_{(x,y) \in \mathcal{D}(\tau_1) \times \mathcal{D}(\tau_2)} p\left(a(x)\right) p\left(b(y)\right) p\left(r(x,y) \mid a(x), b(y)\right) \tag{15}$$

where $p(a(x))$ *is a Bernoulli distribution with parameter* γ_a *for all* $x \in \mathcal{D}(\tau_1)$*,* $p(b(y))$ *is a Bernoulli distribution with parameter* γ_b *for all* $y \in \mathcal{D}(\tau_2)$*, and* $p(r(x,y) \mid a(x), b(y))$ *is a conditional distribution specified parameters* $\gamma_{r|a,b}$*,* $\gamma_{r|\neg a,b}$*,* $\gamma_{r|a,\neg b}$*, and* $\gamma_{r|\neg a,\neg b}$*, corresponding to the four possible value settings of* $a(x)$ *and* $b(y)$*. The Bayesian network is illustrated in Figure 3.*

5.2. Sampling from $\mathcal{G}$

Relation $r(X, Y)$ is an observed relation, whilst $a(X)$ and $b(Y)$ are latent relations. We describe here how our database $\mathbb{D} = \{\mathbb{D}_r\}$ is obtained by sampling $\mathcal{G}$.

For for every $x \in \mathcal{D}(\tau_1)$, randomly generate the value for $a(x)$ by sampling the Bernoulli distribution with parameter γ_a. Similarly, for every $y \in \mathcal{D}(\tau_2)$, randomly generate the value for $b(y)$ by sampling the Bernoulli distribution with parameter γ_b. Then, for every pair $(x, y) \in \mathcal{D}(\tau_1) \times \mathcal{D}(\tau_2)$, where $a(x) = u$ and $b(y) = v$ are previously sampled values, randomly generate a value for $r(x, y)$ according to a Bernoulli distribution with parameter $\gamma_{r|a=u,b=v}$.

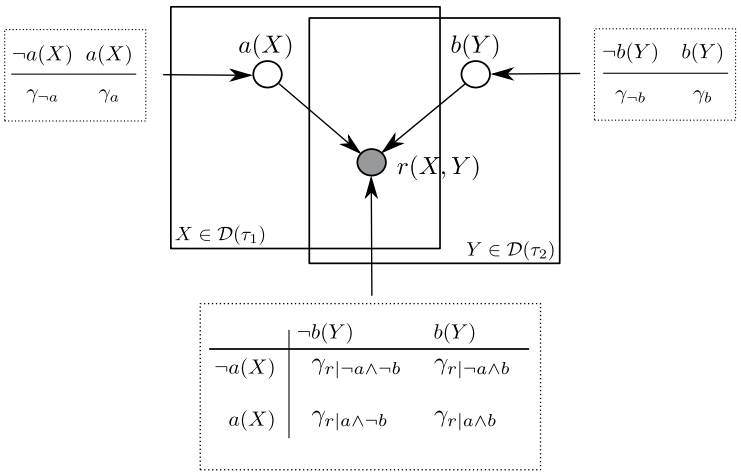


Figure 3: The generative model $\mathcal{G}$ for a simple relational domain shown as a parametrised Bayesian network (in plate notation). Conditional probability tables are shown explicitly. Logical variables X, Y have different types.

Since for all (x, y), $a(x)$ and $b(y)$ are either true or false as a result of sampling $\mathcal{G}$, the correct probability that underlies some particular case $r(x, y)$ is given by

$$\mu_{xy} = P\left(r(x,y)\right) = \begin{cases} \gamma_{r|a,b} & \text{if } a(x) \wedge b(y) \\ \gamma_{r|\neg a,b} & \text{if } \neg a(x) \wedge b(y) \\ \gamma_{r|a,\neg b} & \text{if } a(x) \wedge \neg b(y) \\ \gamma_{r|\neg a,\neg b} & \text{if } \neg a(x) \wedge \neg b(y) \end{cases} \tag{16}$$

which will be referred to when assessing how our models can reproduce the correct probability for answering queries.

5.3. Inference with Observed-relation Models

Assuming that the available data is generated according to $\mathcal{G}$ (Definition 2), this section examines inference with models representing only observed relations. Henceforth we call these models observed-relation models.

The aim is to determine whether observed-relation models can produce the correct answer for probabilistic queries. Let $r(x, y)$ represent the query whose probability is of interest. For the simple domain considered, three possible observed-relation models can be used to answer the query. They are reference classes $\mathbb{C}^{\mathbb{D}}_{\{X,Y\}}(\mathsf{T})$, $\mathbb{C}^{\mathbb{D}}_{\{X,Y\}}(\mathsf{eq\_1}(X, x))$, and $\mathbb{C}^{\mathbb{D}}_{\{X,Y\}}(\mathsf{eq\_2}(Y, y))$, where we have used Boolean relations $\mathsf{eq\_1}(\cdot)$ and $\mathsf{eq\_2}(\cdot)$ to represent equality amongst domain individuals. In the rest of this section we drop the superscript and subscript for reference classes.

The first step towards establishing whether our reference classes can yield the right probabilities is to show that corresponding reference class statistics $\mathbb{P}(r \mid \mathsf{T})$, $\mathbb{P}(r \mid \mathsf{eq\_1}(X, x))$, and $\mathbb{P}(r \mid \mathsf{eq\_2}(Y, y))$ are approximations of marginal

probabilities of $\mathcal{G}$. Marginalising out both $a(X)$ and $b(Y)$ from $\mathcal{G}$ leads to the first marginal

$$\begin{aligned} P(r(X,Y)) &= \sum_u \sum_v P(r(X,Y)|a(X)=u, b(Y)=v) P(a(X)=u) P(b(Y)=v) \\ &= \begin{array}{lll} \gamma_{r|a\wedge b}\gamma_a\gamma_b & + & \gamma_{r|\neg a\wedge b}(1-\gamma_a)\gamma_b \quad + \\ \gamma_{r|a\wedge\neg b}\gamma_a(1-\gamma_b) & + & \gamma_{r|\neg a\wedge\neg b}(1-\gamma_a)(1-\gamma_b) \end{array} \end{aligned} \tag{17}$$

where $u, v \in \{\mathsf{F}, \mathsf{T}\}$. Marginalising out only $b(Y)$ yields

$$P(r(X,Y)|a(X)) = \gamma_{r|a\wedge b}\gamma_b + \gamma_{r|a\wedge\neg b}(1-\gamma_b) \tag{18}$$

$$P(r(X,Y)|\neg a(X)) = \gamma_{r|\neg a\wedge b}\gamma_b + \gamma_{r|\neg a\wedge\neg b}(1-\gamma_b) \tag{19}$$

Finally, marginalising out only $a(X)$ gives

$$P(r(X,Y)|b(Y)) = \gamma_{r|a\wedge b}\gamma_a + \gamma_{r|\neg a\wedge b}(1-\gamma_a) \tag{20}$$

$$P(r(X,Y)|\neg b(Y)) = \gamma_{r|a\wedge\neg b}\gamma_a + \gamma_{r|\neg a\wedge\neg b}(1-\gamma_a) \tag{21}$$

To show that $\mathbb{P}(r \mid \mathsf{T})$, $\mathbb{P}(r \mid \mathsf{eq\_1}(X,x))$, and $\mathbb{P}(r \mid \mathsf{eq\_2}(Y,y))$ approximate Equation 17 to Equation 21, observe that each datum in the dataset $\mathbb{D}_r$ is sampled from $\mathcal{G}$ with four different conditional distributions: $p(r(X,Y) \mid \neg a(X), \neg b(Y))$, $p(r(X,Y) \mid \neg a(X), b(Y))$, $p(r(X,Y) \mid a(X), \neg b(Y))$ and $p(r(X,Y) \mid a(X), b(Y))$. Grouping the data according to their generative distributions we may rewrite $\mathbb{D}_r$ as a union of the partitions, namely $\mathbb{D}_r = D_{\neg a,\neg b} \cup D_{\neg a,b} \cup D_{a,\neg b} \cup D_{a,b}$.

