A Bayesian Multiagent Trust Model for Social Networks

Noel Sardana, Robin Cohen, Jie Zhang, and Shuo Chen

***Abstract*— In this paper, we introduce a framework for modeling the trustworthiness of peers in the setting of online social networks. In these contexts, it may be important to be filtering the wealth of messages that have been sent, which form the ongoing communication within a large community of users. This is achieved by constructing an intelligent agent that reasons about the message and each peer rater of the message, learning over time to properly gage whether a message is good or bad to show a user, based on message ratings, rater similarity, and rater credibility. Our approach employs a partially observable Markov decision process for trust modeling, moving beyond the more traditional adoption of probabilistic reasoning using beta reputation functions. In addition to outlining the technique in full, we present empirical results to demonstrate the effectiveness of our methods, both in simulations featuring head to head comparisons with competitors, and in the context of some existing online social networks where ground truth data are available.**

***Index Terms*— Bayesian approaches, decision making, intelligent agents, peer interaction, social networks, trust modeling.**

I. Introduction

TODAY, users are increasingly engaged with online media via the Internet. A plethora of available information, ranging from discussion boards of massively open online courses to participatory media sites suggests the following question: how can we help users to sift through excess information to retrieve the most relevant objects of interest? In this paper, we develop a new model[1] grounded in Bayesian statistics that adopts a multiagent systems approach, drawing on concepts from partially observable Markov decision processes (POMDPs) in order to learn about users, going beyond other Bayesian approaches like Bayesian Learning to Adapt to Deception in E-Marketplaces (BLADE) [2] to explicitly account for user utilities; we then apply this to message recommendation. The aim is to be offering a solution for users so that they are shown those messages which have high rewards for them to view (setting aside those messages deemed to be of lesser value to them).

In addition to demonstrating the efficacy of our approach in simulation, we evaluate against real-world data from Reddit.com and Epinions.com, websites where users share and rate messages, showing that our model performs well in this realistic setting, in terms of true/false positives and true/false negatives when set against the ground truth. As our model incorporates a modeling of user credibility, it is also interesting from the perspective of trust modeling research. It uses Bayesian learning to derive an observation function that combines user features in a way that is "statistically correct" given the environment and as such begins to address Sen's [3] challenge of resolving trust modeling *use*, i.e., the challenge of clarifying the decision making that should ensue, once trust modeling is performed.

It is important to note that intelligent agents may act as assistants, provided with the autonomy to help to direct the decision making of humans. These agents may, for instance, enable humans to be directed to the most valuable messages in online social media. It then becomes essential for trustworthiness to be modeled well. The framework we outline here provides direction for human users to navigate the waters of social media, setting aside information that may be less valuable or more suspect. Toward the end of this paper, we return to this theme, reflecting briefly as well on the current climate of social networks and the possibility of artificial agents populating the networks with misleading messages. We then elaborate on how our model may be of assistance when this happens. As such, we align with the desiderata set out in [4]: "to examine the antecedents of trust development. . . and repair in . . . online communities. . . (to examine). . . the sources. . . that drive behavior in social media," motivated by the observation that these contexts may well be used to shape the beliefs of individuals.

The reliability of intelligent agents has been examined by a community of artificial intelligence researchers as an issue of trust modeling. These models have primarily adopted a probabilistic method of reasoning, focused on determining whether agents are trustworthy or not, as a kind of binary decision. In the work presented here, we are basically advocating a more principled modeling of trust, grounded in the POMDP formalism. The value of this approach is explained, in comparison with alternate methods for trust modeling, in the context of online social networks. This is performed in order to evaluate interactions where advice is provided. Messages in these networks may include, for instance, comments on an assignment in a Coursera course; peer advice may suggest that this message is worthwhile for others to view. With a vast number of peers participating, it may be challenging for a user to simply be shown all of the conflicting opinions (for example, the most promising methods which can help

Manuscript received May 15, 2018; accepted October 21, 2018. Date of publication November 19, 2018; date of current version December 3, 2018. The work of J. Zhang was supported by the MOE AcRF Tier 1 under Grant M4011894.020. *(Corresponding author: Jie Zhang.)*

N. Sardana was with the University of Waterloo, Waterloo, ON N2L 3G1, Canada. He is now with Facebook, Inc., Menlo Park, CA 94025 USA.

R. Cohen is with the University of Waterloo, Waterloo, ON N2L 3G1, Canada.

J. Zhang and S. Chen are with the Computational Intelligence Laboratory, School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: zhangj@ntu.edu.sg).

Digital Object Identifier 10.1109/TCSS.2018.2879510

[1]An abbreviated first version of this paper was presented at PST 2014 [1].

to solve the current assignment) and reducing the set of messages displayed may be helpful. Central to our framework is the support of belief updates about whether a message is good or bad to show a user, in view of past observations of messages as well as the modeling of user utilities and rewards. The aim is to move beyond heuristic trust modeling, with formulae that may misjudge the point of potential benefit to users; a second value is to cope with dynamically changing, uncertain environments in a more principled way, one that supports effective decision making for the user.

II. Background

Our approach uses a neighborhood of advisors (rating messages) and models similarity, so relates to collaborative filtering recommender systems [5]. However, we are primarily aligned with approaches to multiagent trust modeling that adopt probabilistic reasoning. The beta reputation system (BRS) was foundational for its use of the beta probability distribution to amalgamate peer advice by performing repeated Bayesian updates after discounting peer feedback. Building on the BRS, Zhang and Cohen [6] develop a personalized trust model (PTM) to determine who to listen to among a network of agents. PTM proposes a novel combination of *private* (i.e., personal experience) and *public* (i.e., feedback from third parties) advice to arrive at trust estimates of agents.

A Learning Object Annotation Recommender (LOAR) system was proposed by Champaign *et al.* [7] in order to recommend commentary on learning objects to users in peer-based, online learning environments. LOAR built on concepts from PTM to develop a model for making recommendations; it uses a combination of similarity-adjusted ratings attached to messages and the message author's reputation to determine the predicted benefit of messages to users. As LOAR expands beyond traditional trust models to represent the trustworthiness of objects and to temper trust modeling by similarity of past behavior of advising peers, it served as the motivator and starting point of our work.

Regan *et al.* [2] develop a generalized model called BLADE for learning seller features in e-marketplaces. BLADE uses a Bayesian network to learn advisor evaluation functions (which goes beyond modeling advisor trustworthiness); once learned, advice can be amalgamated in a principled manner by performing inference in the Bayesian network. Beyond BLADE, we integrate a modeling of user utility in the form of reward functions; this then crucially enables our model to be of value for decision making (which was not in focus for BLADE).

In developing our approach, we needed to introduce competitors for our validation. We have chosen to focus on the *stochastic gradient descent* (SGD) approach of latent factor models (LFMs), which work by inferring some number of factors, f, from user ratings on items [8]. In particular, LFM-SGD works by choosing feature vectors $q_i \in \mathbb{R}^f$ (for items) and $p_u \in \mathbb{R}^f$ (for users); it then minimizes the total error between users' predicted ratings on messages (given by the dot product of q_i and p_u) and the ground truth ratings. Estimates are continuously improved by taking steps opposite the gradient, given the current values for q_i and p_u.

A more detailed discussion of these models follows.

A. Personalized Trust Model

Zhang and Cohen [6] suggest a PTM to determine whom to listen to among a network of buyers and sellers in the e-marketplace domain. In particular, they address whether a buyer, b, should purchase a product from a seller, s, based on a combination of *global* advice from other buyers (i.e., advisors, a), and b's own *local* past experiences with s.

The PTM global metric is further broken down to combine *private* and *public* trust estimates of advisors. The intuition is that b may have radically different expectations or preferences regarding s's product than a, and so b should have some notion of how much to trust a. To the extent that b relies on past common experiences to evaluate a's trustworthiness, b uses a *private* trust metric to incorporate a's recommendation. To the extent that b relies on a's similarity to the global rating of various sellers (i.e., how fair are a's ratings), b uses a *public* trust metric to incorporate a's recommendation.

The above-mentioned overview is made more concrete as follows. In PTM, ratings of a product are binary. The beta probability distribution is used to estimate the probability that an advisor will provide a fair rating to b. To estimate the private reputation of advisor a, PTM defines

$$R(a)_{\text{private}} = E[Pr_a(\text{fair rating})] = \frac{\alpha}{\alpha+\beta} \tag{1}$$

where α and β are the parameters to the beta distribution and can be defined as follows:

$$\begin{aligned}\alpha &= 1 + \sum_{s\in S}\sum_{t} \neg(\vec{r}_{b,s,t} \oplus \vec{r}_{a,s,t}) \cdot \vec{1}\\ \beta &= 1 + \sum_{s\in S}\sum_{t} (\vec{r}_{b,s,t} \oplus \vec{r}_{a,s,t}) \cdot \vec{1}.\end{aligned} \tag{2}$$

The notation in these equations has the following meaning.

1) S is the set of sellers rated in common by b and a.
2) $\vec{r}_{b,s,t}$ is a ratings vector of b's experiences with seller $s \in S$ in time window t.
3) $\vec{r}_{a,s,t}$ is a ratings vector for a such that each rating element $r_i \in \vec{r}_{a,s,t}$ corresponds to the most recent rating prior to $r_i \in \vec{r}_{b,s,t}$.
4) $\oplus$ denotes elementwise XOR and $\neg$ denotes elementwise logical NOT.

