Statistical Relational Learning with Soft Quantifiers

Golnoosh Farnadi[1,2], Stephen H. Bach[3], Marjon Blondeel[4],
Marie-Francine Moens[2], Lise Getoor[5], and Martine De Cock[1,6]

[1]Dept. of Applied Mathematics, Computer Science and Statistics, Ghent University, Belgium
[2]Dept. of Computer Science, Katholieke Universiteit Leuven, Belgium
[3]Statistical Relational Learning Group, University of Maryland, USA
[4]Ghent University Global Campus, South Korea
[5]Statistical Relational Learning Group, University of California, Santa Cruz, USA
[6]Center for Data Science, University of Washington Tacoma, USA
golnoosh.farnadi@ugent.be, bach@cs.umd.edu
marjon.blondeel@ugent.be, sien.moens@cs.kuleuven.be
getoor@soe.ucsc.edu, mdecock@uw.edu

Abstract. Quantification in statistical relational learning (SRL) is either existential or universal, however humans might be more inclined to express knowledge using soft quantifiers, such as "most" and "a few". In this paper, we define the syntax and semantics of PSL^Q, a new SRL framework that supports reasoning with soft quantifiers, and present its most probable explanation (MPE) inference algorithm. To the best of our knowledge, PSL^Q is the first SRL framework that combines soft quantifiers with first-order logic rules for modeling uncertain relational data. Our experimental results for link prediction in social trust networks demonstrate that the use of soft quantifiers not only allows for a natural and intuitive formulation of domain knowledge, but also improves the accuracy of inferred results.

1 Introduction

Statistical relational learning (SRL) has become a popular paradigm for knowledge representation and inference in application domains with uncertain data that is of a complex, relational nature. A variety of different SRL frameworks has been developed over the last decade, based on ideas from probabilistic graphical models, first-order logic, and programming languages (see e.g., [20, 24, 10]). Quantification in first-order logic is traditionally either existential ($\exists$) or universal ($\forall$). Given the strong roots of the existing SRL frameworks in (a subset of) first-order logic as a knowledge representation language, it is no surprise that these are the two kinds of quantifications that are known and commonly used in SRL, even though in many application scenarios humans might be more inclined to express knowledge using softer quantifiers, such as *most* and *a few*.

For example, in models for social networks it is common to include the knowledge that the behaviour, beliefs, and preferences of friends all influence each other. How this information can be incorporated depends on the expressivity of the model. In a traditional probabilistic model, a dependency might be included for each pair of friends (corresponding to a universally quantified rule), each expressing the knowledge that it is more probable that two friends share a trait in common. An often cited example in SRL contexts describing smoking behaviour among friends is $\forall X \forall Y Friends(X, Y) \rightarrow (Smokes(X) \leftrightarrow Smokes(Y))$ [24]. This formula states that if two people are friends, then either both of them smoke or neither of them. In this case, the probability that a person smokes scales smoothly with the number of friends that smoke. However, many

traits of interest might not behave this way, but instead exhibit "tipping points" in which having a trait only becomes more probable once *most* or *some* of one's friends have that trait (e.g., smoking behaviour). Expressing this dependency requires a soft quantifier, which none of the existing SRL frameworks allow.

What sets soft quantifiers apart from universal and existential quantification is that expressions that contain them are often true to a certain degree, as opposed to either being true or false. Indeed, the degree to which a statement such as "most of Bob's friends smoke" is true, increases with the percentage of smokers among Bob's friends. This increase is not necessarily linear; in fact, a common approach to compute the truth degree of soft quantified expressions is to map percentages to the scale $[0, 1]$ using non-decreasing piecewise linear functions [29]. Previous SRL work (e.g., [19, 14, 22]) has considered hard quantifiers with thresholds such as *at least k*. Soft quantifiers, on the other hand, do not impose such hard thresholds but allow smooth, gradual transitions from falsehood to truth.

Furthermore, the dependence of predicted probabilities on population size in relational models such as Markov logic networks (MLNs) and relational logistic regression is addressed in [21, 15]. Soft quantifiers not only provide the flexibility of modeling complex relations, but their semantics also do not depend on the absolute population size. Hence soft quantifiers allow us to learn a model for some population size and apply the same model to another population size without the need for changes in the model, e.g. without introducing auxiliary variables to control whether the population size grows.

Many SRL applications could benefit from the availability of soft quantifiers. Collective document classification, for instance, relies on rules such as $\forall D \forall E \forall C(Cites(D, E) \wedge Class(D, C) \rightarrow Class(E, C))$ which expresses that if documents D and E are linked (e.g., by citation), and D belongs to class C, then E belongs to C [2]. Soft quantifiers would allow to classify a document based on *most* of its citing documents instead of one citing document. Similarly, in collaborative filtering, one can rely on the preferred products of a user to infer the behaviour of a similar user, i.e., $\forall U_1 \forall U_2 \forall J(Likes(U_1, J) \wedge Similar(U_1, U_2) \rightarrow Likes(U_2, J))$ [2]. Using soft quantifiers would allow to infer preferences of a user based on *most* of the behaviours of a similar user, or by comparing one user with *most* of the users similar to him.

