Managing Trust in Virtual Communities with Markov Logic

Denis Mauá and Fabio Cozman

Escola Politécnica, Universidade de São Paulo,
Av. Prof. Mello Moraes 2231, São Paulo, SP, Brazil

Abstract. This paper describes advances in the development of a trust management system for web-based virtual communities. We review current research on computational trust and report on a proposal for trust management with Markov Logic, a recently developed probabilistic logic language that aims at unifying statistical relational procedures. This work addresses two complex questions: how to learn trust metrics for a group of interacting agents upon a machine learning perspective, and how to use pairwise trust metrics structured as a trust network to compute personalized trust metrics.

Key words: knowledge discovery, trust management, Markov logic

Denis Mauá is a MSc student with expected thesis defense in September 2008. This paper reflects his current work on trust management and Markov logic.

1 Introduction

Trust is essential in human relationships [1]. Whether these relationships occur in the real world or in the internet, trust plays a major role in how interactions occur. Each individual has its own way of managing the reliability of others. Some techniques are rather elementary, such as the recording of personal experiences and word-of-mouth. As it happens with other information-based tasks, technology can improve the quality of trust metrics and facilitate their management. This management is crucial in virtual environments [2].

To support trust in web-based communities, trust management systems have been extensively used. Auction sites like eBay, for example, provide reliability ratings of their users based on feedbacks reported once a transaction is completed. On peer-review sites such as Epinions.com, trust is explicitly stated among users, and such information is used to filter the huge amount of information a user may receive. Other examples include recommender systems [3], peer-to-peer systems [4], collaborative filtering [5], and public key cryptography [6], just to name a few.

This paper describes ongoing work on the construction of a system capable of acquiring trust metrics from interaction data and of propagating acquired data to provide users with personalized reputation of others. The system is modelled using Markov Logic [7], a powerful statistical relational language, and benefits from an open-source tool freely available.

This paper is organized as follows. Section 2 gives a few necessary definitions. The Markov Logic framework we use to model trust is explained briefly in Section 3. Our approach to trust management is presented in Section 4, and our experimental methodology is given in Section 5. Preliminary results are reported in Section 6. A brief review of our subject is presented in Section 7. Final remarks and topics for further investigations are given in Section 8.

2 Preliminary Considerations on Trust and Reputation

Trust is a vague concept that many researchers have attempted to define [8]. Here, we define trust as *a subjective measurement of an entity on the certainty of the execution of some action by some other entity under a particular situation* [9].

The study of human relationships has described many techniques that people use when dealing with reliability issues. One of the most common is the assignment of *reputation*. Reputation is also a vague concept, as hard to define as trust; we define the reputation of an entity B to another entity A as *an aggregation of information from third-parties regarding B's trustworthiness with A's personal believes about B.* Actually, this definition tries to unify existing concepts, so reputation the way we define it is actually what others have called a *trust metric* [10].

The information fetched by a system about the direct trust that entities have on others can be stored in a graph structure called *trust network*. A trust network is a graph $\mathcal{T} = (\mathcal{V}, \mathcal{E})$, with $\mathcal{V}$ representing the entities and $\mathcal{E}$ denoting direct trust metrics. If trust metrics are not boolean variables but rather real valued, we attach a weight function $\mathcal{W} : \mathcal{E} \times \mathcal{E} \to \mathbb{D} \subset \mathbb{R}$ to the edges of the graph to incorporate such information. In the literature it is common to use the real interval $[0, 1]$ as the weight function domain, but there are cases where the domain can be an interval ranging from 1 to 10 or even discrete values such as "not reliable", "reliable" and "very reliable".

Figure 1 depicts a small trust network representing the relationships among four entities, John, Anna, Mary and Bob. An arc from a vertex u to a vertex v indicates that an entity U trusts an entity V. In the example, we have the direct trust metrics stating that John trusts Anna and Mary, Anna trusts Mary and Bob, and Mary trusts Bob. Reputations not present in the network can be computed through trust propagation. For instance, the reputation of Bob for John, can be computed as an aggregation of the trust that Mary and Anna have on him, plus the trust that John has on Anna and Mary.

