A Priori Trust Inference with Context-Aware Stereotypical Deep Learning

Peng Zhou[‡], Xiaojing Gu[†1], Jie Zhang[§] and Minrui Fei[‡♯]
School of Mechatronic Engineering and Automation, Shanghai University[‡]
School of Information Science and Engineering, East China University of Science and Technology[†]
School of Computer Engineering, Nanyang Technological University[§]
The Shanghai Key Laboratory of Power Station Automation Technology, Shanghai University[♯]
pzhou@shu.edu.cn xjing.gu@ecust.edu.cn ZhangJ@ntu.edu.sg mrfei@staff.shu.edu.cn

Abstract

In multi-agent systems, stereotypical trust models are widely used to bootstrap a priori trust in case historical trust evidences are unavailable. These models can work well if and only if malicious agents share some common features (i.e., stereotypes) in their profiles and these features can be detected. However, this condition may not hold for all the adversarial scenarios. Smart attackers can show different trustworthiness to different agents and services (i.e., launching context-correlated attacks). In this paper, we propose CAST, a novel Context-Aware Stereotypical Trust deep learning framework. CAST coins a comprehensive set of seven context-aware stereotypes, each of which can capture an unique type of context-correlated attacks, as well as a deep learning architecture to keep the trust stereotyping robust (i.e., resist training errors). The basic idea is to construct a multi-layer perceptive structure to learn the latent correlations between context-aware stereotypes and the trustworthiness, and thus can estimate the new trust by taking into account the context information. We have evaluated CAST using a rich set of experiments over a simulated multi-agent system. The experimental results have successfully confirmed that, our CAST can achieve approximately tens of times higher trust inference accuracy in average than the competing algorithms in the presence of context-correlated attacks, and more importantly can maintain a much better trust inference robustness against stereotyping errors.

Keywords: Multi-agent systems, stereotypical trust model, deep learning.

1. Introduction

In multi-agent systems, trust is a vital element and can be used to ensure the security and quality of service. Each agent (i.e., *a trustor*) can assign a trust value to another (i.e., *trustees*) based on the trustor's direct experiences on the trustee in the past (i.e. direct historical trust evidences). The trust can be later used to evaluate the possibility whether the trustee will perform particular services as expected by the trustor in the future [1]. That is, trust is a subjective concept and is with respect to a trustor agent, a trustee agent and a particular service the trustor expects the trustee to perform. All of the three constitute a *trust context*. In general, the trust is established through historical evidences in one context and can only be reused to evaluate future behaviors within the same context [1, 2, 3, 4, 5].

When direct historical evidences are not available in a context, bootstrapping a priori trust for such context is quite challenging. In open and dynamic multi-agent systems, new agents and services may join at any time [5]. These newcomers are necessarily suffering from evidence unavailability. To address this issue, a typical solution is to respect trust recommendations from other trustor agents (i.e., *advisors*) [6], which is known as a *reputation* method. However, this method cannot work well in case the trustee is a newcomer (i.e., all the agents do not have experiences on the newcomer and thus cannot do any recommendations). To bridge this gap, stereotypical trust modelling has been proposed [7, 5]. The idea is to "borrows" the trustworthiness from similar trustee agents through their visible features (i.e., *stereotypes*), such as the agents' geographical location/timezone [7] or their organization and performance indices [5].

Although the stereotypical trust modelling can bootstrap a priori trust for the contexts with new trustees, they cannot cover all the initial cases (e.g., both the trustor and the trustee are newcomers or the service is new), and even worse cannot make an accurate and robust a priori trust inference in the presence of sophisticated adversarial scenarios. In particular, most of the existing stereotypical algorithms can only model stereotypes for the trustees [7, 5, 8, 9, 10]. They have not considered which services the trustees offer and which trustors the trustees serve, and thus fail to capture the malicious patterns when the attackers use different trustworthiness to serve different trustors with different services (i.e., performing context-

[1]Corresponding at (+86) 13916123472.

correlated attacks). Moreover, existing stereotypical trust models have not well addressed the robustness problem (i.e., trust accuracy against stereotyping errors), hence being very difficult to be applied to practical scenarios (in practice, benign agents may set incorrect features due to ignorance or mistakes; adversarial agents may display false features for some malicious purpose, or attack the trust management system itself and thus mislead the system giving high level of trustworthiness to malicious agents [11]).

1.1. Problem statement

We list the main research problems we have to conquer in this paper as follows.

1. How to avoid context-correlated attacks when inferring trust using stereotypical models?
2. How to maintain the robustness of stereotypical trust modelling in the presence of stereotype errors?

1.2. The contributions

To solve the two aforementioned problems, we propose CAST, a new Context-Aware Stereotypical Trust deep learning architecture to extend existing solutions to a complete framework. In contrast to existing stereotypical trust models which only consider stereotypes (visible features) for trustees, we consider stereotypes from context's perspective. That is, given a target context, we generate *context-aware stereotypes* from other contexts where either the trustor or the trustee or the service or a group of them is different. We have seven this kind of stereotypes in total and list them in Figure 1, where each circle (i.e., $n_{i=1,2,3,4}$) represents an agent, each rectangle $s_{k=1,2}$ is a service and each arrow line indicates a trust value. As can be seen, if n_2 attempts to estimate a priori trust for n_1 with respect to s_1, the n_2 can consult the posteriori trust n_2 has already modelled for a different service s_2 (case ①) or agent n_3 (case ②) or a different pair (case ④), or the posteriori trust another agent $n_{3/4}$ has modelled for the same target (case ③) or a different service (case ⑤) or agent (case ⑥) or the pair (case ⑦), through the visible features from different agents and services.

CAST is novel in two aspects. First, CAST can be considered as a generalization and extension of previous research, and can cover a comprehensive set of initial cases. For example, existing trustee stereotype is just the case ②, while the reputation method could be covered by the case ③ or ④ (in case the stereotypical information is ignored). Second, each of the seven stereotypes in our CAST can capture an unique kind of malicious patterns that are correlated to the context. For example, if a malicious trustee treats different trustors differently (i.e., discrimination attack), such patterns can be recognized by the context-aware stereotype ③ in Figure 1. But if the malicious one provides different services with different qualities (i.e., service selection attack), the stereotype ① plays the key role. Moreover, if an attacker combines the two, we should rely on ⑤ for detection.

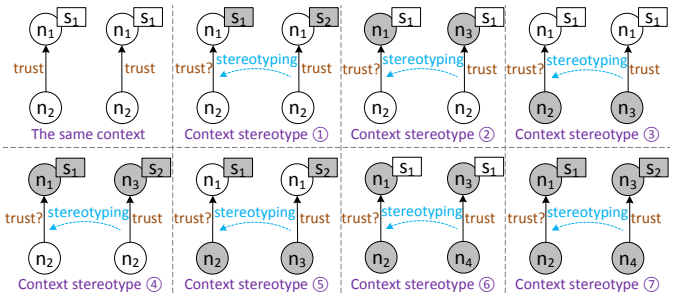


Figure 1: Seven kinds of context-aware stereotypes. In each cubic, the left is the evidence-sparse context (target context) while the right is the evidence-dense one (source context). n_1, n_2, n_3 and n_4 are four different agents while s_1 and s_2 are two different services. The arrow line represents that one agent models the trustworthiness of another agent with respect to a service.

In the design of CAST, we apply a deep learning architecture [12] for trust stereotyping. The deep model deploys a multi-layer perceptive structure [13, 14] to mimic a human's perception and decision making process [15]. It can be trained by learning context-aware stereotypes from the contexts with enough direct historical evidences (i.e., evidence-dense contexts or labelled samples), and later used to estimate a priori trust for the contexts without enough direct historical evidences (i.e., evidence-sparse contexts or test cases). We choose the deep model here since such model can maintain sufficient robustness against training errors. Even if some evidence-dense contexts involve inaccurate or incorrect stereotype information, the deep model can still work well for trust stereotyping.

We summarize the key research contributions we have made in this paper as follows.

1. We have taken into account a comprehensive set of seven context-aware stereotypes for stereotypical trust modelling, and hence being able to avoid context-correlated attacks.
2. We have applied a deep learning architecture to maintain robustness for stereotypical trust modelling.

1.3. Paper structure

The remainder of this paper is organized as follows. We first review related works and point out the novelty of our work in Section 2. We then design CAST with context-aware stereotypes in Section 3 and the deep model in Section 4. After evaluating CAST in Section 5, we conclude the paper in Section 6.

2. Background and related work

In this section, we review state-of-the-art a priori trust inference methods in the literature. We discuss the open

problems of existing methods and thereby motivate the new design.