In this paper we present the first SRL framework that combines soft quantifiers with first-order logic rules for modeling uncertain relational data. A brief overview of our framework is presented in [9]. We start from probabilistic soft logic (PSL) [1], an existing SRL framework that defines templates for hinge-loss Markov random fields [2], and extend it to a new framework which we call PSL^Q. As is common in SRL frameworks, in PSL a problem is defined by a set of logical rules using a finite set of atoms. However, unlike other SRL frameworks whose atoms are Boolean, atoms in PSL can take continuous values in the interval $[0, 1]$. Intuitively, value 0 means *false* and value 1 means *true*, while any value $v \in [0, 1]$ represents a partial degree of truth. PSL has been used in various domains with promising results, including trust propagation [12], drug-target interaction prediction [8], knowledge graph identification [23], semantic textual similarity computation [4] and sentiment analysis in a social network [27], among many others. Reasoning with continuous values has also been addressed in *fuzzy*DL [5], however reasoning is not as efficient as in PSL. Furthermore, using quantifiers in probabilistic logic is addressed in previous works with Boolean atoms, such as in [18, 3, 25], while in this study we address the use of soft quantifiers for continuous atoms.

This paper makes three contributions. First, we introduce PSL^Q, a new SRL framework that supports reasoning with soft quantifiers, such as "most" and "a few". Second, because this expressivity pushes beyond the capabilities of PSL, we introduce new inference and weight learning algorithms for PSL^Q. Finally, as a proof of concept, we

present a PSL^Q model that more accurately predicts trust in social networks than the current state-of-the-art approach.

2 PSL^Q: PSL with Soft Quantifiers

Definition 1. *An **atom** is an expression of the form $p(a_1, a_2, \ldots, a_n)$ where p is a **predicate symbol**, and each argument $a_1, a_2, \ldots, a_n$ is either a constant or a variable. The finite set of all possible substitutions of a variable to a constant for a particular variable a_i is called its **domain** D_{a_i}. If all variables in $p(a_1, a_2, \ldots, a_n)$ are substituted by some constant from their respective domain, then we call the resulting atom a **ground atom**. We call $\neg p(a_1, a_2, ..., a_n)$ a **negated atom** which is the negation of $p(a_1, a_2, ..., a_n)$.*

Definition 2. *A **quantifier expression** is an expression of the form*

$$Q(V, F_1[V], F_2[V]) \tag{1}$$

*where Q is a **soft quantifier**, and $F_1[V]$ and $F_2[V]$ are formulas containing the variable V. A formula is an atom or a negation, conjunction or disjunction of formulas. A **grounded quantifier expression** is obtained by instantiating all variables with constants from their domains except for V.*

Consider as an example the two formulas $Knows(X, T)$ and $Trusts(X, T)$, then $Most(T, Knows(X, T), Trusts(X, T))$ is a quantifier expression. By substituting X with Alice, we obtain the grounded quantifier expression:
$Most(T, Knows(Alice, T), Trusts(Alice, T))$, which can be read as "Alice trusts most of the people she knows".

Definition 3. *A **PSL^Q model** consists of a collection of PSL^Q rules. A **PSL^Q rule** r is an expression of the form:*

$$\lambda_r : T_1 \wedge T_2 \wedge \ldots \wedge T_k \rightarrow H_1 \vee H_2 \vee \ldots \vee H_l \tag{2}$$

where $T_1, T_2, \ldots, T_k, H_1, H_2, \ldots, H_l$ are atoms, negated atoms, quantifier expressions or negated quantifier expressions and $\lambda_r \in \mathbb{R}^+ \cup \{\infty\}$ is the weight of the rule r. We call $T_1 \wedge T_2 \wedge \ldots \wedge T_k$ the body of r (r_{body}), and $H_1 \vee H_2 \vee \ldots \vee H_l$ the head of r (r_{head}). Grounding a PSL^Q rule means instantiating all the variables with constants from their domain except for the variables V in quantifier expressions $Q(V, F_1[V], F_2[V])$.

Remark 1. *A PSL model, i.e., a set of PSL rules, is a PSL^Q model without quantifier expressions. The first 9 rules in Table 1 are an example of a PSL^Q model without quantifier expressions, or a PSL model, while rules $10 - 14$ in Table 1 are examples of PSL^Q rules with quantifier expressions.*

An **interpretation** I is a mapping that associates a truth value $I(s) \in [0, 1]$ to each ground atom s. For example, $I(Knows(Alice, Bob)) = 0.7$ indicates that Alice knows Bob to degree 0.7. The interpretation of PSL^Q rules is based on Łukasiewicz logic [16]. Conjunction $\wedge$ is interpreted by the Łukasiewicz t-norm ($\tilde{\wedge}$), disjunction $\vee$ by the Łukasiewicz t-conorm ($\tilde{\vee}$), and negation $\neg$ by the Łukasiewicz negator ($\tilde{\neg}$), which are defined as follows. For $m, n \in [0, 1]$ we have: $m \tilde{\wedge} n = \max(0, m + n - 1)$, $m \tilde{\vee} n = \min(m + n, 1)$ and $\tilde{\neg} m = 1 - m$. The $\tilde{}$ indicates the relaxation over Boolean values. We can extend the interpretation of atoms to more complex formulas in Łukasiewicz logic as follows. Given an interpretation I, and ϕ_1 and ϕ_2 formulas, we have $I(\phi_1 \wedge \phi_2) = I(\phi_1) \tilde{\wedge} I(\phi_2)$, $I(\phi_1 \vee \phi_2) = I(\phi_1) \tilde{\vee} I(\phi_2)$ and $I(\neg \phi_1) = \tilde{\neg} I(\phi_1)$.

The interpretation of quantifier expressions in PSL^Q relies on quantifier mappings.

