Data-driven Trust Prediction in Mobile Edge Computing-based IoT Systems

Prabath Abeysekara, Hai Dong, *Senior Member, IEEE* and A. K. Qin, *Senior Member, IEEE*

Abstract—We propose a data-driven distributed machine learning approach to scalably predict the trustworthiness of homogeneous IoT services in heterogeneous Mobile Edge Computing (MEC)-based IoT systems. The proposed approach formulates training distributed trust prediction models within an MEC-based IoT system as a Network Lasso problem. We then introduce a variant of the Stochastic Alternating Method of Multipliers framework enriched with the ability for feature selection at each MEC layer. To verify the effectiveness of the proposed approach, we carried out a comprehensive evaluation on three real-world datasets adjusted to exhibit the context-dependent trust information accumulated in MEC environments within a given MEC topology. The experimental results affirmed the effectiveness of our approach and its suitability to predict trustworthiness of IoT services in MEC-based IoT systems.

Index Terms—Trust, Internet of Things, Mobile Edge Computing, Machine Learning.

1 Introduction

Trust in Internet of Things (IoT) systems is an indispensable element, which enables secure interactions between IoT services and their consumers. It also assists people and smart devices alike to consume useful IoT services with an elevated degree of confidence that these interactions will bring favourable outcomes [1]. For instance, in fitness tracking where the personally identifiable information such as social profile, behavioural and location data is shared with third party services, trust provides *assurance* that this collected information will be used as agreed by all parties [2]. In intelligent transport systems, trust allows autonomous vehicles to determine services that provide credible location and traffic information supplied by vehicular networks, etc [3]. As such, trust can be deemed to play an integral part towards ensuring user acceptance towards IoT systems.

Trust information generated in IoT systems takes various shapes such as 1) quality of service (QoS) data of the services, 2) recommendations and ratings of users, 3) statistics monitored by third party monitoring systems in the form of success or failure rates or the number of completed transactions between IoT services and their consumers [4]. In most existing trust evaluation approaches, this trust information is transformed or augmented into various trust metrics within centralized cloud-based computing infrastructures, and used to model trustworthiness of IoT services within a particular application context [1]. These trust models are then used to 1) derive predictions on the probability of success of a transaction, 2) determine the credibility of a given service and 3) filter out services that match the Quality of Experience (QoE) characteristics of service consumers, etc.

However, the evolution of IoT systems in the recent past makes trust modeling strategies used by the existing trust evaluation approaches inherently prohibitive and calls for alternatives that are predominantly *data-driven*. For instance, most existing trust evaluation approaches for IoT services rely heavily on *model-driven* strategies. In such strategies, domain experts model the definition of trust based on trust characteristics of a given application context that are often observed and interpreted using their domain expertise [1][5]. Such approaches require extensive domain knowledge, which is often developed and acquired over years and years of experimentation. Also, model-driven approaches based on manual analysis are particularly prohibitive in settings that require IoT service trust to be *simultaneously* modeled in context-dependent manner for many context-environments within a particular application context [6].

In addition, the rapid rise of the IoT and the growth of IoT services and consumers have resulted in trust information being generated in massive volumes in exorbitant velocities. *The sheer volume of this trust information transmitted into centralized cloud-based infrastructure from the point of origin* is also contributing to the ever-growing stress on current networking infrastructure [7][8]. In addition, *facilitating the trust evaluation requirements of delay-sensitive applications* such as connected autonomous vehicles and video streaming can also challenge the existing infrastructures. Such applications typically require service delivery (i.e. evaluating trustworthiness of useful IoT services) guarantees within a few tens of milliseconds, which cannot be accomplished through existing centralized cloud-based infrastructure [8].

Mobile Edge Computing (MEC) is an emerging paradigm aiming to reduce the rapidly growing network stress on core networks of mobile network providers caused by the high-volume IoT data [7][8]. Characterized by the pooled computing resources sitting in close proximity to mobile devices at the base stations of cellular networks, MEC allows shifting the task of accumulating and processing high-volume data from conventional cloud-based

- *P. Abeysekara, and H. Dong are with School of Computing Technologies, RMIT University, Melbourne, VIC, Australia E-mail: prabath.abeysekara@rmit.edu.au; hai.dong@rmit.edu.au*
- *A.K. Qin is with Department of Computer Science and Software Engineering, Swinburne University of Technology, Hawthorn, VIC, Australia E-mail: kqin@swin.edu.au*

Manuscript received December 19, 2020; revised September 9, 2021; accepted October 5, 2021. (Corresponding author: Hai Dong)

infrastructures to heterogeneous computing environments (i.e. MEC environments) located at the edge of the network. In addition, coupled with the faster network connectivity infrastructure such as 5G, MEC also promises ultra-low communication delays to delay-sensitive applications, which leads to significantly better user experience. Hence, MEC can be viewed as a seemingly befitting solution to address the challenges in conventional cloud-centric IoT systems outlined before [9][10].

Trust prediction in an MEC-based IoT system, however, poses the following key challenges owing to their unique system architecture and other intrinsic characteristics.

1) The inherently *distributed* way MEC environments generate *context-dependent* and potentially *high-dimensional* trust information makes most existing trust prediction strategies less applicable: In a typical MEC-based IoT system, trust information is generated in a *distributed* manner due to its inherently distributed system architecture. This creates a distributed topology of MEC-local data silos across different MEC environments. Even though we can run existing trust prediction strategies against each MEC-local data silo independently, the nature of trust information accumulated in this setting calls for alternative trust prediction strategies that are *distributed* and *collaborative*, as well as encourage trust prediction model training to take place closer to the edge where the trust information is accumulated. Such a *distributed* approach can be instrumental in relieving the network stress on the core networks of mobile networks in-line with the goals of MEC. Meanwhile, allowing *collaboration* among MEC environments during trust prediction model training can help counter the effects of

- noisy or sub-optimally annotated trust information accumulated in a MEC-local siloed dataset. The existence of such noisy data can cause a trust prediction model trained against it to produce unfavorable results.
- unbalanced datasets accumulated in geographically distributed MEC environments. For instance, an MEC environment hosting densely populated sensor service providers and consumers can accumulate trust information in abundance through their interactions, while an MEC environment with sparsely populated service providers and consumers may lack quality trust information degrading the performance of their respective trust prediction models.

In such cases, allowing different MEC environments to collaborate during trust prediction model training can allow them to exchange knowledge between each other and produce more accurate and statistically robust trust prediction models.

Furthermore, the *heterogeneous* characteristics of MEC environments give rise to multiple *context-environments* within a given MEC topology. This, in turns, results in *context-dependent* trust information being generated in these MEC environments. For instance, the network conditions (eg. network congestion), computing and storage resource availability, the use of hardware and software used in different MEC environments hosting homogeneous IoT services can change across different MEC environments [11]. This causes the quantified values of parameters that define trustworthiness, or in other words, trust properties, of an IoT service such as its QoS values and the extent to which a given IoT service can provide credible information may vary from one MEC environment to another. Therefore, the trust information generated by different MEC environments can be interpreted to form varying data distributions (i.e. non-Identically and Independently Distributed (non-IID)) across these *context-environments* formed within a MEC topology for trust evaluation that may not be completely identical to each other [6]. As a result, for a given type of homogeneous services operating from such heterogeneous operating environments, this calls for separate trust prediction models to be fit taking into account the *context-dependent* data distributions with different trust characteristics (i.e. non-IID trust information) accumulated in different MEC environments.

In addition, the context-dependent trust information accumulated in different MEC environments also calls for strategies to *efficiently and automatically capture the most influential properties that define IoT service trust* within a given MEC environment. For instance, in the era of big data coupled with the explosive growth of IoT, modern IoT systems can potentially generate trust information of higher dimensionalities [12] where each dimension is a property that seemingly contributes to IoT service trust. Being similar to the existing centralized IoT systems differing only by its unique distributed system architecture, the same holds true for MEC-based IoT systems as well. In reality, many such trust properties that exist in different MEC environments can force the prediction models to learn overly finer details and noise from the datasets used to train them, causing *overfitting*. Not only does this increase the complexity of the respective trust models but also limit their ability to perform against unseen data.

2) MEC topologies consisting of large numbers of MEC environments and accumulating high-volume trust information require trust prediction strategies that can withstand the scale: As of 2020, the total number of 5G enabled base stations in China and South Korea are known to be approximately 0.78 and 0.01 millions, respectively[1][2]. This demands trust prediction strategies that can scale well to heavily distributed MEC topologies with a large number of MEC environments, without a significant loss of accuracy or computational complexity required to train trust prediction models for comparatively smaller topologies.

In addition, most existing trust prediction strategies attempt to use *batch learning* architectures. They have primarily been developed under the assumptions that the entire training set is available prior to training the trust prediction models, and that it is all utilized during the process to train a trust prediction model once and for all. However, the high-volume trust information generated in MEC-based IoT systems inherently makes trust prediction strategies built on such an assumption computationally expensive. Therefore, trust prediction strategies suitable for MEC-based IoT systems are expected to be scalable in the face of high-volume trust information accumulated in distributed fashion.