For partition $D_{a,b}$, there are $n^+_{a,b}$ cases where $r(X,Y) = \mathsf{T}$, and $n^-_{a,b}$ cases for $r(X,Y) = \mathsf{F}$. The total number of cases is $n_{a,b} = |D_{a,b}| = n^+_{a,b} + n^-_{a,b}$. Similar counts are defined for other partitions of $\mathbb{D}_r$. Given these counts, we can then express reference class estimate $\mathbb{P}(r = w \mid \mathsf{T})$, which implicates all of $\mathbb{D}_r$, as follows

$$\begin{aligned} \mathbb{P}(r \mid \mathsf{T}) &= \frac{n^+_{\neg a,\neg b} + n^+_{a,\neg b} + n^+_{\neg a,b} + n^+_{a,b}}{N} \\ &= \frac{n^+_{\neg a,\neg b}}{n_{\neg a,\neg b}} \frac{n_{\neg a,\neg b}}{N} + \frac{n^+_{a,\neg b}}{n_{a,\neg b}} \frac{n_{a,\neg b}}{N} + \frac{n^+_{\neg a,b}}{n_{\neg a,b}} \frac{n_{\neg a,b}}{N} + \frac{n^+_{a,b}}{n_{a,b}} \frac{n_{a,b}}{N} \end{aligned} \tag{22}$$

By matching terms in Equation 22 with those in Equation 17, one can see that reference class proportion $\mathbb{P}(r \mid \mathsf{T})$ is simply an empirical approximation of Equation 17.

The reference class $\mathbb{C}(\mathsf{eq\_1}(X,x))$ consists of tuples $\langle x', y' \rangle$ such that $\mathsf{eq\_1}(x',x) \wedge \neg\mathsf{eq\_2}(y',y)$ is true (i.e. we exclude the queried pair $\langle x, y \rangle$). Every $r(x',y') = \mathsf{T}$ occurs with probability $\gamma_{a=u,b=v}$, where $a(x') = u$ and $b(y') = v$ happens to be true as a result of the generative process. Let $n(x) = |\mathbb{C}(\mathsf{eq\_1}(X,x))|$ be the number of cases in $\mathbb{C}(\mathsf{eq\_1}(X,x))$, $n^+(x)$ the number of cases with value T, and $n^-(x)$ the number of cases with value F. We further split $\mathbb{C}(\mathsf{eq\_1}(X,x))$ into two sets corresponding to $b(Y) = \mathsf{T}$ and $b(Y) = \mathsf{F}$ respectively. We then have

counts $n_b(x)$, $n_b^+(x)$, $n_b^-(x)$, and $n_{\neg b}(x)$, $n_{\neg b}^+(x)$, and $n_{\neg b}^-(x)$. The reference class statistic $\mathbb{P}(r \mid \mathsf{eq\_1}(X,x))$ can be expressed as

$$\begin{aligned}
\mathbb{P}\left(r \mid \mathsf{eq_1}(X,x)\right) &= \frac{n_b^+(x)+n_{\neg b}^+(x)}{n(x)} = \frac{n_b^+(x)}{n_b(x)}\frac{n_b(x)}{n(x)} + \frac{n_{\neg b}^+(x)}{n_{\neg b}(x)}\frac{n_{\neg b}(x)}{n(x)} \\
&= \frac{\frac{n_b^+(x)}{n_b(x)}\frac{n_b(x)}{N}}{\frac{n(x)}{N}} + \frac{\frac{n_{\neg b}^+(x)}{n_{\neg b}(x)}\frac{n_{\neg b}(x)}{N}}{\frac{n(x)}{N}} \\
&= \frac{\frac{n_b^+(x)}{n_b(x)}\frac{n(x)}{N}\frac{n_b}{N}}{\frac{n(x)}{N}} + \frac{\frac{n_{\neg b}^+(x)}{n_{\neg b}(x)}\frac{n_{\neg b}(x)}{N}\frac{n_{\neg b}}{N}}{\frac{n(x)}{N}} \\
&= \frac{n_b^+(x)}{n_b(x)}\frac{n_b}{N} + \frac{n_{\neg b}^+(x)}{n_{\neg b}(x)}\frac{n_{\neg b}}{N}
\end{aligned} \tag{23}$$

where n_b is the total number of cases where $b(Y)$ is true, and that

$$\frac{n_b(x)}{N} = \frac{n(x)}{N}\frac{n_b}{N}$$

holds because for each case $r(x',y')$ generated, the occurrence of x' and $b(y') = \mathsf{T}$ are probabilistically independent. Following the same steps, for the symmetric case $\mathbb{P}(r \mid \mathsf{eq\_2}(Y,y))$ we obtain

$$\mathbb{P}\left(r \mid \mathsf{eq\_2}(Y,y)\right) = \frac{n_a^+(y)}{n_a(y)}\frac{n_a}{N} + \frac{n_{\neg a}^+(y)}{n_{\neg a}(y)}\frac{n_{\neg a}}{N} \tag{24}$$

Using Equation 22 to 24, we wish to determine when reference class statistics can model the correct probability.

Theorem 1. *Let $r(X,Y)$ be a Boolean relation where X, Y are of types τ_1 and τ_2 respectively. Suppose $r(X,Y)$ is the sole observed relation with respect to a database $\mathbb{D} = \{\mathbb{D}_r\}$ generated using the generative model $\mathcal{G}$ (see Definition 2). Assume that $\gamma_a \in (0,1)$ and $\gamma_b \in (0,1)$ (i.e., they are non-extreme probability values), then*

$$\begin{array}{lll}
(i) & \forall(x,y), \lim\limits_{N\to\infty} \mathbb{P}(r \mid \mathsf{T}) = \mu_{xy} & \textit{iff} \quad \gamma_{r|a\wedge b} = \gamma_{r|a\wedge\neg b} = \gamma_{r|\neg a\wedge b} = \gamma_{r|\neg a\wedge\neg b} \\
(ii) & \forall(x,y), \lim\limits_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq_1}(X,x)) = \mu_{xy} & \textit{iff} \quad \left(\gamma_{r|a\wedge b} = \gamma_{r|a\wedge\neg b}\right) \wedge \left(\gamma_{r|\neg a\wedge b} = \gamma_{r|\neg a\wedge\neg b}\right) \\
(iii) & \forall(x,y), \lim\limits_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq_2}(Y,y)) = \mu_{xy} & \textit{iff} \quad \left(\gamma_{r|a\wedge b} = \gamma_{r|\neg a\wedge b}\right) \wedge \left(\gamma_{r|a\wedge\neg b} = \gamma_{r|\neg a\wedge\neg b}\right)
\end{array} \tag{25}$$

where (x,y) is a pair in $\mathcal{D}(\tau_1)\times\mathcal{D}(\tau_2)$, and $\gamma_\cdot$ are parameters of the generative model $\mathcal{G}$.