In general, a priori trust inference is required when direct historical trust evidences are lacking (here, we do not consider the trust from social relationships [16, 17] and the pre-trusted third parties [18]). One of the most popular solutions is the reputation methods [19, 20, 21, 22, 23], by which the trustor agent will respect the recommendations from some trustworthy advisors to build up initial trust for the same trustee agent with the same service. Although these reputation methods may have some minor differences in their design (e.g., the works [19, 21] may consider the trust is transitive and some others [22] may not), they share the basic idea: trustor agents can infer the trustworthiness for a given target by consulting some trustworthy advisors who have direct experiences to the target.

Although the reputation methods have dominated the research domain for more than two decades, they all cannot bootstrap the trust when the trustees are newcomers. This issue has not been resolved until stereotypical trust models [7, 5, 24, 8, 9, 10] appeared. On the one hand, StereoTrust [7] is perhaps the first work that introduced stereotypical models for a priori trust inference (case ② in Figure 1). It proposed a grouping method to build a membership function for trustees' visible features and then used this function to calculate the trustworthiness for the new trustees. Fang et al. [8] generalized StereoTrust with fuzzy theory. On the other hand, Burnett et al. [5, 9] applied a classification and regression tree (e.g., M5 tree) to learn from existing trustees through their visible features and then used the tree to infer a priori trust for the new ones (case ② in Figure 1). Although Burnett et al. have realized that trustee stereotypical models cannot work if the trustors are newcomers, and have designed a stereotypical reputation method (partially case ⑥ in Figure 1) to address this issue, they have still overlooked that the trustors and the advisors may be treated differently by the trustees even if the advisors are trustworthy enough. Moreover, the work [24] has discussed how to discover more stereotypical sources to model trust in multi-agent systems and pointed out the importance of stereotypes (i.e., visible features) from the trustors (case ③ in Figure 1). However, that work [24] has not designed any practical stereotyping algorithms for trustors. In the most recent research, Sensoy et al. [10] proposed a graph extraction algorithm to mine a rich set of available features for stereotypical trust modelling. But unfortunately, their mining method is still restricted to trustees' profiles (case ② in Figure 1).

In contrast to state-of-the-art stereotypical trust models [7, 5, 24, 8, 9, 10], our CAST goes beyond in two aspects. One is the use of context-aware stereotypes, and the other is learning the context-aware stereotypes using a deep model. More precisely, CAST extends the available sources of stereotypes from trustees to trust contexts which consist of the trustors, the trustees and the services. With this extension, our CAST can naturally generalize existing reputation methods and stereotypical models to a complete framework and thereby is able to detect context-correlated attacks (e.g., discrimination attack, service selection attack or both). CAST also applies a deep architecture [14] for stereotypical trust modelling. As one of the most promising artificial intelligent techniques, the deep model has been proved effective in a broad set of application fields, including but not limited to image and video processing [25], speech recognition [26] and natural language processing [27] and so on. Following these successful experiences, we introduce the deep model, specifically the deep belief network [12], to solve the stereotypical trust inference problem for the first time.

The deep model is a good approximation of human's neocortex [13, 14], while the stereotypical trust inference is a typical human perception and decision making process [28, 15]. The use of a deep model to solve the stereotypical trust inference problem is nature and meaningful. Moreover and more importantly, our deep model can outperform the grouping method [7] and the classification tree [5, 9], because it can abstract the stereotypes in a perceptive manner layer by layer and thus can better represent the latent joint distribution between the stereotypes and trustworthiness, even if the training set (i.e., evidence-dense contexts) contain incorrect and/or inaccurate stereotypical trustworthiness samples. In another word, the deep model can achieve a more robust trust stereotyping than existing algorithms. Note that, we have confirmed this finding through carefully designed experiments in Section 5.4.

3. Context-aware stereotyping

In this paper, we model a multi-agent system as a double bipartite graph $G = (N, N, S, N \times N \times S)$, where N is the set of agents in the system and S is the set of services. In this graph, each edge $(n_i, n_j, s_k) \in N \times N \times S$ denotes an unique context for trust modelling. It can be interpreted as an agent $n_i \in N$ models the trustworthiness of another agent $n_j \in N$ with respect to a service $s_k \in S$.

3.1. Trust Model

Given a context (n_i, n_j, s_k), we model the trust associated to this context using subjective logic [2], which enables agents to express the trustworthiness as degrees of a belief $t^i_{j:k}$, a disbelief $d^i_{j:k}$, an uncertainty $u^i_{j:k}$ and a base rate $a^i_{j:k}$ (a priori degree of belief) about the context. We write it as $(t^i_{j:k}, d^i_{j:k}, u^i_{j:k}, a^i_{j:k})$, where $t^i_{j:k} + d^i_{j:k} + u^i_{j:k} = 1$ and $a^i_{j:k} \in [0, 1]$. We define a direct trust evidence regarding to a context (n_i, n_j, s_k) as a binary event of whether the agent n_i has successfully run the service s_k in another agent n_j as expected. If this is true, the evidence is positive. Otherwise, it is negative. A sequence of independent binary events can be modelled as a Bernoulli experiment, and the posterior distribution of this experiment can be approximated as a beta distribution $Beta(\alpha, \beta)$ [2]. Let $r^i_{j:k}$

be the number of positive historical evidences observed in the context (n_i, n_j, s_k) and $s^i_{j:k}$ be the number of negative ones. We have $\alpha = r^i_{j:k}+1$, $\beta = s^i_{j:k}+1$ and thus can calculate the belief (i.e., posterior trustworthiness), the disbelief and the uncertainty using Eqn. (1).

$$t^i_{j:k} = \frac{r^i_{j:k}}{r^i_{j:k}+s^i_{j:k}+2}, d^i_{j:k} = \frac{s^i_{j:k}}{r^i_{j:k}+s^i_{j:k}+2}, u^i_{j:k} = \frac{2}{r^i_{j:k}+s^i_{j:k}+2}. \quad (1)$$

In Eqn. (1), if $r^i_{j:k}+s^i_{j:k}$ is small, the uncertainty $u^i_{j:k}$ could be quite large and the resulting $t^i_{j:k}$ (and $d^i_{j:k}$) are not with enough confidences. In this case, the trustworthiness can be corrected using the base rate $a^i_{j:k}$, which is a priori trust value purely inferred without any historical trust evidences. The correction function is $\widehat{t}^i_{j:k} = t^i_{j:k} + a^i_{j:k} \cdot u^i_{j:k}$ [2]. When $r^i_{j:k} + s^i_{j:k}$ is large enough, the trustworthiness can be mainly determined by $t^i_{j:k}$. But if $r^i_{j:k} + s^i_{j:k}$ is too small, $a^i_{j:k}$ becomes the dominant factor. In this paper, our ultimate goal is to infer $a^i_{j:k}$ for the context where $r^i_{j:k} + s^i_{j:k} \leq th$ by learning the available $t^i_{j:k}$ values from the context where $r^i_{j:k} + s^i_{j:k} > th$. In the following of this paper, we will regard the contexts with $r^i_{j:k} + s^i_{j:k} \leq th$ as *evidence-sparse contexts* while the ones with $r^i_{j:k}+s^i_{j:k} > th$ as *evidence-dense contexts*. The th parameter is a threshold we should choose to classify whether a context is evidence dense or sparse. Normally and simply, we choose $th = 0$ for this paper.

3.2. Context-Aware Stereotypes

It is known that we cannot calculate the trustworthiness for evidence-sparse contexts using the Equation (1) directly. Instead, we should infer $t^i_{j:k} \simeq a^i_{j:k}$ with the help of other evidence-dense contexts. The basic idea is to learn a supervised model from evidence-dense contexts to capture the latent correlations between the trustworthiness and visible features, and then use this model to infer a priori trust for the evidence-sparse contexts. Since visible features play the role as a mould to convey trust information, we also call them *stereotypes*. In this paper, our innovative design is to consider stereotypes with respect to the trust contexts. We let Fn and Fs be the set of visible features (i.e., stereotypes) for the agents and services, respectively.