3) Distributed trust prediction model training in MEC-based IoT systems requiring coordination and collaborative knowledge sharing amongst distributed models

1. https://www.statista.com/statistics/1119453/china-5g-base-station-number/

2. https://www.statista.com/statistics/1121538/south-korea-5g-base-stations-number/

should not cause excessive network stress: Even though MEC helps relieve the network stress on the core networks significantly, *the process of training trust prediction models across an inherently decentralized MEC topology can still burden the core networks*. For instance, distributed prediction strategies, many of which are iterative methods, often require some form of coordination and collaboration amongst the prediction models trained in distributed fashion for multiple purposes [13][14]. Some may require such coordination only to coordinate the iterations of the corresponding distributed prediction strategy or share metadata across the distributed prediction models trained. Other approaches, in the meantime, would encourage active collaboration amongst distributed trust prediction either directly or indirectly for knowledge sharing and clustering, etc. While collaboration among MEC environments can bring favorable results such as improved accuracy of the trained models, it requires communication amongst themselves across their network boundaries. Such communication contributes to the network stress on the core mobile networks. There is, however, an opportunity for inter-MEC communication avoiding the transmission of data through the core networks, most pragmatic MEC application models are likely to facilitate collaboration amongst MEC environments indirectly via the centralized cloud due to the complexities associated with existing inter-MEC communication models [8][15]. This requires such collaboration to take place via the core mobile networks, which bridges MEC environments to the centralized cloud. Therefore, trust prediction strategies suitable for such a setting should be communication-efficient and avoid overloading the core networks of mobile network providers.

To address the aforementioned limitations, we propose *a data-driven distributed and stochastic trust prediction model to evaluate the trustworthiness of homogeneous IoT services in heterogeneous MEC environments*. The specific contributions that address the challenges outlined previously are summarized below.

1) To address **challenge 1**,

- we model the trust prediction problem in MEC-based IoT systems as a distributed optimization (i.e. Network Lasso) problem over a set of non-IID distributions of context-dependent trust information;
- we propose a data-driven distributed trust prediction approach to evaluate trustworthiness of IoT services in MEC-based IoT systems. The proposed approach *automatically* filters out the most prominent trust features that define trust in IoT services within a given MEC environment. Hence, it avoids the need for manual exploration or additional pre-processing of data prior to learning trust models from data.

2) We propose a scalable parallel algorithm to train a distributed family of trust prediction models within MEC environments modelled as the network lasso problem, using Stochastic Alternating Method of Multipliers (S-ADMM), which

- efficiently processes the large volumes of trust information accumulated in MEC environments, and scale well to the potentially large number of MEC environments that form MEC topologies, to address **challenge 2**;
- causes lesser network stress on the core networks of mobile networking providers, to address **challenge 3**.

3) We report results of our exhaustive evaluation of the proposed approach carried out against the state-of-the-art distributed and centralized trust prediction approaches in the current literature.

The rest of the paper is structured as follows: Section 2 reviews the prior research our work builds on. Section 3 formally defines the problem setting we focused on, and conceptually models a mathematical framework for data-driven distributed and stochastic trust prediction in MEC-based IoT environments. Section 4 elaborates the proposed solution, which is an implementation of the mathematical framework modelled in Section 3. Section 5 comprehensively details out the experiments and evaluation of the proposed solution. Section 6 concludes our work and discusses possible future work.

2 Related Work

Trust evaluation in MEC based IoT systems was investigated in a handful of prior studies [16][17][18][19]. Out of which, [16] employs a probabilistic graphical model to predict trustworthiness of sensors in an IoT environment, and also proposes an algorithm to mobilize Mobile Edge Nodes (MENs) to reduce the energy consumption in the corresponding sensor topology. [19] proposes a similar approach based on crowdsourcing atop MEC. Meanwhile, [17] trains a distributed trust prediction model based on Network Lasso parallelized by ADMM to predict trustworthiness of MEC-based IoT services. Although [17] supports training context-dependent trust prediction models across distributed MEC environments in a given MEC topology, it fails to address two essential challenges outlined in Section 1. For instance, being an algorithm that runs in batch-mode, it lacks the ability to efficiently tackle the high-volume trust information accumulated across different MEC environments. In addition, it also lacks support for efficiently tackling high-dimensional trust information that can potentially be available within MEC environments in modern MEC-based IoT systems. On the other hand, [16] and [19] utilize mobility-oriented mobile edge nodes to collect and process trust information, which are characteristically different from the stationary MEC environments providing immovable computing and storage resources to interested IoT services and service consumers that we focus on in this work.

Furthermore, Trust evaluation in non MEC-based distributed settings is a well-studied problem in the existing literature. For instance, a trust evaluation approach for large-scale P2P systems based on multiple trust metrics was introduced in [20] whereas a more generic approach that can be applied to a variety of distributed network types had been introduced in [21]. Meanwhile, trust evaluation in distributed IoT environments was explored in [22][23]. However, the aforementioned approaches predominantly focus on direct node-to-node approach (where a node can be a sensor, smart-device or an agent in a multi-agent environment) for trust information dissemination and learn the reputation of the nodes operating in a distributed networked setting through which each node evaluates the trustworthiness of its peers. Such approaches are less-suitable for IoT service consumers as they typically do not engage in

direct communication with other similar service consumers operating in close proximity.

Meanwhile, trust in IoT systems, in general, is an actively researched domain. A trust model based on fuzzy theory was proposed for IoT systems consisting of a large number of networked sensors in [24]. In addition, social and QoS semantics of IoT networks were explored in the respective trust models proposed in [25][26]. However, they did not put enough emphasis into addressing the scalability requirements of trust in IoT, which is vital to realize the objectives of trust in such systems. This key limitation was addressed in [27]. Furthermore, a high-level layered framework was introduced in [28] to model trust in IoT systems. [29][30][31][32] used Support Vector Machines (SVMs) to model trust as a classification problem. In addition, using Artificial Neural Networks (ANNs) was also attempted in [33][34] in order to train a Multilayer Perceptron-based trust classifier using back-propagation. However, all the aforementioned approaches focussed on training a single global trust prediction model for centralized IoT systems, which are of limited applicability in an inherently decentralized setting such as an MEC-based IoT system. In addition, the majority of these models rely on domain experts to model trust, and also lack support for efficiently handling potentially high-dimensional trust information except for [32] and [29]. Therefore, it is apparent that none of the aforementioned state-of-the-art trust modelling approaches in the current literature can comprehensively address the challenges outlined in Section 1 together, which calls for a comprehensive and consolidated trust prediction approach for MEC-based IoT systems that can holistically address the previously introduced challenges are data-driven and less reliant on domain expertise.

3 Problem Formulation

In a typical IoT system, the trust between services and their consumers is composed of multiple distinct factors, which we identify as *trust features*. They typically include parameters that are indicative of the quality of the IoT services, quality of the experience expectations of the service consumers and misbehaving service providers amongst many others [35]. These trust features are combined in multiple different ways to come up with trust prediction models to predict trustworthiness of a given IoT service. The service consumers can then use these predictions to decide whether to take part in transactions with a given IoT service or not. In a *supervised setting*, we suppose a simple mathematical formulation could be derived, which can represent any such arbitrary trust prediction model, as below.

Given a set of arbitrary trust features organized into a vectorized form $x_i \in \mathbb{R}^d$ where d represents the dimensionality of x_i, the impact of each trust feature towards the overall trust value denoted as elements of a coefficient vector $w \in \mathbb{R}^d$, and a mapping function tr defining how each trust feature and their respective weight coefficients can be consolidated to come up with an overall trust value $\hat{y_i}$, any arbitrary trust model could be represented, as below.

$$\hat{y_i} = \text{tr}(x_i; w) \quad \text{where} \quad tr : \mathbb{R}^d \times \mathbb{R} \Rightarrow \mathbb{R} \tag{1}$$

Then, by applying the basic machine learning theory, and given a labelled trust dataset $X = (x_i, y_i), i = 1, \ldots, n$, the problem of deriving a supervised trust prediction model can be introduced as inferring the *best* set of values for w from X that minimizes the cumulative deficit between the observed trust y_i and output of $tr(x_i; w)$ against every training example in X. For mathematical convenience, let us denote the aforementioned deficit in the form of a loss function $f(w, \xi^i) = \ell(y_i, \hat{y_i})$ where $\xi^i = (x_i, y_i)$, or, in other words, a training example taken from X. Then, finding the best set of values for w can *generally* be defined in the form of the following loss minimization problem, where $\mathbb{E}$ denotes the expected value of the losses being minimized.

$$\underset{w \in \mathbb{R}^d}{\text{minimize}} \quad \mathbb{E}_\xi[f(w, \xi)] \tag{2}$$

Most existing machine learning based trust prediction models for IoT systems attempt to solve the problem (2) on top of datasets accumulated *centrally* in cloud-based data centres. It could be interpreted that most existing strategies assume that the trust information is generated from a single context-environment, or, in other words, a single trust region where the same trust characteristics are uniformly distributed. Therefore, they predominantly focus on deriving a single global model to predict trustworthiness of IoT services or(and) their providers. However, the aforementioned assumption is inherently obsolete and too restrictive in the context of MEC-based IoT environments, where each MEC environment could be thought of as a separate context-environment. Further, in a typical MEC topology, trust information generated from transactions between service producers and consumers is generally persisted and processed in an entirely decentralized manner within MEC-local data-centres. This demands us to re-formulate the problem (2) to fit into a distributed setting, as below.

$$[w_1^k, w_2^k, ..., w_M^k] = \underset{w_m \in \mathbb{R}^d}{\text{minimize}} \quad \sum_{m=1}^{M} \mathbb{E}_{\xi_m}[f_m(w_m, \xi_m)] \tag{3}$$

where MEC environments within a given MEC topology are indexed with $m \in (1, .., M)$; w_m denotes the parameters of the trust prediction model learnt at m^{th} MEC environment in response to the trust-based interactions between service providers and consumers described by ξ_m; f_m denotes the loss function used to learn a trust prediction model in m^{th} MEC environment under influence of ξ_m.