Proof. For all $(x, y) \in \mathcal{D}(\tau_1) \times \mathcal{D}(\tau_2)$, μ_{xy} is the probability $\gamma_{r|a=u\wedge b=v}$, representing the true probability of $r(x, y) = \mathsf{T}$ where $a(x) = u \wedge b(y) = v$ is entailed by database $\mathbb{D}$. Let $N = |\mathbb{D}_r|$ be the number of observed cases of $\mathbb{D}_r$.

We assess how reference class statistic $\mathbb{P}(r \mid \mathsf{T})$ can represent μ_{xy} in the limit as N approaches ∞. Firstly, $\mathbb{P}(r \mid \mathsf{T})$ has the limiting value

$$\lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{T}) = \gamma_{r|a\wedge b}\gamma_a\gamma_b + \gamma_{r|a\wedge\neg b}\gamma_a\gamma_{\neg b} + \gamma_{r|\neg a\wedge b}\gamma_{\neg a}\gamma_b + \gamma_{r|\neg a\wedge\neg b}\gamma_{\neg a}\gamma_{\neg b}$$

Now solve $\lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{T}) = \mu_{xy}$, i.e.

$$\gamma_{r|a\wedge b}\gamma_a\gamma_b + \gamma_{r|a\wedge\neg b}\gamma_a\gamma_{\neg b} + \gamma_{r|\neg a\wedge b}\gamma_{\neg a}\gamma_b + \gamma_{r|\neg a\wedge\neg b}\gamma_{\neg a}\gamma_{\neg b} = \mu_{xy} \tag{26}$$

Equation 26 is ill-posed; there are an unbounded number of solutions. However, for Equation 26 to hold for all $(x, y) \in \mathcal{D}(\tau_1) \times \mathcal{D}(\tau_2)$, where μ_{xy} can be one of $\gamma_{r|a\wedge b}$, $\gamma_{r|a\wedge\neg b}$, $\gamma_{r|\neg a\wedge b}$, or $\gamma_{r|\neg a\wedge\neg b}$, it must be the case that

$$\gamma_{r|a\wedge b} = \gamma_{r|a\wedge\neg b} = \gamma_{r|\neg a\wedge b} = \gamma_{r|\neg a\wedge\neg b}$$

which yields condition (i) of Equation 25.

To prove (ii) in Equation 25, note that the value of $\mathbb{P}(r \mid \mathsf{eq\_1}(X, x))$ (Equation 23) as $N \to \infty$ is

$$\lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq\_1}(X, x)) = \begin{cases} \gamma_{r|a\wedge b}\gamma_b + \gamma_{r|a\wedge\neg b}\gamma_{\neg b} & \text{if } a\,(x) = \mathsf{T} \\ \gamma_{r|\neg a\wedge b}\gamma_b + \gamma_{r|\neg a\wedge\neg b}\gamma_{\neg b} & \text{if } a\,(x) = \mathsf{F} \end{cases} \tag{27}$$

Next we solve $\lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq\_1}(X, x)) = \mu_{xy}$. Suppose $a(x) = \mathsf{T}$ happens to be true as a result of the generative process, and assume $\mu_{xy} = \gamma_{r|a\wedge b}$, we solve

$$\begin{aligned} \lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq\_1}(X, x)) &= \gamma_{r|a\wedge b} \\ \gamma_{r|a\wedge b}\gamma_b + \gamma_{r|a\wedge\neg b}\gamma_{\neg b} &= \gamma_{r|a\wedge b} \\ (\gamma_b - 1)\gamma_{r|a\wedge b} &= (\gamma_b - 1)\gamma_{r|a\wedge\neg b} \\ \therefore \quad \gamma_{r|a\wedge b} &= \gamma_{r|a\wedge\neg b} \end{aligned}$$

This shows that since γ_b is a non-extreme probability, it follows that $\lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq\_1}(X, x)) = \mu_{xy}$ if and only if $\gamma_{r|a\wedge b} = \gamma_{r|a\wedge\neg b}$. Similarly, if $a(x) = \mathsf{F}$ happens to be true and $\mu_{xy} = \gamma_{r|\neg a\wedge b}$, we solve

$$\begin{aligned} \lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq\_1}(X, x)) &= \gamma_{r|\neg a\wedge b} \\ \gamma_{r|\neg a\wedge b}\gamma_b + \gamma_{r|\neg a\wedge\neg b}\gamma_{\neg b} &= \gamma_{r|\neg a\wedge b} \\ (\gamma_b - 1)\gamma_{r|\neg a\wedge b} &= (\gamma_b - 1)\gamma_{r|\neg a\wedge\neg b} \\ \therefore \gamma_{r|\neg a\wedge b} &= \gamma_{r|\neg a\wedge\neg b} \end{aligned}$$

which implies that $\lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq\_1}(X, x)) = \mu_{xy}$ if and only if $\gamma_{r|\neg a\wedge b} = \gamma_{r|\neg a\wedge\neg b}$.

As the actual value of $a(x)$ in the world is not observed, it is therefore not known whether $\mu_{xy} = \gamma_{r|a\wedge b}$ or $\mu_{xy} = \gamma_{r|\neg a\wedge b}$. Therefore, $\lim_{N\to\infty} \mathbb{P}(r \mid$

$\mathsf{eq\_1}(X,x)) = \mu$ is guaranteed for all $(x,y) \in \mathcal{D}(\tau_1) \times \mathcal{D}(\tau_2)$ if and only if $(\gamma_{r|a\wedge b} = \gamma_{r|a\wedge\neg b}) \wedge (\gamma_{r|\neg a\wedge b} = \gamma_{r|\neg a\wedge\neg b})$, and thus proving *(ii)* in Equation 25.

To prove *(iii)* in Equation 25, note that $\mathbb{P}(r \mid \mathsf{eq\_2}(Y,y))$ has the limiting value

$$\lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq\_2}(Y,y)) = \begin{cases} \gamma_{r|a\wedge b}\gamma_a + \gamma_{r|\neg a\wedge b}\gamma_{\neg a} & \text{, if } b\,(y) = \mathsf{T} \\ \gamma_{r|a\wedge\neg b}\gamma_a + \gamma_{r|\neg a\wedge\neg b}\gamma_{\neg a} & \text{, if } b\,(y) = \mathsf{F} \end{cases} \tag{28}$$

Since $\mathbb{P}(r \mid \mathsf{eq\_2}(Y,y))$ is a symmetric case of $\mathbb{P}(r \mid \mathsf{eq\_1}(X,x))$, the same arguments used to derive condition *(ii)* of Equation 25 can followed to show that $\lim_{N\to\infty} \mathbb{P}(r \mid \mathsf{eq\_2}(Y,y)) = \mu$ if and only if $(\gamma_{r|a\wedge b} = \gamma_{r|\neg a\wedge b}) \wedge (\gamma_{r|a\wedge\neg b} = \gamma_{r|\neg a\wedge\neg b})$, thereby proving condition *(iii)* of Equation 25.