The overall trust level of advisor a is derived by combining $R(a)_{\text{private}}$ and $R(a)_{\text{public}}$ according to a weight function. Analagous to the derivation of global trust values for advisors, the overal trust value for a seller is determined via a weighted combination of global and local reputations.

B. Learning Object Annotation Recommendations

Champaign *et al.* [7] (see also [9]) develop a model for recommending commentary (annotations) on learning objects (texts or videos) to users in a peer-based, online learning environment, which we term LOAR.

In LOAR, when viewing learning objects, users are allowed to rate the attached annotations as valuable (1) or not (0). The "current" user is presented with peer commentary in a way that is customized according to each annotation's predicted learning benefit for that user. An annotation's predicted benefit is calculated using a combination of the annotator's reputation and explicit ratings the given annotation has recieved. An annotator's reputation is derived as follows.

1) An author q has created a set of annotations $A_q = \{a_1, \ldots, a_n\}$, each of which has an associated set of ratings $R_{a_i} = \{r_1, \ldots, r_{m_i}\}$ left by some number, m_i, of students who have experienced the annotation.
2) Compute a set of average ratings, $V = \{v_{a_1}, \ldots, v_{a_n}\}$, corresponding to each annotation using the associated rating set, i.e., $v_{a_i} = (1/m_i) \sum r_i$.
3) The annotator reputation, T_q, is the mean average rating, i.e., $(1/n) \sum v_{a_i}$.

In addition, in LOAR what constitutes a "local" annotation reputation depends on the number of votes it receives. In particular, votes for (vF_a) and against (vA_a), an annotation a are weighted according to the similarity between the current user and peer voter, which is calculated according to prior votes the pair have cast in common. The global and local annotation reputations are then combined to derive the predicted benefit for the current user (where those with highest predicted benefit are then shown). In particular, a tally-based option ends up focusing on the "local" reputation, by normalizing $(vF_a - vA_a/vF_a + vA_a)$. LOAR fundamentally determines which messages from peers to present, in order to improve the learning achieved by a user. In this respect, it should be suitable to be applied to the task of recommending messages in online participatory media networks, as well.

C. Credibility-Based Trust Modeling

CredTrust [10], [11] is another model to which we compare our new approach. CredTrust draws inspiration from PTM and TRAVOS [12] in the use of beta distributions, but continues to model the benefit of annotation objects themselves, as in LOAR, and introduces *credibility* in order to reason about the *right* annotations to show to users. CredTrust works as follows.

Determining whether an annotation or message will be well received (i.e., is beneficial) can be modeled as a Bernoulli process. That is, if M is the event that a message is well received, then we seek to determine $\psi = Pr(M)$. We represent this parameter as a random variable and use Bayes' theorem to update prior probability distributions over ψ; we represent the prior $Pr(\psi)$ with a beta distribution,[2] so that $\psi \sim Beta(\alpha^*, \beta^*)$.

Since we model the trustworthiness of messages (not annotators), the user begins with a uniform prior belief about the message. Accordingly, we update this belief by looking to the experiences of peers, as in LOAR.

When a user solicits feedback about a message, his peers report binary ratings, i.e., peers report parameters[3] α_p and β_p such that $\alpha_p + \beta_p = 1$, where $\alpha_p, \beta_p \in \{0, 1\}$. To combine peer reports, we model the similarity between users i and j using Hamming distance; normalizing, we obtain the Hamming ratio, denoted h_{ij} (the Hamming distance divided by the length of the binary strings, i.e., the number of common ratings). In particular, a Hamming distance of 0(1) means that the two strings are identical (opposite). Accordingly, a Hamming ratio of 0 suggests, we simply take a peer report as given; in contrast, if the Hamming ratio is 1, we swap the values reported for α_p and β_p. This captures the fact that nonsimilar peers can still deliver useful information; perfect negative correlations are just as informative as positive ones. We formalize this combination as follows:

$$\alpha^* = 1 + \sum_{p \in P} (1 - h_{\mathrm{sp}}) \cdot \alpha_p + h_{\mathrm{sp}} \cdot \beta_p$$
$$\beta^* = 1 + \sum_{p \in P} (1 - h_{\mathrm{sp}}) \cdot \beta_p + h_{\mathrm{sp}} \cdot \alpha_p \tag{3}$$

where P is the set of all peers. This combination capitalizes on the fact that the beta distribution is well defined for all real-valued parameters $\alpha, \beta > 0$. It allows us to extend peer reports to include expectations on message trust values: a report $r \in [0, 1]$ can be translated into parameters $(\alpha, \beta) = (r, 1 - r)$ so that a report of $r = 1$ corresponds to $\alpha = 1, \beta = 0$, a report of $r = 0.5$ corresponds to $\alpha = \beta = 0.5$, and $r = 0$ to $\alpha = 0, \beta = 1$. Thus, a user can solicit feedback from peers about an annotation even if those peers have yet to personally experience the annotation. This is useful if, for example, the current user has no or limited ratings in common with peers who have rated the annotation in focus. (That is, it might be more useful to use a report of expected benefit from a peer who is highly similar to the current user rather than use an explicit report from a peer with whom the user has no history and thus no notion of similarity).

CredTrust employs the scheme detailed in Algorithm 1 for combining peer advice. This algorithm computes a trust metric by discounting peer reports by their credibility (c_p), except when they report negatively on the given annotation. When this happens, the peer's negative rating is weighted using a combination of similarity and credibility. In particular, the role that similarity plays in blending the reported message rating is linearly reversed as the peer's credibility approaches 1 (i.e., perfect credibility).

D. Bayesian Network Trust Model (BLADE)

Our proposed solution draws inspiration from POMDP models. The following trust model is thus quite relevant: it also tries to learn the evaluation functions of advisors as a stand in for trust modeling and with a Bayesian perspective. As will be revealed in Section VI-D, the primary difference between our approach and BLADE is our consideration of user utilities and the inherent decision-making aspect of our POMDP classification model.

Regan *et al.* [2] developed a model called "BLADE." In BLADE, sellers are evaluated on the basis of several features, say k of them. Accordingly, each seller feature, denoted F_i^s, can be viewed as a random variable that follows a multinomial distribution; each F_i^s is, therefore, characterized by a corresponding probability vector $\vec{\Theta}_i^s$. The goal of a prospective buyer is to learn each $\vec{\Theta}_i^s$. In particular, $\vec{\Theta}_i^s \sim \mathrm{Dir}(\vec{\alpha})$, where the Dirichlet hyperparameters are interpreted as the

[2]When α^* is high and β^* is low (α^* is low, β^* is high), the user is very confident that the message should be seen (not seen).

[3]A report of 1 corresponds to $(\alpha_p, \beta_p) = (1, 0)$.

Algorithm 1: CredTrust

Input: The current user, u, his set of peers, P, their credibility scores, $c_p \in [0, 1]$, and their ratings for the annotation in focus, $r_p \in \{0, 1\}$

Output: Parameters α^* and β^* to a beta distribution describing trust in the current annotation

$\alpha^* = \beta^* = 1$
foreach $p \in P$ **do**
  $h_{up} \longleftarrow computeHammingRatio(u, p)$
  `// Discounted Bayesian update`
  if $r_p == 0$ **then**
    `// Adjust sim by credibility`
    $\alpha^* += h_{up}(1 - c_p)$
    $\beta^* += 1 - h_{up} \cdot (1 - c_p)$
  else
    `// Dampen update by credibility`
    $\alpha^* += c_p \cdot (1 - h_{up})$
    $\beta^* += c_p \cdot h_{up}$
  end
end

number of past observations of corresponding events, and buyers perform Bayesian updates on the basis of evidence they observe with each interaction. Then, given beliefs about $\vec{\Theta}^s = \{\vec{\Theta}_i^s\}_{i=1}^k$, buyers can reason about $Pr(\vec{F}^s = \vec{f}|\vec{\Theta}^s)$, and can, furthermore, make purchasing decisions that depend arbitrarily on utility derived from a given realization of seller features.

If buyers lack sufficient past interactions to draw upon when inferring seller feature distributions, they make use of reports from advisors who provide ratings based on their own past interactions with that seller. BLADE can handle equally well advisors who report truthfully or deceptively, by learning that advisor's private rating function (though faring better when advisors offer consistent or deterministic ratings, and less well when rating functions are stochastic).

Ultimately, BLADE can be seen as a generalization of several other trust models. In particular, BRS [13] and TRAVOS [12] can both be described by a particular realization of the BLADE Bayesian network wherein each seller has a single binary feature variable F^s that corresponds to the seller's overall reputation/trustworthiness (i.e., trustworthy or not) and that is fully determined by a single parameter Θ^s. By contrast, BLADE allows buyer utilities to depend arbitrarily on various seller features, and allows advisors to report ratings using whatever (independent) scale they deem most suitable.

