SALT: transfer learning-based threat model for attack detection in smart home

Pooja Anand1, Yashwant Singh1, Harvinder Singh2,3, Mohammad Dahman Alshehri4 & Sudeep Tanwar5

The next whooping revolution after the Internet is its scion, the Internet of Things (IoT), which has facilitated every entity the power to connect to the web. However, this magnifying depth of the digital pool oil the wheels for the attackers to penetrate. Thus, these threats and attacks have become a prime concern among researchers. With promising features, Machine Learning (ML) has been the solution throughout to detect these threats. But, the general ML-based solutions have been declining with the practical implementation to detect unknown threats due to changes in domains, different distributions, long training time, and lack of labelled data. To tackle the aforementioned issues, Transfer Learning (TL) has emerged as a viable solution. Motivated by the facts, this article aims to leverage TL-based strategies to get better the learning classifiers to detect known and unknown threats targeting IoT systems. TL transfers the knowledge attained while learning a task to expedite the learning of new similar tasks/problems. This article proposes a learning-based threat model for attack detection in the Smart Home environment (SALT). It uses the knowledge of known threats in the source domain (labelled data) to detect the unknown threats in the target domain (unlabelled data). The proposed scheme addresses the workable differences in feature space distribution or the ratio of attack instances to a normal one, or both. The proposed threat model would show the implying competence of ML with the TL scheme to improve the robustness of learning classifiers besides the threat variants to detect known and unknown threats. The performance analysis shows that traditional schemes underperform for unknown threat variants with accuracy dropping to 39% and recall to 56.

Internet of Things (IoT) is getting matured by employing an umbrella of technologies working together to connect smart devices to the Internet. With such expansion, there is a need to have ubiquitous control over these smart devices1. Such as our things, fridges, homes, TVs, cars, cameras, transportation, medicals, surgeries, cities, grids, agriculture, industries, and defense. Moreover, IoT is a blend of diverse technologies working together in an IoT communication stack to bestow the preeminent services in diversified application areas. According to the predicting agency, the number of smart devices will increase up to 41 billion by 2027. Such a massive increase in the usage of smart devices will end up generating the data up to Geopbytes (10^30) even. Moreover, embracing such a technological shift in our lives has intensified the risk of cyber-attacks and privacy breaches. For example, the Palo Alto Threat report under, Unit 42 substantiates the declination of IoT with supporting numbers for growing attacks3. The study claims the unencrypted IoT data traffic is ruling with 98% and 83% medical devices running the antiquated operating systems. Additionally, depicting the smart security cameras ushering with 33% security concerns, and the smart printers ruling out with 24%4. In relation to growing cyber-attacks in IoT, that too has turned to billions following the IoT with roundabout a three-fold accretion in attack traffic5,6.

These statistics have been ruled out in the IoT realm, as these systems are integrated with resource constraint nodes and generally located in unattended environments. Thus, deploying traditional security mechanisms to maintain privacy, access control, integrity, and authentication in smart systems becomes a challenging task7.

1Department of Computer Science and Information Technology, Central University of Jammu, Rahya Suchani, Jammu and Kashmir 181143, India. 2Leaders Institute, Woolloongabba, Brisbane, QLD -4102, Australia. 3Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, Fortitude Valley, QLD 4006, Australia. 4Department of Computer Science, College of Computers and Information Technology, Taif University, P.O. Box 11099, Taif 21944, Saudi Arabia. 5Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad, Gujarat 382481, India. 6email: harvinder_07@yahoo.com; sudeep.tanwar@nirmauni.ac.in
Moreover, these smart devices become part of countless smart services with inherent vulnerabilities as an easy entry point to penetrate. And the negligent manufacturers and unaware users of these smart devices aid in easy playing. The successful launch of several IoT-based botnet attacks also exhilarates its feeble state by shutting down the system with denial of service (DoS). Even the hackers earn well with cyber assaults collecting good ransom with ransomware attacks. With its many more variants, cryptocurrency could be mined, and privacy could be hindered. All these factors looming on the IoT horizon pose a pronounced risk to the sustainability of IoT in this digital era. The society being served with the vital services nurtured by IoT necessitates the accelerated development of robust, resilient security mechanisms.