Given an evidence-sparse context (n_i, n_j, s_k), we can group available evidence-dense contexts into seven categories by considering that the n_i (trustor) or n_j (trustee) or s_k (service) or multiple of them are different. We define the set of available evidence-dense/sparse contexts as C_d/C_s, respectively:

$$\begin{array}{l} C_d \triangleq \{(n_i, n_j, s_k) \in N \times N \times S, r^i_{j:k} + s^i_{j:k} > th\} \\ C_s \triangleq \{(n_i, n_j, s_k) \in N \times N \times S, r^i_{j:k} + s^i_{j:k} \leq th\} \end{array} \quad (2)$$

Obviously, $C_d \cap C_s = \varnothing$ and $C_d \cup C_s = N \times N \times S$. Therefore, given a $(n_i, n_j, s_k) \in C_s$, we can classify C_d into seven subsets $C_d = \bigcup_{l=1}^{7} C^l_d(n_i, n_j, s_k)$ (see Figure 1) as:

$$\begin{array}{c} C^1_d(n_i, n_j, s_k) \triangleq \{(n_i, n_j, s_{k'}) \in C_d, s_{k'} \neq s_k\} \\ C^2_d(n_i, n_j, s_k) \triangleq \{(n_i, n_{j'}, s_k) \in C_d, n_{j'} \neq n_j\} \\ C^3_d(n_i, n_j, s_k) \triangleq \{(n_{i'}, n_j, s_k) \in C_d, n_{i'} \neq n_i\} \\ C^4_d(n_i, n_j, s_k) \triangleq \{(n_i, n_{j'}, s_{k'}) \in C_d, n_{j'} \neq n_j, s_{k'} \neq s_k\} \\ C^5_d(n_i, n_j, s_k) \triangleq \{(n_{i'}, n_j, s_{k'}) \in C_d, n_{i'} \neq n_i, s_{k'} \neq s_k\} \\ C^6_d(n_i, n_j, s_k) \triangleq \{(n_{i'}, n_{j'}, s_k) \in C_d, n_{i'} \neq n_i, n_{j'} \neq n_j\} \\ C^7_d(n_i, n_j, s_k) \triangleq \{(n_{i'}, n_{j'}, s_{k'}) \in C_d, n_{i'} \neq n_i, n_{j'} \neq n_j, s_{k'} \neq s_k\} \end{array} \quad (3)$$

If we infer $t^i_{j:k}$ using C^1_d, the model input in the training phase should be $(t^i_{j:k'}, V(Fs_{k'}))$ where $t^i_{j:k'}$ is the label and $V(Fs_{k'})$ is the value vector of $s_{k'}$'s features. But if we consider C^7_d, the model input can be extended to $(t^{i'}_{j':k'}, V(Fn_{i'} \times Fn_{j'} \times Fs_{k'}))$ where $t^{i'}_{j':k'}$ is the label and $V(Fn_{i'} \times Fn_{j'} \times Fs_{k'})$ is the value vector of the combination of the features from $n_{i'}$, $n_{j'}$ and $s_{k'}$. We define $F_7 \triangleq Fn_{i'} \times Fn_{j'} \times Fs_{k'}$ as a set of context-aware stereotypes for C^7_d. By this way, we can also define context-aware stereotype sets for the other six C^l_ds as follows:

$$\begin{array}{c} F_1 \triangleq Fs_{k'}, \quad F_2 \triangleq Fn_{j'}, \quad F_3 \triangleq Fn_{i'} \\ F_4 \triangleq Fn_{j'} \times Fs_{k'}, \quad F_5 \triangleq Fn_{i'} \times Fs_{k'}, \quad F_6 \triangleq Fn_{i'} \times Fn_{j'} \end{array} \quad (4)$$

In this paper, we consider a homogeneous multi-agent system, in which each agent (and each service) has the same feature set, that is $Fn_{i'} = Fn_{j'} = Fn$ and $Fs_{k'} = Fs$.

We differentiate the seven context-aware stereotypes in our design, because they can preserve more precise discriminative information to correlate the visible profiles of agents and services with the potential trustworthiness. By this way, our solution can detect not only the malicious patterns shown in trustee's profile but also the smart adversaries who hide their patterns among trustors, trustees and the services (i.e., context-correlated attacks).

It is worth noting that, if we infer a priori trust based on evidence-dense contexts from C^3_d, C^5_d, C^6_d and C^7_d, the agent $n_{i'}$ plays the role as an advisor. Our method cannot detect malicious advisors when they recommend honest agents as attackers or recommend attackers as honest ones. To cope with this challenge, we need a set of trustworthy advisors, denoted as $R \subseteq N$, in the system. We then restrict $n_{i'} \in R$ to refine the definition of C^3_d, C^5_d, C^6_d and C^7_d. Many existing trust models, such as [20, 23], can accurately model the trustworthiness of advisors from which R can be chosen.

3.3. A Priori Trust Inference Algorithm

To infer a priori trust for a given evidence-sparse context, we can train seven individual inference models based on the seven different kinds of evidence-dense contexts. In each training phase, the trustworthiness (from evidence-dense contexts) is the label and the value vector of context-aware stereotypes are the model inputs. We then use the seven models one by one to infer the trustworthiness of the target evidence-sparse context. We select the minimal value of outputs as the final result, because the lower value reported from the model on C^l_d indicates that some malicious patterns are more likely being captured in C^l_d (in

case $n_{i'} \in R$). We list the details in Algorithm 1. As can be seen, the while loop (from line 3 to line 7) traversals all the seven kinds of evidence-dense contexts given a target evidence-sparse context. Inside the loop, the code in line 4 is used to train an inference model using one kind of context-aware stereotype samples, while the code in line 5 is to use the trained model to infer a new trust for the target, and meanwhile keep the smallest inferred trust as the final output. It is worth noting that, we implement the $InferModel$ in the algorithm using a deep architecture [12] in this paper.

Input: $(n_i, n_j, s_k) \in C_s$
Output: $t^i_{j:k}$
1 $l \leftarrow 1$;
2 $t^i_{j:k} \leftarrow 1$;
3 **while** $l \leq 7$ **do**
4 $\quad InferModel \leftarrow Training(C^l_d(n_i, n_j, s_k))$;
5 $\quad t^i_{j:k} \leftarrow \min(t^i_{j:k}, InferModel(n_i, n_j, s_k))$;
6 $\quad l \leftarrow l + 1$;
7 **end**
8 return $t^i_{j:k}$;

Algorithm 1: Context-aware stereotypical trust inference algorithm

4. The stereotypical deep model

As a key step, a priori trust inference algorithm should implement a function (i.e., the $InferModel$ in Algorithm 1) to map context-aware stereotypes to trustworthiness. In this paper, we choose the deep learning model, particularly the deep belief network (DBN) [12], for this task due to two reasons. First, DBN is a probability generative model without any a priori assumption. It can learn the empirical samples layer by layer and eventually can discover more latent informative probabilities which can hardly be achieved by other learning algorithms (such as the group methods and the classification tree). Second, despite the DBN requires the labelled samples to fine tune the whole model, it has a pre-training phase which is aimed to reconstruct layer-wise connections using restricted Boltzmann machine (RBM). RBM is built with Gibbs sampling in order to utilize unlabeled samples for the training. The Gibbs sampling is a typical Markov chain Monte Carlo algorithm which can obtain a sequence of samples from a specified and practical sample set [29]. This algorithm is able to re-construct a target sample set (e.g., the training set) to approximate the real distribution, and can therefore lead the deep model to a much better robustness against sampling errors in the training set (i.e., the set of evidence-dense contexts).

4.1. Stereotypical Deep Model Overview

In general, a deep belief network (DBN) consists of an input layer H^0 in the bottom, several hidden layers H^n in the middle and a label layer L on the top. Each of two adjacent layers are fully interconnected, while the units in the same layer are disconnected. By this design, the units in each layer can independently interact with the units in their adjacent layers. This architecture can well simulate the physical structure of the human's neocortex.

In this paper, we design a new stereotypical DBN model to suit our trust inference problem. In particular, we build a meaningful deep architecture with three hidden layers, each of which aims to take a particular task from initial impressions to the final decision-making. We choose three hidden layers, because such design can well reason about the human's brain and has many successful applications in solving real-world problems [12, 30]. We present the overview of our architecture in Figure 2.

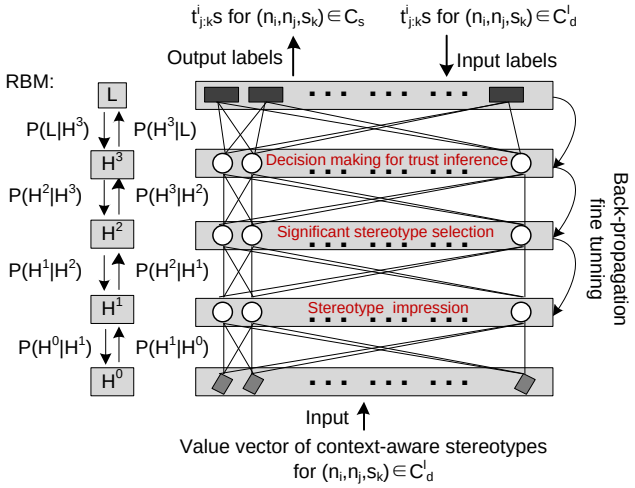


Figure 2: Deep architecture for context-aware stereotypical trust inference.