III. Principled Bayesian Trust Modeling

Models such as BRS [13] and LOAR [7] encode particular functions when combining peer feedback. However, such functions might underperform if the mix of agents and ratings are inappropriate for the heuristic functions each model encodes. We developed CredTrust as another heuristic model to replace LOAR's use of similarity with one that also models credibility (to deflect cases of misguided, similar peers), motivated by the credibility model Bayesian Credibility Modeling [14]. We now move on to develop our BayesTrust model, which uses a more principled method for deciding which messages to recommend. We return to compare its performance against these models as competitors.

A. POMDP Classification Model

The process of imputing a benefit to each message by modeling the trustworthiness of advisors can be accomplished by designing a decision-making agent that can decide whether or not to recommend a message to a particular user. To begin with, consider the task of classifying messages as useful to show or not as mapped to a POMDP.

Formally, classifying messages would be done on the basis of an eight tuple, $(S, A, O, T, \Omega, R, \gamma, h)$, defined as follows.

1) S is the set of states (e.g., {good, bad}).
2) A is the set of actions (e.g., {recommend, reject, poll}).
3) O is the observation space $\{(f_1, \ldots, f_n)\}$ where each f_i is a feature that is correlated with the message state.
    a) For, e.g., $O = \{R, M, C\}$ in our simulations; each observation consists of a peer rating (R), that peer's similarity (M) $\in [0, 1]$ to the current user (two users are similar if they are likely to rate a message the same), and that peer's credibility (C).
4) $T = P(s'|s, a)$ is the probability function that describes transitions between states given the action.
5) $\Omega = P(o'|s', a)$: the probability of an observation given the state and action.
6) $R : S \times A \to \mathbb{R}$: a reward function that encodes the desirability (utility) of each state for a particular agent.
7) Discount factor $0 \leq \gamma < 1$.
8) Infinite horizon h.

This POMDP describes a process in which a decision-making agent participates to determine the desirability of a message. This process can be applied for each message in the network, individually, while still exploiting the overall POMDP structure. For example, when making recommendations for a particular user, a given message can be viewed as having an underlying state that is either "good" or "bad." This state is static, albeit itself unobservable. The agent can choose to either recommend the message, reject it, or poll for advice about it. This then yields an observable state, for example: an advisor rating of a message (1 for positive, 0 for negative); the similarity of the rater to the user (based on their previous rating behavior); and credibility (a value from 0 to 1, where highly credible peers have higher scores). This POMDP model is the ideal message recommending framework. We immediately make some simplifying assumptions, in order to move rapidly to an illustration and defense of this design.

For example, in our simulations, we use for T an identity function, since the underlying message state does not change and assume for R that there is a single global reward function (encoding relative utilities) where polling offers a higher reward than recommending or rejecting; the reward for polling is equivalent in all states (reflecting that the price of obtaining more information from advisors is the same, regardless of the agent's belief about the underlying message state). At this

TABLE I
OBSERVATION FUNCTION CONDITIONAL PROBABILITY TABLE $Pr(O \mid S, A = \text{POLL})$ (VALUES ARE IN [0, 1])

Observation			State	
R	M	C	Good	Bad
1	1	1	0.3	0.025
1	1	0	0.15	0.1
1	0	1	0.025	0.3
1	0	0	0.05	0.05
0	1	1	0.025	0.3
0	1	0	0.1	0.15
0	0	1	0.3	0.025
0	0	0	0.05	0.05

point, our model now resembles Naive Bayes. We return to discuss possible extensions as future work in Section VII. We refer to our model as BayesTrust.

The overall process is one where a series of observations about messages will yield insights into whether a brand new message should be shown to the user or not through prediction grounded on past experiences. This kind of policy evaluation is sketched in more detail through an example shown below. As will be seen, using a POMDP model has the advantage of providing a good tradeoff between exploitation and exploration, as part of the agent's decision making. The intelligent agent can decide how many ratings it should collect for each message in order to achieve good recommendations for its user, i.e., the agent knows when to stop gathering ratings from peers. This is in contrast with other approaches which may produce considerable delay for a user while waiting for a base of peer ratings to be amassed.

B. Example

We now illustrate how the BayesTrust model can be used to classify a message, m_1, for a given user u. For the sake of this example, we make the following assumptions about the BayesTrust model and about the messages.

1) The underlying state for message m_1 is "good," denoted G (m_1 should be recommended).
2) The observation function for the poll action is encoded in Table I: e.g., the first row shows the probability of seeing $\langle R, M, C\rangle = \langle 1, 1, 1\rangle$ given that the message is *Good* (*Bad*) is 0.3 (0.025) (values fabricated for the example).

To illustrate the framework for this particular example, we assume that the reward function is one where recommending good (bad) messages yields a reward of 1 (-1), while rejecting bad (good) messages yields a reward of 1 (-1). Polling for advice (whether the message is good or bad) is set to provide a higher reward than either recommending or rejecting. This is the strategy we use in our simulations, explicitly to determine whether our solution for processing advice from peers is working well.

Note that Table I's encoding of the conditional probability is intended to capture the following: "if the state is *Good* (*Bad*), what is the probability with which we could have seen the given $\langle R, M, C\rangle$?." The only observations that we have in the real world are the ratings provided by peers. This then requires us to figure out similarity (M) and credibility (C). We shed more light on how to do this in Section III-D in the following.

The agent starts with belief $b^{(m_1)}(s_0) = 0.5$, namely, that the message is equally likely to be good or bad. In other words, the agent has no *a priori* knowledge about the message, and therefore, begins with a uniform belief.[4] From this starting belief state, the agent makes observations about the underlying message by polling for peer advice. For the purposes of this example, we do not demonstrate how the polling decision might be made; we simply assume that the agent follows a policy to always poll if possible, i.e., if there are peers left who have not been polled (indeed, this strategy is optimal given that polling offers a higher reward than either recommending or rejecting a message). The agent can update its belief about the true message state through repeated updates

$$b_{t+1}(s_{t+1}) \propto \Pr(o_{t+1} \mid s_{t+1}, a_t) \sum_{s_t} \Pr(s_{t+1} \mid s_t, a_t) b_t(s_t). \tag{4}$$

For example, suppose that polling elicits feedback corresponding to a positive rating from a similar and credible advisor ($\langle R, M, C\rangle = \langle 1, 1, 1\rangle$). Then, we can update our belief about m_1 as follows:

$$b_1(G) \propto 0.3 \cdot (1.0 \cdot 0.5 + 0.0 \cdot 0.5) = 0.15$$

$$b_1(B) \propto 0.025 \cdot (0.0 \cdot 0.5 + 1.0 \cdot 0.5) = 0.0125.$$

After normalizing (so that the belief over all states sums to 1), we arrive at the belief $b_1 = [0.92 \;\; 0.08]$, namely, that the message is "good" with probability 0.92.

After belief updates are complete, decision-making agents have to advise their users. To do so, the agent would perform the following expected utility calculations (and choose the action that yields the highest expected utility):

$$\begin{aligned} EU_{\text{recommend}} &= b_{t_{\text{final}}}(G) \cdot R_{\text{recommend}}(G) \\ &\quad + b_{t_{\text{final}}}(B) \cdot R_{\text{recommend}}(B) \end{aligned} \tag{5}$$

$$EU_{\text{reject}} = b_{t_{\text{final}}}(G) \cdot R_{\text{reject}}(G) + b_{t_{\text{final}}}(B) \cdot R_{\text{reject}}(B). \tag{6}$$

C. Learning the Observation Function

We now examine the treatment of observations under this model. The observation kernel, Ω, encodes the function $Pr(o \mid s, a)$. We can denote this function with $|A| \times |S|$ multinomial distributions

$$\vec{\theta} = \begin{bmatrix} \theta_{s_1,a_1} & \cdots & \theta_{s_1,a_{|A|}} \\ \vdots & \ddots & \vdots \\ \theta_{s_{|S|},a_1} & \cdots & \theta_{s_{|S|},a_{|A|}} \end{bmatrix}$$

where each distribution contains $|O|$ parameters:[5] $Pr(o \mid s_i, a_j) \sim \theta_{s_i,a_j} = \text{Mult}(m_1, m_2, \ldots, m_{|O|})$ Since the

[4]A domain expert could augment the starting belief with prior knowledge. For example, past experience with a message author might provide evidence against a uniform prior belief.

[5]Note that $|O|$ is intended to mean "the size of the observation space."

BayesTrust model only produces observations when the agent chooses to poll for advice, we only need to encode a single multinomial for each state, namely, $\vec{\theta} = [\theta_{\text{good}} \;\; \theta_{\text{bad}}]$ when the state space is {good, bad}. If each of the parameters is distributed according to a Dirichlet distribution, $\theta_i \sim \text{Dir}(\alpha_1, \ldots, \alpha_{|O|})$ then we can refine each of these distributions as the agent traverses the POMDP and retrieves evidence from the user. That is, whenever a user provides feedback about a message as to the true state of the message, we perform a Bayesian update to determine the posterior distribution $Pr(\theta_i \mid d) = (Pr(d \mid \theta_i)Pr(\theta_i)/Pr(d))$. We can perform an update for each peer rating on a message when the current user informs the agent about the true state of the message (i.e., using supervised learning).