Several signature-based and learning-based mechanisms exist in the literature to shield the IoT system. Moreover, threat vectors and attacks are also evolving rapidly, making them inefficient to the growing needs of an IoT system. Thus, opening the ways for intelligent detection engines to adopt the paradigm of transferring knowledge as humans do, to learn with the changing attack pattern and behavior, employed under transfer learning. Even though the learning models are the germane tools for discerning the ‘regular’ and ‘irregular’ behavior of nodes in an IoT system and the interaction among them to provide vital services. These machine learning (ML) classifiers fail to detect unknown threats, usually the mutations of forgoing known threats. Classifiers generally consider them the same and sentence to detect the unknown threats without any specific mechanism. The study articulating the compounding security implications evidenced in smart systems demands the shift towards identifying the known and unknown threats. Consequently, in this research work, we have proposed a TL-based threat model with baseline ML classifiers to detect the known and unknown threats in the smart home environment.

State-of-the-art threat modelling methods like STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, DoS, and Elevation of Privilege), DREAD (Damage, Reproducibility, Exploitability, Affected Users, and Discoverability), etc., have been the generalized schemes, initially developed for IT systems and not cover the application of intelligent trends for threat detection. Although, the identification of different known threats under the specific categorization has been generalized. But unknown threats have never been part of those threat models. The studied literature evidence that the tools like Microsoft threat modelling tools are used to develop threat models based on the basic STRIDE approach for the smart environment. However, as per the authors’ knowledge, no specific smart threat modelling approach has been proposed to cover both known and unknown threats in smart systems. For example, authors in present an application of TL to understand well variants of threat insinuations detecting over the general learning methods. In continuation, Sameera et al. detected unknown threats by exploiting manifold alignment in the deep transductive TL model. Later, Zhang et al. proposed a domain adversarial NN-based TL approach to detect network attacks in smart grid.

Motivated by the aforementioned discussion, in this paper, we have proposed the TL-based smart threat modelling approach to detect the known and unknown threats in the smart home environment by analyzing the network traffic and leveraging the knowledge acquired while detecting the known threats. Figure 1 shows the various stages of the proposed threat modelling process, which are further elaborated in "The proposed threat modelling methodology" section. In SALT, the targeted smart application/device with identified assets is taken as input by the intelligent detection engine for further system analysis. For known threats, the collected data of the smart device undergoes preprocessing, followed by preprocessed feeds to learning models to detect the system status of being under threat or not. Specifically, for unknown threats, a domain adaptation-based TL scheme is chosen, which facilitates the transfer of knowledge from a domain containing adequate training data, i.e., the source domain, to a similar target domain having very less or no labelled training data at all.

The literature evidence that the growing depth of the digital pool with innumerable IoT devices has made the infant smart world more susceptible to penetrations and cyber-attacks. The key motivation for this study to leverage the gap in the literature are listed below:

Figure 1. Stages of threat modelling process.
Later, Kumar et al. proposed an intelligent decentralized scheme for threat detection, employed on fog nodes to detect both known and unknown threat variants, tested on NSL-KDD dataset. Then, Javeed et al. proposed an approach by bringing together a decision tree (C5 classifier) and support vector machine (One-class) for threat detection. In another recent work, Al-Hawawreh et al. employed a deep learning-based pattern extractor for IoT networks with threat intelligence-based mechanisms for threat-type detection, evaluated on TON IoT and N-BAIOT. However, none of the state-of-the-art work targeted unknown threat variants. Moreover, the applicability of the few discussed works in TL is limited due to the use of structured dataset like NSL-KDD or validating through hypotheses on different threats within the same dataset. The TL-based proposed solution can improve the competence of learning models for self-adaptation to detect the different variants of threats.

The rest of the paper is organized as follows. “State-of-the-art” section provides the related work for this study, covering both known threats and unknown threats with learning. “The proposed threat modelling methodology” section discusses the threat modelling methodology with DFDs and the proposed intelligent threat model; SALT is detailed in “SALT: transfer learning-based threat model for attack detection in smart homes” section. “Results and discussion” section discusses the theoretical proof for evaluating the proposed system, and finally, “Conclusion” section concludes the paper.