3 Markov logic

Markov logic offers a way to soften the hard constraints that first-order logic imposes to knowledge representation. Take the example of representing the knowledge that birds often fly. The great majority of birds are able to fly, but there

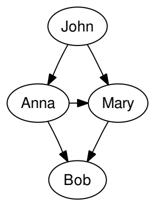

Fig. 1. Example of a small trust network involving four entities. An arc from a vertex A to a vertex B represents the notion that entity A has trust in entity B. Reputations (indirect trust metrics) can be computed propagating trust through paths.

is a small number of them such as the penguin that cannot fly. If we would like to represent this in standard logic we would have to state that any bird that it is not a penguin and it is not any other non-flying bird (we have to explicitly cite all of them) flies. This can be a burden, and even unfeasible if we were not able to list all the exceptions. Moreover, even if we could list all non-flying birds, if suddenly a new kind of non flying bird is discovered our application may become useless. Markov logic deals with such issues by attaching probabilities to first-order formulas. Thus, we can just state that there is 60% probability that a bird flies and that penguins are birds that do not fly.

Formally, we define a knowledge base in Markov logic as a set of (implicitly conjoined) weighted first-order formulas. The probability of a particular interpretation is given by

$$P(X = x) = \frac{1}{Z}\exp\left(\sum_i w_i f_i\right), \tag{1}$$

where Z is the partition function given by $\sum_{x\in X} \exp\left(\sum_i w_i f_i\right)$, and w_i denotes the weight attached to the ith grounded formula f_i.

Markov Logic can also be seen as template language for building large Markov Networks. Let L denote a Markov Logic consisting in a set of pairs (f_i, ω_i), where f_i is a first-order logic formula, and ω_i is a real-valued variable, representing the ith formula weight. Along with a constant domain C, a Markov Network $M_{L,C}$ is defined as follows:

1. There is one binary node in $M_{L,C}$ for every grounded predicate in L. The node values 1 if the grounded atom is true, and 0 otherwise. If two predicates appear in the same formula then an edge connecting them in the graph is added.
2. There is one feature in $M_{L,C}$ for every grounded formula $f_i \in L$. The feature values 1 if the grounded formula is true, and 0 otherwise. The weight of the feature is the variable ω_i attached to f_i.

The birds example can be represented in Markov Logic by the formula w `Bird(x)` $\Rightarrow$ `Fly(x)`. According to Equation (1), the probability that a specific individual `A` is a flying bird is $Pr(\texttt{Bird(A)}, \texttt{Fly(A)}) = \frac{e^w}{1+3e^w}$, and the probability that the same individual do not fly is $Pr(\texttt{Bird(A)}, \neg\texttt{Fly(A)}) = \frac{1}{1+3e^w}$. One can see that A is e^w times more likely of flying than of not flying, showing the greater flexibility of Markov Logic in relation to traditional logic when representing uncertain knowledge.

4 Trust management

A trust management system consists of two tasks: trust metrics acquisition and trust metrics propagation. In the first task, the system must apply some technique to learn trust metrics from user interaction data or resort to user statements. The second task must aggregate information collected in the former stage and manipulate it to provide trust metrics of any two entities, specially for those that no direct trust metric is available.

There are two approaches to managing trust metrics. In the *centralized* approach, the system tracks all interactions of the users within the system and use such information to produce a global reputation score for its users. Thus, users are assessed by the amount of interactions with other users, but no personalization is taken into account. For example, if a user A have many good interactions with another user B but a few bad interactions with user C, the system provides the same score (probably a high one) to user A disregarding whether which user is requiring it. This can be appropriate to abort attempts of biasing some user score, but can result in poor accuracy in some situations. In [10], for example, the authors concluded, after conducting some experiments with the Epinions.com data, that a decentralized approach suited better the case of controversial users, that is, users that have been rated both as a trustworthy and as a untrustworthy entity by different raters. Yet, their study reported that for non-controversial users, both approaches exhibited similar performance.

In *decentralized* management systems, trust metrics are personalized in that they differ according to the user that requests them. In the former example, a decentralized system would provide user C with a different metric of user A (probably a lower one) than it would for user B. Commonly, centralized systems are more concerned with the acquisition of trust metrics and do not deal with the propagation of trust; decentralized systems, on the other hand, concerns with the propagation of trust and often do not perform further exploitations of the trust acquisition procedure.

4.1 Trust acquisition

When trust is obtained via trust statements of the user, there is no much work left. These statements are translated into trust metrics in a straightforward manner. For binary-valued trust functions, one may simple added an edge from vertex A to vertex B in the trust graph if user A states that she trusts user B. For real-valued trust metrics, we may repeat the operation attaching the stated amount of trust as the edge weight. However, in order to leverage metrics, it may be interesting to apply some normalization of the statements before converting them into metrics.

Obtaining trust from interaction data is a difficult issue. From a pattern recognition point-of-view, we wish to predict the value of the relationship of every possible pair of entities in the system, based on attributes of entity pairs. For example, the system may record the rates users give to other users' posts and use this information to predict whether a user has trust on another one.