D. Belief Updates for Observations

Since the similarity and credibility feature variables, denoted M and C, are continuous in the interval [0, 1], it is nonobvious how such metrics should index the observation function. One technique that can be used to incorporate these metrics is to consider them as parameters to binomial distributions,[6] namely, θ_M and θ_C. Then, whenever the agent solicits advice, it simply samples from the similarity and credibility distributions, respectively, to obtain a discrete, binary observation. For example, if similarity were 0.7 and credibility were 0.9, then this technique would entail observing a similarity of 1 with probability 0.7 and a credibility 1 with probability 0.9. This results in a single, discrete observation. Observe that the mix of sampled discrete observations converge to the parameter from which those observations were sampled in the limit (i.e., as the number of advisors with similarity s and credibility c goes to infinity, a consequence of the central limit theorem), so that this sampling technique performs the right combination of belief updates in the limit.

Since ratings in real-world data sets are often sparse, another technique [7] would be to blend different belief states together by extending (4) as follows:

$$
\begin{aligned}
&b_{t+1}(s_{t+1}) \\
&\quad \propto \Pr(o_{t+1} \mid s_{t+1}, a_t) \sum_{s_t} \Pr(s_{t+1} \mid s_t, a_t) b_t(s_t) && (7)\\
&\quad = \Pr(o_{t+1} \mid s_{t+1}, a_t) \cdot \kappa && (8)\\
&\quad = \sum_{m_{t+1}} \sum_{c_{t+1}} \Pr(r_{t+1} \wedge m_{t+1} \wedge c_{t+1} \mid s_{t+1}, a_t) \cdot \kappa && (9)\\
&\quad = \sum_{m_{t+1}} \sum_{c_{t+1}} [\Pr(m_{t+1} \wedge c_{t+1} \mid s_{t+1}, a_t) \\
&\qquad \cdot \Pr(r_{t+1} \mid m_{t+1}, c_{t+1}, s_{t+1}, a_t) \cdot \kappa] && (10)\\
&\quad = \sum_{m_{t+1}} \sum_{c_{t+1}} \theta_{m_{t+1}} \cdot \theta_{c_{t+1}} \Pr(r_{t+1} \mid m_{t+1}, c_{t+1}, s_{t+1}, a_t) \cdot \kappa && (11)
\end{aligned}
$$

where (8) introduces a constant κ to represent the summation in (7) for notational simplicity; (9) encodes the fact that the observation function is a joint probability distribution over the elements of the observation tuple, and hence, we can marginalize out the unobserved variables M_{t+1} and C_{t+1}; (10) follows from the chain rule of probability, and; (11) encodes the heuristic in our mechanism, namely, that the similarity and credibility random variables are independent.

[6] This is done because M and C are continuous; we cannot simply index them, since the observation function is discrete.

[7] We used this technique for our experiments. Note that o_{t+1} in the following equation contains the discrete rating r and continuous belief for M and C.

TABLE II

SEPARATE, DISCRETE BELIEF UPDATE RESULTS

Observation			State	
R	M	C	$b_{t+1}(good)$	$b_{t+1}(bad)$
1	1	1	0.923	0.077
1	1	0	0.6	0.4
1	0	1	0.077	0.926
1	0	0	0.5	0.5

The above-mentioned calculations are intended to use beliefs about an agent's similarity and credibility in order to arrive at an actual observation to use. In summary, there are two methods which can be used. The first is to use the belief as a parameter to a binomial distribution and select from the distribution to get a concrete observation (in the form of a three tuple). The second is to marginalize out the unknown M and C and do a belief update directly (without ever having to construct an actual concrete three tuple). The former is a suggested option, while the latter (marginalizing out probabilties) is the way we adopted.

To make this more concrete, consider the following small example. Suppose that the observation function is once again given in Table I and that we receive a positive rating about a message ($r_{t+1} = 1$) from a peer for whom we have calculated a similarity score of 0.7 and a credibility score of 0.9. The belief updates for the different discrete assignments to observation variables are exhibited in Table II. To arrive at a final belief, we combine the beliefs according to the weights given by the similarity and credibility parameters (shown in parentheses)[8]

$$
\begin{aligned}
b_{t+1}(G) &= (0.7 \cdot 0.9) \cdot 0.923 + (0.7 \cdot 0.1) \cdot 0.6 \\
&\quad + (0.3 \cdot 0.9) \cdot 0.077 + (0.3 \cdot 0.1) \cdot 0.5 = 0.664 \\
b_{t+1}(B) &= (0.7 \cdot 0.9) \cdot 0.077 + (0.7 \cdot 0.1) \cdot 0.4 \\
&\quad + (0.3 \cdot 0.9) \cdot 0.923 + (0.3 \cdot 0.1) \cdot 0.5 = 0.336.
\end{aligned}
$$

E. Additional Insights for Social Media

Polling is turned ON in our simulations in order to observe healthy patterns of information exchange between peers. The reward function, in general, will not always prefer polling to take place. Rejecting messages can play out, in the end, as either filtering these messages out to reduce the overload to the user (not showing) or perhaps bundling less valuable messages together into some repository which the user can view at their leisure. For the future work, it is worth exploring further the ultimate preferences of users participating in social media where misleading information may abound, in order to

[8] These weights/parameters were chosen arbitrarily for the sake of this example.

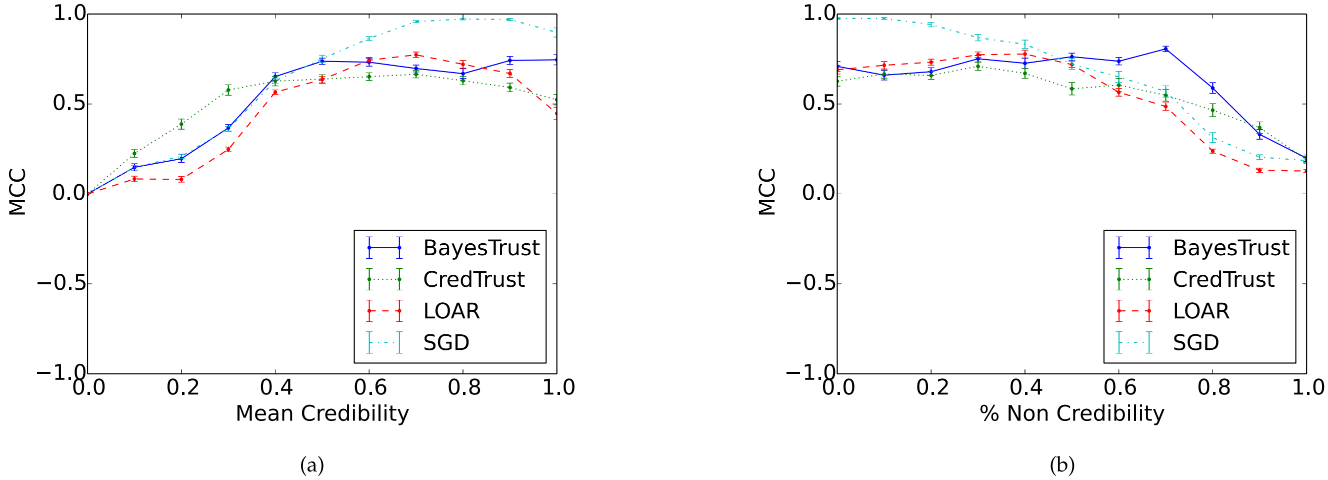

Fig. 1. MCC performance for various agent environments (higher MCC is better). (a) MCC versus mean credibility for randomly generated agent credibilities. (b) MCC versus percentage of noncredible advisors for dichotonomous agent credibilities.

specify more precisely the final proposed decision making of the intelligent agent assistant. Regardless, our intelligent agent approach helps to emphasize the messages of potential high benefit to the user. This trust modeling strategy is offered as an avenue for assisting with the navigation of today's current challenging social media environments.

IV. Simulations

A. Experimental Design

To evaluate BayesTrust, we conducted simulations and compared the performance of BayesTrust versus CredTrust, LOAR, and an SGD-latent factor model. We simulate an environment with 20 agents, each of which create messages and rate messages created by other the agents.

When authoring messages, credibility scores influence the "underlying message credibility" of the messages the agents create. For example, when an agent has a credibility of 0.5, approximately half of the messages it authors will be simulated to be beneficial and approximately half of the messages will have a "flaw" that detracts from agents' utilities if read.[9] In addition to credibility, agents are randomly assigned a type $\theta_a \in [0, 1]$. The agent's type is a parameter that influences similarity; agents of the same type tend to like the same messages. Moreover, messages have a type $\theta_m \in [0, 1]$ in order to appeal to different agents. In particular, we simulate agents rating messages more highly when those messages correspond to their type. However, agents' evaluation of the credibility of each message is modeled by flipping biased coins with probabilities proportional to their own credibilities; if an agent considers a message to be credible, and that message closely matches the agent's type, it will rate the message highly. The result is that less credible agents tend to rate messages they like highly, irregardless of any misinformation, or flaws contained within the message.

Each agent randomly produces between 1 and 10 messages and rates all of the messages produced by other agents. In order to evaluate the quality of the inferred benefits for messages, we randomly partition messages into a training and validation set. The training set is composed of approximately 70% of the messages and is used for the purpose of learning the observation function (in BayesTrust through repeated Bayesian updates), and determining peer similarities (for LOAR/CredTrust).