**State-of-the-art**

In this section, the relevant works corresponding to a domain of threat modelling known and unknown threats in the smart environments are retrospected. The literature manifests that detecting cyber-threats in IoT mainly focuses on known threats. For example, Pajouh et al. employed Linear Discriminant Analysis and Principal Component Analysis for dimensionality reduction, with a two-tier classification approach comprising K-Nearest Neighbor (certainty Factor version) and Naïve Bayes. Gao et al. designed an extreme learning machine to identify suspicious behaviors in smart set-ups. Later, Kumar et al. proposed an intelligent decentralized scheme for threat detection, employed on fog nodes. Muder et al. used the recursive neural networks for threat detection. Khraisat et al. employed a decision tree (C5 classifier) and support vector machine (One-class). In another recent work, Al-Hawawreh et al. employed a deep learning-based scheme. But they mainly focused on known threats, whose signatures are either stored in the database or the learning model has learned their behavior. However, unknown threats remain undetected by the learning model and the system collapses. This is because most machine learning algorithms assume that the training data and the testing data have same distribution and they are in the same feature space. This assumption does not hold in most real life scenarios. Thus, problem becomes more serious for smart devices, due to their exponential number, constrained resources, and inherent vulnerabilities. And they become easy targets to be the part of a botnet. From last few years, TL has emerged as a potential ground in areas of image identification and natural language processing, but studies using TL in threat detection are still lacking. Moreover, the applicability of the few discussed works in TL is limited due to the use of structured dataset like NSL-KDD or validating through hypotheses on different threats within the same dataset. The TL-based proposed solution can improve the competence of learning models for self-adaptation to detect the different variants of threats.

The key contributions of this work are as follows.

- We present a threat modelling methodology, working in six well-illustrated phases with data flow diagrams (DFDs) up to level-2, about the smart-home environment to identify whether a smart application/device is under known/unknown threat or not.
- We propose SALT, a TL-based threat model for attack detection in smart homes, to make the proposed model intelligent and adaptable to detect the new evolving variants of threats. This can be achieved by applying the domain adaptation paradigm of the TL scheme with customized classifiers for the problem. Also, the correlation analysis of different threats on network dataset of TON-IoT is given.
- We present an algorithm for the proposed threat model, SALT with ML classifier to detect the system is under known threat / not and TL-based classifier to detect the unknown threats. Finally, the theoretical proof to detect whether the system is under the known or unknown threat is given with the threshold decidability factor. Also, the performance analysis of general ML-based solutions on TON-IoT dataset to detect known and unknown threat variants is given.

Authors across the globe have presented many solutions to detect cyber-threats in IoT. Pajouh et al. employed a two-tier classification approach comprising K-Nearest Neighbor (certainty Factor version) and Naïve Bayes. Gao et al. designed an extreme learning machine to identify suspicious behaviors in smart set-ups. Later, Kumar et al. proposed an intelligent decentralized scheme for threat detection, employed on fog nodes. Muder et al. used the recursive neural networks for threat detection. Khraisat et al. employed a decision tree (C5 classifier) and support vector machine (One-class). In another recent work, Al-Hawawreh et al. employed a deep learning-based scheme. But they mainly focused on known threats, whose signatures are either stored in the database or the learning model has learned their behavior. However, unknown threats remain undetected by the learning model and the system collapses. This is because most machine learning algorithms assume that the training data and the testing data have same distribution and they are in the same feature space. This assumption does not hold in most real life scenarios. Thus, problem becomes more serious for smart devices, due to their exponential number, constrained resources, and inherent vulnerabilities. And they become easy targets to be the part of a botnet. From last few years, TL has emerged as a potential ground in areas of image identification and natural language processing, but studies using TL in threat detection are still lacking. Moreover, the applicability of the few discussed works in TL is limited due to the use of structured dataset like NSL-KDD or validating through hypotheses on different threats within the same dataset. The TL-based proposed solution can improve the competence of learning models for self-adaptation to detect the different variants of threats.
From the last few years, TL has emerged as a potential ground to form an invariant representation of data traffic from different domains to understand the threat insinuations of unknown types well. For example, Zhao et al. proposed a linear transformation-based TL (HeTL) model for detecting the network attacks. Still, this model requires hyper-parameters to be pre-set manually. Following this work, the authors proposed another TL model to detect unknown attacks using a cluster-based approach (CeHTL). On a similar approach, Zhang et al. proposed a domain adversarial NN-based TL approach to detect the network attacks in a smart grid, with experimentation on the created testbed. In the same line, Li et al. proposed an adaptation regularization-based TL model to detect new unseen variants of known malware. Then, Vu et al. proposed an auto-encoder-based TL approach to detect the network attacks in IoT. Later, Singla et al. used adversarial domain adaptation to develop the TL model to use the knowledge attained from one NID dataset for another domain/network with the less labeled dataset. The existing literature essentially focused either on modelling the known threats or the common vulnerabilities being part and parcel of smart devices. The authors have proposed several learning-based approaches that can classify known and unknown threats. However, some threats which do not belong to the class of the threats on which the model is being trained, remains undetected. Their work could be further improved by applying TL. The applications of TL are widely seen in areas of image identification and natural language processing, but studies using TL in threat detection are still lacking. Moreover, the applicability of the few discussed works in TL is limited due to the use of structured datasets like NSL-KDD or validating through hypotheses on different threats within the same dataset. Therefore, improving the competence of learning models for self-adaptation in diverse operating scenarios with TL for their deployment in different smart applications is the main goal of this work.