In this model, we use the first hidden layer H^1 to simulate a human's "initial impression" about the stereotypes. That is, given an object, a human will quickly skim all the visible features of this object in order to establish a first-hand impression. We deploy $|F_l|$, the number of available stereotypes in F_l, hidden units in this layer. If we take $F_{l=7}$ as an example, we have $|F_7| = x^2 \cdot y$ where $x = |F_n|$ is the number of visible features associated to each agent and $y = |F_s|$ is the number of visible features for each service.

For the second layer H^2, we use it to mimic a human selecting significant stereotypes for decision-making based on his/her logical thinking. Although there may exist a numerous stereotypes associated to a trust context, only a few are significant and effective to characterize the context. This phenomenon follows real-life scenarios, where humans can abstract significant features to make a rational decision. As a result, we set $\sigma \cdot |F_l|, 0 \leq \sigma < 1$ hidden units in the layer H^2. We can choose appropriate σ based on our evaluations in Section 5.2.

We employ the last hidden layer H^3 to mimic a human making a decision to rate the context. It is well known

that humans cannot provide fine-grained ratings. For example, many online rating systems (such as Epinions and Amazon) restrict their users to rate products in 10 levels (from 1 star to 5 stars with each step in 0.5 star). Moreover, to mimic humans' divergent thinking to some extent, we use 10 hidden nodes to represent each trust level and thus involve 100 hidden nodes in H^3.

In multi-agent systems, the agents' and services' features (i.e., stereotypes) may have diverse value ranges and types. In order to input the values from different stereotypes, we should normalize them at first. As the deep model is designed to accept quantitative data only, we consider two typical quantitative data value types: discrete values and continuous values [31]. To normalize discrete-valued stereotype, we can directly convert them to a binary vector, where the size of the vector is the logarithm (based on 2) of possible values of this feature. For example, if a feature has 10 candidate values, the length of the vector is $\lceil log_2 10 \rceil = 4$. In case the stereotype takes the 2nd candidate value, we can convert it to 0010. For the continuous-valued stereotypes, we can discrete them using a segmentation method. The segmentation criterion can be set according to the value distribution of each continuous-valued stereotype. To sum up, we formalize the input normalization process as follows. Let B_f be the binary vector of a context-aware stereotype $f \in F_l$ which is associated to a context (n_i, n_j, s_k). When we build the deep model for a sample context $(n_i, n_j, s_k) \in C_d^l$, we can generate the input $B(F_l)$ for this context by concatenating the binary value vectors $B_f, f \in F_l$ of (n_i, n_j, s_k) into one vector: $B(F_l) = Concat_{f \in F_l} B_f$.

For the label layer, we determine the number of units depending on how precise we expect the deep model can output. Given a context $(n_i, n_j, s_k) \in C_d^l$, the label is the trustworthiness (or belief) $t_{j:k}^i \in [0, 1]$. As a result, if we expect the output precision is 0.1, we can use 10 units in the label layer. But if we attempt to get 0.01 precision, we need 100 units. In this paper, we choose the 100-unit opinion. Note that, since $t_{j:k}^i$ is a continuous value, we need to discrete it and convert it to a binary vector at first.

We list the main learning steps of our stereotypical DBN model given a C_d^l as follows.

Step 1 We prepare the input binary vectors $B(F_l)$ for all the $(n_i, n_j, s_k) \in C_d^l$ (or including $(n_i, n_j, s_k) \in C_s$ to get more samples to rebuild the empirical distribution), and hence generate the empirical distribution of the input layer.

Step 2 After the structure of the lower layer is determined, we reconstruct the upper layer and the layer-wise connections between the two adjacent layers using a restricted Boltzmann machines.

Step 3 We repeat Step 2 until all the layer-wise connections are constructed.

Step 4 We fine-tune the whole deep model using the labelled samples (n_i, n_j, s_k) with $t_{j:k}^i$ from C_d^l.

4.2. Layer-Wise Reconstruction via RBM

Given the input layer H^0, our model can reconstruct the first hidden layer H^1 and the layer-wise connections between H^0 and H^1 using a restricted Boltzmann machine (RBM). We repeat this step to reconstruct H^2 given H^1, H^3 given H^2, and L given H^3, and eventually build the whole deep model. The target of each layer-wise reconstruction is to find a set of optimal layer-wise parameters $\theta = (\mathbf{W}, \mathbf{o}, \mathbf{u})$ which can minimize the state energy for each layer pair in RBM [32]. We will show the details of RBM using the layer-wise reconstruction of the first hidden layer H^1 given the input layer H^0 as follows. The layer-wise reconstructions for other layer pairs are similar.

Let $\mathbf{h}^0 = B(F_l)$ be the input binary vector of H^0 and $\mathbf{h}^1$ be the hidden unit vector of H^1. The layer-wise parameters of this layer pair is $\theta^0 = (\mathbf{W}^0, \mathbf{o}^0, \mathbf{u}^0)$, where $\mathbf{W}^0$ is the connection weight matrix between the layers H^0 and H^1, $\mathbf{o}^0$ is the biased vector for H^0 and $\mathbf{u}^0$ is the biased vector for H^1. $\mathbf{W}_{ij}^0 \in \mathbf{W}^0$ represents the connection weight between the i-th unit in $\mathbf{h}^0$ and the j-th unit in $\mathbf{h}^1$. $\mathbf{o}_i^0 \in \mathbf{o}^0$ is the bias of the i-th unit in $\mathbf{h}^0$, and $\mathbf{u}_j^0 \in \mathbf{u}^0$ is the bias of the j-th unit in $\mathbf{h^1}$.

We can calculate the state energy of this layer pair as

$$E(\mathbf{h}^0, \mathbf{h}^1, \theta^0) = -(\mathbf{h}^1)' \mathbf{W}^0 \mathbf{h}^0 - (\mathbf{o}^0)' \mathbf{h}^0 - (\mathbf{u}^0)' \mathbf{h}^1 \tag{5}$$

and then generate a layer-by-layer joint distribution for RBM as

$$P(\mathbf{h}^0, \mathbf{h}^1, \theta^0) = \frac{e^{-E(\mathbf{h}^0, \mathbf{h}^1, \theta^0)}}{\sum_{\mathbf{h}^0} \sum_{\mathbf{h}^1} e^{-E(\mathbf{h}^0, \mathbf{h}^1, \theta^0)}} \tag{6}$$

By factorizing the RBM's joint distribution, we can get two RBM associated layer-to-layer conditionals:

$$\begin{aligned} P(\mathbf{h}_i^0 = 1 | \mathbf{h}^1) &= s(\sum_k \mathbf{W}_{ik}^0 \mathbf{h}_k^1 + \mathbf{o}_i^0) \\ P(\mathbf{h}_j^1 = 1 | \mathbf{h}^0) &= s(\sum_k \mathbf{W}_{kj}^0 \mathbf{h}_k^0 + \mathbf{u}_j^0) \end{aligned} \tag{7}$$

where $s(x) = \frac{1}{1+exp(-x)}$ and each unit value (i.e., $\mathbf{h}_i^0$ and $\mathbf{h}_j^1$) is a binary value. Using Equation (7), we can eventually have the conditional probability distribution $P(\mathbf{h}^0 | \mathbf{h}^1)$ and $P(\mathbf{h}^1 | \mathbf{h}^0)$. With these two conditionals, we can apply Gibbs sampling method to sample H^1 given H^0, and then sample H^0 given H^1 and then repeat the two steps t times. This process forms a Markov chain [33], which can generate $\mathbf{h}^1(t)$ only depending on $\mathbf{h}^1(t-1)$. $\mathbf{h}^1(t)$ is the t-th state of $\mathbf{h}^1$ in the Markov chain and the train starts at $\mathbf{h}^0(t=0)$.