The categorization of user interactions are particular to the domain of the application. In practice they can range from binary interactions, denoting only positive or negative classes, to ternary interactions, such as negative, positive and neutral, to real-valued interactions. Here, we restrict our attention to applications with finite discrete interaction categories and binary-valued trust metrics.

The knowledge base below presents a model for learning trust from interaction data categorized as integers.

$$\begin{array}{lll} w_1 & \mathtt{interaction}(\mathtt{a},\mathtt{b},\mathtt{1}) \Rightarrow & \mathtt{trusts}(\mathtt{a},\mathtt{b}) \\ \vdots & \vdots & \vdots \\ w_n & \mathtt{interaction}(\mathtt{a},\mathtt{b},\mathtt{n}) \Rightarrow & \mathtt{trusts}(\mathtt{a},\mathtt{b}) \ , \end{array}$$

where the weights w_i represent how much influence the i-th type of interaction has on the reputation of b for a. For example, it is common to have negative interactions with higher absolute weight values than positive ones. Note that trust is not a symmetric concept, thus `trusts(a,b)` and `trusts(b,a)` represent distinct relationships of the world. In the same way, interactions are not symmetric; the same interaction can be categorized as a good one to one of the entities, and as a bad one to the other.

Trust acquisition is actually a relational classification problem. Interaction weights are learned from a training data set and then used to compute trust metrics by making probabilistic inferences. The probability that a given entity A trusts another entity B given their past interactions is denoted

$$Pr(\mathtt{trusts}(\mathtt{A},\mathtt{B})|\mathtt{interactions}) \ ,$$

where `interactions` is the set of the atoms representing past interactions involving these two entities. If the above probability is greater than 0.5 we assume that there is a trust relationship from A to B, and otherwise we assume that there is no relationship.

4.2 Trust propagation

Trust propagation is central in the computation of reputations. Basically, all trust propagation techniques work by assuming that trust is a transitive concept. In other words, we assume that if A trusts B and B trusts C than it is likely that A will trust C as well. This assumption is supported by the balance theory of social networks analysis [11].

Trust propagation can be modeled in Markov Logic as

$$w\ \texttt{trusts}(\texttt{a},\texttt{b}) \wedge \texttt{trusts}(\texttt{b},\texttt{c}) \Rightarrow \texttt{trusts}(\texttt{a},\texttt{c})\ .$$

The w weight dictates the strength of trust transitivity and is particular of a domain. Its value can be find empirically or learned from data.

5 Methodology

To validate our models, we intend to perform experiments with both synthetic and real-world datasets. Synthetic datasets have the advantage of allowing highly controlled experiments, so that side effects can be isolated and studied in depth. For example, the density of a network, or its centrality can affect the overall performance of trust propagation algorithms. The amount of interaction data recorded can be crucial to the success of trust acquisition procedures. Generating data for trust propagation is simply a question of creating parameterized graphs. On the other hand, generating data for trust acquisition is a much more difficult task, and as far as we know, there is no example of in the literature. If we follow a generative approach, all we need is to generate a trust network graph and from this graph sample interaction data. However, in order to have data with a real-world profile, further investigations are needed.

Trust acquisition can be assessed as traditional classification tasks. Two of the most common metrics in machine learning are *precision*, which measures the amount of correct guesses over the total number estimations, and *recall*, which measures the amount of correct guesses over the maximum number of correct estimations.

Trust propagation can also benefit from machine learning assessment techniques. In particular, *leave-one-out* is highly appropriate because the strong relational character of trust makes dataset splitting a hard task [12]. Let $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ denote a given trust network. Then, for every edge $(u, v) \in \mathcal{E}$ we compute the probability $P = Pr(\texttt{trusts}(\texttt{U},\texttt{V})|\mathcal{T}')$ of u trusting v using network $\mathcal{T}' = (\mathcal{V}, \mathcal{E}')$, where $\mathcal{E}' = \mathcal{E} - \{(u, v)\}$. If $P \geq 0.5$ we assume that there is an edge pointing from u to v in the estimated trust network. Precision is then computed as the fraction of correct estimated trust edges over the total number of estimations. Standard trust propagation procedures often use a coverage metric to measure the amount of trust metrics that the algorithm was able to estimate at some degree. This is because many approaches requires that a path from u to v exists in the graph in order to be capable of computing a trust metric $T(u, v)$. However, our model in Markov Logic does not have this constraint, so our algorithm coverage is always maximum.

6 Preliminary Results

Relational data preprocessing is a burden that requires careful work, so as to maintain basic data properties when creating datasets for evaluation.