In a batch learning setting, a reasonable approximation to (3) can be obtained by letting each MEC environment use a sufficiently large set of independent training samples to minimize $f_i(w_i, \xi_i)$, as below.

$$[w_1^k, w_2^k, ..., w_M^k] = \underset{w_i \in \mathbb{R}^d}{\text{minimize}} \quad \sum_{m=1}^{M} \left(\frac{1}{N_m} \sum_{j=1}^{N_m} f_i(w_m; \xi_m^j) \right) \tag{4}$$

However, given the scale at which typical IoT systems operate in the era of big data, iterating over large batches of training samples to derive a sufficiently accurate solution to w_i can be significantly time and resource consuming [36]. To address the aforementioned challenge, we aim to adopt a *stochastic* approach in which one randomly chosen training sample corresponding to a trust-based transaction between

an IoT service and a consumer is used iteratively to derive and update a solution to w_m for each MEC environment. At each iteration k, we will be solving a minimization problem of the form,

$$[w_1^k, w_2^k, ..., w_M^k] = \underset{w_m^k \in \mathbb{R}^d}{\text{minimize}} \quad \sum_{m=1}^{M} f_m^k(w_m^k; \xi_m^k) \tag{5}$$

Solving a problem of the form (5) can inherently be componentized into multiple sub-problems, each of which can be solved in parallel and isolation within individual MEC environments. In other words, each sub-problem f_i in problem (5) can conveniently be solved at the i^{th} MEC environment in a given MEC topology on top of dataset accumulated within itself. In addition to the inherent parallelism enforced by the aforesaid approach, such a strategy also minimizes the movement of data towards the cloud layer This is particularly important as it significantly reduces the stress on the core networks of the mobile networking providers, which would have otherwise been triggered by most existing centralized cloud based trust prediction approaches.

In a typical MEC-based IoT system, sensor service providers and consumers can mobilize among adjacent MEC environments [9]. Not only that, it is also possible that same sensor provider or those that exhibit similar characteristics (e.g. sensors manufactured by a particular vendor under the same specification) can operate from MEC environments in close proximity to each other. As a result, we hypothesize that the MEC environments that are either adjacent or close to each other may accumulate similar trust information. Therefore, allowing such MEC environments to collaborate with each other may assist *similarly-poised* MEC environments derive more accurate trust prediction models by *borrowing strength* from each other (see Fig 2). At the same time, we are also interested in penalizing neighbouring prediction models that are differently-poised in order to prevent them from collaborating. Consequently, problem (3) can be further modified as the following *regularization problem* in which trust models carrying significantly different trust features are penalized while incentivizing those that carry similar trust features.

$$\underset{w_i \in \mathbb{R}^d}{\text{minimize}} \quad \sum_{i=1}^{M} f_i(w_i, \xi_i) \quad + \quad \sum_{i=1}^{M} \Big(\sum_{j \in N(i)} a_{ij}.g(w_i, w_j) \Big) \tag{6}$$

where g denotes a function that assesses the similarity of two prediction models trained by two adjacent MEC environments, (w_i, w_j) are parameters of those prediction models and a_{ij} a non-negative regularization parameter defined as $\{a_{ij} \in \mathbb{R} | a_{ij} = 0 \text{ if } w_i = w_j, a_{ij} > 0 \text{ if } w_i \neq w_j\}$.

However, the regularization term introduced in problem (6), in its default form, pollutes the parallelism enforced by problem (5) as it involves computing a global sum of the differences between the model parameters w_i learnt by MEC environments. These differences are computed in a pair-wise manner between two adjacent MEC environments, over the entire MEC topology. As a result, we can no longer easily isolate solving an independent sub-problem at the i^{th} MEC environment.

4 SOLUTION

This section provides a comprehensive overview of the proposed solution and the theoretical foundation upon which it is developed. Section 4.1 lays out an easy-to-comprehend summary of the proposed solution void of rigorous mathematical notations. We later expand this to provide more details in Section 4.3. Section 4.2 lists out the key precursors and their respective mathematical foundations that our solution depends on. Section 4.3, then, incrementally builds the proposed solution on top of the technical preliminaries introduced in Section 4.2 as well as comprehensively explores the networked communication structures of MEC-based IoT systems as well as their characteristics, and also explains how the proposed solution is effectively applied for trust prediction in MEC-based IoT systems in harmony of the aforesaid network communication structures.

4.1 Solution Overview

We propose a parallel and iterative stochastic algorithm with embedded feature selection for distributed data-driven trust prediction in MEC-based IoT systems. The proposed algorithm utilizes a two-tier hierarchical communication architecture with the global cloud and MEC environments forming its tiers (see Fig. 1). The information flows between the two aforementioned tiers over the backhaul links that exist between the global cloud layer and each distributed MEC environment.

In this proposed approach, the global cloud layer plays the role of a *facilitator* to perform two key important tasks, namely, 1) acting as a centrally available iteration coordinator for the proposed iterative algorithm. 2) maintaining a *logical* map of the predetermined links between each MEC environment and their neighbours derived based on proximity. 3) enforcing knowledge sharing among neighbors of a given MEC environment and computing other important metadata required by the proposed iterative algorithm. On the other hand, each distributed MEC environment trains an MEC-local trust prediction model atop its locally accumulated data, subjected to the knowledge shared of the neighbouring MEC environments by the global cloud layer.

Meanwhile, the proposed algorithm runs in six key steps, which are elaborated below.f

Step 1: First, each distributed trust prediction model initializes their MEC-local trust prediction models. At the same time, the global model coordinator initializes and shares the initial estimations of the models of the neighboring MEC trust prediction models, as well as other metadata required by an MEC environment to train its trust prediction model (see Fig. 1(a)).

Step 2: Then, each MEC environment executes their trust prediction model training strategy atop the locally accumulated data, subjected to the knowledge and metadata shared by the global model coordinator. In addition, determining the key features associated with the MEC-local trust prediction models is also done, simultaneously (see Fig. 1(b)).

Step 3: Once trained, the model parameters of the trained trust prediction models are, then, shared with the global model coordinator (see Fig. 1(c)).

Step 4: The global model coordinator, then, accumulates all the model parameters it has received from the all the

distributed MEC environments, and enforces knowledge sharing among neighboring MEC environments, and relevant other metadata is computed (see Fig. 1(d)).

Step 5: Once knowledge sharing and the computation of the metadata is done, the global model coordinator then shares the learnt knowledge and metadata associated with each neighboring MEC environments with a given MEC environment (see Fig. 1(e)).

Step 6: The procedure formed by steps 2) to 5) are then repeated iteratively until the algorithm converges to a user-defined application-specific error margin, at which point, we consider that the training of the trust prediction models is concluded and ready to be used.

4.2 Technical Preliminary

For completeness, we provide a brief systematic exposition below on the Network Lasso problem and Stochastic ADMM as well as euclidean projection onto the l-1 norm ball, which are the key precursors upon which the proposed solution is developed.

4.2.1 Network lasso problem

Network lasso is a framework to solve large-scale optimization problems formulated as a graph structure, allowing simultaneous clustering and optimization [37]. Given an undirected networked graph $G = (V, E)$ in which nodes are denoted by $V = \{1, \ldots, N\}$, and their connectivity with each other is denoted by the edges $E = \{(v_1, v_2) : v_1, v_2 \in V, v_1 \neq v_2\}$, the network lasso problem is mathematically expressed, as below.

$$\text{minimize} \quad \sum_{i \in V} f_i(w_i) + \lambda \sum_{(j,k) \in E} a_{jk} ||w_j - w_k||_2. \tag{7}$$

In this optimization problem, w_i represents model parameters of a loss function denoted as $f_i = \{(w_i, f(w_i)) : w_i \in \mathbb{R}^n\}$. Each loss function f_i defined over the input-output space $f_i : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ is local to a node $v_i \in V$ in the graph G. These loss functions are used to estimate model parameters of each node in the graph $v_i \in V$ by formulating an optimization problem. Meanwhile, λ is a regularization parameter that scales the edge objectives relative to the node objectives, a_{jk} represents an impact factor of a particular edge, i.e. (v_j, v_k), on the finite-sum problem computed over the loss functions of all nodes participating in the optimization problem. It is also noteworthy that w_j and w_k correspond to the parameters of the models associated with two adjacent nodes v_j and v_k in the graph, respectively. Furthermore, the regularization parameter λ and impact factor a_{jk} alongside the L^2-norm computed over the difference of model parameters between the two nodes connected by the edge (v_j, v_k) form a penalty factor. This compels the contrast between two connected nodes to be zero, strengthening the cohesion among those that carry similar model parameters (i.e. $w_j = w_k$).

4.2.2 Stochastic ADMM

Stochastic ADMM (S-ADMM) is a variant of the ADMM algorithm that promotes solving a linearly constrained regularized optimization problem iteratively under a stochastic setting. The specialty of S-ADMM over its predecessor ADMM stems from, at each iteration, w in problem (8) is updated based on one noisy sample drawn from the underlying dataset. The key advantage this brings in is that the updates are less time and resource consuming than that of the conventional batch based ADMM. S-ADMM does so by substituting a first-order approximation of the optimization goal $f(w; \xi)$ used in the augmented Lagrangian. S-ADMM algorithm primarily intends to take on problems of the type

$$\underset{w,z}{\text{minimize}} \quad f(w, \xi) + g(z) \quad \text{s.t.} \quad Aw + Bz = c \tag{8}$$

where $w \in \mathbb{R}^n$, $z \in \mathbb{R}^m$ represent two variables of dimensions n and m optimized independently by f and g, respectively. $A \in \mathbb{R}^{p \times n}, B \in \mathbb{R}^{p \times m}$ and $c \in \mathbb{R}^p$ represent two matrices and a vector defining the linear constraints associated with the problem (8) and p denotes the number of constraints. It is assumed that the functions denoted by $f(w, \xi)$ and $g(z)$ are typically convex [13].