By using Gibbs sampling to reconstruct H^0 and H^1, the layer-wise parameters θ^0 should be optimized as well. The optimization goal is to minimize a so-called log-likelihood $\log P(\mathbf{h}^0)$. We have $P(\mathbf{h}^0) = \sum_{\mathbf{h}^1} P(\mathbf{h}^0, \mathbf{h}^1, \theta^0)$ and can write its log version (i.e., the log-likelihood of $P(\mathbf{h}^0)$) as

$$\log P(\mathbf{h}^0) = \log \sum_{\mathbf{h}^1} e^{-E(\mathbf{h}^0, \mathbf{h}^1, \theta^0)} - \log \sum_{\mathbf{h}^0} \sum_{\mathbf{h}^1} e^{-E(\mathbf{h}^0, \mathbf{h}^1, \theta^0)} \tag{8}$$

We cannot minimize $\log P(\mathbf{h}^0)$ by finding the optimal θ^0 directly, because the parameters depend on each other (i.e., the partial derivative of $\log P(\mathbf{h}^0)$ with respect to one parameter contains other parameters). To tackle this challenge, we apply a gradient descent (GD) method with the line search. The GD method can calculate the derivative of $\log P(\mathbf{h}^0)$ with respect to the parameter $\theta^0 = (\mathbf{W}^0, \mathbf{o}^0, \mathbf{u}^0)$ as

$$\begin{aligned}\frac{\partial \log P(\mathbf{h}^0(0))}{\partial \theta^0} = &- \sum_{\mathbf{h}^1(0)} P(\mathbf{h}^1(0)|\mathbf{h}^0(0)) \frac{\partial E(\mathbf{h}^1(0)|\mathbf{h}^0(0))}{\partial \theta^0} \\ &+ \sum_{\mathbf{h}^1(t)} \sum_{\mathbf{h}^0(t)} P(\mathbf{h}^1(t)|\mathbf{h}^0(t)) \frac{\partial E(\mathbf{h}^1(t)|\mathbf{h}^0(t))}{\partial \theta^0}\end{aligned} \tag{9}$$

To determine how long the Markov chain is deserved (i.e., determining the t), we refer to a contrastive divergence algorithm [34], which can take a small t (typically take $t = 1$) to run the chain by referring a difference between two Kullback-Leibler divergences. The Kullback-Leibler divergence is a typical measure of the information loss when one probability distribution is used to approximate another [35]. The authors [34] chose the Kullback-Leibler divergence in their design in order to ensure the minimized information loss when the Markov chain is constructed. By doing so, we can calculate the derivative of $\log P(\mathbf{h}^0)$ with respect to the parameter $\mathbf{W}^0$ as

$$\frac{\partial \log P(\mathbf{h}^0(0))}{\partial \mathbf{W}^0} = \langle \mathbf{h}^1(0)' \mathbf{h}^0(0) \rangle_d - \langle \mathbf{h}^1(1)' \mathbf{h}^0(1) \rangle_m \tag{10}$$

where, $\langle \cdot \rangle_d$ is the expectation regarding to the data distribution, and $\langle \cdot \rangle_m$ is the expectation for the model distribution which is sampled by one step Gibbs sampling. We therefore can update the parameter $\mathbf{W}^0$ as

$$\mathbf{W}^0(t{=}1) = \varepsilon_W \mathbf{W}^0(0) + \rho_W (\langle \mathbf{h}^1(0)' \mathbf{h}^0(0) \rangle_d - \langle \mathbf{h}^1(1)' \mathbf{h}^0(1) \rangle_m) \tag{11}$$

where, ε_W is the momentum for smoothness and used here to mitigate overfitting. ρ_W is the learning rate of $\mathbf{W}^0$ (i.e., the step length of gradient descent for $\mathbf{W}^0$). Similarly, we can update the other two parameters $\mathbf{o}^0$ and $\mathbf{u}^0$ as

$$\begin{aligned}\mathbf{o}^0(t=1) &= \varepsilon_o \mathbf{o}^0(0) + \rho_o (\mathbf{h}^0(0) - \mathbf{h}^0(1)) \\ \mathbf{u}^0(t=1) &= \varepsilon_u \mathbf{u}^0(0) + \rho_u (\mathbf{h}^1(0) - \mathbf{h}^1(1))\end{aligned} \tag{12}$$

where, ε_o and ε_u are momentums. ρ_o and ρ_u are the learning rate of $\mathbf{o}^0$ and $\mathbf{u}^0$ respectively. Our model uses the same values for the momentums and learning rates as the typical deep learning method [32] used.

4.3. Supervised Global Fine-Tuning

As described in Section 4.2, the RBM reconstruction is an unsupervised learning progress and thus cannot utilize the labelled samples. To avoid this issue and make a full utilization of the labelled samples in our learning process, we implement a supervised mean-field posterior approximation algorithm through back-propagation [12]. This method can fine-tune the deep model with labelled samples (i.e., evidence-dense contexts). The tuning goal is to minimize the model error by further optimizing the parameter $\theta = (\mathbf{W}, \mathbf{o}, \mathbf{u})$ from the label layer back to the input layer (down-pass). The model error can be measured in terms of a cross-entropy between the probability distribution of the true trustworthiness and that of the inferred trustworthiness. The cross-entropy is a typical measure that can be used to calculate the average number of error bits when one distribution is used to approximate another [36]. In our CAST, we can calculate the cross-entropy as $-\sum t^i_{j:k} \log \bar{t}^i_{j:k}$, where $t^i_{j:k}$ is the correct trustworthiness label of a context (n_i, n_j, s_k) in the evidence-dense context training set C^l_d, and $\bar{t}^i_{j:k}$ is the output trustworthiness label when inputting the stereotypes of the evidence-dense context (n_i, n_j, s_k) to the model.

5. Evaluation

In this section, we conduct a rich set of simulation-based experiments to confirm the effectiveness of our CAST. In particular, we first prove the effectiveness of context-aware stereotypes against context-correlated attacks. We then validate the effectiveness of our deep architecture in making our stereotypical trust model robust. We also evaluate the performance overheads of CAST at the end.

5.1. Experiment Setup

In our experiments, we simulate a multi-agent system with $|N| = 100$ agents and $|S| = 50$ services for each agent. We then have $|N \times N \times S| = 500,000$ distinct trust contexts in total. If we exclude the context where the trustor and the trustee are the same agent, we can obtain a reasonable definition $|N| \cdot (|N| - 1) \cdot |S| = 495,000$. That is, each agent will select any other agents to consume their services in every simulated time slot. Moreover, we randomly choose $|R| = 10$ agents from N as trustworthy advisors, and consider that each agent can trust the recommendations from these advisors with an equal belief. We set 100 visible features (i.e., stereotypes) in each agent's profile and 20 features in each service's profile. Each feature contains a binary value (the same value setting is used in [5]).

Moreover, we simulate context-correlated attacks and investigate whether CAST can make an accurate a priori trust inference against this kind of attacks. Since all the stereotypical trust inference models share a common assumption that they can work if and only if malicious patterns can be expressed in visible features [7, 5, 8, 9, 10], we follow this assumption in our experiments but classify the malicious patterns induced by the context-correlated attacks into seven types (we list them in Table 1). By this classification, our seven kinds of context-aware stereotypes can capture each type of the malicious patterns one to one. In this paper, we use the malicious pattern to describe the attacking behaviors depending on the contexts. For example, the malicious pattern ① in Table 1 indicates that the attack will happen if and only if the targeted service contains some specific stereotypes (i.e., visible features), while the malicious pattern ② means the attack

should be launched according to the trustee's stereotypes. Further, we sample the posterior trustworthiness $t^i_{j:k}$ under attack using a normal distribution $\mathcal{N}(0.1, 0.01)$, and the posterior trustworthiness $t^i_{j:k}$ without being attacked from $\mathcal{N}(0.9, 0.01)$. Our posterior trustworthiness sampling method is reasonable and feasible, because it can well simulate the uncertainty of belief required by the subjective logic. We use the posterior trustworthiness as the ground truth when we apply stereotypical models to infer a priori trust. In our simulation, we consider the percentage $P_c \in (0, 1]$ of trust contexts are malicious and the malicious pattern is expressed in the percentage $P_f \in (0, 1]$ of visible features. Since our investigation indicates that our CAST and existing stereotypical models are all not sensitive to P_c and P_f (the CAST case has been shown in Section 5.2), we simply set $P_c = P_f = 20\%$ for our experiments.

Table 1: Context-correlated malicious patterns.