Seeing the potential of TL and the limitation of general ML-based classifiers to detect unknown threat variants, we have proposed a TL-based threat model. The proposed threat model employed the basic ML classifier with deep transductive learning. In the proposed model, the knowledge of known threats is used to detect unknown threat types, with the TON dataset comprising different attack types.

The proposed threat modelling methodology

This section covers the threat modelling process with DFDs up to level 2.

Threat Modelling is the structured process of identifying the threats in a system, communicating them to the organization's stakeholders or end-users, and the mitigation controls to handle them. A threat can be anything that has the potential to exploit the vulnerabilities and breach the system's security to hamper the object of interest in multiple ways. Figure 2 shows the relationship between the basic terms like vulnerability, threat, attack surface, attack, and risk in threat modelling. The different threat modelling methods like STRIDE, DREAD, VAST, etc., are widely used for identifying the threats in a system. These methods are integrated into TM tools like Microsoft Threat Modeling tool, ThreatModeler, etc. The Microsoft TM tool designs the DFD for the system and then employs the STRIDE methodology to identify the threats at the design level. The STRIDE model covers the threats under spoofing, tampering, repudiation, information disclosure, DoS, and elevation of privileges.

On the contrary, the ThreatModeler tool designs process flow diagrams for the system under investigation and then employs the VAST methodology to identify the threats at the application/process level. The VAST model does not categorize the threats and is defined as visual, agile, and simple threat modelling. But none of these TM processes cover identifying unknown threats in their threat modelling process. The proposed threat modelling methodology works in six phases discussed as follows.

a. **Find Target of Evaluation (ToE):** The first step while designing the security structure/model is to understand the environment your device works within and identify the use case that will be called your target of evaluation. In the proposed methodology, the smart home application and the smart devices are our ToE. In particular, we can start with the smart camera and the applications aiding the user interface as ToE.

b. **Identify Assets:** After identifying your evaluation target, identify the components of the targeted device that need to be protected, which will be called the assets of a smart device. For example, in the context of the smart camera, the identified assets include log-in credentials, private keys, firmware, system configurations,
Figure 3. DFD: The level-0 DFD.

The identified threats are forwarded to the mitigate controller, which further addresses the threats like data integrity by applying the secure data dissemination scheme. Then, the IKT module undergoes a validation process to know the performance of the trained ML model. The trained ML model only detects the known threats with significant results on validation. The unknown threats remain undetected as the model is not familiar with the new variants of threats with different data distribution depicted by results in "SALT: the theoretical proof" section. Thus, for handling unknown threats, another TL-based learning module is added. The TL-based model takes the input as labeled data from the source domain data repository and unlabeled data from the target domain data repository. As an output, it generates the labels for the target unlabeled data and stores them in the corresponding repository. And thus sends the threat signal for the target labels which are categorized to be threats. The trained TL model also undergoes a validation process; this module will be covered in future work.

network communications, event logs, and resources of the targeted device like battery, computing power, storage, and bandwidth, etc. A list of identified assets will be stored in the database. Additionally, identify the users and other entities interacting with the smart device/application.

c. Identify Threats: Once the targeted device and assets are identified, known and unknown threats will be detected by the intelligent module based on the network traffic generated by the device and its behaviour under threat. The respective threat signal will be sent to the alert generator if the system is under threat. Also, the identified threats will be communicated to the mitigate controller, and the list will be stored in the database for further assistance.

d. Generate Alert: Suppose the system/device is found under some known/unknown threat. In that case, the threat identification module will generate an alert to communicate to the organization’s security team or end-users about the same through email/ text message. Also, the logs of the generated threats will be stored in the database.

e. Mitigate Controller: After the identified threats are communicated to the mitigate controller, it addresses the threats by applying the control measures from the database. Additionally, if the new security measures are created to mitigate the unknown threats, they are stored in the controller database.

f. System Validation: The modules created for identifying known/unknown threats and those for mitigating these threats undergo a validation process before deployment in real systems by uploading on the raspberry-Pi working on the smart home gateway. The validation results are shown in performance metrics on the display unit.