□

Theorem 1 shows that, assuming $\mathcal{G}$ is the actual process of the data, reference class statistics $\mathbb{P}(r \mid \mathsf{T})$, $\mathbb{P}(r \mid \mathsf{eq\_1}(X,x))$ and $\mathbb{P}(r \mid \mathsf{eq\_2}(Y,y))$ can represent the true probability of $r(x,y) = \mathsf{T}$ in the limit under specific conditions about $\mathcal{G}$. Figures 4 and 5 provide an instructive view of Theorem 1, showing several configurations of parameters of $\mathcal{G}$. In these illustrations, the true probabilities (conditional probabilities $\gamma_{r|\cdot,\cdot}$ in $\mathcal{G}$) for the query are shown inside of the boxes, whilst marginal probabilities of $\mathcal{G}$ are shown outside the boxes. The marginal probabilities are calculated under the assumption that $\gamma_a = \gamma_b = 0.5$ for simplicity.

In the first case, Figure 4, no conditions stated in Theorem 1 hold. As such, none of the marginal probabilities (outside of the box) correspond to any of the true probabilities (inside the box). Since we have shown that the marginal probabilities are the limiting values of our three reference classes, we conclude that these reference classes cannot represent the correct probabilities, even in the limit of infinite data.

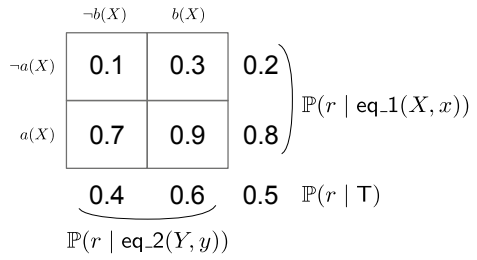


Figure 4: An example parameter configuration of generative model $\mathcal{G}$: numbers inside of the boxes are conditional probability parameters for $r(X,Y)$ (the correct probabilities), and those outside of the boxes are marginal probabilities obtained by marginalising out $a(X)$ and $b(Y)$ separately, where $\gamma_a = \gamma_b = 0.5$ is assumed. The marginal probabilities are limiting values of reference class statistics indicated adjacently.

Figure 5(a) again shows an example where Theorem 1 is not met. However, it is interesting as the most general reference class statistic $\mathbb{P}(r \mid \mathsf{T})$ yields

the correct probability for those cases corresponding to $\neg a(X) \wedge \neg b(Y)$ and $a(X) \wedge b(Y)$. Although this is true, $\mathbb{P}(r \mid \mathsf{T})$ does not represent the correct probability for all cases of $r(X, Y)$. $\mathbb{P}(r \mid \mathsf{T})$ will yield the correct probability for all queries if condition (i) of Theorem 1 is met. An interesting case occurs in Figure 5(b), where parameters of the generative model is configured such that all reference class statistics produce the same answer, but are all incorrect.

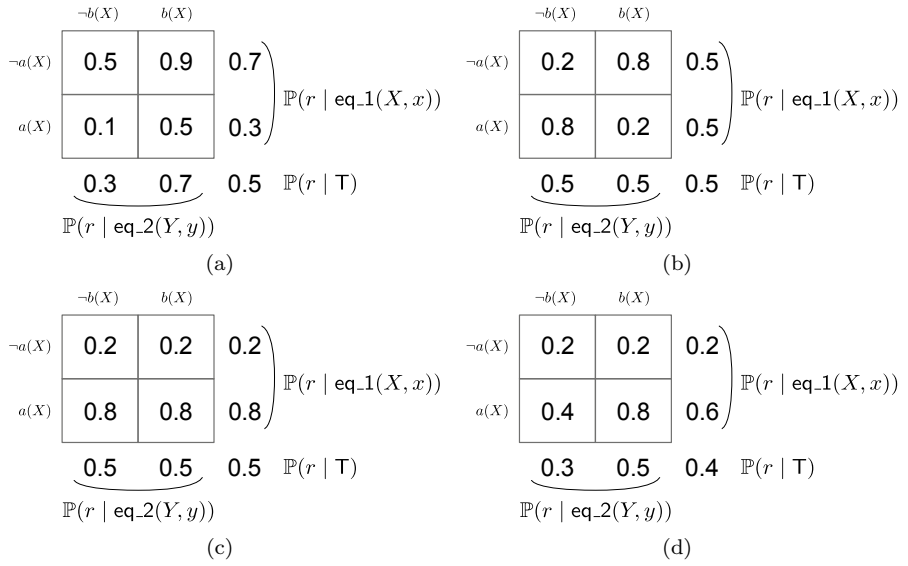


Figure 5: Example parameter configurations of generative model $\mathcal{G}$: numbers inside of the boxes are conditional probability parameters for $r(X, Y)$ (the correct probabilities), and those outside of the boxes are marginal probabilities obtained by marginalising out $a(X)$ and $b(Y)$ separately, where $\gamma_a = \gamma_b = 0.5$ is assumed. The marginal probabilities are limiting values of reference class statistics indicated adjacently.

Figure 5(c) shows an example where condition (ii) holds, allowing reference class statistic $\mathbb{P}(r \mid \mathsf{eq\_1}(X, x))$ to entail the correct probability in the limit. Figure 5(c) also reflects condition (iii) of Theorem 1 as it is symmetric to condition (ii). Figure 5(d) shows a special case where the generative probabilities are such that $\mathbb{P}(r \mid \mathsf{eq\_1}(X, x))$ can represent the correct probability for cases where $\neg a(X)$ is true in the world. Since the value of $a(X)$ is not observed, $\mathbb{P}(r \mid \mathsf{eq\_1}(X, x))$ will be incorrect for cases where $\neg a(X)$ is true.

Overall, conditions $(i) \sim (iii)$ are restrictive, as the success of reference class statistics $\mathbb{P}(r \mid \mathsf{T})$, $\mathbb{P}(r \mid \mathsf{eq\_1}(X,x))$, and $\mathbb{P}(r \mid \mathsf{eq\_2}(Y,y))$ depend crucially on parameter configurations of $\mathcal{G}$. Namely, the reference class statistics can be used to obtain the right probability if and only if $\mathcal{G}$ satifies the conditions stated in Theorem 1, and fails otherwise. Put simply, for domains where $\mathcal{G}$ does not meet any of the stated conditions, the use of reference classes will lead to incorrect probabilities in inference.

It could be argued that our definition of $\mathcal{G}$ over-simplifies real-world relational domains; where realistic generative processes are much more complex. Whilst this is true, our analysis shows that even such simple domains, reference classes can fail to entail the true probability in inference.

5.4. Inference with Latent-property Models

Here we examine whether latent-property models can produce the correct answers for probabilistic queries in domain where $\mathcal{G}$ (Definition 2) is the underlying generative model. The database $\mathbb{D} = \{\mathbb{D}_r\}$ is generated from $\mathcal{G}$, where $\mathbb{D}_r$ is the set of all observed ground instances of $r(X,Y)$. Let $r(x,y) = \mathsf{T}$ denote some query proposition of interest.