Malicious pattern	Malicious features			Captured by
	Trustor	Trustee	Service	
Pattern ①			√	$C_d^{l=1}$
Pattern ②		√		$C_d^{l=2}$
Pattern ③	√			$C_d^{l=3}$
Pattern ④		√	√	$C_d^{l=4}$
Pattern ⑤	√		√	$C_d^{l=5}$
Pattern ⑥	√	√		$C_d^{l=6}$
Pattern ⑦	√	√	√	$C_d^{l=7}$

When we have established the ground truth trustworthiness for every trust context, we will simulate one context as a evidence-sparse context and meanwhile assume all the others are evidence-dense. We repeat this simulation by traversing all the contexts as evidence-sparse one time, and measure a priori trust inference accuracy in reverse of a root-mean-square deviation (RMSD) [37]. The RMSD is a very popular measurer of the differences between the true values and the values predicted by a model. A lower RMSD means a higher trust inference accuracy. In our simulation, we calculate RMSD as

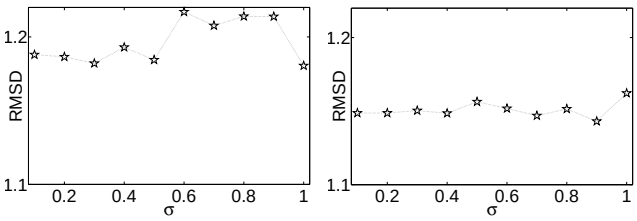
$$RMSD = \sqrt{\frac{\sum_{(n_i, n_j, s_k) \in C_s} (\bar{t}^i_{j:k} - t^i_{j:k})^2}{|C_s|}} \quad (13)$$

where, $t^i_{j:k}$ is the ground truth and $\bar{t}^i_{j:k}$ is the trustworthiness inferred by a model. Since we iterate every context as evidence-sparse one by one, the C_s could include all the trust contexts in our experiments. In the following of the evaluations, we will run each experiment 100 rounds and show the average RMSD. Without specified explanation, we will simply use RMSD to represent the average RMSD in Figure 3 to Figure 5.

We acknowledge that there exist another famous error measure, the mean absolute error (MAE) in the literature [37]. Compared with MAE which calculates the average error of a set using the same weight to each sample error in this set, the root-mean-square deviation (RMSD) can assign a higher weight to a larger sample error when computing the set error [37]. In our problem, we target to infer the trustworthiness through context-aware stereotypes for the agents without enough trust evidences. A higher trust inference error necessarily leads to a higher negative impact when the others relying on the inferred trustworthiness to make a security-related decision. For example, given an agent with the true trustworthiness 0.9, if we incorrectly infer the trustworthiness as 0.89, this 0.01 error could be tolerable. But if we mistakenly infer it as 0.1, such 0.8 error may make a significant security risk with a very high probability. To take into account this weighting requirement, we choose RMSD as the accuracy/robustness measurer in this paper.

5.2. Parameter Selection

As discussed in Section 3, we need to choose a parameter $\sigma \in (0, 1]$ in order to determine the number of hidden units in the second hidden layer, which is designed to model the significant feature selection process. To achieve this goal, we investigate how accurate the trust inference we can make when σ is increased from 0.1 to 1 with 0.1 as the step. In particular, we calculate the RMSD for each tested σ value by considering the seven types of context-correlated malicious patterns one by one and report its mean value in Figure 4. As can be seen, from case $P_c = P_f = 20\%$ to case $P_c = P_f = 80\%$, the RMSD is changed from 1.19 to 1.16 in average (the difference is less than 2.5%). We thereby confirm that CAST is not sensitive to P_c and P_f. Moreover, we find that the RMSD gets the smallest value 1.14 when $\sigma = 0.9$ in case $P_c = P_f = 80\%$. We thus use this value in the following experiments.

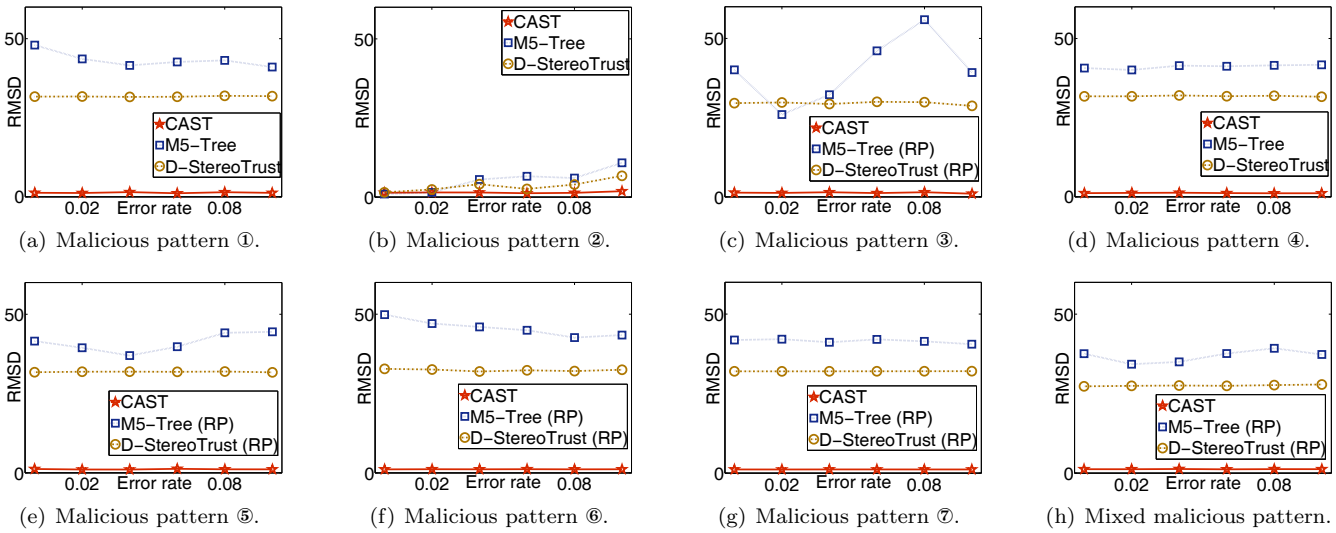


(a) $P_c = 20\%$ and $P_f = 20\%$. (b) $P_c = 80\%$ and $P_f = 80\%$.

Figure 4: The selection of σ to determine how many hidden nodes (i.e., $\sigma \cdot |F_l|$) we need to put in the second hidden layer in our deep model. The accuracy is measured in terms of root-mean-square deviation (RMSD). A lower RMSD indicates a more accurate trust inference.

5.3. Reasons for Context-Aware Stereotypes

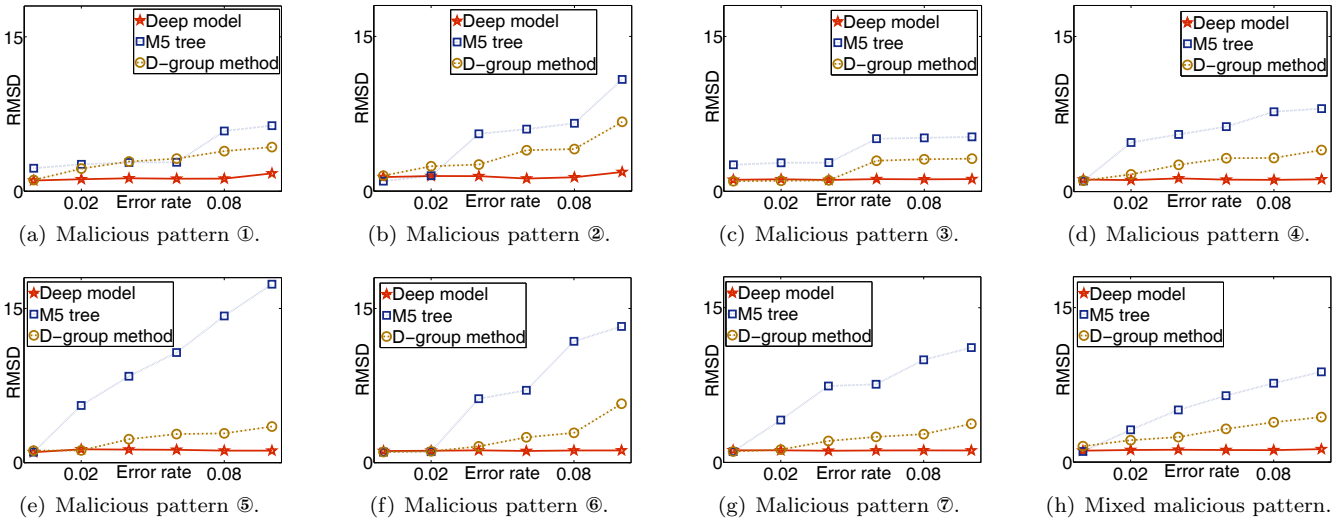
To demonstrate the effectiveness of the use of context-aware stereotypes, we compare our CAST with two state-of-the-art trustee stereotypical trust models [5, 7]. In [5], the authors proposed to establish a classification and regression tree, known as M5 tree, to learn from evidence-dense contexts and then use the tree to infer a priori trust



(a) Malicious pattern ①. (b) Malicious pattern ②. (c) Malicious pattern ③. (d) Malicious pattern ④.