The proposed threat modelling process is also illustrated with the DFDs upto level-2. Initially, DFDs were developed for system engineers to visualize better how a system works with multiple applications in terms of moving data, applying operations on data, and storing data. In DFDs, the circle represents a process, the closed rectangle represents an entity, the rectangle with open endings represents a datastore, and the arrow represents a data flow. Figure 3 shows level-0 DFD gives us the abstract view of the entire system. It represents the entire threat model as a single process, i.e., an AI-based threat model with its relationship to external entities, i.e., smart home devices and end-users. With normal and attack data as input from the smart home devices and sending an alert as an output to the end user. Figure 4 shows level-1 DFD decomposes level-0 process into different sub-processes, which are briefly explained as the six phases of the threat modelling methodology. Thus, DFD-1 highlights the system's main functions with processes, external entities, inputs, outputs, and data stores. At this level, the data stores are added for storing the list of threats, list of threat detection algorithms, list of alerts, and the list of threat mitigation algorithms.

In level-2 DFD, the processes of level-1 DFD are dug one step deeper to get a clear understanding of the sub-processes further work and if they are further decomposed to the sub-modules/processes to achieve the desired functionality. At this level, the necessary details about the exact functioning of the system are recorded. Figure 5 shows the proposed threat model with level-2 DFD. The end-user selects the smart home application to check whether it is under threat or not. A specific smart home device like a smart camera or ecobee thermostat is targeted for evaluation. Then, the assets for ToE are identified to be fed to another sub-process, i.e., Identifying known threats (IKT). Also, IKT gets the data of the targeted smart device as input from the respective benign data and attack data repositories. This sub-process uses ML models to analyze the data further and generates the threat signal if the system is found under threat. These threat signals are stored in the alert logs repository, and the alert is sent to the end-user.

Suppose the system/device is found under some known/unknown threat. In that case, the threat identification module will generate an alert to communicate to the organization's security team or end-users about the same through email/ text message. Also, the logs of the generated threats will be stored in the database.

Identify Threats: Once the targeted device and assets are identified, known and unknown threats will be detected by the intelligent module based on the network traffic generated by the device and its behaviour under threat. The respective threat signal will be sent to the alert generator if the system is under threat. Also, the identified threats will be communicated to the mitigate controller, and the list will be stored in the database for further assistance.

Generate Alert: Suppose the system/device is found under some known/unknown threat. In that case, the threat identification module will generate an alert to communicate to the organization's security team or end-users about the same through email/ text message. Also, the logs of the generated threats will be stored in the database.

Mitigate Controller: After the identified threats are communicated to the mitigate controller, it addresses the threats by applying the control measures from the database. Additionally, if the new security measures are created to mitigate the unknown threats, they are stored in the controller database.

System Validation: The modules created for identifying known/unknown threats and those for mitigating these threats undergo a validation process before deployment in real systems by uploading on the raspberry-Pi working on the smart home gateway. The validation results are shown in performance metrics on the display unit.
**SALT: transfer learning-based threat model for attack detection in smart homes**

Figure 6 shows the threat model with learning-based robust mechanisms for threat detection in an IoT environment created in the smart home. The smart home application (SHm) comprises multiple smart devices/nodes (Nq) connected via a network (Nw). Each smart device has multiple assets (At) like firmware, network communication, log-in credentials, device resources, system configuration, certificates, and unique keys that the adversaries target. The smart networks have been the victim of known and unknown threat variants rooted by the network of intruders (Id) working on common attack surfaces like gateways, user interfaces, open ports, and weak authentication mechanisms. It is alarming to entrust human lives and their wealth to utilize the exploding smart technologies with poor robust security mechanisms. In the given threat model (Msy), the known threats (Km) are detected by training the ML model with data collected from the smart home network. On the contrary, the unknown threat variants (Zr) for which the machine learning classifiers (MLc) underperforms, for the common classifiers like ANN and SVM, could be detected by employing domain adaptation in the TL paradigm. It is evident from the literature that unknown attack variants could be detected by leveraging the knowledge from known attack variants.