The latent-property model in question represents the observed relation $r(X,Y)$ as well as latent properties $\alpha(X)$ and $\beta(Y)$, where $r(X,Y)$ probabilistically depends on $\alpha(X)$ and $\beta(Y)$. We assume that $\alpha(X)$ and $\beta(Y)$ are Boolean. The model also contains parameters $\theta_{r|\alpha=u,\beta=v}$ for all $u, v \in \{\mathsf{F}, \mathsf{T}\}$, each of which is given by

$$\theta_{r|\alpha=u,\beta=v} = \frac{\tilde{\#}_{\mathbb{D}}\left(r(X,Y) \wedge \alpha(X) = u \wedge \beta(Y) = v, Q\right)}{\sum_z \tilde{\#}_{\mathbb{D}}\left(r(X,Y) = z \wedge \alpha(X) = u \wedge \beta(Y) = v, Q\right)} \tag{29}$$

and a probability function Q representing the posterior values of latent properties, i.e. Q define a probability distribution for all ground instances of $\alpha(X)$ and $\beta(Y)$. (See Section 4 for a discussion of latent property models.)

Assume that the correct probability of the query is $\mu_{xy} = P(r(x,y) = \mathsf{T})$ (Equation 16). Then, to assess whether our latent-property model can reproduce μ_{xy}, we drectly examine the answer given by the latent-property model (see Section 4.2):

$$\begin{aligned} \hat{P}\left(r(x,y)\right) = \quad & \theta_{r|\alpha,\beta} Q(\alpha(x)) Q(\beta(y)) \quad + \\ & \theta_{r|\alpha,\neg\beta} Q(\alpha(x)) Q(\neg\beta(y)) \quad + \\ & \theta_{r|\neg\alpha,\beta} Q(\neg\alpha(x)) Q(\beta(y)) \quad + \\ & \theta_{r|\neg\alpha,\neg\beta} Q(\neg\alpha(x)) Q(\neg\beta(y)) \end{aligned} \tag{30}$$

where $\hat{P}(\cdot)$ denotes a probability as given by our model.

First we consider a hard-clustering model, where Q returns 0 or 1. By comparing terms in Equation 30 and Equation 16, our hard-clustering model

predicts the correct probability, i.e. $\hat{P}(r(x,y)) = \mu_{xy}$, if the following are true

$$\begin{aligned}
&(i) \quad \theta_{r|\alpha=u,\beta=v} = \gamma_{r|a=u,b=v} \\
&(ii) \quad Q(\alpha(x)=u) = \begin{cases} 1 & \text{if } a(x)=u \\ 0 & \text{otherwise} \end{cases} \\
&(iii) \quad Q(\beta(y)=v) = \begin{cases} 1 & \text{if } b(y)=v \\ 0 & \text{otherwise} \end{cases}
\end{aligned} \tag{31}$$

Conditions (ii) and (iii) ensure that Q selects the correct statistic $\theta_{(\cdot)}$ (Equation 30) to be the prediction. These two conditions also ensure that each $\theta_{(\cdot)}$ is calculated correctly, i.e. that condition (i) is also satisfied. Namely, Q partitions the data into four disjoint sets, i.e. $\mathbb{D}_r = D_{\alpha,\beta} \cup D_{\alpha,\neg\beta} \cup D_{\neg\alpha,\beta} \cup D_{\neg\alpha,\neg\beta}$, and conditions (ii) and (iii) ensure that the induced partitions are the same as those given by $\mathcal{G}$. For each partition $D_{\alpha=u\wedge\beta=v}$, the associated proportion $\theta_{r|\alpha=u\wedge\beta=v}$ (Equation 29) is an empirical estimate (maximum likelihood) of the conditional probability $P(r(X,Y) \mid \alpha(X)=u, \beta(Y))$. Since the induced partitions are correct, the proportion is also a maximum likelihood estimate of the true conditional probability $P(r(X,Y) \mid a(X)=u, b(Y)=v) = \gamma_{r|a=u,b=v}$. Thus, as the number of data points grow, $\theta_{r|\alpha=u\wedge\beta=v}$ approaches $\gamma_{r|a=u,b=v}$ and satisfies (i).

Inferring the probability function Q to satisfy conditions (ii) and (iii) of Equation 31 can be a difficult task. For instance, computing the most likely value of $\alpha(x)$ requires marginalising over the values of all other ground instances of $\alpha(X)$ and those of $\beta(y)$, where the marginalisation is particularly costly when there are dense correlations amongst all of the latent properties. To alleviate the computation cost, approximate inference methods such as variational Bayes are preferred for computing approximate posterior probabilities for each ground instance of α and β (e.g. in [12, 46] for similar models). Another approach is to perform Markov chain Monte Carlo sampling, adopted in well-known latent-property models such as the infinite relational model [49] and the infinite hidden relational model [15]. Despite the computational hurdles, hard-clustering models do not preclude the correct probability for queries.

For soft-clustering models to entail the correct probability μ_{xy}, we require that

$$\begin{aligned}
&(i) \quad \theta_{r|\alpha=u,\beta=v} = \gamma_{r|a=u,b=v} \\
&(ii) \quad Q(\alpha(x)=u) = P(\alpha(x)=u \mid \mathbb{D}) \\
&(iii) \quad Q(\beta(y)=v) = P(\beta(y)=v \mid \mathbb{D})
\end{aligned} \tag{32}$$

That is, Q represents the exact marginal posterior distribution for each latent property. As such, Equation 30 becomes a *Bayes optimal* predictor, and approach the correct probability in the limit of infinite data. Again, similar to the hard-clustering case, computing the *exact* posterior probabilities confronted by a major computational bottleneck due to the dense correlation over many latent variables. Approximate inference techniques are therefore necessary. Same as

hard-clustering models, the correct probability for queries are within the search space of soft-clustering models.

The above shows that both the hard-clustering and soft-clustering type of latent-property models can entail the correct probability of a given query, if the posterior value (probabilities, in the soft-clustering case) can be inferred exactly. Whilst exact inference of latent property values is in general computationally prohibitive, the main point is that latent-property models do not preclude the correct probability as observed-relation models do. Whether latent-property models are better models of the correct probabilities in practise depends on the learning algorithms used and the problem at hand. This question is addressed empirically in experiments described in Section 6.

6. Experiments

We evaluate predictive accuracy using simple domains containing only one observed relation, and report empirical loss, specifically *log-loss* (or negative log-likelihood), where lower loss values indicates higher accuracy. Since we are interested in measuring discrepancies in probability estimates, empirical loss is an appropriate measure because in the limit of infinite data, the true probability underlying the data is the minimiser of loss [45].

Although Section 5 shows that latent-property models can achieve the correct probabilities with less restrictive conditions than reference classes, our experiments aim to quantify this advantage in practise. To do so we compare the losses incurred by reference classes and two implementations of latent-property models from literature. The reference classes we use are sufficient to represent the kind of observed-relation models described in Section 3 for the simple domains we consider.

Our first experiment uses synthetic data simulated from $\mathcal{G}$ (Definition 2). We validate by generating many examples of $\mathcal{G}$ with randomly generated latent properties and parameters. Results obtained in our synthetic domains quantify how much we gain by incorporating the right modelling assumptions, i.e. modelling latent properties. It serves as a baseline result for comparing with real-world domains; that when latent properties are important in the real world, we can hypothesise that similar gains can be achieved as was in synthetic domains.

In the second experiment, we use real-world relational data from the WebKB project[6] which describes hyperlinks between web pages from five academic world wide web domains.