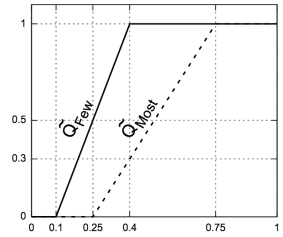


Fig. 1: Examples of quantifier mappings

Definition 4. *A **quantifier mapping** $\tilde{Q}$ is a $[0,1] \rightarrow [0,1]$ mapping. If $\tilde{Q}$ is non-decreasing and satisfies the boundary conditions $\tilde{Q}(0) = 0$ and $\tilde{Q}(1) = 1$, it is called a **coherent** quantifier mapping [7].*

We assume that for every soft quantifier Q an appropriate quantifier mapping $\tilde{Q}$ can be defined, i.e. a function that represents the meaning of Q. Using two thresholds $\alpha \in [0,1]$ and $\beta \in [0,1]$, where $\alpha \leq \beta$, the following equation defines a parametrized family of such quantifier mappings:

$$\tilde{Q}_{[\alpha,\beta]}(x) = \begin{cases} 0 & \text{if } x < \alpha \\ \frac{x-\alpha}{\beta-\alpha} & \text{if } \alpha \leq x < \beta \\ 1 & \text{if } x \geq \beta \end{cases} \tag{3}$$

Figure 1 depicts a possible coherent quantifier mapping for the soft quantifier "a few" as $\tilde{Q}_{Few} = \tilde{Q}_{[0.1,0.4]}$ and for the soft quantifier "most" as $\tilde{Q}_{Most} = \tilde{Q}_{[0.25,0.75]}$. Note how $\tilde{Q}_{Few}$ is more relaxed than $\tilde{Q}_{Most}$. For example, using these mappings, the statement "a few friends of Bob smoke" is true to degree 1 as soon as 40% of Bob's friends are smokers, while 75% of Bob's friends are required to be smokers for the statement "most friends of Bob smoke" to be fully true. The evaluation section contains a detailed analysis on the effect of the choice of the thresholds α and β on the results obtained with MPE inference. In practice friendship is not necessarily a black-and-white matter, i.e., people can be friends to varying degrees. For instance, $I(Friend(Bob, Alice)) = 1$ and $I(Friend(Bob, Chris)) = 0.2$ denote that under interpretation I, Alice is a very close friend of Bob, while Chris is a more distant friend. Similarly, Chris might be a heavy smoker, while Alice might be only a light smoker. All these degrees can and should be taken into account when computing the truth degree of statements such as "a few friends of Bob smoke" and "most friends of Bob smoke". We define the interpretation of a grounded quantifier expression based on the Zadeh approach [29]. Zadeh suggested to calculate the truth value of "Q A's are Bs", with $A : \mathcal{U} \rightarrow [0,1]$ and $B : \mathcal{U} \rightarrow [0,1]$ fuzzy sets in a universe $\mathcal{U}$, as: $\tilde{Q}\left(\frac{|A \cap B|}{|A|}\right)$, where $A \cap B$ is a fuzzy set defined as: $A \cap B : \mathcal{U} \rightarrow [0,1] : u \mapsto A(u) \tilde{\wedge} B(u)$. In this expression, the **cardinality** of a fuzzy set $S : \mathcal{U} \rightarrow [0,1]$ is defined as: $|S| = \sum_{u \in \mathcal{U}} S(u)$.

Definition 5. *For a given interpretation I, **the interpretation of a grounded quantifier expression** $Q(V, F_1[V], F_2[V])$ is defined as*

$$I(Q(V, F_1[V], F_2[V])) = \tilde{Q}\left(\frac{\sum_{x \in D_V} I(F_1(x)) \,\tilde{\wedge}\, I(F_2(x))}{\sum_{x \in D_V} I(F_1(x))}\right) \tag{4}$$

with $\tilde{Q}$ a quantifier mapping modeling Q and D_V the domain of V.

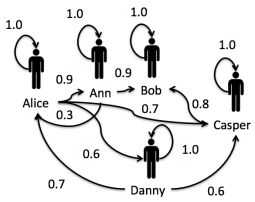


Fig. 2: Sample trust network between five users

Example 1. *Let's consider an interpretation I in a sample trust network as shown in Figure 2. Nodes represent users and each edge represents the trust relation between two users. Since a trust relation is asymmetric, the direction of the trust relation is shown with an arrow. The degree of the trust links are shown with a value under/above the links, e.g., $I(Trusts(Alice, Ann)) = 0.9$.*
To calculate $I(Most(X, Trusts(Alice, X), Trusts(X, Bob)))$, we calculate:
$\sum_x$(I(Trusts$(Alice, x)) \tilde{\wedge}$ I(Trusts$(x, Bob))) = 1.3$ and $\sum_x$ I(Trusts$(Alice, x)) = 3.2$, thus we have $\tilde{Q}\left(\frac{1.3}{3.2}\right) \sim \tilde{Q}(0.41)$. By using the quantifier expression mapping of "most" in Figure 1, we obtain $\tilde{Q}_{[0.25,0.75]}(0.41) = 0.32$. Thus, under interpretation I, "most trustees of Alice trust Bob" to degree 0.32.

Remark 2. *In Łukasiewicz logic, the formula $\phi_1 \rightarrow \phi_2$ where $\rightarrow$ is implication, is logically equivalent to the formula $\neg\phi_1 \vee \phi_2$, thus the interpretation of a grounded PSL^Q rule r is as follows:*

$$I(r) = I(r_{body} \rightarrow r_{head}) = \tilde{\neg} I(r_{body}) \tilde{\vee} I(r_{head}) \tag{5}$$

Definition 6. *The distance to satisfaction $d_r(I)$ of a rule r under an interpretation I is defined as:*

$$d_r(I) = \max\{0, I(r_{body}) - I(r_{head})\} \tag{6}$$

By using Remark 2, one can show that a rule r is fully satisfied, i.e. satisfied to degree 1, when the truth value of its head is at least as high as the truth value of its body. Thus, the closer the interpretation of a grounded rule is to 1, the smaller its distance to satisfaction.