Each algorithm then runs against the testing set, classifying messages as good or bad. We compute the number of correctly classified messages (i.e., correctly recommended or correctly rejected) by comparing to the "correct" message classifications (based on the known benefits of each message to each agent) and report Matthew's correlation coefficient (MCC), which relates the true positive, false positive, false negative, and true negative rates.[10] In these simulations, we evaluate four algorithms for predicting benefit, which include BayesTrust, CredTrust, LOAR, and SGD.

We run four separate simulations as follows.

1) *Mean Credibility*: Agents are assigned randomly generated credibility scores at the outset.
2) *Dichotonomous Agents*: Agents are partitioned into one of two sets: low-credibility agents or high-credibility agents.
3) *Sparsity (No Bias)*: Agents flip a coin to determine whether to rate a message.
4) *Biased Sparsity*: Agents rate messages they like; otherwise, they flip a coin to decide whether to rate the message. This captures the bias "if you don't have something nice to say, don't say anything at all."

Beyond this, we use a simple greedy policy that polls for as much advice as possible,[11] and then takes the action that maximizes the expected utility for agents based on the belief state.

B. Results

The results of our simulations are depicted in Figs. 1 and 2. Overall, the new BayesTrust model does very well with MCC

[9]This treatment of authorship allows for credible (noncredible) agents to potentially create poor (good) messages, as in LOAR.

[10]The MCC metric is in the interval $[-1, 1]$, with higher numbers being better. In particular, 1 implies perfect classification accuracy and 0 means no better than random accuracy.

[11]This is equivalent to setting the reward for polling to be higher (1.5) than the reward for recommending/rejecting a message (1).

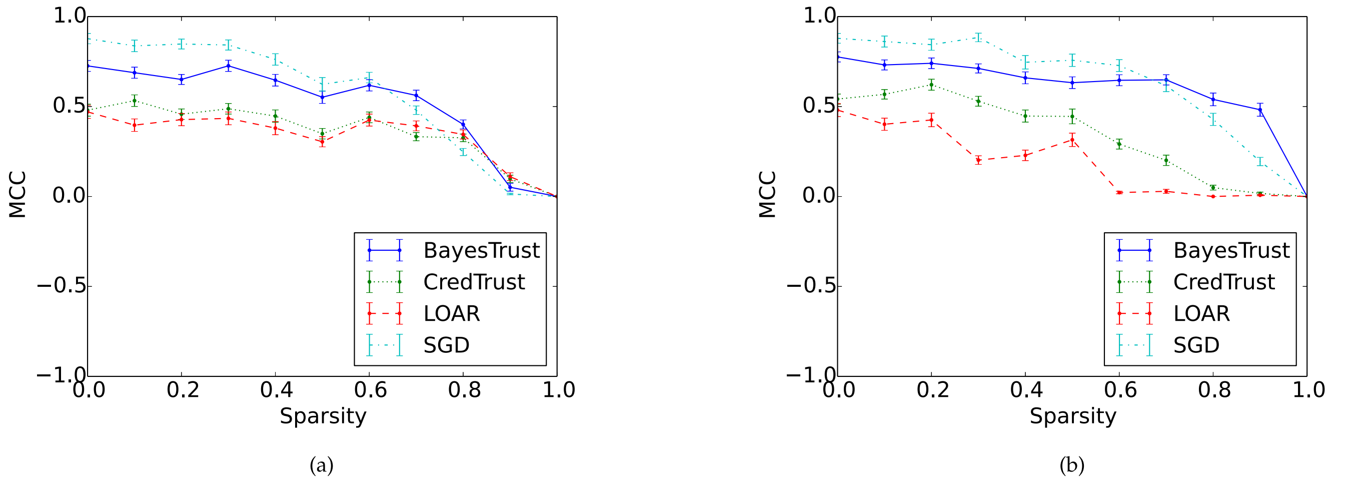

Fig. 2. MCC performance for sparse ratings environments (higher MCC is better). (a) MCC versus sparsity. Agents do not discriminate between messages they like and dislike. (b) MCC versus sparsity. Agents tend to only rate the messages they like.

and tends to outperform in all of the experiments, which makes sense given that it uses Bayesian updates to extract exactly the right amount of information from a given set of peer advice. The BayesTrust model only underperforms CredTrust in one instance: the lower spectrum in Fig. 1. However, in each of Figs. 1 and 2(a) and (b), BayesTrust is an upper bound on the performance of both heuristic models LOAR and CredTrust. We also compare the performance of BayesTrust versus a LFM using SGD (see Section II), which we dub "SGD" model,[12] by repeating the same four simulations. We chose parameters $f = 10$ and $\lambda = 3$ as the parameters that yielded the best results for the SGD model, after tuning by considering several values for regularization parameters and a number of latent features.

The results are depicted in the same figures as before and are slightly more mixed. In Fig. 1, BayesTrust and SGD trace a similar performance curve until credibility in the system reaches 0.5, after which the SGD algorithm significantly outperforms the BayesTrust model. Fig. 1 tells a slightly different story; in the dichotomous case, SGD outperforms until the percentage of noncredible peers reaches 50%, after which point the BayesTrust model outperforms. The sparsity simulations (Fig. 2) depict a smaller difference in performance between the two approaches, however, the story remains similar in that SGD outperforms in lower sparsity settings while the BayesTrust model modestly outperforms in the higher sparsity settings.

V. Real-World Experiments

We now turn our attention to validation using real-world data available from Reddit.com and Epinions.com. Reddit.com is a website that allows users to create and post messages for other users to see; other users in the Reddit community can then choose to "upvote" the message (i.e., rate it positively) or "downvote" the message (rate it negatively). Messages are then more or less likely to be seen by other users based on the mix of positive and negative feedback the messages have received. Epinions.com is a website that allows users to submit reviews about real-world products they have used. Other users can then read and rate the reviews (not the products themselves) on a five-point scale (1–5 stars), as well as provide concrete, written feedback. Data from these websites have been made available for research purposes.

The Reddit.com data contains a total of approximately 7.4 million user ratings by approximately 31 thousand different users on approximately 2.0 million messages.[13] Of these ratings, approximately 76% were positive. Table III provides a more detailed summary of the data set by reporting the means, standard deviations, and quartiles for various features. There were 31 553 users connected with this data set; the first row, "count," shows the number of users over which the remaining statistics were calculated. "User Ratings" in this table summarizes the number of ratings left by each user in the data set. "Common Ratings" summarizes the average number of ratings each user had in common with each other user. "Advisors" summarizes the average number of ratings that were left on messages rated by a given user.[14] Q_1, Q_2, Q_3 means that 25%, 50%, 75% of the users had the given number or lower. These summary statistics make clear that most users rate a small number of messages and that users tend to rate a small number of common ratings (perhaps because users tend to rate a small number of messages at all). The Epinions.com data contains a total of approximately 13.7 million user ratings by 132 thousand users on approximately 1.56 million different

[12]Strictly speaking, our SGD model implementation does not perform *stochastic* gradient descent, as the cost function is evaluated in "batch" fashion, however, this distinction is pedantic and we hereafter refer to the LFM as "SGD."

[13]The raw, anonymized Reddit data can be retrieved from http://www.reddit.com/r/redditdev/comments/dtg4j/want_to_help_reddit_build_a_recommender_a_public/

[14]To draw a distinction between "Common Ratings" and "Advisors," consider the case where a user A has rated only one message. It may be the case that all other users also rated that message, in which case the user has a large number of advisors (for that message). However, since A only rated one message, it can only have on average 1 common rating with each other user.

TABLE III
REDDIT.COM DATA SET DESCRIPTIVE STATISTICS

	User Ratings	Common Ratings	Advisors
count	31553	31553	31553
mean	234.66	2.3269	2315.99
std	446.64	4.1893	3285.83
min	1	0.0000	0
Q_1	3	0.0000	0
Q_2	20	1.1176	320
Q_3	194	2.5579	4058
max	2000	188.00	13322

TABLE IV
EPINIONS.COM DATA SET DESCRIPTIVE STATISTICS

	User Ratings	Common Ratings	Advisors
count	119,901	119,901	119,901
mean	112.571	2.27976	538.931
std	1052.70	5.92368	1794.91
min	1	0.0000	0
Q_1	1	1.0000	19
Q_2	3	1.0471	67
Q_3	12	1.5192	258
max	159,607	148.14	68,563

messages.[15] As earlier, Table IV illustrates a more detailed summary of the data set. As with the Reddit.com data, most users rate a small number of messages and tend to rate a small number of common ratings.

A. Experimental Design

For these experiments, we chose to compare BayesTrust against LOAR and BLADE [an apt comparator, since it adopts a principled statistical approach to combining peer advice (though without user utilities)]. In order to evaluate the effectiveness of these models against the Reddit.com and Epinions.com data sets, we use repeated random subsampling validation. In particular, we run 10 trials for each algorithm, and on each trial, we randomly partition the data into a training and testing sets consisting of 70% and 30% of the ratings, respectively. The training data are used for the purpose of determining the similarities between peers (for both LOAR and BayesTrust) and updating the BayesNet Dirichlet priors in BLADE. The testing set is withheld from the training phase and then used to evaluate the efficacy of each model. In particular, after training the models, each algorithm develops predictions for the ratings that users give to messages in the testing set. The actual ratings that users give to messages in the testing set is known, and therefore, considered to be the ground truth message benefit. For the Epinions data set, we substitute the LOAR similarity metric with a standard cosine similarity metric.