Thirdly, we use a movie-rating dataset from the EachMovie project[7], which provides ratings by a approximately 60,000 users for 1,600 movies.

[6] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-11/www/wwkb/index.html
[7] http://www.grouplens.org/node/76

6.1. Models

6.1.1. Observed-relation Predictors

In the kind of single-relation domains we consider here, we again use the three reference class predictors $\mathbb{P}(r \mid \mathsf{T})$, $\mathbb{P}(r \mid \mathsf{eq\_1}(X, x))$ and $\mathbb{P}(r \mid \mathsf{eq\_2}(Y, y))$, where r represents the sole observed relation, e.g. *hyperlink* in the WebKB domain. The query is denoted by $r(x, y) = \mathsf{T}$.

Since there is only one observed relation in the domains of interest, relational probabilistic models described in Section 3.2 amount to the reference class $\mathbb{C}(\mathsf{T})$, and thus the reference class statistic $\mathbb{P}(r \mid \mathsf{T})$ is sufficient to represent such models in our experiments.

We construct two prediction methods with our reference classes. The first of which, called REF, simply chooses the best of the three reference class predictions *after* the log-loss is computed. As such, REF is an inadmissible predictor, but is a gold standard for single reference class predictions, and any method that can outperform REF can outperform all other reference class predictors in the set.

A second class of reference class predictors, called POOL, combines all three reference classes by linear interpolation (e.g. weighted combination with weights summing to 1). However, we again construct a gold standard for this type of reference class predictors by adopting the following scheme. Suppose for each test case $r(x, y)$ that the true probability $P(r(x, y)) = \eta$ is known (this is true for synthetic data). Let δ_{min} be the minimum of $\mathbb{P}(r \mid \mathsf{T})$, $\mathbb{P}(r \mid \mathsf{eq\_1}(X, x))$ and $\mathbb{P}(r \mid \mathsf{eq\_2}(Y, y))$, and δ_{max} the maximum. Then, if $\eta \in [\delta_{min}, \delta_{max}]$, we assume a perfect interpolator POOL outputs η. If $\eta \notin [\delta_{min}, \delta_{max}]$, then REF is used. In the case that the true probability is unknown, $P(r(x, y)) \in \{0, 1\}$ (e.g. in real-world experiments), POOL again defaults to REF.

6.1.2. Latent-property Models

We use two examples of latent property models: the infinite relational model (IRM) [15] and one based on Definition 2 which we call the latent relational model (LRM) for these experiments.

The IRM models latent properties to explain relational data and is a *non-parametric* model that stochastically generates the number of values for latent properties, as well as the value of latent properties for each individual. The IRM is a hard-clustering model, as the IRM generates deterministic values for latent properties.

Given a query $r(x, y) = w$, an IRM prediction is based on the latent property values of x and y respectively. Suppose that $\alpha_1(x) = u$ and $\alpha_2(y) = v$ in the IRM (or, x belongs to cluster u and y belongs to cluster v in the IRM nomenclature), the probability ascribed to $r(x, y) = w$ is the empirical proportion given by Equation 4.

The LRM used here is of the same structure as the generative model described by in Definition 2 where the latent properties are assumed Boolean. Predictions generated by LRM follow Equation 30. We consider a soft-clustering LRM here, where the marginal probability function over latent properties as

well as all model parameters are learned using an approximate EM algorithm proposed by [3] or that used in [46].

6.2. Protocol

6.2.1. Synthetic Relations

We simulate 2000 datasets, each generated by sampling a generative model in the form of $\mathcal{G}$ (see Section 5). Parameters of the generative model are generated randomly. There are two types, τ_1 and τ_2, where both $|\mathcal{D}(\tau_1)|$ and $|\mathcal{D}(\tau_2)|$ are restricted to be between 50 to 150. The two latent properties $a(X)$ and $b(Y)$ can have between 2 and 10 values. Each generated dataset contain observed cases for the observed relation $r(X, Y)$ only. A supervised learning framework is assumed, where 90% of the observed cases are used for training, whilst 10% are reserved for testing.

Training of the IRM uses Markov chain Monte Carlo sampling with default parameters specified in the accompanying software[8], whilst the LRM training is done over 30 restarts at 100 iterations per restart, or until convergence, whichever comes first. The best LRM over the restarts is returned.

6.2.2. WebKB

The WebKB contains web-page hyperlinks in the domain of five universities. The data consists of instances of the relation $\mathsf{link}(URL_1, URL_2)$ as well as features based on the words appearing in each web page. Word features are omitted and we use only the hyperlink data in these experiments.

Models REF, IRM and LRM are used for this experiment. The IRM and LRM were trained using the same settings as described in the previous experiment. 2000 randomly sampled groups of 200 web pages are generated from the original data, each forming a standalone dataset. In each sample, 90% of data are used for training, and 10% for prediction.

6.2.3. Movie Ratings

We repeat our previous WebKB experiment with the EachMovie dataset[9]. The rating values in this dataset range from 1 to 5, but we threshold these labels to be Boolean-valued by the global mean of all ratings. We take 500 independent sub-samples of the rating data and run independent experiments on each subsample. 90% of ratings in each sample are used for training, whilst 10% are used for testing.

6.3. Results

6.3.1. Synthetic Relations

The results from our 2000 experiments are binned according to the percentage of observed tuples to the maximum number of possible tuples for $r(X, Y)$.

[8] http://www.psy.cmu.edu/ ckemp/code/irm.html
[9] http://www.grouplens.org/node/76

In other words, we group the experiments by the amount of data generated for each experiment. Figure 6 shows a plot of log-losses of REF, POOL and IRM and LRM, where each point in the graph represents the average log-loss in one bin of experiments using one predictor. Each bin contains 250 to 300 data sets. Standard error is also shown.

The outcome indicate a significant advantage towards the LRM and IRM (i.e. lower log-loss). The log-loss of both and the LRM and IRM improves with the number of data points, indicating an ability to exploit information as they becomes available. The reference class approaches REF and POOL, on the other hand, appears to be insensitive to the amount of observed data. The ability of LRMs to minimise loss when more information becomes available suggests that sufficient statistics encoded in the LRMs are better approximations of those in the underlying model than REF and POOL. The advantage of the LRM over the IRM is likely due to the soft-clustering nature of LRMs compared to the hard-clustering nature of IRM.

6.3.2. WebKB

We take the average log-loss over the 2000 experiments with REF, and IRM, shown in Table 1

ALG.	$\mathbf{L}_{train}$	$\mathbf{L}_{test}$
REF	0.0216 ± 0.000402	0.0210 ± 0.000549
IRM	0.0179 ± 0.000268	0.0190 ± 0.000475
LRM	0.0181 ± 0.000260	0.0185 ± 0.000431

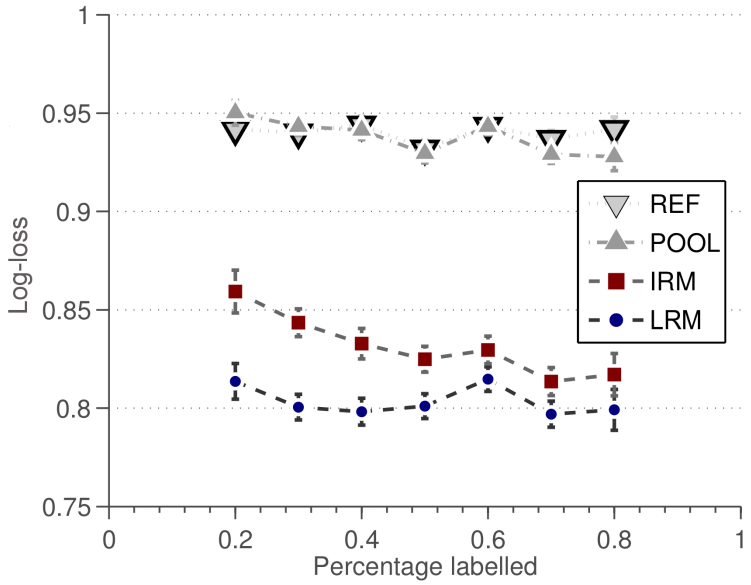
Table 1: Average log-loss over 2000 sampled datasets from the WebKB domain for REF, IRM and LRM on both the training and test sets.