Algorithm 1 presents the steps to find the known and unknown threats by applying the domain adaptation paradigm of TL. The overall time and space complexity of the given algorithm is $O(n \log n)$ and $O(nd)$ respectively, where $n$ represents the total number of data samples and $d$ represents the number of features representing each data sample in the dataset. In the given algorithm, the source domain (Ds) with labeled data (Ld) depicting known threats, whereas the target domain (Dt) with unlabelled data (Ud) depicting unknown threat variants are taken as input. The TL-based model (Tlm) is used to transfer the knowledge of known threats $K_m$ from the source domain, i.e., $D_s \in E_d$ to detect the unknown threats $Z_r$ in the target domain, i.e., $D_t \in E_d$, by bringing the data into the common latent space by reducing the distance between the two distributions. $L_d$ and $U_d$ are taken as input by $Tlm$ and produce $Z_i$ as output. Further, the domain invariant representation ($Z_i$) is taken as input by the learning classifier to label the target data $D_t$ as a threat or normal. Thus, TL techniques have been employed in SALT. Such schemes have the grounds to reduce the dependence on labeled data, expedite the learning speed, and strengthen the learning classifiers' robustness against different wireless environments. The proposed threat model is elaborated in detail as follows.

---

**Figure 4.** DFD: The level-1 DFD.
Figure 5. DFD: The level-2 DFD.

Figure 6. SALT: The Threat Model.
A. Attackers Space: Smart homes are one of the reimbursements of the technological advancements in ICT with technological shifts in our personal lives. Transforming homes into smart homes with IoT and other ubiquitous technologies has widened the attack surface for the adversaries to move into the smart home remotely. This is because attackers and our smart devices are working on the same common platform, i.e., the Internet, and thus easily accessible. The attack surface of the smart home mainly includes sensors and actuators, smart devices firmware, an application installed on smartphones, the web interface, gateways, hubs, and the requisite communications within the smart home. A single intruder or chain of intruders can plan bigger attacks whose aftermaths result in bigger ransoms or denial of crucial services. Some of the threats they pose are summarized like the basic function of a smart device can be hampered by maliciously modifying the Integrated Circuit using a hardware trojan. The secret keys could even be generated by using the side channel information about the device like power dissipation, processing time, and power consumption. The clones of smart nodes are added to mislead the entire smart system and can drain the battery of sensor nodes to make them non-functional. Also, it can have an active ear over the communication links to gain the control information like node configurations, node identities, and shared network passwords. Even the communication links could be jammed, and the fraudulent packets are made the part of communication.34

B. Network of Smart Home Devices: A smart home is a housing environment equipped with several technologies to facilitate the users of interest to control home appliances and systems from anywhere. For this, smart homes communicate in real-time, whether it is about sending the sensor’s data/camera’s feeds to the user or receiving the command from the user to set the temperature at home. These intelligent services are provided by communication between the devices at home and the related environments. In the proposed threat model, the smart home is made up of devices like a smart lighting system, smart hue bulb, ecobee thermostat, Enno doorbell, smart camera, baby monitor, smart door lock, temperature/humidity sensors, and amazon echo plus. The user can access these devices through his mobile application, as they all are connected to a gateway for remote access. As per the report from Stratcast, numbers stipulate the market growth of smart homes to exceed 7 billion dollars by 2025.35 But these devices have significant inherent vulnerabilities and are easy targets for adversaries to play with. For example, TVs and refrigerators were hacked to send spam and phishing messages. On a similar note, leading to threats like data forgery, gaining unauthorized access, and control over the smart device. Thus, the assets of the smart home environment, whether physical or information, must be handled carefully. The physical assets cover smart devices, IoT hubs, gateways, cloud servers, sensors, and actuators. The information collected by sensors, user credentials, encryption keys, certificates, installed assets, location information, etc., come under the information assets.