Since information about message authors is unavailable for the Reddit data set, we assume that each message is submitted by an anonymous author. Hence, in BLADE, users are unable to develop strong posterior beliefs about the message authors, since at most each user interacts only once with a given anonymous author (namely, either the user rates a message in the training set or not). In BLADE, advisor evaluation functions are learned on the basis of strong posterior beliefs. Projected into our context, this requires us to model beliefs about authors. In our setting, since there is at most a single data point with which to arrive at a posterior belief about a given anonymous author, the hope is that there is sufficient additional training data (i.e., additional common ratings) to improve estimates of advisor evaluation functions.

For the Epinions data set, we hybridized the BLADE model. If the message author was known, then BLADE was set up to learn the author's features; if the message author was unknown, BLADE proceeded as in the Reddit case and assumed that the message was authored anonymously. In this case, BLADE also made use of posterior belief about authors when reasoning about the probability of author features for messages in the testing set. (That is, if a message in the testing set was known to have been authored by a particular user, then that user's posterior feature distribution was used as a prior in addition to the explicit advisor ratings on the message for the purpose of classifying the message). For both data sets, we use the maximum likelihood to make recommendation decisions in the context of BLADE.

B. Results

For the Reddit experiment, we report the results of two separate BayesTrust instantiations, corresponding to two separate global reward functions. The first ("Bayes") rewards true positives slightly more than true negatives to reflect the positive skew of the data set. The second ("Bayes2") rewards true positives and true negatives equally. We begin by reporting the positive/negative hit rates: 1) LOAR achieves accuracies of 95% and 24%, respectively; 2) BLADE achieves 88% and 46%; and 3) BayesTrust achieves 81% and 56% (63% and 79%) for the first (second) instantiation. In both instantiations, we see that BayesTrust achieves a significantly higher true negative rate at the expense of a lower true positive rate versus both LOAR and BLADE. Accordingly, based on hit rates alone, it is difficult to distinguish which algorithms perform better.[16] The Matthew's correlation coefficient is another important metric to report since it balances the true positive and true negative rates to arrive at an overall accuracy that corrects in some sense for class skewness. For this data, the MCC is important to examine because a scheme that simply recommends all messages will achieve true positive rate of 100% and an overall accuracy of 76% (since 76% of the messages were rated positively in the original data set). Table V demonstrates how the first Bayes instantiation has a 23% higher MCC versus LOAR and a 6% lower MCC versus BLADE. On the other hand, the second instantiation (in parentheses) outperforms LOAR by 31%, and achieves a not significantly different MCC versus BLADE. Overall, BayesTrust performs very well against the Reddit data, despite

[15]The raw, anonymized Epinions data can be retrieved from http://www.trustlet.org/wiki/Extended_Epinions_data set, and was generously donated to the scientific community for research purposes as a result of the work of Paolo Massa [15].

[16]In many scenarios, correctly rejecting bad messages (i.e., a higher true negative rate) is a more important consideration than ensuring that all good messages are shown (i.e., a higher true positive rate).

TABLE V
REAL-WORLD EXPERIMENT RESULTS (MCC FOR REDDIT AND MAE FOR EPINIONS). BAYES2 IS SHOWN IN PARENTHESES

	Reddit.com MCC	Epinions.com MAE
LOAR	0.279	0.225
BLADE	0.365	0.300
BayesTrust	0.34 (0.365)	0.241

using a single, global reward function (versus tailoring a reward function for individual users).

For the Epinions experiment, we extend the two-state BayesTrust to be able to classify messages on a scale of 1 through 5 stars. (This is done very simply by extending the number of states to 5 and increasing the transition and observation functions accordingly). We only run a single instantiation of BayesTrust with a specific global reward function for all agents that accounts for the skewness of the Epinions data (e.g., by emphasizing a higher reward for classifying messages as "5 star"). As a result, we observe that BayesTrust achieves higher individual hit rates and is able to correctly classify messages with an overall accuracy of 82% versus LOAR (80%) and BLADE (78%). In addition, Table V depicts the mean absolute error (MAE) achieved by each algorithm (a measure of the average distance each classification is from the true message classification, where a low MAE is desirable). We can see that BayesTrust outperforms BLADE by approximately 20%, while it slightly underperforms LOAR by 7%. Again, BayesTrust performs very well against the Epinions data despite using a single, global reward function for all advisors. In all, the results of these experiments demonstrate that BayesTrust performs very well in real-world scenarios, outperforming LOAR in the Reddit experiment and BLADE in the Epinions one.

VI. DISCUSSION OF RESULTS AND RELATED WORK

We discuss in greater detail the results of our experiments, and include additional comparison with related work.

A. BayesTrust Versus CredTrust

Note that BayesTrust learns an appropriate observation function in order to amalgamate advice from peers in the "right" way given the environment and distribution of users (namely, the mix of user credibilities and preferences). It is understandable that this model outperforms CredTrust in nearly all cases. However, BayesTrust slightly underperforms CredTrust in the simulation depicted in Fig. 1 when the mean credibility of peers in the simulation falls below 0.4. We believe this to be the case because CredTrust discounts, and hence, filters out, the advice of less credible peers whose ratings tend to be more stochastic. In particular, in such cases, BayesTrust sometimes learns an incorrect observation function because the ratings of peers are less deterministic, and hence, less indicative of the true probabilities. Perhaps seeding the POMDP model with an appropriate prior for the observation functions could help to boost BayesTrust's performance in this particular simulation (and in such circumstances). The development of an appropriate prior is typically considered a domain-specific challenge; this is another area that could be considered as the future work.

B. BayesTrust Versus LOAR

Our approach outperforms LOAR in all of the simulations and real-world experiments we performed (see Figs. 1 and 2 and Table V). In some sense, BayesTrust can thus be viewed as an evolution of LOAR; it corrects for the biases that LOAR introduces in the form of heuristic updates. It is important to note that BayesTrust was able to make use of both similarity and credibility information, whereas LOAR only makes use of user similarities. However, BayesTrust can be easily made to handle any additional information (e.g., message-specific features such as message length or topic), as long as there is sufficient data from which to learn the observation function. In order to boost the performance of BayesTrust, it is, therefore, important to identify features that are strongly correlated with the underlying message states.

C. BayesTrust Versus SGD

The SGD model learns a number of hidden features of both messages and users. Essentially, the SGD model solves a nonlinear optimization problem to find a least-squares solution that situates each user and each message in a vector space for users and messages, respectively. Accordingly, we believe that part of the reason why the SGD model outperforms (see Fig. 2) is because it performs a number of computations that the BayesTrust model does not. For example, BayesTrust does not consider any message-specific features. Furthermore, in the simulations, BayesTrust only considered two user-specific features: user similarity and credibility (whereas SGD learned 10 latent features). We postulate that BayesTrust could be improved by considering message-specific features in addition to user features, and by searching for additional, informative, and correlated metrics. Moreover, our implementation of BayesTrust used a single, global reward function for all users. We postulate that using a user-specific reward function that caters to each user's individual preferences could help to boost the model's performance.

As a final point, consider that SGD requires finding the least-squares minimizer solution given an $M \times N$ ratings matrix (M being the number of messages and N being the number of users). Whenever a new message is created or a new user enters the system, their associated feature variables are unknown. Accordingly, a new solution needs to be computed every time the system changes (additional messages are created or new users enter); given that real-world participatory media networks like Reddit can potentially have many thousands of messages created and millions of new votes cast per day, recomputing a least-squares solution may be prohibitive, and thus the SGD approach may not be the most desirable solution for such dynamic contexts. In contrast, the BayesTrust approach can be done entirely online (and, thus, is perhaps more appropriate for online message recommendations). It would be interesting to perform a head-to-head experiment in the future to empirically validate this hypothesis and quantify the relative benefits of these two methods for different environments.

D. BayesTrust Versus BLADE

The BayesTrust model has many parallels to the Bayesian approach for inferring trust in e-marketplaces espoused by Reagan *et al.* [2] in their BLADE model. The most striking similarity is in the use of probability distributions to encode uncertainty about the underlying parameters that drive inference about trust in the network. Both approaches essentially learn the "right" way to treat peer advice. That is, BayesTrust performs Bayesian updates to learn the correct observation function for a given user. Likewise, BLADE performs Bayesian updates to learn the evaluation functions of peers in the system.