(e) Malicious pattern ⑤. (f) Malicious pattern ⑥. (g) Malicious pattern ⑦. (h) Mixed malicious pattern.

Figure 3: We show the effectiveness of context-aware stereotypes by comparing trust inference accuracy between CAST and trustee stereotypical methods such as M5 tree [5] and D-StereoTrust [7]. The accuracy is measured in terms of root-mean-square deviation (RMSD). A lower RMSD indicates a more accurate inference. The error rate is the percentage of mis-classified labelled samples in the training set. The mixed malicious pattern consists of the seven malicious patterns with an equal probability.



(a) Malicious pattern ①. (b) Malicious pattern ②. (c) Malicious pattern ③. (d) Malicious pattern ④.

(e) Malicious pattern ⑤. (f) Malicious pattern ⑥. (g) Malicious pattern ⑦. (h) Mixed malicious pattern.

Figure 5: We show the effectiveness of deep model by comparing trust inference accuracy when we applying context-aware stereotypes to the deep model, the M5 tree [5] and the dichotomy enhanced group method [7]. The accuracy is measured in terms of root-mean-square deviation (RMSD). A lower RMSD indicates a more accurate inference. The error rate is the percentage of mis-classified labelled samples in the training set. The mixed malicious pattern consists of the seven malicious patterns with an equal probability.

for the evidence-sparse contexts. We refer it as M5-tree in our comparison. While in [7], the authors designed a group-based method for trust stereotyping as well as a dichotomy based enhancement. We choose the dichotomy enhancement (referred as D-StereoTrust) in our comparison because this one has better inference performance than its original version. As described in [7], the authors claimed that the D-StereoTrust's effectiveness mainly relies on how accurate the membership functions can be approximated when grouping stereotypes, but they do not give the details on how to generate an appropriate membership function in their design. Instead, we simply assume the optimal membership function can be obtained in advance and thus can show the best inference performance (i.e., theoretical upper bound) of D-StereoTrust. Note that, we do not need to compare our method with [8] and [10]. The former one [8] suggested a fuzzy model as the membership functions to approximate the D-StereoTrust's upper bound performance for trust inference, but we have already assumed the best membership function to show

the upper bound. The latter one [10] discussed how to extract more available features from trustees for the trust inference. In our design, we just consider that the set of visible features is readily prepared.

In Figure 3, we compare CAST with M5-tree and D-StereoTrust against the seven types of context-correlated malicious patterns (listed in Table 1) as well as a mixed malicious pattern which involves the seven kinds of patterns with an equal probability. We append (RP) to M5-tree and D-StereoTrust in Figures 3(c) and 3(e)-3(h), because these malicious patterns, which indicate the malicious trustee agents launch attacks according to different trustors stereotypes, can only make effects when consulting advisors for trust inference. In these cases, we need to use the stereotypical reputation method [5] in our comparison. The (RP) represents the trustworthy advisors run stereotypical methods such as M5-tree and D-StereoTrust for the target trustees on behalf of the trustors, and finally recommend the results to the trustors. The error rate is the percentage of mis-classified labelled samples in the training set and will be elaborated on in Section 5.4.

As shown in Figure 3, CAST demonstrates a much higher accuracy (the RMSD keeps in around 1 for all the sub figures) than both the M5-tree and D-StereoTrust (the RMSD is about 45 and 30 respectively, except in the sub figure 3(b)). Due to the overlook of context-aware stereotypes, the trustee stereotypical methods (including M5-tree and D-StereoTrust) cannot discover the malicious patterns in Figures 3(a) and 3(c)-3(h), hence getting a more than 30 times accuracy reduction compared with our design. While for the sub figure 3(b), the malicious pattern only appears in trustee's features and thus can be captured by the trustee stereotypical methods. In this case, our CAST can still make a trust inference with a slightly higher accuracy but the level is much lower than other cases. We will further investigate this case with more details in Section 5.4.

5.4. Reasons for Applying Deep Model

Although Section 5.3 has proved that our CAST can do a much better job to infer a priori trust compared with the state-of-the-art techniques [5, 7] which do not take into account context-aware stereotypes, it is still unclear how much benefit we can obtain due to the use of the deep model. We fill this gap in this section. In particular, we purposely introduce modify existing trustee stereotypical models to accept context-aware stereotypes and then compare our deep model with these modified versions using the same set of context-aware stereotypes. In another word, we can use the M5 tree [5] and dichotomy based group method (D-group method in short) [7] to implement the $InferModel$ function in Algorithm 1, and then use these models to compare with CAST which implements the $InferModel$ function using a deep model.

Our comparison is mainly against an error rate which is the percentage of mis-classified labelled samples in the training set. This rate can well reflect a common practical phenomenon where some contexts with malicious patterns may be regarded with a high trustworthiness due to the attacks that are targeted trust management system itself. A stereotypical trust inference model that maintains sufficient accuracy against a higher error rate can be considered with a sound robustness.

We show our results in Figure 5. As can be seen, in the presence of all the seven types of context-correlated malicious patterns as well as the mixed pattern, our deep model remains in a high accuracy (the RMSD is at around 1 for all the cases), while the accuracy of M5 tree and D-group method decreases sharply (the RMSD grows up at least three times) when the error rate is increasing from 0 to 0.1, despite the M5 tree and D-group method can sometimes have a slightly better result when the error rate is 0. Therefore, we have successfully confirmed that our deep model can achieve a better trust inference robustness than previous solutions.

5.5. Performance overheads

We analyze the time complexity of CAST (specifically the deep model) as well as evaluate its practical performance overheads in this section. In general, CAST's time complexity largely depends on the structure of the deep model, such as the number of layers and the number of units in each layer. As described in Section 4, our stereotypical deep model includes five layers: one input layer, three hidden layers and an output layer. As the input layer is for the value vector of context-aware stereotypes, it has $|V(F_l)|$ units inside. Moreover, the number of units in the first hidden layer equals to the number of context-aware stereotypes $|F_l|$. The second hidden layer has $\sigma \cdot |F_l|$ units and the third hidden layer has 100 units. The output layer has 100 units. Then the numbers of layer-wise connections for each layer pair are $|V(F_l)| \cdot |F_l|$, $\sigma \cdot |F_l|^2$, $100 \cdot \sigma \cdot |F_l|$ and 10000 respectively.

In the training phase, the RBM should run two times of Gibbs sampling for each layer pair and the supervised fine-tuning should run from the output layer back to the input layer one time, each layer-wise connection should be calculated three times. Therefore, we can compute the time complexity of training phase as $O(k \cdot 3 \cdot (|V(F_l)| \cdot |F_l| + \sigma \cdot |F_l|^2 + 100 \cdot \sigma \cdot |F_l| + 10000))$, where k is the number of iterations we will run in the training phase. In our experiments, we use $k = 1000$. In the trust inference phase, the calculation is from the input layer to the output layer one time. We thus can get the time complexity of this phase as $O(|V(F_l)| \cdot |F_l| + \sigma \cdot |F_l|^2 + 100 \cdot \sigma \cdot |F_l| + 10000)$.

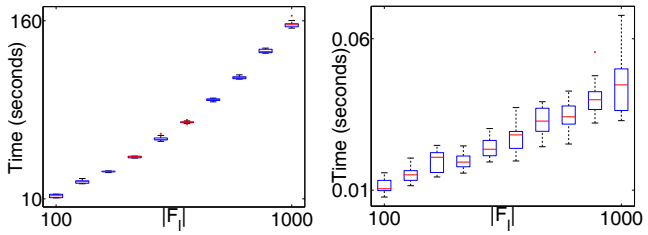
As can be seen, CAST's time complexity is mainly determined by $|F_l|$. We then evaluate CAST's performance overheads from $|F_l| = 100$ to $|F_l| = 1000$ in our experiments. The computer we use for this evaluation has a 8-core CPU at 1.73 GHz each and 8 GB memory. We run the evaluation for each setting 20 times and report the results using boxplot in Table 2 and Figure 6. In the experimental results, we confirm that the CAST's performance overhead can grow up when the $|F_l|$ becomes larger.

Moreover, the overheads induced by the training phase is about 3000 times bigger in average than those induced by the inference phase. This is in accordance with our time complexity analysis: the training phase has a $3 \cdot k = 3000$ times larger complexity than the inference phase.