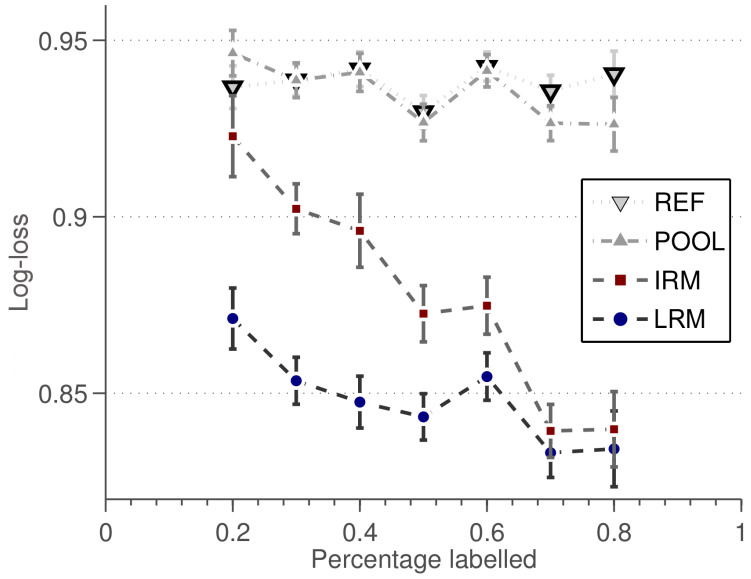
The first observation is that each method performs well overall, scoring low log-losses. (Note random guessing of values will yield log-loss of 1, using log of base 2, whilst the best possible score is 0). This indicates that the sampled datasets present easy prediction problems, due to the sparse linkage patterns in these sets. The fact that IRM and LRM again achieves significantly (in the statistical sense) better log-loss than REF emphasises the value of modelling latent properties, particular in these simple data samples where relative limited information is available. In turn, this suggests that the IRM and LRM are effective in exploiting information that is available. The separation between LRM and IRM are notstatistically significant in this case.

6.3.3. Movie Ratings

Tables 2 lists the training and test performance of each method measured in log-loss.

The EachMovie domain contains a more complex relational structure than the WebKB dataset, and the linkage density (e.g. ratings per user) is greater





(b) Test set.

Figure 6: Log-loss for the IRM, LRM, and reference class predictions REF and POOL. Losses measured on (a) training data and (b) test data are shown for 2000 sets of simulated datasets. Each point in the figure corresponds to an average loss for bins of 200 $\sim$ 300 datasets (with standard error shown). The bins are sorted in increasing order of percentage of observed data. Lower log-loss indicates higher accuracy.

ALG.	$\mathbf{L}_{train}$	$\mathbf{L}_{test}$
REF	0.7373 ± 0.00844	0.7340 ± 0.00852
IRM	0.0173 ± 0.00819	0.4359 ± 0.01261
LRM	0.4728 ± 0.00818	0.5372 ± 0.00888

Table 2: Average log-loss over 500 sampled datasets from the EachMovie domain for REF, IRM and LRM on both the training and test sets.

than that of WebKB (links per web page). As such, it represents a more difficult relational prediction problem, which is reflected in the overall increase of losses. The IRM's ability to exploit latent clusters of individuals likely contributes to its superior score, as on average it returned between 3 and 8 clusters of users (and movies), compared to all other methods tested. The LRM learned using LRM is restricted to modelling two clusters for users and movies and yields higher loss, but still maintains its advantage relative to REF.

It is possible that the generative assumptions used in our analyses (which mirrors those behind LRMs) may hold in the world, thus contributing to LRMs' higher accuracy relative to reference classes. However, it is unlikely that the generative assumptions embodied in LRMs fully reflect the complexities of the true generative model in the world, thus highlighting the potential of modelling with latent properties.

7. Discussion and Related Work

We have explicated the connection between reference classes and models used in relational learning. Further, we showed how inference and learning with probabilistic relational models relate to methods discussed in philosophy for reference classes. This work discusses the impact of problems associated with reference classes on relational learning, motivated by works of Henry Kyburg on reference classes and uncertain reasoning in artificial intelligence and machine learning [22, 23, 21, 24].

As relational probabilistic models can be expressed in terms of reference classes (in Section 3 and 4) philosophical issues affecting reference classes thus affect relational probabilistic models. Instead of revisiting philosophical issues brought about by reference classes, we opt to directly assess whether our relational probabilistic models can represent the correct probabilities for probabilistic queries, under an assumed generative model of the relational domain.

Our analysis showed that when there are properties about individuals that are hidden from observation, relational probabilistic models built solely from observed relations are only capable of representing the correct probabilities under restrictive conditions on the underlying generative process. On the other hand, explicitly postulating about the hidden properties can overcome this shortcoming.

Although our analysis has focused on domains with only one observed relation (where real-world domains often have many observed relations), one can argue that this setting is representative of the case when all available observed relations have been conditioned on. For instance, the root of the tree in Figures 1 and 2 can just as well represent the data remaining after the observed relations have been exhausted.

Empirically, we found that that modelling latent properties about individuals yield significantly better accuracy in inference – on both training and test data – compared to models that do not. Related contributions also support these empirical results, e.g. [46, 34, 49], which were carried out in the same kind of single-relation domains we have considered. In these works, comparisons are essentially made between two models only: one that represents the most general reference class, e.g. $\mathbb{C}(\mathsf{T})$, and one that models latent properties. By contriving gold-standard reference class models in our experiments and showing that they are surpassed by latent-property models, our experimental results implicate a broader range of reference classes than those considered in the existing work mentioned.

Due to the link between reference classes and latent-property models, the inclusion of latent properties should be seen as an alternative way of constructing reference classes that goes beyond the traditional approach of including only observed properties and relations. As observations are never complete, it is questionable whether the most specific reference class (constructed from observed properties and relations only) may entail the correct probability. Latent properties can be seen as a way to fill in the missing knowledge and, as explained in Section 5, and we show it can lead to the correct probability where using only observed properties and relations can preclude this possibility.

8. References

[1] Airoldi, E. M., Blei, D. M., Fienberg, S. E., Xing, E. P., 2008. Mixed membership stochastic blockmodels. Journal of Machine Learning Research 9, 1981–2014.

[2] Bacchus, F., Grove, A. J., Halpern, J. Y., Koller, D., 1996. From statistical knowledge bases to degrees of belief. Artificial Intelligence 87 (1-2), 75–143.