However, there are some differences too. One difference is that our model incorporates some notion of user utilities[17] for making decisions about messages by using a reward function. In particular, BayesTrust computes the expected utility for recommending a message versus rejecting it, and chooses the action that maximizes a user's overall expected utility. On the other hand, BLADE does not incorporate any model for reasoning about decisions or actions that agents should take.[18]

Accordingly, in order to make recommendations using the BLADE model (for our experiments), we used a maximum likelihood estimate based on BLADE's prediction of author features in order to classify messages. This strategy worked reasonably well in the case of Reddit.com (where BLADE matched the MCC performance of BayesTrust). However, BLADE underperformed in the Epinions experiment, where ratings sparsity is higher as a result of a larger number of classification classes (i.e., 1 through 5 stars); this hampered BLADE's ability to learn a good evaluation function, which resulted in lower overall performance in terms of MAE and individual hit rates. In contrast, BayesTrust's use of utilities gives rise to a natural means for making recommendation decisions, and was shown to perform very well in both simulation and against real-world data. Moreover, the model could potentially be extended to explicitly account for individual user preferences; individualized reward functions could serve to boost BayesTrust's performance. (One potential avenue that could be explored for developing appropriate user reward functions was suggested in the SGD discussion earlier; another approach could be to specifically ask users about their preferences and encode their responses using suitable reward functions).

A second difference is that BayesTrust is able to make use of a number of different features that could be correlated with the underlying message state. In particular, BayesTrust can be extended to make use of message-specific features, social network information, and so on, simply by extending the observation function. In fact, it is quite possible that BayesTrust could operate in environments where there are no ratings provided for messages. BLADE, on the other hand, is tuned to make use of the evaluation functions of advisors to learn seller features, which are represented by Dirichlet priors; it offers no way to take advantage of other features.

Finally, BLADE offers little insight into how one might prune the advisor space; if all peers in a network provide feedback about a given message (or if implicit feedback is used to impute ratings), BLADE offers no insight as to how to choose the ones that are most important to listen to. BayesTrust, on the other hand, can follow a policy that "short circuits" once it develops a strong enough belief about a message state, thereby potentially avoiding a large amount of unnecessary computation. For example, consider a case where there exist hundreds of thousands of imputed ratings for a given message. In such a scenario, BLADE will consider every single rating before coming to a decision. BayesTrust, on the other hand, might only need to view a small percentage of the ratings set, at which point it might obtain a belief about the message that is strong enough to make the expected utility for recommending/rejecting the message high enough that further polling becomes unnecessary.[19] Of course, this approach does not go as far as the work of Gorner *et al.* [17], which explicitly considers a careful selection of advisors to boost the overall accuracy of trust imputation.

E. Contribution to a Comprehensive Trust Module

To date, trust literature has largely focused on the evaluation of agent trustworthiness and has placed little emphasis on the other facets of a comprehensive trust module (CTM) as proposed by Sen [3]. Our BayesTrust model is, therefore, novel with respect to the CTM, as it explores the *usage* of trust information to select future actions by incorporating notions of user utilities that are associated with different outcomes (e.g., recommending a "good" message or rejecting a "bad" message). BayesTrust also provides a novel approach to the *evaluation* of agent trustworthiness through the belief update mechanism inherent to POMDPs. Taken together, our work offers insight into various aspects of Sen's CTM [3] and is, therefore, we feel, of particular interest and value to trust researchers.

F. Comparison With Other POMDP Trust Models

Irissappane *et al.* [18] presented a POMDP-based model for buyers to select quality sellers in an electronic marketplace. It allows buyers to optimally decide whether to ask from other buyers (called advisors) for more information about the quality of sellers or take the action of either choosing or not choosing a particular seller. With this model, buyers will know when to stop querying advisors. Later, they extend the idea to a hierarchical POMDP-based model [19] for the domain of wireless sensor networks where sensor nodes need to choose a suitable next-hop neighbor to route packets. To scale up their models, they further propose the mixture of POMDP experts technique [20], which exploits the inherent structure of trust-based domains by aggregating the solutions of smaller sub-POMDPs. These works are distinct from ours in the following

[17]Note that we use utilities interchangeably with the rewards from the reward function specified by BayesTrust.

[18]We consider [2] to be the standard reference to BLADE. We note, however, that Regan's thesis [16] includes a future work section that expands on possible decision making solutions. Interestingly, these suggest the use of POMDPs. We developed our BayesTrust model independently (and later discovered the overlap with [16]).

[19]This is, of course, less of a concern when considering explicit ratings alone, given the sparse nature of real-world ratings on data.

two aspects. To begin with, our model additionally considers a peer's similarity, thus provides the agent with a richer set of observations for reasoning about whether to follow the advice provided by the peers. Second, the model complexity of our BayesTrust model is much lower; since the underlying message state does not change, the state space S is smaller and more manageable.

G. Sources of Misleading Information in Online Social Media

In an invited lecture at WWW 2016 [21], Gruzd highlights a growing concern with online social media: the rise of artificial agents populating these networks with possibly misleading messages. While the framework outlined here is general enough to be applied in a number of different contexts, we feel that it may be just one part, but perhaps an important part, of the ongoing thread of research aimed at mitigating misinformation when messages may not have originated from human peers. The fact that our model aims to evaluate the credibility and similarity of the raters of the messages in these networks should be of particular value.

Current research examining trends in social media environments confirms the need for models such as ours, to detect messages with questionable value. Reference [22] shows that ratings are often recorded by users without proper forethought. Reference [23] reveals the importance of trust on persuasion and the value of trying to quantify its representation. Reference [24] suggests showing users only the tweets that are most important, which is similar to our suggestion of recommending the most reliable messages. Finally, several researchers have been examining the spread of rumors in social media (e.g., [25]), confirming that information quality is indeed an important concern, because certain posts are quite different from credible messages.

VII. Conclusion

In this paper, we presented BayesTrust, a model motivated by POMDPs to account for peer ratings on messages. BayesTrust learns an appropriate function for combining peer feedback rather than encoding specific heuristic combination functions. We demonstrated the merits of this approach in simulation versus LOAR and CredTrust (wherein BayesTrust outperformed), and also considered its performance against a class of LFMs (wherein BayesTrust stood its ground). Apart from simulation, we also validated our approach against real-world data. Taken together, our efforts make clear the value of BayesTrust.

The benefits of using our approach to help classify messages are manifold. First, this model intrinsically encodes the uncertainty in our knowledge of the world and in particular, the desirability of each message for each user. It allows us to express this uncertainty using probability distributions over our beliefs about the state of the world. Moreover, we can using reinforcement learning techniques to learn the correct distributions over advisory ratings, which allows us to correctly evaluate evidence that we receive from advisors in a principled manner. Furthermore, this formulation is tractable, since the space over states, actions, and observations is small and the process is Markovian. Ultimately, we believe our research provides a novel perspective on the problem of message recommendation through the application of Bayesian learning (which would culminate in using POMDPs) and through the exploration of user credibilities in order to help users sift through the raft of information available to them in online messaging environments. In this way, we provide a framework for decision making in the context of trust modeling, which ensures that users are exposed to the right information.

For the future, it would be interesting to explore the use of additional features, such as author-specific ones, to further improve the recommendations made. In particular, considering author-specific features and message-specific features could help to boost performance. Author history might also provide a basis for using nonuniform priors when performing belief updates. Considering implicit feedback, as in the work of Hu *et al.* [26] could also be of benefit, especially in domains where explicit feedback is sparse or unavailable (e.g., Twitter). Furthermore, learning a good reward function presents an interesting direction for future work; our experiments used an arbitrary, global reward function for all users. However, using standard regression and drawing on inverse reinforcement techniques [27], [28], or using preference elicitation as in [29], could yield more appropriate reward functions for individual users, resulting in better overall performance. We could as well explore slight variations to the existing reward function used in the experiments, to continue to assess the value of the framework, in the short term.[20]

There would also be value in exploring the selection of specific advisors, rather than soliciting advice from every advisor in the network. Our model could be extended to include actions for polling specific advisors; Monte Carlo Tree Search [30] could then be applied to optimize a policy that dictates which advisors to listen to over the more complex model. Research studies such as [31] and [32] then offer insight into how to transitively elicit advice when restricting neighbors to a subset of trusted advisors. Moreover, [33] use a subset of description logics combined with Dempster–Shafer theory to reason about uncertain information and delete and/or discount peer advice; such work could provide insight into determining which peers should be heeded. We could also be closer to full POMDPs by considering *recommend* or *reject* actions instead of simply polling, and experimenting with this less restrictive model.

Finally, future experiments will also consider other state-of-the-art models, like HABIT [34], which adopts a BLADE-like Bayesian model but with statistical inference leveraged by third-party relationships. HABIT also integrates user behavior modeling, considering user utilities when making decisions on the basis of evaluated trustworthiness. Our model remains separate given its explicit consideration of user utilities in the decision-making framework that we provide. That said, a more in-depth evaluation would be interesting to conduct.

[20]For example, we could explore a reward function where rejecting a good message and recommending a bad message both have a penalty of -5, with polling yielding a reward of 2 and then rejecting bad messages or recommending good messages providing a reward of 5. After several polling actions, recommending or rejecting can become more likely.

At the IJCAI 2016 panel on Human-Aware Artificial Intelligence, Eric Horvitz of Microsoft claimed that human–machine collaboration will dominate the influence of artificial intelligence in the future world. This paper has outlined a framework that would enable machines to assist humans in the processing of messages in social networks. This partnership will be increasingly important as these environments continue to offer an overabundance of information. We advocate trust modeling as a central component of the solution; performing this modeling well will, in turn, engender trust in the agents who are providing assistance. This has been borne out by experimental results suggesting that our algorithms are, in fact, delivering results that users do appreciate. Our final comment is that this trust modeling should, in turn, be of additional value in order to address scenarios where the messages have, in fact, been generated by agents and not by humans; our particular trust modeling will still be aiming to filter out what should not be shown to the humans in these networks. There is undoubtedly a challenging but vital path ahead with research on this topic, for the years to come.