To solve the constrained optimization problem (8) as an unconstrained optimization problem, the augmented Lagrangian associated with it $L_\rho(w, z, \mu)$ is first obtained, in which μ represents the dual variable or, in other words, Lagrange multiplier associated with the augmented Lagrangian [37]. Then, by applying dual-ascent iteratively, S-ADMM alternatingly minimizes $L_\rho(w, z, \mu)$ with the following steps.

$$w^{k+1} = \underset{w \in \mathbb{R}^n}{\operatorname{argmin}} L_\rho(w, z^k, \mu_k) \tag{9a}$$

$$z^{k+1} = \underset{z \in \mathbb{R}^m}{\operatorname{argmin}} L_\rho(w^{k+1}, z, \mu_k) \tag{9b}$$

$$\mu^{k+1} = \mu^k + \rho \nabla L_\rho(w^{k+1}, z^{k+1}, \mu) \tag{9c}$$

where ρ acts as a penalty parameter and step-size for the dual variable update carried out by sub-problem (9c) and ∇L_ρ represents the gradient of $L_\rho(w^{k+1}, z^{k+1}, \mu)$ with respect to μ.

When $f(w, \xi)$ and $g(z)$ are separable into multiple sub-problems, each solved over a partition of the training data population, the aforesaid iterations can be carried out to solve each sub-problem independently in parallel. The enhanced network lasso algorithm proposed in this work utilizes this exact behaviour to learn a family of trust prediction models over a potentially large topology of MEC environments as a distributed stochastic optimization problem.

4.2.3 Euclidean projection onto the l-1 norm ball

The most basic form of projection task we consider can be formally described as the following optimization problem.

$$\underset{w \in R^n}{\text{minimize}} \quad \|w - v\|_2^2 \quad \text{s.t.} \quad \|w\|_1 < z \tag{10}$$

where $w, v \in R^n$ and $z \in R$. If $\|v\|_1 \leq z$ then, the solution to problem (10) is $w = v$. Therefore, in this particular context, we assume that $\|v\|_1 > z$, so that, the optimal solution to the problem (10) lies on the boundary of the constraint set, and thus we can replace the inequality $\|v\|_1 \leq z$ with the equality constraint $\|v\|_1 = z$. The solution to the aforementioned optimization problem returns a sparser solution on w, which has only a few non-zero components corresponding to the most prominent features that are instrumental to representing the data distribution upon which w is derived.

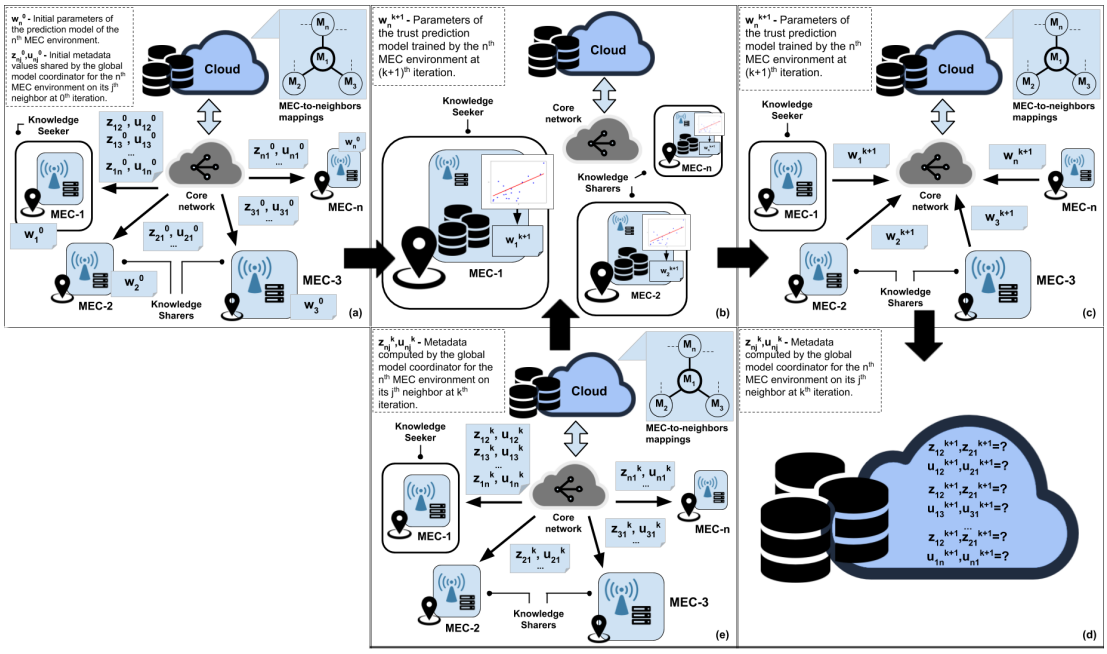

Fig. 1: A visualization of the information flow associated with the stochastic Network Lasso for MEC-based IoT systems.

4.3 Stochastic Network Lasso with feature selection for MEC-based IoT systems

We consider the problem (6) to be close to the general form of the Network Lasso framework [37]. However, the Network Lasso framework has originally been proposed for simultaneous clustering and optimization of a family of distributed prediction models in a *batch setting*. Our work (i.e. the problem setting (6) formulated in 3 differs from the original Network Lasso problem formulation in a way that it aims to tackle the challenge of *stochastically* fitting a distributed trust prediction models over the *context-dependent* data accumulated across MEC environments.

As introduced in Section 1, a typical MEC-based IoT system accumulates data originated from the transactions between IoT services and consumers within individual geographically distributed MEC environments. Therefore, training trust prediction models for each distributed MEC environment, closer to where the data originated and is kept, is paramount in terms of reducing the network stress on the core mobile networks. To achieve a similar goal, [37] proposed using ADMM to derive an algorithm, which runs in *batch-mode* to solve the Network Lasso problem in parallel, closer to where the data is persisted. We draw inspiration from this approach, and attempt to use S-ADMM to transform problem (6) to a form that can be solved in parallel, as below.

$$\text{minimize} \sum_{i \in V} f_i(w_i, \xi_i) + \lambda \sum_{(j,k) \in E} a_{jk} ||w_j - w_k||_2. \quad (11)$$

where $f_i(w_i, \xi_i)$ is a time-varying loss function, and the rest of the parameters remain the same as they were in (7).

The S-ADMM framework is predominantly applied to the problems of the form (8), which are bounded by some arbitrary linear constraints. Therefore, to convert the problem (11) into a form, which allows us to apply S-ADMM, copies of both w_i and w_j (i.e. z_{ij} and z_{ji}, respectively) are introduced at every edge (v_j, v_k) in the graph as per [37]. The transformed problem can now be viewed as,

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in V} f_i(w_i, \xi_i) + \sum_{(j,k) \in E} a_{jk} ||w_j - w_k||_2. \\ \text{s.t.} \quad & w_i = z_{ij} \end{aligned} \quad (12)$$

Now, by applying S-ADMM against (12), we then decompose the aforementioned optimization problem into three sub-problems that can be iteratively solved in parallel, as below.

$$\begin{aligned} w_i^{k+1} = \underset{w_i}{\text{argmin}} \Big(& w_i^T . f_i'(w_i^k, \xi_i^{k+1}) \\ & + \sum_{j \in N(i)} \frac{\rho}{2} \|w_i - z_{ij}^k + \mu_{ij}^k\|_2^2 + \frac{\|w_i - w_i^k\|_2^2}{2\eta} \Big) \end{aligned} \quad (13a)$$

$$\begin{aligned} z_{ij}^{k+1}, z_{ji}^{k+1} = \underset{z_{ij}, z_{ji}}{\text{argmin}} \Big(& a_{ij} \|z_{ij} - z_{ji}\|_2 \\ & + \frac{\rho}{2} (\|w_i^{k+1} - z_{ij} + \mu_{ij}^k\|_2^2 \\ & + \|w_i^{k+1} - z_{ji} + \mu_{ji}^k\|_2^2) \Big) \end{aligned} \quad (13b)$$

$$\mu_{ij}^{k+1} = \mu_{ij}^k + (w_i^{k+1} - z_{ij}^{k+1}) \quad (13c)$$

where, μ is a scaled dual variable used for mathematical convenience, $\rho (> 0)$ is the penalty parameter, while k denotes the k^{th} time the above steps were run.

Out of these sub-problems, problem (13a), or in other words, the w_i-update performed at each distributed agent concerns training a prediction model over its locally accumulated data. Next, to enforce feature selection, or in other words, obtain a sparser solution that better fits the context-dependent trust prediction in a given MEC environment, we

then apply (10) onto the optimal solution returned at each iteration, w_i^{k+1}, as below.

$$w_i^{k+1} = \underset{w_i \in R^n}{\text{minimize}} \ \|w_i^{k+1} - v_i\|_2^2 \quad \text{s.t.} \ \|w_i^{k+1}\|_1 < z_i \tag{14}$$

Even though deriving a parallel algorithm to solve the problem (6) is of utmost importance, that alone is unlikely to help us completely realize the goals announced in Section 1. Therefore, it is also important to analyze how the above solution harmonizes with the network architecture of MEC, as well. Typically, individual MEC environments tend to operate independently of each other at the application layer avoiding backhaul and core network layers. However, recent work exploring the synergy between the MEC and cloud layers promises pathways towards better quality services to the end users [15]. This makes it a compelling case to fully utilize the centralized cloud in our work, which all MEC environments possibly connect to, in order to establish a *logical* MEC network allowing them to communicate indirectly with each other (see Fig. 2). In such a setting, each MEC environment can be indirectly connected to multiple neighbouring MEC environments via the centralized cloud for collaborative training of trust prediction models. There are multiple ways we can model a network of MEC environments exploiting various characteristics and properties of different application contexts [8]. However, for simplicity, we used a simple model where each MEC environment connects to a set of other MEC environments determined based on proximity. The problem (6) can then be modeled over the graph resulting from this topology (see Fig. 2).