Epinions.com is a peer-review web site where users are encouraged to rate other users' posts in order to system improve usability. Besides being able to rate contents, users can also state the users they trust, and such statements are used to filter information when visualizing reviews on the site. The full data consists of a graph of $131,828$ nodes and $840,799$ edges representing trust statements. The network has an average degree of 12.756 and it is composed by $75,753$ strongly connected components. The great majority of these components stands for isolate users (users with null in- and out-degree). In fact, there is one big component with $36,490$ nodes and $602,518$ edges, whereas the other components has orders ranging from 1 to 15 nodes and sizes ranging from 0 to 59 edges.

Although our objective is to work with the largest component, because it is better representative of real world, Markov Logic has at this point scalability problems and does not suit well massive data scenarios (at least with the implementation in Alchemy). Thus, the final goal is to reduce the main component to an order of $\sim 5,000$ nodes so Markov Logic can work well. However, as discussed earlier, dealing with relational data is not a trivial issue. We are currently engaged in the development of appropriate methods for data preprocessing. Meanwhile, we conducted experiments with the second largest component, composed of 15 nodes and 42 edges. The network graph is depicted in Figure 2.

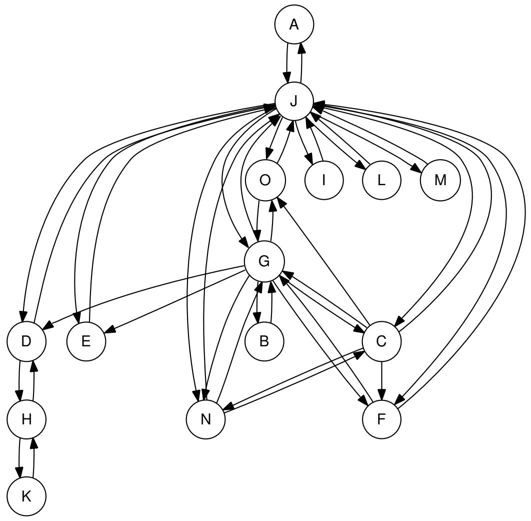

Fig. 2. Second largest component of Epinions.com dataset containing 15 nodes, 42 edges and an average degree of 5.6.

Using the model described in subsection 4.2 with a value of $w = 0.4$ empirically chosen, we implemented a leave-one-out procedure for the trust propagation task and measured the precision. This procedure is described as Exp_1 in table 1. Most of the errors were due to edges that when removed make the graph disconnected. In other words, the procedure fails for group of individuals that do not share the transitivity property of trust. For example, in Figure 2 the edge (K, H) is the only path connecting node K to H, so removing it makes the model incapable of predicting the trust from K to H via third parties. Thus, we repeated the experiment ignoring edges that lead to disconnected graphs. In total, 16 edges were ignored. This approach had 100% precision as shown in the results table (Exp_2). This result was quite amazing, but the small number of individuals considered makes it untrustworthy. Experiments with bigger datasets are expected to better evaluate our method.

Table 1. Trust propagation performance of the data depicted in Figure 2. In Exp_1 a full leave-one-out procedure was performed. In Exp_2 edges that lead to disconnected graphs were ignored from the evaluation, what greatly improved performance.

Procedure	Precision
Exp_1	60.95%
Exp_2	100.0%

7 Related work

One of the earlier attempts to deal with the trust concept under a computation point of view is due to [13], where the author summarizes past studies of the trust concepts in several fields.

In [9], the authors propose a trust management system based on entity interactions. They define the trust metric as a categorical variable and provide an algorithm to compute a trust score of others entities based on past interactions. Witness trust is used to compute source reliability and not only to propagate trust through a chain of trust metrics. The work is one of the first to attempt to provide a trust management in virtual communities.

A brief proposal on how agents should learn trust from experience based on graph-based probabilistic models is in [14]. The authors also attempt to the propagation of trust through a network of recommendations, and discuss some other sources of trust such as social role. However, as much of the work in the area, this is an agent-centric treatment of trust, where the main concern is in how an agent should behave in a risky environment regarding collaboration issues.

Recently, researchers have focused at the propagation of trust metrics aiming at assigning trust to sources in the semantic web vision. In [15], the authors present a good explanation on the common graph-based techniques to manage trust and present a new algorithm based on Markov chains to compute personalized trust scores. In the data used in the work, the authors noted no substantial difference in the overall performance of both methods.

Graph-based techniques are also explored in [16], where the authors describe an ontology for the trust concept to be used within the semantic web, based on a widespread ontology of the acquaintance concept. Experiments with standard propagation techniques are conducted with data gathered from a virtual community and the author reports on the success of such methods for the task in question. The authors also suggest some applications that could profit from trust management techniques such as an e-mail filtering system that ranks incoming e-mails by their sender reputations.