These performance evaluation results indicate that, our CAST is more appropriate to be run in powerful agents, since the training phase will induce a large delay (around 160 seconds when $|F_l| = 1000$). For the lightweight agents, a better solution is to out source the training phase to another powerful third parties (such as a public cloud services) and only run the inference phase in the lightweight agents themselves. We leave a future work for this study.

Table 2: Performance overheads (in terms of time cost) induced by CAST.

$\|F_l\|$	Lower|Median|Upper time cost (seconds)	
	Training phase	Inference phase
100	10.81|12.61|13.93	0.010|0.011|0.016
200	22.71|24.49|27.13	0.012|0.015|0.021
300	32.12|33.04|33.54	0.014|0.019|0.025
400	44.38|45.56|46.00	0.016|0.020|0.025
500	58.43|60.01|63.42	0.019|0.024|0.030
600	73.40|74.45|75.67	0.020|0.028|0.037
700	91.72|93.68|94.86	0.024|0.033|0.039
800	110.88|112.32|114.46	0.025|0.034|0.043
900	132.74|134.22|136.99	0.032|0.040|0.048
1000	153.80|156.65|160.23	0.033|0.045|0.068



(a) Training phase. (b) Trust inference phase.

Figure 6: Boxplots for performance overheads (in terms of time cost) induced by CAST.

6. Conclusion and Future Work

In this paper, we have proposed CAST, a novel context-aware stereotypical trust model, to address the the challenging a priori trust inference problem in the literature. CAST has successfully extended the malicious pattern detection from a trustee's profiles to a trust context perspective, hence being able to capture context-correlated attacks. This kind of attacks could cause seriously negative effects to a priori trust inference but cannot be well addressed by previous solutions. Moreover, CAST has been designed with a new stereotypical deep model. The use of deep model to solve trust problems is promising, because the deep model can well approximate how a human is reasoning and making decisions with trust. To the best of our knowledge, our work is the first attempt that models trust using deep representations. At the end, we have conducted a rich set of experiments to evaluate the effectiveness and performance of CAST in practical settings. Our design and the experimental results have successfully confirmed that, (1) CAST is the first trust inference solution that can be resist context-correlated attacks, and (2) CAST can make a more accurate and robust trust inference with a moderate performance overhead than the state-of-the-art.

In the future, we will attempt to remove the assumptions we have made in our current work. For example, we will design new methods to extend CAST to suit the conditions where trustworthy advisors are not presented. We will also design some stereotype (i.e., feature) authentication mechanism to avoid noise and unreliable profile features. Moreover, we will investigate how to apply CAST to some real multi-agent applications, especially these whose posteriori trust is too expensive to obtain, such as the wireless sensor networks [38] and the Internet of things [39].

Acknowledgement

The work was partially supported by National Natural Science Foundation of China under grant No.61205017, the Fundamental Research Funds for the Central Universities and the ASTAR/I2R-SERC, Public Sector Research Funding (PSF) Singapore (M4070212.020).

References

[1] Trung Dong Huynh, Nicholas R Jennings, and Nigel R Shadbolt. An integrated trust and reputation model for open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 13(2):119–154, 2006.

[2] Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision support systems*, 43(2):618–644, 2007.

[3] WT Luke Teacy, Jigar Patel, Nicholas R Jennings, and Michael Luck. Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12(2):183–198, 2006.

[4] Yonghong Wang and Munindar P Singh. Formal trust model for multiagent systems. In *Proceedings of the 20th the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 7, pages 1551–1556, 2007.

[5] Chris Burnett, Timothy J Norman, and Katia Sycara. Bootstrapping trust evaluations through stereotypes. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 241–248, 2010.

[6] Jordi Sabater and Carles Sierra. Review on computational trust and reputation models. *Artificial intelligence review*, 24(1):33–60, 2005.

[7] Xin Liu, Anwitaman Datta, Krzysztof Rzadca, and Ee-Peng Lim. Stereotrust: a group based personalized trust model. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 7–16, 2009.

[8] Hui Fang, Jie Zhang, Murat Sensoy, and Nadia Magnenat Thalmann. A generalized stereotypical trust model. In *Proceedings of the 11th International Conference on Trust, Security and*

Privacy in Computing and Communications (TrustCom), pages 698–705, 2012.

[9] Chris Burnett, Timothy J Norman, and Katia Sycara. Stereotypical trust and bias in dynamic multiagent systems. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(2):26, 2013.

[10] Murat Şensoy, Burcu Yilmaz, and Timothy J Norman. Stage: Stereotypical trust assessment through graph extraction. *Computational Intelligence*, 2014.

[11] Isaac Pinyol and Jordi Sabater-Mir. Computational trust and reputation models for open multi-agent systems: a review. *Artificial Intelligence Review*, 40(1):1–25, 2013.

[12] Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[13] Guy Wallis and Heinrich Bülthoff. Learning to recognize objects. *Trends in cognitive sciences*, 3(1):22–31, 1999.

[14] Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[15] C Neil Macrae and Galen V Bodenhausen. Social cognition: Categorical person perception. *British Journal of Psychology*, 92(1):239–255, 2001.

[16] Peng Zhou, Xiapu Luo, Ang Chen, and Rocky KC Chang. Sgor: Trust graph based onion routing. *Computer Networks*, 57(17):3522–3544, 2013.

[17] Peng Zhou, Xiapu Luo, and Rocky KC Chang. Inference attacks against trust-based onion routing: Trust degree to the rescue. *computers & security*, 39:431–446, 2013.

[18] Xiaojing Gu and Xingsheng Gu. On the detection of fake certificates via attribute correlation. *Entropy*, 17(6):3806–3837, 2015.

[19] Bin Yu and Munindar P. Singh. An evidential model of distributed reputation management. In *Proceedings of International Autonomous Agents and Multi Agent Systems (AAMAS)*, pages 294–301, 2002.

[20] W. T. Luke Teacy, Jigar Patel, Nicholas R. Jennings, and Michael Luck. Coping with inaccurate reputation sources: Experimental analysis of a probabilistic trust model. In *Proceedings of Fourth International Autonomous Agents and Multiagent Systems (AAMAS)*, 2005.

[21] Sepandar D Kamvar, Mario T Schlosser, and Hector Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*, pages 640–651, 2003.

[22] Li Xiong and Ling Liu. Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):843–857, 2004.

[23] Siwei Jiang, Jie Zhang, and Yew-Soon Ong. An evolutionary model for constructing robust trust networks. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems (AAMAS)*, pages 813–820, 2013.

[24] Chris Burnett, T Norman, and Katia Sycara. Sources of stereotypical trust in multi-agent systems. In *Proceedings of the 14th International Workshop on Trust in Agent Societies*, page 25, 2011.

[25] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 184–199. 2014.

[26] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[27] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008.

[28] Elaine Rich. User modeling via stereotypes. *Cognitive science*, 3(4):329–354, 1979.

[29] Walter R Gilks and Pascal Wild. Adaptive rejection sampling for gibbs sampling. *Applied Statistics*, pages 337–348, 1992.

[30] Yan Liu, Sheng-hua Zhong, and Wenjie Li. Query-oriented multi-document summarization via unsupervised deep learning. In *Proceedings of the 27th national conference on Artificial intelligence (AAAI)*, 2012.

[31] Kenneth Rosen. *Discrete Mathematics and Its Applications 7th edition*. McGraw-Hill Science, 2011.

[32] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[33] Walter R Gilks. *Markov chain monte carlo*. Wiley Online Library, 2005.

[34] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[35] Rainer Dahlhaus. On the kullback-leibler information divergence of locally stationary processes. *Stochastic Processes and their Applications*, 62(1):139–168, 1996.

[36] John E Shore and Rodney W Johnson. Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Transactions on Information Theory*, 26(1):26–37, 1980.

[37] J Scott Armstrong and Fred Collopy. Error measures for generalizing about forecasting methods: Empirical comparisons. *International journal of forecasting*, 8(1):69–80, 1992.

[38] Peng Zhou, Siwei Jiang, Athirai Irissappane, Jie Zhang, Jianying Zhou, and Joseph Chee Ming Teo. Toward energy-efficient trust system through watchdog optimization for wsns. *IEEE Transactions on Information Forensics and Security*, 10(3):613–625, 2015.

[39] Yosra Ben Saied, Alexis Olivereau, Djamal Zeghlache, and Maryline Laurent. Trust management system design for the internet of things: a context-aware and multi-service approach. *Computers & Security*, 39:351–365, 2013.