Acknowledgment

The authors would like to thank A. Irissippane and P. F. Wang for their valuable feedback on an earlier version.

References

[1] N. Sardana and R. Cohen, "Validating trust models against realworld data sets," in *Proc. Privacy, Secur. Trust Conf. (PST)*, Jul. 2014, pp. 355–362.

[2] K. Regan, P. Poupart, and R. Cohen, "Bayesian reputation modeling in e-marketplaces sensitive to subjectivity, deception and change," in *Proc. 21st Nat. Conf. Artif. Intell. (AAAI)*, 2006, pp. 1206–1212.

[3] S. Sen, "A comprehensive approach to trust management," in *Proc. Int. Conf. Auto. Agents Multi-Agent Syst. (AAMAS)*, May 2013, pp. 797–800.

[4] V. F. Mancuso, J. C. Christensen, J. Cowley, V. Finomore, C. Gonzalez, and B. Knott, "Human factors in cyber warfare ii: emerging perspectives," in *Proc. Human Factors Ergonom. Soc. Annu. Meeting*, vol. 58, no. 1, 2014, pp. 415–418.

[5] C.-W. Hang and M. P. Singh, "Generalized framework for personalized recommendations in agent networks," *Auto. Agents Multi-Agent Syst.*, vol. 25, no. 3, pp. 475–498, Nov. 2012.

[6] J. Zhang and R. Cohen, "Evaluating the trustworthiness of advice about seller agents in e-marketplaces: A personalized approach," *Electron. Commerce Res. Appl.*, vol. 7, no. 3, pp. 330–340, 2008.

[7] J. Champaign, J. Zhang, and R. Cohen, "Coping with poor advice from peers in peer-based intelligent tutoring: The case of avoiding bad annotations of learning objects," in *Proc. User Modeling Adaptation Personalization (UMAP)*, Jul. 2011, pp. 38–49.

[8] Y. Koren, R. M. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[9] J. Champaign, "Peer-based intelligent tutoring systems: A corpus-oriented approach," Ph.D. dissertation, David R. Cheriton School Comput. Sci., Univ. Waterloo, Waterloo, ON, Canada, 2012.

[10] N. Sardana and R. Cohen, "Credibility-based trust in social networks," in *Proc. Int. Conf. Auto. Agents Multi-Agent Syst. (AAMAS)*, 2014, pp. 1423–1424.

[11] N. Sardana and R. Cohen, "Demonstrating the value of credibility modeling for trust-based approaches to online message recommendation," in *Proc. 12th Annu. Int. Conf. Privacy, Secur., Trust (PST)*, Jul. 2014, pp. 363–370.

[12] W. T. L. Teacy, J. Patel, N. R. Jennings, and M. Luck, "TRAVOS: Trust and reputation in the context of inaccurate information sources," *Auto. Agents Multi-Agent Syst.*, vol. 12, no. 2, pp. 183–198, Mar. 2006.

[13] A. Jøsang and R. Ismail, "The beta reputation system," in *Proc. 15th Bled Electron. Commerce Conf.*, 2002, pp. 324–337.

[14] A. Seth, J. Zhang, and R. Cohen, "Bayesian credibility modeling for personalized recommendation in participatory media," in *Proc. Int. Conf. User Modeling, Adaptation Personalization (UMAP)*, 2010, pp. 279–290.

[15] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proc. ACM Conf. Recommender Syst.*, Oct. 2007, pp. 17–24.

[16] K. Regan, "A social reputation model for electronic marketplaces sensitive to subjectivity, deception and change," M.S. thesis, David R. Cheriton School Comput. Sci., Univ. Waterloo, Waterloo, ON, Canada, 2006.

[17] J. Gorner, J. Zhang, and R. Cohen, "Improving trust modelling through the limit of advisor network size and use of referrals," *Electron. Commerce Res. Appl.*, vol. 12, no. 2, pp. 112–123, Apr. 2013.

[18] A. A. Irissappane, F. A. Oliehoek, and J. Zhang, "A pomdp based approach to optimally select sellers in electronic marketplace," in *Proc. 13th Int. Conf. Auto. Agents Multiagent Syst. (AAMAS)*, 2014, pp. 1329–1336.

[19] A. A. Irissappane, J. Zhang, F. Oliehoek, and P. S. Dutta, "Secure routing in wireless sensor networks via POMDPs," in *Proc. 25th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2015, pp. 2617–2623.

[20] A. A. Irissappane, F. A. Oliehoek, and J. Zhang, "A scalable framework to choose sellers in e-marketplaces using POMDPs," in *Proc. 30th AAAI Conf. Artif. Intell. (AAAI)*, 2016, pp. 158–164.

[21] A. Gruzd, "Who are we modelling: Bots of humans?" in *Proc. 25th Int. Conf. Companion World Wide Web (WWW)*, 2016, p. 551.

[22] M. Glenski, C. Pennycuff, and T. Weninger, "Consumers and curators: Browsing and voting patterns on reddit," *IEEE Trans. Comput. Social Syst.*, vol. 4, no. 4, pp. 196–206, Dec. 2017.

[23] E. Shmueli, V. K. Singh, B. Lepri, and A. Pentland, "Sensing, understanding, and shaping social behavior," *IEEE Trans. Comput. Social Syst.*, vol. 1, no. 1, pp. 22–34, Mar. 2014.

[24] S. Khater, D. Gračanin, and H. G. Elmongui, "Personalized recommendation for online social networks information: Personal preferences and location-based community trends," *IEEE Trans. Comput. Social Syst.*, vol. 4, no. 3, pp. 104–120, Sep. 2017.

[25] Y. Liu and S. Xu, "Detecting rumors through modeling information propagation networks in a social media environment," *IEEE Trans. Comput. Social Syst.*, vol. 3, no. 2, pp. 46–62, Jun. 2016.

[26] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 263–272.

[27] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2000, pp. 663–670.

[28] J. Choi and K.-E. Kim, "Inverse reinforcement learning in partially observable environments," in *Proc. J. Mach. Learn. Res.*, 2011, pp. 691–730.

[29] C. Boutilier, "A pomdp formulation of preference elicitation problems," in *Proc. AAAI/IAAI*, Jul. 2002, pp. 239–246.

[30] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2010, pp. 2164–2172.

[31] C. Burnett and N. Oren, "Position-based trust update in delegation chains," in *Proc. Int. Workshop Trust Agent Soc. (TRUST)*, 2013, pp. 51–62.

[32] C.-W. Hang, Z. Zhang, and M. P. Singh, "Generalized trust propagation with limited evidence," *IEEE Comput.*, vol. 46, no. 3, pp. 78–85, Mar. 2012.

[33] M. Sensoy *et al.*, "Reasoning about uncertain information and conflict resolution through trust revision," in *Proc. 12th Int. Conf. Auto. Agents Multiagent Syst.*, 2013, pp. 837–844.

[34] W. T. L. Teacy, M. Luck, A. Rogers, and N. R. Jennings, "An efficient and versatile approach to trust and reputation using hierarchical Bayesian modelling," *Artif. Intell.*, vol. 192, pp. 149–185, Dec. 2012.

Noel Sardana received the master's degree from the Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada, in 2014, under the supervision of Prof. R. Cohen. He completed this research when he was at the University of Waterloo.

He was a Software Engineer with Amazon Web Services, Inc, Seattle, WA, USA. He is currently a Software Engineer with Facebook, Inc., Menlo Park, CA, USA. His current research interests include trust modeling, social implications of computing, machine learning, distributed systems, and functional programing.

Mr. Sardana was a recipient of the Wilfrid Laurier University School of Business and Economics Alumni Gold Medal in 2012.

Robin Cohen is currently a Professor with the David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada, where she has been a Faculty Member since 1984. Her current research interests include trust modeling in multiagent systems and improving the experience of users in online social networks.

Ms. Cohen has been an Active Member of the AAMAS and Artificial Intelligence Communities since their inception. She became a Senior Member of AAAI in 2014. She is also a Founding Member of the User Modeling community. She was a recipient of the Canadian Artificial Intelligence Association's Lifetime Achievement Award in 2018.

Jie Zhang received the Ph.D. degree from the Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada, in 2009.

He is currently an Associate Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. He is also an Associate of the Singapore Institute of Manufacturing Technology, Singapore. He has authored or co-authored several papers of top journals and conferences.

Dr. Zhang is an Active Member of research communities. He was a recipient of the Alumni Gold Medal at the 2009 Convocation Ceremony, the Gold Medal is awarded once a year to honor the top Ph.D. graduate from the University of Waterloo, several best paper awards, and the Prestigious NSERC Alexander Graham Bell Canada Graduate Scholarship rewarded for top Ph.D. students across Canada.

Shuo Chen received the bachelor's degree from the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, China, in 2011, and the master's degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, in 2014. He is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, Nanyang Technological University, Singapore.

His current research interests include the planning and decision making in multiagent system, trust modeling, and security in the Internet of Things.