A PSL^Q model, i.e., a set of PSL^Q rules, induces a distribution over interpretations I. Let R be the set of all grounded rules, then the probability density function is:

$$f(I) = \frac{1}{Z} \exp[-\sum_{r \in R} \lambda_r (d_r(I))^p] \tag{7}$$

where λ_r is the weight of rule r, Z is a normalization constant

$$Z = \int_I \exp[-\sum_{r \in R} \lambda_r (d_r(I))^p]$$

and $p \in \{1, 2\}$. These probabilistic models are instances of hinge-loss Markov random fields (HL-MRF). For further explanation we refer to [2]. Choosing $p = 1$ (i.e., linear) favors interpretations that completely satisfy one rule at the expense of higher distance from satisfaction for conflicting rules, and $p = 2$ favors interpretations that satisfy all rules to some degree (i.e, quadratic). Note that in Section 3 we only consider $p = 1$, since by [2] the results can be extended for $p = 2$.

3 Inference and Weight Learning in PSL^Q

Expressing soft quantifiers pushes beyond the capabilities of inference and weight learning methods in PSL. In this section, we introduce new methods for inference based on the most probable explanation inference method (MPE inference) and weight learning with maximum-likelihood estimation (MLE) in PSL^Q.

3.1 Inference

The goal of MPE "most probable explanation" inference is to find the most probable truth assignments I_{MPE} of unknown ground atoms given the evidence which is defined by the interpretation I. Let X be all the evidence, i.e., X is the set of ground atoms such that $\forall x \in X, I(x)$ is known, and let Y be the set of ground atoms such that $\forall y \in Y, I(y)$ is unknown. Then we have

$$I_{MPE}(Y) = arg\ max_{I(Y)} P(I(Y)|I(X)) \tag{8}$$

and by Equation 7 it follows that the goal of optimization is to minimize the weighted sum of the distances to satisfaction of all rules. Using the particular semantics of Łukasiewicz logic we can translate this optimization problem to a set of linear constraints.

Remark 3. *Suppose we want to optimize a $f : [0,1]^n \to [0,1]$ function consisting of applications of only piecewise linear functions, fractions of piecewise linear functions, $\min : [0,1]^2 \to [0,1]$ and $\max : [0,1]^2 \to [0,1]$. We can transform such an optimization problem as follows. For every expression of the form $\min(\phi, \psi)$, we introduce a variable $v_{\min(\phi,\psi)}$ and add the constraints $0 \leq \phi, \psi, v_{\min(\phi,\psi)} \leq 1$, $v_{\min(\phi,\psi)} \leq \phi$ and $v_{\min(\phi,\psi)} \leq \psi$. Similarly, for every expression of the form $\max(\phi, \psi)$, we introduce a variable $v_{\max(\phi,\psi)}$ and add the constraints $0 \leq \phi, \psi, v_{\max(\phi,\psi)} \leq 1$, $v_{\max(\phi,\psi)} \geq \phi$ and $v_{\max(\phi,\psi)} \geq \psi$. Define the function g as the original function f but all minima and maxima are replaced by their corresponding variables. Optimizing f is then equivalent to optimizing g under these constraints.*

By the particular piecewise linear form of $d_r(I)$ (see Equation 6) and Remark 3, standard PSL's underlying HL-MRFs have log concave density functions and hence finding an MPE assignment is a convex optimization problem, which is solvable in polynomial time. Standard PSL only supports linear constraints to preserve convexity. Hence, standard PSL potentially can support linear aggregates.

Definition 7. *An **aggregate** is a $[0,1]^n \to [0,1]$ mapping. If it is a linear mapping, it is called a **linear aggregate**, otherwise it is called a **non-linear aggregate**.*

As an example, $f : [0,1]^n \to [0,1] : (t_1, .., t_n) \mapsto \frac{t_1+t_2+...+t_n}{n}$ is a linear aggregate. A PSL^Q program allows expressions that contain quantifier expressions. Since the interpretation of a grounded quantifier expression (see Equation 4) is based on a non-linear aggregate, finding a MPE assignment of a PSL^Q program with quantifier expressions is beyond the capabilities of the standard PSL MPE-solver. To deal with this, we will first categorize different types of grounded quantifier expressions, given the interpretation I denoting the evidence.

Definition 8. *A grounded quantifier expression $Q(V, F_1[V], F_2[V])$, where for every $s \in D_V$, it holds that all ground atoms in the formulas $F_1[s]$ and $F_2[s]$ are in X, is called a **fully observed grounded quantifier expression (FOQE)**.*

For instance, in a social network where the age and the friends of all users are known, by grounding $Most(X, Friend(A, X), Young(X))$, we obtain FOQEs. Note that for a FOQE $Q(V, F_1[V], F_2[V])$, we have that $I(Q(V, F_1[V], F_2[V]))$ is a known value in $[0, 1]$.

Definition 9. *A grounded quantifier expression $Q(V, F_1[V], F_2[V])$, where for every $s \in D_V$, it holds that all ground atoms in the formula $F_1[s]$ are in X and there exists $t \in D_V$ such that at least one ground atom in the formula $F_2[t]$ is in Y, is called a* ***partially observed grounded quantifier expression of type one (POQE$^{(1)}$)****.*

Suppose all friendship relations are known and the goal is to infer the age of all users based on the age of some, then by grounding $Most(X, Friend(A, X), Young(X))$, we obtain POQE$^{(1)}$s. Node labelling applications can benefit from the use of POQE$^{(1)}$s. Note that for a POQE$^{(1)}$ $Q(V, F_1[V], F_2[V])$, we have that $I(Q(V, F_1[V], F_2[V])) = \tilde{Q}(f(Y))$ where f is a piecewise linear function in variables belonging to Y.