In a typical MEC network topology such as the one depicted in Fig. 2, the sub-problems denoted by (13) can be solved iteratively at different layers between local MEC and the centralized cloud infrastructures (see Fig. 1) with a six-step procedure, as elaborated below. The mathematical representations associated with each of these six-steps have also been consolidated into an algorithmic framework and depicted in Algorithm 1. We also provide pointers to the reader to better map each following step into their corresponding mathematical representations in Algorithm 1, where appropriate.

Step 1: To begin training the trust prediction models, we first initialize each MEC-local trust prediction model corresponding to an individual MEC environment first (see Lines [2-3]). As noted in (13), the (13a) or in other words, the w-update, depends on the values z_{ij}^k and μ_{ij}^k, that were generated in the previous iteration (i.e. k) of this six-step procedure (see Line 6 in Algorithm 1). We will later see in Step 3 and Step 5 that the aforementioned values are generated and cached within the centralized cloud layer by the global model coordinator. However, since this is the beginning of the trust prediction model training process and there is no previously executed iteration of the underlying six-step procedure, the global model coordinator initializes and shares a set of initial estimations (as opposed to sharing the values from the previous iterations) of the models of the neighboring MEC trust prediction models (see Fig. 1(a)).

Step 2: Then, each MEC environment executes their trust prediction model training strategy atop the locally accumulated data, subjected to the knowledge and metadata (i.e. z_{ij}^k and μ_{ij}^k) shared by the global model coordinator (see Line 6 in Algorithm 1). In addition, determining the key features associated with the MEC-local trust prediction models is also done (see Fig. 1(b) and Line 7 in Algorithm 1). This not only helps determine the most influential parameters that the context-dependent IoT service trust of a given MEC-environment depends on, and also avoids *overfitting* of the trained MEC-local trust prediction model on the slack parameters that might exist in each MEC environment.

Step 3: Once trained, the model parameters of the trained trust prediction models are, then, synchronized with the global model coordinator (see Fig. 1(c) and Line 8 in Algorithm 1).

Step 4: Next, the global model coordinator accumulates all the model parameters received from all the distributed MEC environments. It, then, enforces knowledge sharing among neighboring MEC environments (i.e. z_{ij}^{k+1} and z_{ji}^{k+1}), and other relevant metadata (i.e. μ_{ij}^{k+1} and μ_{ji}^{k+1}) is computed (see Fig. 1(d) and Lines [9-11] in Algorithm 1). This is carried out by executing (13b) and (13c) of the three sub-problems denoted by problem (13).

Step 5: Once knowledge sharing and the computation of the metadata is done, the global model coordinator then shares the learnt knowledge and metadata (i.e. z_{ij}^{k+1} and μ_{ji}^{k+1}) associated with each neighboring MEC environment back with a given MEC environment (see Fig. 1(e) and Line 10 in Algorithm 1).

Step 6: The procedure formed by steps 2) to 5) are then repeated iteratively (see Line 4 in Algorithm 1) until the algorithm converges to a machine learning practitioner-defined application-specific error margin, at which point, we consider that the training of the trust prediction models is concluded, and the corresponding trust prediction models are ready to be used.

It is noteworthy that this enhanced network lasso algorithm for MEC-based IoT systems preserves the convergence properties it inherits from its parent models (i.e. ADMM, and network lasso). In other words, the above proposed algorithm converges to a global optimum in all possible cases as the underlying problem it attempts to solve is convex. In addition, as with many practical optimization problems, the more iterations the algorithm runs for, the more slowly it reaches precision accuracy. For use-cases where precision accuracy can be traded off against faster convergence time, primal and dual residuals provide a tuner to control the desired outputs.

We used a soft-margin Support Vector Machine (SVM) [38] as the reference implementation of our network lasso based machine learning architecture for MEC-based IoT environments. SVM had already been widely used and shown to work well in prior trust research for developing classification- and regression-based trust prediction models [29][31]. In fact, [5] proposed an SVM based classification model for predicting trustworthiness in IoT services as well, which aligns quite well with the primary scope of this study. This background provided us with a rational basis to adopt SVM as the local trust prediction problem to be solved as part of each sub-task running in the local MEC layers of the reference implementation. In that, each local MEC environment trains its own SVM-based binary classifier to

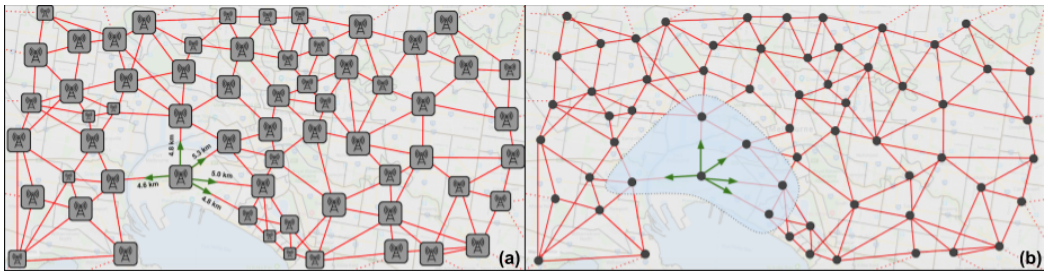

Fig. 2: A hypothetical deployment of MEC environment, which shows how the neighbouring MEC environments are linked based on proximity forming a partial mesh network.

Algorithm 1 Stochastic network lasso for MEC-based IoT systems

parameters: M-MEC environments, E-Links among MEC environments, ρ-Penalty parameter
for all $m \in M$ **do** ▷ Loop over MECs in Cloud layer
  Send initial z_{ij}, z_{ji} and μ_{ij} to m
for $k = 1$ to K (or until convergence) **do**
  for all $m \in M$ **do** ▷
  Distributed loop over MECs in parallel
    $w_i^{k+1} \leftarrow \underset{w_i}{\text{argmin}} \Big(w_i^T . f_i'(w_i^k, \xi_i^{k+1}) + \sum_{j \in N(i)} \frac{\rho}{2} \|w_i - z_{ij}^k + \mu_{ij}^k\|_2^2 + \frac{1}{2\eta} \|(w_i - w_i^k\|_2^2 \Big)$
    $w_i^{k+1} \leftarrow \underset{w_i^{k+1}}{\text{argmin}} \|w_i^{k+1} - v_i\|^2 s.t. \|w_i^{k+1}\| < z_i$
    Send w_i^{k+1} to cloud layer
  for all $e \in E$ **do** ▷
  Loop over all edges among MECs, in cloud
    $z_{ij}^{k+1}, z_{ji}^{k+1} \leftarrow \underset{z_{ij}, z_{ji}}{\text{argmin}} \Big(a_{ij} \|z_{ij} - z_{ji}\|_2 + \frac{\rho}{2} (\|w_i^{k+1} - z_{ij} + \mu_{ij}^k\|_2^2 + \|w_i^{k+1} - z_{ji} + \mu_{ji}^k\|_2^2) \Big)$
    $\mu_{ij}^{k+1} \leftarrow \mu_{ij}^k + (w_i^{t+1} - z_{ij}^{k+1})$

predict untrustworthy IoT services. Each trained classifier classifies an input as either "benign" or "harmful" (denoted by "1" and "-1" respectively) indicating whether the IoT service in concern is trustworthy or not.

In a batch setting, the task of obtaining the optimal separating hyperplane of the underlying soft-margin SVM, which separates the two classes that the classifier is trained for can be formally modelled as a minimization problem, as below [38].

$$\underset{w \in \mathbb{R}^d}{\text{minimize}} \quad \frac{\lambda}{2} \|w\|_2^2 + \frac{1}{N} \sum_{i=1}^{N} max(0, 1 - y_i x_i^T . w + b). \tag{15}$$

where w denotes a weight vector corresponding to the model parameters to be learnt, x_i identifies the feature vector associated with i^{th} training sample fed into the learning model, y_i corresponds to the label associated with it and b is a bias. However, in a typical stochastic setting, we are primarily interested in evaluating one randomly drawn sample at each iteration of the stochastic network lasso algorithm. This gives rise to the following modified version of the problem (15).

$$\underset{w \in \mathbb{R}^d}{\text{minimize}} \quad \frac{\lambda}{2} \|w\|_2^2 + \max(0, 1 - y_i x_i^T . w + b) \tag{16}$$

5 Evaluation

This section is divided into four parts. Section 5.1 describes the experiments designed to evaluate the suitability of the proposed approach to address the challenges outlined in Section 1. Section 5.2 provides a technical summary of the state-of-the-art approaches that the proposed model was compared against while Section 5.3 briefly describes the datasets used. Finally, Section 5.4 shows and discusses the results of the experiments carried out. The concrete implementation of the proposed approach (i.e. represented by Algorithm 1) used for the evaluation is available at: https://github.com/prabathabey/mec-trust

5.1 Experiments

We conducted a series of experiments to comprehensively and empirically evaluate the following key aspects of the proposed trust prediction strategy. These key aspects we focused on together with the experiments designed to evaluate them are as follows.

1) Convergence properties of the proposed trust prediction strategy: The necessary and sufficient optimality conditions for the ADMM framework to converge include two key criteria, namely,

- **Primal feasibility**: making sure that the quantity $r^{k+1} = Aw^{k+1} + Bz^{k+1} - c \rightarrow 0$ as $k \rightarrow \infty$.
- **Dual feasibility**: making sure that the quantity $s^{k+1} = \rho A^T B(z^{k+1} - z^k) \rightarrow 0$ as $k \rightarrow \infty$.

where k represents the iterations of the algorithm. Therefore, the convergence of the proposed stochastic Network Lasso based trust prediction strategy was predominantly assessed by observing if the primal and dual residuals - denoted by r^k and s^k, respectively - are converging to a given error bound via a decreasing sequence of *errors* over a finite number of iterations carried out sequentially across all three datasets outlined in Section 5.3 to provide evidence on the generality of the said convergence properties of the proposed algorithm.