[3] Chiang, M., Poole, D., 2009. Incremental EM for latent relational models. In: Proceedings of the International Workshop on Statistical Relational Learning.

[4] De Raedt, L., 2008. Logical and Relational Learning. Springer.

[5] Dempster, A. P., Laird, N. M., Rubin, D. B., 1977. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society 34, 1–38.

[6] Fetzer, J. H., 1977. Reichenbach, reference classes, and single case probabilities. Synthese 34 (2).

[7] Friedman, N., Getoor, L., Koller, D., Pfeffer, A., 1999. Learning probabilistic relational models. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence. pp. 1300–1309.

[8] Getoor, L., Taskar, B. (Eds.), 2007. Introduction to Statistical Relational Learning.

[9] Halpern, J. Y., 1990. An analysis of first-order logics of probability. Artificial Intelligence 46, 311–350.

[10] Handcock, M. S., Raftery, Adrian, E., Tantrum, J. M., March 2007. Model-based clustering for social networks. Journal of the Royal Statistical Society 170 (2), 301–354.

[11] Hofman, J. M., Wiggins, C. H., 2008. Bayesian approach to network modularity. Physical Review Letters 100 (25).

[12] Hofmann, T., Puzicha, J., 1999. Latent class models for collaborative filtering. In: Proceeding of the 16th International Joint Conference on Artificial Intelligence. pp. 688–693.

[13] Jaeger, M., 1997. Relational Bayesian networks. In: Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann, pp. 266–273.

[14] Kahneman, D., Slovic, P., Tversky, A., 1982. Judgment Under Uncertainty: Heuristics and Biases. Cambridge University Press.

[15] Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., Ueda, N., 2006. Learning systems of concepts with an infinite relational model. In: Proceedings of the 21st National Conference on Artificial Intelligence.

[16] Kersting, K., De Raedt, L., 2000. Bayesian logic programs. In: Cussens, J., Frisch, A. (Eds.), Proceedings of the 10th International Conference on Inductive Llogic Programming (Work-in-progress track). pp. 138–155.

[17] Kersting, K., De Raedt, L., 2002. Basic principles of learning Bayesian logic programs. Tech. rep., University of Freiburg.

[18] Kok, S., Domingos, P., 2005. Learning the structure of Markov logic networks. In: Proceedings of the 22th International Conference on Machine Learning. pp. 441–448.

[19] Koller, D., Friedman, N., 2009. Probabilistic Graphical Models: Principles and Techniques. MIT Press.

[20] Koren, Y., 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, New York, NY, USA, pp. 426–434.

[21] Kyburg, H., 1983. The reference class. Philosophy of Science 50, 374–397.

[22] Kyburg, H. E., 1974. Logical Foundations of Statistical Inference. D. Reidel Publishing Co.

[23] Kyburg, H. E., 1977. Randomness and the right reference class. Journal of Philosophy 74, 501–502.

[24] Kyburg Jr., H. E., 1994. Believing on the basis of the evidence. Computational Intelligence 10, 3–20.

[25] Lavrac, N., Dzeroski, S., 1994. Inductive Logic Programming: Techniques and Applications. Ellis Horwood, New York.

[26] Levi, I., 1977. Direct inference. The Journal of Philosophy 74 (1), pp. 5–29.

[27] Levi, I., 1981. Direct inference and confirmational conditionalization. Philosophy of Science 48 (4), 532–552.

[28] Marlin, B., Zemel, R. S., 2004. The multiple multiplicative factor model for collaborative filtering. In: Proceedings of the 21st international conference on Machine learning. ICML '04. ACM, New York, NY, USA, pp. 73–.

[29] McGrew, T., 2001. Direct inference and the problem of induction. The Monist 84.

[30] Milch, B., Marthi, B., Russell, S., Sontag, D., Ong, D. L., Kolobov, A., 2005. BLOG: Probabilistic models with unknown objects. In: 19th International Joint Conference on Artificial Intelligence. pp. 1352–1359.

[31] Muggleton, S., 1995. Inverse entailment and Progol. New Generation Computing, Special issue on Inductive Logic Programming 13 (3-4), 245–286.

[32] Muggleton, S., 1995. Stochastic logic programs. In: De Raedt, L. (Ed.), Proceedings of the 5th International Workshop on Inductive Logic Programming. Department of Computer Science, Katholieke Universiteit Leuven, p. 29.

[33] Muggleton, S., De Raedt, L., 1994. Inductive logic programming: Theory and methods. Journal of Logic Programming 19/20, 629–679.

[34] Neville, J., Jensen, D., 2005. Leveraging relational autocorrelation with latent group models. In: IEEE International Conference on Data Mining. pp. 322–329.

[35] Newman, M. E. J., 2003. The structure and function of complex networks. SIAM Review 45, 167–256.

[36] Nienhuys-Cheng, S., de Wolf, R., 1997. Foundations of Inductive Logic Programming. Lecture notes in Artificial Intelligence. Springer-Verlag, Germany.

[37] Pearl, J., 1988. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Reasoning. Morgan Kaufmann, San Mateo.

[38] Poole, D., 1993. Probabilistic Horn abduction and Bayesian networks. Artificial Intelligence 64 (1), 81–129.

[39] Poole, D., 1997. The independent choice logic for modelling multiple agents under uncertainty. Artificial Intelligence 95(1-2), 7–56.

[40] Quinlan, J. R., 1990. Learning logical definitions from relations. Machine Learning 5, 239–266.

[41] Reichenbach, H., 1949. The Theory of Probability. University of California Press.

[42] Richardson, M., Domingos, P., 2006. Markov logic networks. Machine Learning 62 (1-2), 107–136.

[43] Sampson, S., 1968. A novitiate in a period of change: An experimental and case study of social relationships. Ph.D. thesis, Cornell University.

[44] Sato, T., Kameya, Y., 1997. PRISM: a symbolic-statistical modeling language. In: Proceedings of the 15th International Joint Conference on Artificial Intelligence. pp. 1330–1335.

[45] Shen, Y., 2005. Loss functions for binary classification and class probability estimation. Ph.D. thesis, University of Pennsylvania.

[46] Taskar, B., Segal, E., Koller, D., 2001. Probabilistic classification and clustering in relational data. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence. pp. 870–878.

[47] Ungar, L. H., Foster, D. P., 1998. Clustering methods for collaborative filtering. In: Americal Association for Artificial Intelligence Workshop on Recommendation Systems. AAAI Press.

[48] Xu, Z., Tresp, V., Rettinger, A., Kersting, K., 2009. Social network mining with nonparametric relational models. In: Zhang, H., Smith, M., Giles, L., Yen, J. (Eds.), Advances in Social Network Mining and Analysis. LNCS. Springer.

[49] Xu, Z., Tresp, V., Yu, K., Kriegel, H., 2006. Infinite hidden relational models. In: Proceedings of the 22nd Conference on Uncertainty in Aritificial Intelligence.

[50] Yedidia, J. S., Freeman, W. T., Weiss, Y., 2004. Constructing free energy approximations and generalized belief propagation algorithms. IEEE Transactions on Information Theory 51, 2282–2312.

[51] Zhang, N. L., Poole, D., 1996. Exploiting causal independence in Bayesian network inference. Journal of Artificial Intelligence Research 5, 301–328.