Definition 10. *A grounded quantifier expression $Q(V, F_1[V], F_2[V])$, for which there exists $t \in D_V$ such that at least one ground atom in the formula $F_1[t]$ is in Y, is called a* ***partially observed grounded quantifier expression of type two (POQE$^{(2)}$)****.*

Note that for a POQE$^{(2)}$ $Q(V, F_1[V], F_2[V])$, we have that $I(Q(V, F_1[V], F_2[V])) = \tilde{Q}(f(Y))$ where f is a fraction of piecewise linear functions in variables belonging to Y. In link prediction applications, such as trust link prediction, we mostly deal with POQE$^{(2)}$s. By grounding the rules $10 - 14$ in Table 1 using unknown trust relations, we obtain complex examples of POQE$^{(2)}$s.

In the following proposition we give an equivalent definition for the membership function in Equation 3. By applying Remark 3 we will then be able to show that a PSL^{Q} program can be transformed to a linear fractional program (LFP).

Proposition 1. *The membership-function defined in Equation 3 where $\alpha \in [0, 1]$, $\beta \in [0, 1]$, and $\alpha \leq \beta$ can be rewritten as:*

$$\tilde{Q}_{[\alpha,\beta]}(x) = \max(0, \frac{x - \alpha}{\beta - \alpha}) + \min(\frac{x - \alpha}{\beta - \alpha}, 1) - \frac{x - \alpha}{\beta - \alpha} \tag{9}$$

After grounding a PSL^{Q} program we can obtain a mixture of FOQE's, POQE$^{(1)}$'s and POQE$^{(2)}$'s. Recall that for a FOQE $Q(V, F_1[V], F_2[V])$, we have that $I(Q(V, F_1[V], F_2[V])) \in [0, 1]$. On the other hand, for a POQE$^{(1)}$ $Q(V, F_1[V], F_2[V])$, we have that $I(Q(V, F_1[V], F_2[V])) = \tilde{Q}(f(Y))$ where f is a piecewise linear function in variables belonging to Y and for a POQE$^{(2)}$ $Q(V, F_1[V], F_2[V])$, we have that $I(Q(V, F_1[V], F_2[V])) = \tilde{Q}(g(Y))$ where g is a fraction of piecewise linear functions in variables belonging to Y. By applying Proposition 1 and Remark 3 it then follows that a PSL^{Q} program using piecewise linear quantifier mappings such as in Equation 3 can be transformed to a linear fractional program (LFP). Note that this is only a worst case scenario: if the grounded PSL^{Q} program has no POQE$^{(2)}$'s then we obtain a linear program. We can then use a transformation similar to the approach of Isbell and Marlow [13] to replace a LFP to a set of linear programs by establishing a convergent iterative process. The linear program at each iteration is determined by optimization of the linear program at the previous iteration.

The algorithm we propose for the MPE inference (Algorithm 1) starts by initializing all random variables to zero (i.e., line 2). Then, an iterative process starts by grounding

Algorithm 1 Iterative MPE inference in PSL^Q

Require: PSL^Q program P, evidence variables X and random variables Y

1: $R \leftarrow \emptyset$
2: $I^{(0)}(Y) \leftarrow 0$
3: **for** $i := 1$ **to** k **do**
4: **for** $r \in P$ **do**
5: $R_g \leftarrow$ **ground**(r)
6: **for** $r_g \in R_g$ **do**
7: **for** every Q of type POQE$^{(2)}$ in r_g **do**
8: $I(Q) \leftarrow \tilde{Q}(I(X) \cup I^{i-1}(Y))$
9: **end for**
10: $d_{r_g}(I) \leftarrow 1 - I(r_g)$
11: **if not** $d_{r_g}(I) = 0$ **then**
12: $R \leftarrow R \cup r_g$
13: **end if**
14: **end for**
15: **end for**
16: $f(I) \leftarrow$ **generate**(R)
17: $G(I) \leftarrow$ **transform**$(f(I))$
18: $I^{(i)}(Y) \leftarrow$ **optimize**$(G(I))$
19: **end for**

all rules in a PSL^Q program (i.e., line 3-5). For every grounded quantifier expression Q of type POQE$^{(2)}$, the value of Q is initialized by calculating the value over the known values ($I(X)$) and the current setting of the unknown values ($I^{(0)}(Y)$). In the algorithm, we use the notation $\tilde{Q}(I(X) \cup I^{i-1}(Y))$ to denote this new interpretation of Q at iteration i (i.e., line 7-9). For each rule r_g we then calculate the distance to satisfaction (i.e., line 10). Note that $I(r_g)$ and hence also $d_{r_g}(I)$ can be piecewise linear functions in Y, but here $d_{r_g}(I)$ does not contain fractions of piecewise linear functions since we calculate values for the POQE$^{(2)}$s. Next, we exclude the satisfied grounded rules (i.e., we exclude rules r_g such that $d_{r_g}(I) = 0$) from the optimization since their values will not change the optimization task (i.e., line 11-13). For the optimization task, $f(I)$ (Equation 7) is calculated using the distance to satisfaction of all grounded rules (i.e., line 16). Since $f(I)$ does not contain fractions of piecewise linear functions, it can be transformed to a linear program (i.e., line 17). Finally, the inner optimization in PSL^Q is solved with PSL's scalable, parallelizable message-passing inference algorithm [2] (i.e., line 18). In each iteration, the values of Qs get updated by the most probable assignment of random variables in the previous iteration ($I(X) \cup I^{(i-1)}(Y)$) (i.e., line 8). This process is iteratively repeated for a fixed number of times (i.e., k).