For different values of a_{ij}, r^k and s^k were monitored over many sequential iterations of the Algorithm 1, until a

pre-configured stopping criterion was reached. This stopping criterion was set using two parameters, absolute (0.01) and relative (0.001) error tolerances of the algorithm based on the guidelines provided in [39]. In addition, to tune ρ, we also adopted the *varying penalty parameter* strategy, which not only improves the rate of convergence, but also ensures that the convergence as well as the performance of the proposed algorithm rely less on the initial choice of ρ [13].

2) Effectiveness of the proposed approach to predict trustworthiness of IoT services in MEC-based IoT systems: These experiments were aimed at comparing the performance of our approach against the state-of-the-art trust evaluation methods outlined in Section 5.2 and justifying the ability of the proposed method to address the key challenges in predicting the trustworthiness of MEC-based IoT services outlined in Section 1. The aforementioned experiments are organized into the following categories.

- **Effectiveness of the context-dependent and data-driven trust prediction (Challenge 1)**: This particular experiment was designed to evaluate the ability of the proposed approach to address two seemingly related key aspects in predicting the trustworthiness of IoT services in MEC-based IoT systems outlined in Section 1 in relation to Challenge 1. As part of it,
  1) To assess *the ability to effectively tackle context-dependent trust prediction*, we compared the results of our approach against both state-of-the-art trust prediction strategies that promote context-dependent trust prediction in a distributed setting as well as centralized approaches that assume the entire MEC topology to be a single global context-environment for trust prediction.
  2) To assess *the effectiveness of the data-driven trust prediction* component of the proposed approach and its impact on the overall performance, we compared the performance results obtained in the previous step against a variant of our approach with feature selection disabled. To independently evaluate the effectiveness of the data-drivenness of our approach thereby facilitating a fair comparison, we disabled the ability of our approach to transfer knowledge among trust prediction models trained by neighboring MEC environments by setting a_{ij}s in Algorithm 1 to be 0. Setting $a_{ij} = 0$ forces our proposed approach to train a family of non-collaborative binary SVMs across the simulated MEC topology.

  In addition, we carried out further experiments to assess *the impact of knowledge sharing via collaboration of MEC environments* on the overall performance of our approach. For that, we re-enabled the knowledge sharing component by allowing a_{ij}s in Algorithm 1 to be > 0 and compared its performance against the other state-of-the-art distributed trust evaluation approaches selected in Section 5.2. In addition, the performance was also compared against the non-collaborative MEC-local binary SVMs trained in the third step, as well. To provide further evidence on knowledge sharing, we provide data on the *degree of consensus* among trust prediction models trained by different MEC environments, computed by examining the resulting trust prediction model structures as well. If two neighboring models end up adapting the same trust prediction model, we call the two respective models to be *in consensus*.
- **Ability to tackle scalability (Challenge 7):** This experiment was designed to evaluate the ability of our approach to address the two key scalability criteria outlined in Section 1 in relation to Challenge 2.
  1) To assess *the ability to scale well to growing topology sizes*, we monitored the average prediction accuracy across all distributed trust prediction models in a given MEC topology as well as the average number of communication rounds required till convergence when the number of MEC environments in the underlying MEC topology is gradually increased. The other non-distributed state-of-the-art models were left out from this experiment as they use only a single global model that does not scale across a given MEC topology.
  2) To assess *the ability to efficiently process large datasets accumulated across MEC environments*, we monitored and compared the total time taken to train the trust prediction models by each compared state-of-the-art trust prediction models.
- **Ability to cause less network stress on the core networks of mobile network providers (Challenge 5):** To evaluate this, the number of *rounds of communication* needed during the end-to-end process that includes trust information accumulation and prediction model training between the centralized cloud and distributed MEC layers was measured and analysed. Here, the aforementioned metric takes into account the number of times the data has been transmitted between the MEC environments and centralized cloud layer during the end-to-end process that spans across data accumulation as well as trust prediction model training. Therefore, it was assumed to be indicative of the network stress on core mobile networks of mobile network providers.

5.2 Models Compared

We compared our approach against a comprehensive set of state-of-the-art machine learning-based trust prediction models proposed against several application domains. The details of their simulations used in our experiments are described below. Furthermore, TABLE 1 provides a concise comparison of features of the evaluated models against the trust prediction expectations highlighted in Section 1.

Abeysekara et al. [17]: To evaluate and compare this approach, which is seemingly similar to our approach, an identical experimental set-up and datasets were used. However, batch Gradient Descent (GD) was used to train each MEC-local trust prediction model as opposed to Stochastic Gradient Descent (SGD) used in our approach.

MTES [16]: To facilitate a fairer comparison, we considered a scenario where the MENs (which are computational nodes equivalent to MEC environments in our context) are stationary. In addition, we also made a key assumption that rather than MENs pinging sensor nodes for information, the sensor nodes publish statistics derived from transactions

between other sensor nodes to MEC environments. Then, to evaluate the performance of the two models, we reshaped the datasets described above, to simulate a sensor network publishing information into a topology of MEC environments.

Jayasinghe et al. [5]: A binary SVM with a Radial Bias Function (RBF) kernel was used to train a global trust prediction model for IoT services based on five key trust features via k-fold (k=5) cross validation, as recommended in [5]. Principal Component Analysis (PCA) was, then, used to filter the most influential features. To avoid possible hidden optimization routines and approximations thereby allowing a fairer comparison, we implemented our own version of the SVM algorithm with an RBF kernel extending the problem (16).

Lopez et al. [29]: A three-class SVM with an RBF kernel was used to train a trust prediction model for a given *trust context* as recommended in [29] via k-fold (k=5) cross validation. Meanwhile, the hyper-parameters of the SVM (i.e. λ and C in problem (16)) were optimized via a grid-search by incrementing the parameter values with a step size of $2^{0.25}$. We used our own implementation of an SVM with an RBF kernel to allow a fairer comparison.

5.3 Datasets

For the experiments described above, we used multiple public IoT datasets in our simulations. A comprehensive overview of the structure of these datasets is given below.

UNSW-NB15[3]**:** This dataset consists of transaction data (each containing 49 numerical and categorical features, i.e. $\in \mathbb{R}^{49}$) traced from a simulated intrusion detection system (IDS). Each record in the dataset corresponds to a transaction indicating either a benign behaviour and or one of nine types of attack scenarios. We labelled each sample as *benign* or *harmful* based on whether they correspond to a benign or attack scenario. This dataset was first normalized and then divided into 100 randomly-sized ($n \in [200, 2000]$) smaller datasets forming an aggregate of 110892 examples.

Bot-IoT[4]**:** This dataset consists of labelled data that can be used for network forensic analysis, developed on a realistic testbed environment [40]. It carries records ($\in \mathbb{R}^{49}$) corresponding to both legitimate and simulated IoT network traffic, as well as ones that are representative of multiple types of botnet attacks such as, probing, denial-of-service and information-theft attacks. We consolidated and relabelled all records related to malicious activity as *harmful*, and those that represent legitimate network traffic as *benign*. The resulting dataset was first normalized and then divided into 100 randomly-sized ($n \in [1000, 20000]$) smaller datasets forming an aggregate of [need to fill] examples.

N-BaIoT[5]**:** This dataset consists of network traffic data previously used to detect Mirai and BASHLITE attacks within an IoT setting [41]. Under each family of attacks, there were multiple individual attack types of which the records ($\in \mathbb{R}^{115}$) were consolidated under the label *harmful*. In addition, the records related to legitimate network traffic ($\in \mathbb{R}^{115}$) were classified under the label *benign*. The resulting dataset was normalized and split into 100 randomly-sized ($n \in [1000, 20000]$) smaller datasets for each simulated MEC environment.

Random noise was also added to each dataset via flipping the labels of randomly picked samples to mimic a Non-IID dataset. The resulting datasets (with a training-to-test split ratio of 70:30) were used to train the distributed prediction models for each simulated MEC-environment.

5.4 Results and Discussion

This section provides comprehensive details on the results observed at each experiment we carried out and sufficient reasoning as well as justification on how they demonstrate the proclaimed benefits of the proposed approach.

5.4.1 Convergence properties of the proposed trust prediction strategy

The results obtained for both primal and residuals of the proposed algorithm atop the three datasets used did show that the algorithm converged to a relatively stationary point *iteratively* for all values of λ (see Fig. 3) over multiple iterations of the Algorithm 1. A more granular look at the convergence rates of the residuals r^k and s^k can be seen in Fig. 4 against an arbitrarily picked value of λ (=0.002). This affirms the convergence of the algorithm, and therefore, confirms that the proposed approach is algorithmically robust.

Furthermore, inheriting the properties of ADMM, the proposed solution based on S-ADMM converges to a modest accuracy fast, and high accuracy slowly [13], [37]. In addition, the fact that S-ADMM uses one noisy sample for gradient updates while training each distributed MEC-local prediction model, its convergence can further slow down as evident from our results depicted in TABLE 4. In our problem context, the proposed solution was observed to attain significantly higher accuracy compared to most baseline models it was evaluated against, at a lower time and communication cost (see TABLE 2).