There are some nice reviews on the literature of the subject in [17] and [18], where the authors summarize the research on trust definition in the computational realm, trust propagation issues and trust management applications.

8 Conclusion and Future Work

The internet brings new challenges to trust management applications. Indeed, with the spread of the semantic web, the question of how to deal with trust tends to increase in importance. The present research addresses such needs, presenting a new approach to the problems of trust acquisition and trust propagation. In particular, the use of an expressive language supported by a sound probabilistic framework represents a significant advance to existing methods. As one can see from the examples in the paper, models in Markov Logic can be very concise and readable.

There is still much work to do, but our preliminary results show that Markov Logic can be quite an appropriate language to use in our setting. Because we do not use graph-theoretic techniques, our method does not have coverage problems, being superior in this criterion to other methods. However, extra care should be taken during data preparation and scalability issues in order to not bias data.

For the future, we intend to focus on synthetic data generation and relational data preprocessing techniques, aiming at good benchmarks for trust management techniques. In this paper only dichotomous trust metrics were considered; extending trust functions to allow real-valued metrics can give us more flexibility and leverage our technique with the majority of techniques in the literature, which are based on real-valued trust metrics.

Acknowledgments.

The authors gratefully acknowledge generous support from HP Labs.

References

1. Buskens, V.: The social structure of trust. Social Networks **20** (July 1998) 265–289(25)
2. Resnick, P., Zeckhauser, R., Swanson, J., Lockwood, K.: The value of reputation on eBay: A controlled experiment. Experimental Economics **9**(2) (June 2006) 79–101

3. Massa, P., Avesani, P.: Trust-aware recommender systems. In: RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems, New York, NY, USA, ACM (2007) 17–24
4. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in p2p networks. In: WWW '03: Proceedings of the 12th international conference on World Wide Web, New York, NY, USA, ACM (2003) 640–651
5. Kong, J.S., Boykin, P.O., Rezaei, B.A., Sarshar, N., Roychowdhury, V.P.: Scalable and reliable collaborative spam filters: Harnessing the global social email networks. In: CEAS. (2005)
6. Abdul-Rahman, A.: The pgp trust model. the Journal of Electronic Commerce (1997)
7. Domingos, P., Kok, S., Lowd, D., Poon, H., Richardson, M., Singla, P.: Markov logic. In Raedt, L.D., Frasconi, P., Kersting, K., Muggleton, S., eds.: Probabilistic Inductive Logic Programming. Volume 4911 of Lecture Notes in Computer Science., Springer (2008) 92–117
8. Hussain, F.K., Chang, E.: An overview of the interpretations of trust and reputation. In: AICT, IEEE Computer Society (2007) 30
9. Abdul-Rahman, A., Hailes, S.: Supporting trust in virtual communities. In: HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 6, Washington, DC, USA, IEEE Computer Society (2000) 6007
10. Massa, P., Avesani, P.: Controversial users demand local trust metrics: An experimental study on epinions.com community. In Veloso, M.M., Kambhampati, S., eds.: AAAI, AAAI Press / The MIT Press (2005) 121–126
11. Wasserman, S., Faust, K.: Social network analysis. Cambridge University Press, Cambridge (1994)
12. Macskassy, S.A., Provost, F. In: A Brief Survey of Machine Learning Methods for Classification in Networked Data and an Application to Suspicion Scoring. Volume 4503. Springer Berlin / Heidelberg (2007) 172–175
13. Marsh, S.: Formalising Trust as a Computational Concept. PhD thesis, University of Sterling (1994)
14. Esfandiari, B., Chandrasekharan, S.: On how agents make friends: Mechanisms for trust acquisition. In: Proceedings of the Fifth International Conference on Autonomous Agents Workshop on Deception, Fraud and Trust in Agent Societies. (2001) 27–34
15. Richardson, M., Agrawal, R., Domingos, P.: Trust management for the semantic web. In: Proceedings of the Second International Semantic Web Conference, Sanibel Island, Florida (2003)
16. Golbeck, J., Parsia, B., Hendler, J.A.: Trust networks on the semantic web. In Klusch, M., Ossowski, S., Omicini, A., Laamanen, H., eds.: CIA. Volume 2782 of Lecture Notes in Computer Science., Springer (2003) 238–249
17. Sabater, J., Sierra, C.: Review on computational trust and reputation models. Artif. Intell. Rev. **24**(1) (2005) 33–60
18. Artz, D., Gil, Y.: A survey of trust in computer science and the semantic web. Web Semant. **5**(2) (2007) 58–71