3.2 Weight Learning

The goal of weight learning based on maximum likelihood estimation (MLE) is to maximize the log likelihood of the rules' weight based on the training data in Equation 7. Hence, the partial derivatives of log likelihood with respect to λ_i of rule $r_i \in R$ is

$$-\frac{\delta log(f(I))}{\delta \lambda_i} = E_\lambda[\sum_{r \in R_g i} (d_r(I))^p] - \sum_{r \in R_g i} (d_r(I))^p \tag{10}$$

with E_λ the expected value under the distribution defined by λ, and $R_g i$ is the set of grounded rules of rule r_i. The optimization is based on the voted perception algorithm [6], in which approximation is done by taking fixed-length steps through the

direction of gradient and averaging the points after all steps; out of the scope steps are projected back into the feasible region. To make the approximation tractable, a MPE approximation is used that replaces the expectation in the gradient with the corresponding values in the MPE state. We use our proposed MPE approach for transforming POQE$^{(1)}$s and POQE$^{(2)}$s in our MLE algorithm. We omit the pseudocode of the MLE algorithm for a PSL^{Q} program to save space.

4 Evaluation: Trust Link Prediction

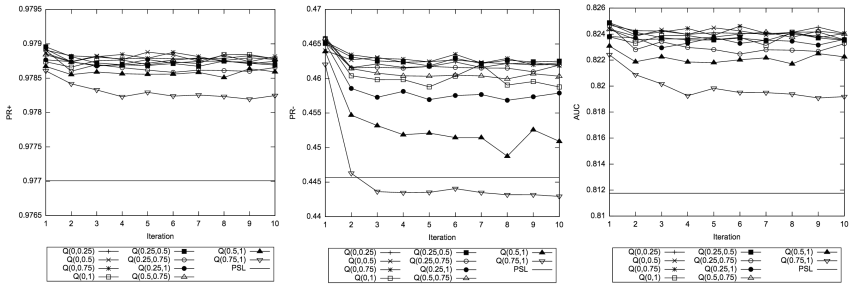
Studies have shown that people tend to rely more on recommendations from people they trust than on online recommender systems which generate recommendations based on anonymous people similar to them. This observation has generated a rising interest in trust-enhanced recommendation systems [26]. The recommendations generated by these systems are based on an (online) trust network, in which members of the community express whether they trust or distrust each other. In practice these networks are sparse because most people are connected to relatively few others. Trust-enhanced recommendation systems therefore rely on link prediction. In [12], trust relations between social media users are modeled and predicted using a PSL model based on structural balance theory [11]. Structural balance theory implies the transitivity of a relation between users. Based on this theory, users are more prone to trust their neighbors in the network rather than unknown other users. In [2][1], this PSL model was evaluated on data from *Epinions*[2], an online consumer review site in which users can indicate whether they trust or distrust each other. Throughout this section, we will use the same sample of Epinions [17]. The sample dataset includes 2,000 users with 8,675 relations, namely 7,974 trust relations and only 701 distrust relations. We systematically perform 8-fold cross-validation and to evaluate the results, we use three metrics, *AUC*: the area under the receiver operating characteristic curve, *PR+*: the area under the precision-recall curves for trust relations, and *PR-*: the area under the precision-recall curves for distrust relations. In each fold, we first learn the weights of the rules based on 7/8 of the trust network and then apply the learned model on the remaining 1/8 to infer the trust/distrust relations. Bach et al. used the model of [12] which is composed of twenty PSL rules in order to predict the degree of trust between two individuals. Sixteen rules from these rules encode possible stable triangular structures involving the two individuals and a third one. For example, an individual is likely to trust people his or her friends trust. The model of [12] is used to predict unobserved truth-values of $Trusts(A, B)$ for pairs of individuals. The results of this model are shown in the first line in Table 2. In this

[1] Source code available at http://psl.umiacs.umd.edu

[2] www.epinions.com

Table 1: PSL^{Q} model for trust link prediction

	Transitive rules
$(R\#1)$	$Knows(A, B) \wedge Trusts(A, B) \wedge Knows(B, C) \wedge Trusts(B, C) \wedge Knows(A, C) \rightarrow Trusts(A, C)$
$(R\#2)$	$Knows(A, B) \wedge \neg Trusts(A, B) \wedge Knows(B, C) \wedge Trusts(B, C) \wedge Knows(A, C) \rightarrow \neg Trusts(A, C)$
$(R\#3)$	$Knows(A, B) \wedge Trusts(A, B) \wedge Knows(B, C) \wedge \neg Trusts(B, C) \wedge Knows(A, C) \rightarrow \neg Trusts(A, C)$
$(R\#4)$	$Knows(A, B) \wedge \neg Trusts(A, B) \wedge Knows(B, C) \wedge \neg Trusts(B, C) \wedge Knows(A, C) \rightarrow Trusts(A, C)$
	Cyclic rule
$(R\#5)$	$Knows(A, B) \wedge Trusts(A, B) \wedge Knows(B, C) \wedge Trusts(B, C) \wedge Knows(C, A) \rightarrow Trusts(C, A)$
	Complementary rules
$(R\#6)$	$Knows(A, B) \wedge Knows(B, A) \wedge Trusts(B, A) \rightarrow Trusts(A, B)$
$(R\#7)$	$Knows(A, B) \wedge Knows(B, A) \wedge \neg Trusts(B, A) \rightarrow \neg Trusts(A, B)$
$(R\#8)$	$Knows(A, B) \wedge Average(\{Trusts\}) \rightarrow Trusts(A, B)$
$(R\#9)$	$Knows(A, B) \wedge Trusts(A, B) \rightarrow Average(\{Trusts\})$
	PSL^{Q} rules based on the transitive rules
$(R\#10)$	$Q(X, Knows(A, X) \wedge Trusts(A, X), Knows(X, C) \wedge Trusts(X, C)) \wedge Knows(A, C) \rightarrow Trusts(A, C)$
$(R\#11)$	$Q(X, Knows(A, X) \wedge \neg Trusts(A, X), Knows(X, C) \wedge Trusts(X, C)) \wedge Knows(A, C) \rightarrow \neg Trusts(A, C)$
$(R\#12)$	$Q(X, Knows(A, X) \wedge Trusts(A, X), Knows(X, C) \wedge \neg Trusts(X, C)) \wedge Knows(A, C) \rightarrow \neg Trusts(A, C)$
$(R\#13)$	$Q(X, Knows(A, X) \wedge \neg Trusts(A, X), Knows(X, C) \wedge \neg Trusts(X, C)) \wedge Knows(A, C) \rightarrow Trusts(A, C)$
	PSL^{Q} rule based on the cyclic rule
$(R\#14)$	$Q(X, Knows(A, X) \wedge Trusts(A, X), Knows(X, C) \wedge Trusts(X, C)) \wedge Knows(C, A) \rightarrow Trusts(C, A)$