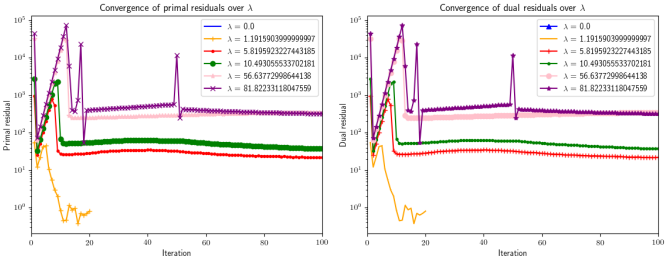

Fig. 3: Convergence of primal (r^k) and dual (s^k) residuals against different values of λ (in log scale).

5.4.2 Effectiveness of the proposed approach to predict trustworthiness of IoT services in MEC-based IoT systems

Results of the experiments affirmed that the proposed stochastic Network Lasso-based distributed trust prediction

3. https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/
4. https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/bot_iot.php
5. https://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT

Feature	Proposed approach	MTES	Abeysekara et al.	Jayasinghe et al.	Lopez et al.	Global SVM	Local SVMs
Data-driven trust evaluation	✓	✗	✗	✓	✓	✓	✓
Distributed trust evaluation model training at the *edge* close to where the data originates	✓	✓	✓	✗	✗	✗	✓
Distributed and context-dependent trust modelling	✓	✗	✓	✗	✗	✗	✓
Knowledge sharing among trust evaluation models	✓	✗	✓	✗	✗	✗	✗
Scales to high-volume trust information	✓	✗	✗	✗	✗	✓	✓
Scales to MEC topologies with large number of MEC environments	✓	✓	✓	✗	✗	✗	✓

TABLE 1: Comparison of features among the proposed approach and other state-of-the-art models evaluated.

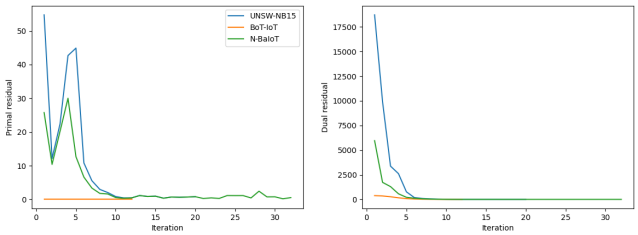

Fig. 4: A snapshot of the convergence of primal (r^k) and dual (s^k) residuals.

model lived up to the expectations set forth in Section 1. Below, we discuss the results gathered against the experiments that evaluated the suitability and effectiveness of the proposed approach to predict trustworthiness of IoT services in MEC-based IoT systems.

Effectiveness of context-dependent and data-driven trust prediction:

1) Effect of context-dependent trust prediction: The results of our experiments showed that that the context-dependent binary SVM classifiers trained by the proposed approach consistently outperformed all the state-of-the-art trust evaluation strategies that promote context-dependency across MEC environments, *except Abeysekara et al.'s* on all three datasets used for the experiments (see TABLE 3). However, the approach proposed by Abeysekara et al. [17], showed marginally better results by 3.1%, 4.2% and 3.8% on UNSW-NB15, Bot-IoT and N-BaIoT datasets, respectively.

We explain this behaviour by taking into consideration that the approach proposed by Abeysekara et al. [17] uses a standard Gradient Descent (GD) solver to solve problem (13a) in each distributed MEC environment as elaborated in Step 2 of the Section 4.3. The standard GD, which runs in batch-mode, inherently takes each and every example in the dataset used for training the underlying prediction model. As a result, the distributed trust prediction strategy proposed by Abeysekara et al. [17] produces more robust updates during the phase of the model training process that is equivalent to the Step 2 of Section 4.3, at each iteration.

2) Effect of data-driven trust prediction: The results of our experiments that compared the proposed approach against a variant of it with feature selection disabled showed that the former produced significantly better results. In other words, the proposed approach *with* feature selection enabled showed 40% more accuracy compared to the one *without* it against all λ values the two models were executed against. Given the two aforementioned approaches were evaluated under an identical experimental settings, we attribute this performance difference to the ability of the proposed approach to select only *the most explicable features* that represent context-dependent trust in different MEC environments via the feature selection, in comparison to the variant of the proposed approach that only uses the default S-ADMM (without feature selection) to solve the trust prediction problem formulated in Section 3.

3) Effect of knowledge sharing: The average prediction accuracy recorded (over multiple rounds of experiments against 100 simulated MEC environments) for the collaborative SVMs trained by the proposed approach and the non-collaborative local SVMs showed 15.29%, 15.47% and 0.33% higher accuracy against the UNSW-NB15, BoT-IoT, N-BaIoT datasets, respectively (see TABLE 2). The fact that both collaborative and non-collaborative SVMs were run under identical environmental settings, the above accuracy gain of the collaborative SVMs can be attributed to the effect of collaboration through knowledge sharing enforced by the proposed approach.

Type	Model	UNSW-NB15	BoT-IoT	N-BaIoT
Batch	Local SVMs	82.17	50.23	73.69
	Global SVM	95.12	53.34	73.56
	Abeysekara et al.	100	61.89	76.7
	MTES	65.54	41.47	53.98
	Jayasinghe et al.	90.58	55.49	69.04
	Lopez et al.	78.12	47.86	59.54
Stochastic	**Proposed approach w/o feature selection**	**94.2**	**55.46**	**69.53**
	Proposed approach	**97**	**59.42**	**73.93**

TABLE 2: Average prediction accuracy (%) of the evaluated models running in batch and stochastic modes atop UNSW-NB15, N-BaIoT and Bot-IoT datasets with the number of MEC environments in the underlying MEC topology kept at 100.

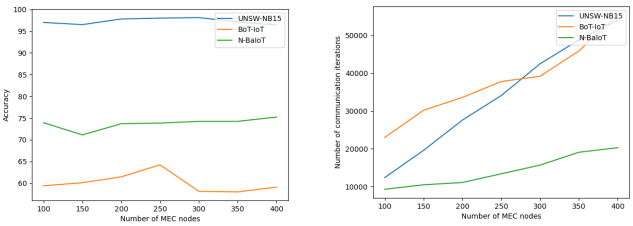

Fig. 5: Average accuracy and number of rounds of communication taken to converge over MEC topologies of different sizes.

Ability to tackle scalability:

1) Ability to scale well in the face of growing topology sizes: The experiments on the scalability of the proposed approach revealed noteworthy results, as well. As depicted in Fig. 5, the average accuracy of the entire MEC topology remained relatively stable as the number of MEC environments in the underlying MEC topology was increased,

across all the three datasets used. On the other hand, the number of rounds of communication needed till convergence was observed to rise relatively linearly as the number of MEC environments in the underlying MEC topology was increased, as well. This indicates that the number of rounds of communication is rather independent of the number of training examples accumulated in each MEC node, but the total number of MEC environments in the underlying MEC topology. Further, it also affirms that the number of communication iteration does not exponentially increase when the number of MEC environments of the MEC topology is increased, which can be considered a favourable result.

Dataset	Model	No. of MEC environments						
		100	150	200	250	300	350	400
UNSW-NB15	Abeysekara et al.	100	100	99.5	100	100	100	100
	MTES	65.54	60.12	68.85	65.22	63.28	65.46	65.52
	Lopez et al.	78.12	79.5	77.27	70.87	78.66	78.39	80.12
	Proposed approach	**97**	**96.12**	**99.12**	**99.8**	**100**	**100**	**100**
N-BaIoT	Abeysekara et al.	100	100	99.5	100	100	100	100
	MTES	65.54	60.12	68.85	65.22	63.28	65.46	65.52
	Lopez et al.	78.12	79.5	77.27	70.87	78.66	78.39	80.12
	Proposed approach	**97**	**96.12**	**99.12**	**99.8**	**100**	**100**	**100**
Bot-IoT	Abeysekara et al.	61.89	63.95	63.52	68.11	68.89	61.89	73.21
	MTES	41.47	53.98	43.21	42.3	40.14	45.97	45.21
	Lopez et al.	47.86	45.32	46.15	47.86	47.86	47.86	47.86
	Proposed approach	**59.42**	**63.12**	**64.22**	**60.12**	**60.64**	**66.34**	**65.12**

TABLE 3: Average prediction accuracy (%) of the evaluated distributed trust prediction models atop UNSW-NB15, N-BaIoT and Bot-IoT datasets with the number of MEC environments in the MEC topology gradually increased.

2) Ability to efficiently handle large trust datasets: The results gathered from the experiments on the time taken for a given iteration in the proposed approach against that of Abeysekara et al.'s showed that our approach completed a given iteration significantly faster (see TABLE 4). On average, within the first 10 iterations of the two algorithms compared had taken 20 and 514 milliseconds to complete a single iteration whereas, on average, 254 and 2135 milliseconds were taken for every other iteration.

However, the accuracy improvement gained by Abeysekara et al. comes at a significant computational cost (measured as a function of time taken for a given iteration) as shown in TABLE 4. In fact, the approach proposed by Abeysekara et al. was observed to take 2500% more time to complete a single iteration within the first 10 iterations of their respective algorithm, and 841% for every other iteration. Such a significant computational cost can be inherently prohibitive in the context of trust prediction in modern IoT systems, which accumulate information from the transactions between IoT services and their consumers in exorbitant volume and at rapid velocities.

Ability to cause less network stress on the core networks of mobile network providers: The results of our experiments revealed that the network stress imposed by the proposed approach is significantly less than that of the global SVM-based baseline model we used in our evaluation as well as the one proposed by Jayasinghe et al, which were trained under an identical environmental setting (see TABLE 4).

In our experiments, the communication-efficiency was only compared between the proposed approach, a global SVM-based baseline model and the model proposed by Jayasinghe et al. [32]. This is because none of the other models required the data to be transmitted out of the network boundaries within which the data was accumulated and model training took place.