(a) (b) (c)

Fig. 3: (a) PR+ , (b) PR-, and (c) AUC of changing α and β in the quantifier mapping $\tilde{Q}$

paper, we propose a model based on 4 transitive rules (rules 1–4 in Table 1) and one rule which models the cyclic relation between 3 users (rule 5 in Table 1). Rules 6-9 in Table 1 are complementary rules that we refer to [12] for further explanation. The atom $Average(\{Trusts\})$ in rules 8 and 9 is a constant which refers to the global average value of observed trust scores. This atom is useful for the disconnected parts of the trust network without any known trust relation. These four rules are also used in the PSL model of [2]. To investigate whether we can improve the accuracy of the predictions by introducing rules with soft quantifier expressions, we construct PSL^Q rules based on a triad relation over a set of users instead of a third party (rules 10-14). The full PSL^Q model then consists of all rules displayed in Table 1.

We examine what happens when changing the thresholds for the quantifier mappings $\tilde{Q}$ (Equation 3). We have investigated ten different quantifier mappings by changing the values of α and β by steps of 0.25. In this way, we obtain ten different PSL^Q programs. For every program, we applied Algorithm 1 for all $k \in \{1, 2, \ldots, 10\}$. Note that for $k = 1$ the output of the MPE inference is equivalent to the output generated by a PSL^Q model with only FOQEs by ignoring the unknown values. Figure 3 presents changes of the three metrics of these ten PSL^Q models with different quantifier mappings. All ten PSL^Q models outperform the PSL model (shown with a line) in all iterations and in all three metrics, except for the PSL^Q model with $\tilde{Q}_{[0.75,1]}$ in $PR-$ after the first two iterations. An explanation for this is the fact that people trust/distrust a third party as soon as a few/some of their trusted/distrusted friends trust/distrust that person and not most of them, i.e., more than 75%. Interestingly, by decreasing both α and β values, results get better. The model with the best predicting scores is PSL^Q with $\tilde{Q}_{[0,0.25]}$ as a quantifier mapping representing "a few"(see Table 2).

Figure 4 emphasizes the importance of the PSL^Q rules with quantifier expressions (rules 10–14) after the weight learning phase. Bars represent average and error bars

Table 2: Values with a $*$ are statistically significant with a rejection threshold of 0.05 and values in bold are statistically significant with a rejection threshold of 0.1 using a paired t-test w.r.t. the PSL model [2]. Distrust prediction is more challenging than trust prediction (i.e., PR- values are overall lower than PR+ values) because of the unbalanced nature of the data (7,974 trust vs. 701 distrust relations)

Method	PR+	PR-	AUC
PSL	0.977	0.446	0.812
PSL^Q ($\tilde{Q}_{[0,0.25]}$), ($k = 1$)	**0.979**$*$	**0.467**$*$	**0.825**$*$
PSL^Q ($\tilde{Q}_{[0,0.25]}$), ($k = 10$)	**0.979**$*$	**0.463**	**0.824**$*$

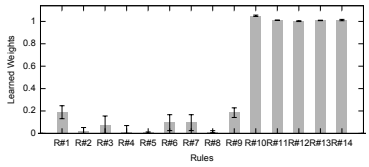


Fig. 4: Learned values of the weight of the 14 rules of the PSL^Q model

represent minimum and maximum weights of the rules learned in 8 folds for the PSL^Q model with quantifier mapping $\tilde{Q}_{[0,0.25]}$. These results show that using soft quantifiers not only improves the accuracy of trust and distrust predictions but also that the rules containing soft quantifiers, i.e. rules 10-14, play a major part in this by dominating all other rules in terms of weight. In these experiments, we used one quantifier mapping for all the quantifiers in a PSL^Q program; however it is possible to use different mapping functions for each quantifier expression in a PSL^Q model, which is an interesting direction for future research.

5 Conclusion

In this paper, we have introduced PSL^Q, the first SRL framework that supports reasoning with soft quantifiers, such as "most" and "a few". PSL^Q is a powerful and expressive language to model uncertain relational data in an intuitive way. Since this expressivity pushed beyond the capabilities of existing PSL-MPE solvers, we have introduced and implemented new inference and weight learning algorithms that can handle rules with soft quantifiers. We have shown how the higher expressivity of PSL^Q can lead to better results in practice by extending an existing PSL model for link prediction in social trust networks with rules that contain soft quantifiers. We have presented the effects of using different interpretations of soft quantifiers in our trust model. As a next step, we want to include an automatic way of learning the best interpretation for each quantifier expression in a PSL^Q model. Besides trust link prediction, many other applications could benefit from the use of soft quantifiers. Exploring the effects of using soft quantifiers in PSL^Q models for other AI applications is therefore another promising research direction. Furthermore, in addition to the approach of Zadeh that we have used in this paper, other approaches for soft quantifiers have been proposed, most notably Yager's OWA-operators [28]; we plan to investigate them in our future work.

Acknowledgements: We would like to thank the anonymous reviewers for their helpful comments and suggestions. This work was funded in part by the SBO-program of the Flemish Agency for Innovation by Science and Technology (IWT-SBO-Nr. 110067) and NSF grant IIS1218488. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

References

1. S. H. Bach, M. Broecheler, B. Huang, and L. Getoor. Hinge-loss markov random fields and probabilistic soft logic. arXiv:1505.04406 [cs.LG], 2015.
2. S. H. Bach, B. Huang, B. London, and L. Getoor. Hinge-loss Markov random fields: Convex inference for structured prediction. In *Proceedings of the Uncertainty in Artificial Intelligence (UAI)*, 2013.
3. I. Beltagy and K. Erk. On the proper treatment of quantifiers in probabilistic logic semantics. *IWCS 2015*, page 140, 2015.

4. I. Beltagy, K. Erk, and R. J. Mooney. Probabilistic soft logic for semantic textual similarity. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL-14)*, pages 1210–1219, 2014.
5. F. Bobillo and U. Straccia. fuzzyDL: An expressive fuzzy description logic reasoner. In *FUZZ-IEEE*, pages 923–930, 2008.
6. M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 Conference on Empirical methods in Natural Language Processing*, 2002.
7. M. Delgado, D. Sánchez, and M. A. Vila. Fuzzy cardinality based evaluation of quantified sentences. *International Journal of Approximate Reasoning*, (1):23–66, 2000.
8. S. Fakhraei, B. Huang, L. Raschid, and L. Getoor. Network-based drug-target interaction prediction with probabilistic soft logic. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2014.
9. G. Farnadi, S. H. Bach, M. F. Moens, L. Getoor, and M. De Cock. Extending psl with fuzzy quantifiers. In *Proceedings of StarAI 2014: Fourth International Workshop on Statistical Relational AI at AAAI*, 2014.
10. L. Getoor and B. Taskar. *Introduction to statistical relational learning*. MIT press, 2007.
11. F. Heider. The psychology of interpersonal relations. *New York: Wiley*, 1958.
12. B. Huang, A. Kimmig, L. Getoor, and J. Golbeck. A flexible framework for probabilistic models of social trust. *Social Computing, Behavioral-Cultural Modeling and Prediction*, pages 265–273, 2013.
13. J. R. Isbell and W. H. Marlow. Attrition games. *Naval Research Logistics Quarterly*, 3(1-2):71–94, 1956.
14. D. Jain, A. Barthels, and M. Beetz. Adaptive Markov Logic Networks: Learning Statistical Relational Models with Dynamic Parameters. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, pages 937–942, 2010.
15. S. M. Kazemi, D. Buchman, K. Kersting, S. Natarajan, and D. Poole. Relational logistic regression. In *International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 2014.
16. G. Klir and B. Yuan. *Fuzzy sets and fuzzy logic*. Prentice Hall New Jersey, 1995.
17. J. Leskovec, D. Huttenlocher, and J. Kleinberg. Signed Networks in Social Media. In *Proceedings of the 28th ACM Conference on Human Factors in Computing Systems (CHI)*, 2010.
18. D. Lowd and P. Domingos. Recursive random fields. In *IJCAI*, pages 950–955, 2007.
19. B. Milch, L.S. Zettlemoyer, K. Kersting, M. Haimes, and L. P. Kaelbling. Lifted probabilistic inference with counting formulas. In *Proceedings of the International Conference on Artificial Intelligence (AAAi)*, volume 8, pages 1062–1068, 2008.
20. S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, pages 629–679, 1994.
21. D. Poole, D. Buchman, S. M. Kazemi, K. Kersting, and S. Natarajan. Population size extrapolation in relational probabilistic modelling. In *Scalable Uncertainty Management*, pages 292–305. Springer, 2014.
22. D. Poole, D. Buchman, S. Natarajan, and K. Kersting. Aggregation and population growth: The relational logistic regression and markov logic cases. In *Proceedings of the UAI-2012 Workshop on Statistical Relational AI*, 2012.
23. J. Pujara, H. Miao, L. Getoor, and W. Cohen. Knowledge graph identification. In *International Semantic Web Conference (ISWC)*, 2013.
24. M. Richardson and P. Domingos. Markov logic networks. *Machine learning*, (1-2):107–136, 2006.
25. G. Van den Broeck, W. Meert, and A. Darwiche. Skolemization for weighted first-order model counting. *arXiv preprint arXiv:1312.5378*, 2013.
26. P. Victor, C. Cornelis, and M. De Cock. Trust and recommendations. In *Recommender Systems Handbook*. Springer, 2011.
27. R. West, H. S. Paskov, J. Leskovec, and C. Potts. Exploiting social network structure for person-to-person sentiment analysis. *Transactions of the Association for Computational Linguistics (TACL)*, 2:297–310, 2014.
28. R. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on Systems, Man and Cybernetics.*, (1):183–190, 1988.
29. L. A. Zadeh. A computational approach to fuzzy quantifiers in natural languages. *Computers & Mathematics with Applications*, (1):149–184, 1983.