Dataset Type	Model	UNSW-NB15		BoT-IoT		N-BaIoT	
		Cost	Time(ms)	Cost	Time(ms)	Cost	Time(ms)
Batch	Global SVM	110892	N/A	452023	N/A	98245	N/A
	Jayasinghe et al.	110892	N/A	452023	N/A	98245	N/A
	Abeysekara et al.	2400	370.14	8600	609.39	1800	351.18
Stochastic	**Proposed approach**	**12400**	**44.86**	**23000**	**61.14**	**9300**	**42.23**

TABLE 4: The number of rounds of communication between MEC and centralized cloud layer observed at the point of achieving the maximum accuracy and the total time taken (in milliseconds) evaluated in an MEC topology of 100 simulated MEC environments.

6 Conclusion and Future Work

This paper proposes a scalable and distributed machine learning architecture based on Stochastic Alternating Direction Method of Multipliers (S-ADMM) for trust prediction in MEC-based IoT systems. We made a proposition that training a distributed trust prediction model in an MEC-based IoT system could be interpreted as a distributed convex optimization problem. In addition, we also discussed the possibility of adopting the Network Lasso framework parallelized by S-ADMM as a solution. Finally, the feasibility of our approach was affirmed via simulated experiments.

Our future work includes investigating the adoptability of other distributed optimization frameworks such as Federated Learning in the concerning problem setting, and rigorously evaluating the resulting models against the proclaimed features and performance of the proposed approach to be able to understand their suitability for predicting trustworthiness of MEC-based IoT sensor services.

References

[1] Z. Yan, P. Zhang, and A. V. Vasilakos, "A survey on trust management for internet of things," *Journal of network and computer applications*, vol. 42, pp. 120–134, 2014.

[2] J. Daubert, A. Wiesmaier, and P. Kikiras, "A view on privacy and trust in iot," in *Communication Workshop (ICCW), 2015 IEEE International Conference on*, Conference Proceedings, pp. 2665–2670.

[3] L. Xie, Y. Ding, H. Yang, and X. Wang, "Blockchain-based secure and trustworthy internet of things in sdn-enabled 5g-vanets," *IEEE Access*, vol. 7, pp. 56 656–56 666, 2019.

[4] Y. Wang, "Trust quantification for networked cyber-physical systems," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2055–2070, 2018.

[5] U. Jayasinghe, G. M. Lee, T.-W. Um, and Q. Shi, "Machine learning based trust computational model for iot services," *IEEE Transactions on Sustainable Computing*, 2018.

[6] Y. Wang, R. Chen, J.-H. Cho, A. Swami, Y.-C. Lu, C.-T. Lu, and J. J. Tsai, "Catrust: Context-aware trust management for service-oriented ad hoc networks," *IEEE Transactions on Services Computing*, vol. 11, no. 6, pp. 908–921, 2016.

[7] M. T. Beck, M. Werner, S. Feld, and S. Schimper, "Mobile edge computing: A taxonomy," in *Proc. of the Sixth International Conference on Advances in Future Internet*, Conference Proceedings, pp. 48–55.

[8] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5g," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[9] X. Sun and N. Ansari, "Edgeiot: Mobile edge computing for the internet of things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22–29, 2016.
[10] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016.
[11] S. Khan, S. Parkinson, and Y. Qin, "Fog computing security: a review of current applications and security solutions," *Journal of Cloud Computing*, vol. 6, no. 1, p. 19, 2017.
[12] R. Zuech, T. M. Khoshgoftaar, and R. Wald, "Intrusion detection and big heterogeneous data: a survey," *Journal of Big Data*, vol. 2, no. 1, p. 3, 2015.
[13] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.
[14] P. Abeysekara, H. Dong, and A. K. Qin, "Distributed machine learning for predictive analytics in mobile edge computing based iot environments," in *2020 International Joint Conference on Neural Networks (IJCNN)*, Conference Proceedings, pp. 1–8.
[15] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5g networks: New paradigms, scenarios, and challenges," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 54–61, 2017.
[16] T. Wang, H. Luo, W. Jia, A. Liu, and M. Xie, "Mtes: An intelligent trust evaluation scheme in sensor-cloud enabled industrial internet of things," *IEEE Transactions on Industrial Informatics*, 2019.
[17] P. Abeysekara, H. Dong, and A. Qin, "Machine learning-driven trust prediction for mec-based iot services," in *2019 IEEE ICWS*, 2019, Conference Proceedings, pp. 188–192.
[18] X. Ma and X. Li, "Trust evaluation model in edge computing based on integrated trust," in *Proceedings of the 2018 International Conference on Algorithms, Computing and Artificial Intelligence*, Conference Proceedings, pp. 1–6.
[19] T. Wang, H. Luo, X. Zheng, and M. Xie, "Crowdsourcing mechanism for trust evaluation in cpcs based on intelligent mobile edge computing," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 6, pp. 1–19, 2019.
[20] X. Li, F. Zhou, and X. Yang, "A multi-dimensional trust evaluation model for large-scale p2p computing," *Journal of Parallel and Distributed Computing*, vol. 71, no. 6, pp. 837–847, 2011.
[21] Y. L. Sun, Z. Han, W. Yu, and K. R. Liu, "A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks," in *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*. IEEE, Conference Proceedings, pp. 1–13.
[22] S. Guleng, C. Wu, X. Chen, X. Wang, T. Yoshinaga, and Y. Ji, "Decentralized trust evaluation in vehicular internet of things," *IEEE Access*, vol. 7, pp. 15 980–15 988, 2019.
[23] C. V. L. Mendoza and J. H. Kleinschmidt, "A distributed trust management mechanism for the internet of things using a multi-service approach," *Wireless Personal Communications*, vol. 103, no. 3, pp. 2501–2513, 2018.
[24] D. Chen, G. Chang, D. Sun, J. Li, J. Jia, and X. Wang, "Trm-iot: A trust management model based on fuzzy reputation for internet of things," *Computer Science and Information Systems*, vol. 8, no. 4, pp. 1207–1228, 2011.
[25] F. Bao and I.-R. Chen, "Dynamic trust management for internet of things applications," in *Proceedings of the 2012 international workshop on Self-aware internet of things*, Conference Proceedings, pp. 1–6.
[26] F. Bao and R. Chen, "Trust management for the internet of things and its application to service composition," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a*, Conference Proceedings, pp. 1–6.
[27] F. Bao, R. Chen, and J. Guo, "Scalable, adaptive and survivable trust management for community of interest based internet of things systems," in *Autonomous Decentralized Systems (ISADS), 2013 IEEE Eleventh International Symposium on*, Conference Proceedings, pp. 1–7.
[28] C. Fernandez-Gago, F. Moyano, and J. Lopez, "Modelling trust dynamics in the internet of things," *Information Sciences*, vol. 396, pp. 72–82, 2017.
[29] J. Lopez and S. Maag, "Towards a generic trust management framework using a machine-learning-based trust model," in *Trustcom/BigDataSE/ISPA, 2015 IEEE*, vol. 1, Conference Proceedings, pp. 1343–1348.
[30] K. Zhao and L. Pan, "A machine learning based trust evaluation framework for online social networks," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2014 IEEE 13th International Conference on*, Conference Proceedings, pp. 69–74.
[31] R. Akbani, T. Korkmaz, and G. Raju, "Emltrust: an enhanced machine learning based reputation system for manets," *Ad Hoc Networks*, vol. 10, no. 3, pp. 435–457, 2012.
[32] U. Jayasinghe, A. Otebolaku, T.-W. Um, and G. M. Lee, "Data centric trust evaluation and predication framework for iot," 2017.
[33] M. Graña, J. D. Nuñez-Gonzalez, L. Ozaeta, and A. Kamińska-Chuchmała, "Experiments of trust prediction in social networks by artificial neural networks," *Cybernetics and Systems*, vol. 46, no. 1-2, pp. 19–34, 2015.
[34] K. Zolfaghar and A. Aghaie, "Evolution of trust networks in social web applications using supervised learning," *Procedia Computer Science*, vol. 3, pp. 833–839, 2011.
[35] R. Chen, J. Guo, and F. Bao, "Trust management for soa-based iot and its application to service composition," *IEEE Transactions on Services Computing*, vol. 9, no. 3, pp. 482–495, 2016.
[36] H. Ouyang, N. He, L. Tran, and A. Gray, "Stochastic alternating direction method of multipliers," in *International Conference on Machine Learning*, 2013, Conference Proceedings, pp. 80–88.
[37] D. Hallac, J. Leskovec, and S. Boyd, "Network lasso: Clustering and optimization in large graphs," in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, Conference Proceedings, pp. 387–396.
[38] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for svm," *Mathematical programming*, vol. 127, no. 1, pp. 3–30, 2011.
[39] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.
[40] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.
[41] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-baiot—network-based detection of iot botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, 2018.

Prabath Abeysekara received his B.Sc. (Hons) of Engineering degree from University of Moratuwa, Sri Lanka in 2010. He is currently a PhD candidate at School of Computing Technologies in RMIT University, Melbourne, Australia. His primary research interests include Machine Learning, Cyber Security and Distributed Computing.

Hai Dong received a Bachelor's degree from Northeastern University, China in 2003 and a PhD degree from Curtin University of Technology, Australia in 2010. He is currently a lecturer at School of Computing Technologies in RMIT University, Australia. His primary research interests include: Service-Oriented Computing, Distributed System, Cyber Security, Machine Learning and Data Analytics. He is a senior member of the IEEE.

A. K. Qin received the BEng degree from Southeast University, China, in 2001, and the PhD degree from Nanyang Technological University, Singapore, in 2007. He is currently a full professor with Swinburne University of Technology, Australia. His major research interests include evolutionary computation, machine learning, image processing, GPU computing, and services computing. He is a senior member of the IEEE.