In the proposed smart home threat model, there is a set of different entities $E = \{E_a, E_n, E_m, E_k\}$, where an IoT application $A_p \in \{A_1, A_2, \ldots, A_a\} \in E_a$, with multiple IoT nodes $N_q \in \{N_1, N_2, \ldots, N_n\} \in E_n$ is connected via a network $N_w \in E_w$. These smart home nodes $N_q$ have been the victim of known threats $K_m \in \{K_1, K_2, \ldots, K_k\} \in E_k$, and unknown threats $Z_r \in \{Z_1, Z_2, \ldots, Z_z\} \in E_z$. Since with time ‘t’ different variants i.e., $Z_r$ of known threats $K_m$ are emerging which can have different marginal distribution $M_d$ in $E_m$ and feature space $F_s \in E_s$. The relationship among the aforementioned entities in the proposed threat model is represented as follows.

```plaintext
Algorithm 1 Known and Unknown Threat Detection in Smart Home.

Input: Entities $SH_m, D_1, D_2, M_{d_1}(f_1), M_{d_2}(f_2)$.
Output: Successful Threat Detection using traffic analysis and Domain Adaptation.

1: procedure SALT: THE THREAT MODEL($SH_m$)
2: System Initialization
3: Read the data
4: $D_1 \leftarrow$ FetchSourceDomainData($SH_m$)
5: $D_2 \leftarrow$ FetchTargetDomainData($SH_m$)
6: $PD_1 \leftarrow$ CALL(PREPROCESSOR($D_1$))
7: $PD_2 \leftarrow$ CALL(PREPROCESSOR($D_2$))
8: Calculate Marginal Probability Distribution of source and target data.
9: $M_{d_1}(f_1) \leftarrow PD_1$
10: $M_{d_2}(f_2) \leftarrow PD_2$
11: if ($M_{d_1}(f_1) == M_{d_2}(f_2)$ and $D_1 == D_2$) then
12: Status $\leftarrow$ CALL(KNOWLEDG)(($D_1$))
13: else
14: Status $\leftarrow$ CALL(UNKNOWN)(($D_1$, $D_2$))
15: end if
16: if (Status $==$ 1) then
17: User $\leftarrow$ Alert()
18: print("Smart Home environment is under the threat")
19: end if
20: end procedure
```
\[ K_m \rightarrow \sum_{i=1}^{\nu} A_p \text{ and } K_m \rightarrow \sum_{j=1}^{\nu} N_q \]  

(1)

\[ Z_r \rightarrow \sum_{i=1}^{\nu} A_p \text{ and } Z_r \rightarrow \sum_{j=1}^{\nu} N_q \]  

(2)

subject to constraints

\[ z \ll k \text{ and } z \geq 0 \]  

(3)

\[ v \leq a \text{ and } w \leq n \]  

(4)

\[ a, m, f, n, w, k \geq 1; d \neq 0 \]  

(5)

where \( \alpha \) is the association of a known threat \( K_m \) with the IoT application \( A_p \), and \( \beta \) signifies relation with IoT nodes \( N_q \). Similarly, the association of the unknown threat is shown by \( \rho \) and \( \sigma \). Further, the domain and the task are defined as follows. A domain \( D \) is defined with two terms, i.e., the feature space \( F \) and the marginal distribution \( M_d(f) \), such that their relationship is represented as follows.

\[ D = \{F, M_d(f)\}, \forall F = \{f_1, f_2, \ldots, f_n\} \in F, \]  

(6)

where \( n \) is the number of feature vectors.

For different domains say \( D_t \) and \( D_s \), as considered in this paper, either the feature space will be different or the marginal distribution represented as follows.

\[ F_i \neq F_t \text{ or } M_{ts}(f_i) \neq M_{dt}(f_t) \]  

(7)

(8)

Given domain \( D \), a task \( T \) is also defined with two terms, i.e., the label space \( L \) and the prediction function \( P(f_t) \), which learns from the pair of feature vector and the label, i.e., \( \{f_t, l_t\} \) such that \( f_t \in F \) and \( l_t \in L \). Thus, the task is represented as follows.

\[ T = \{L, P(f_t)\}, \forall L = \{l_1, l_2, \ldots, l_m\} \in L, \]  

(9)

where \( m \) is the number of different labels.

For different tasks, say \( T_s \) and \( T_t \), either the label space will be different or the conditional distributions for the source and the target domains will be represented as follows.

\[ L_s \neq L_t \text{ or } C_{st}(l_k(f_t)) \neq C_{ds}(l_k(f_t)) \]  

(10)

(11)

C. Detecting known threats with machine learning: The proposed threat model leverages ML/DL to find the status of a smart system of being under threat or not. It accepts the traces of network traffic as input from the gateway of the smart home environment that could be benign samples or attack traces of DDoS attacks like Mirai or bashlite (scanning, udp, tcp, ack, syn). The labeled source data \((U_d)\), with \( n \) values, undergo data cleaning/preprocessing (outlier removal, handling missing value, encoding, and normalization) before being fed to the ML model to increase the prediction accuracy in threat detection. The procedures involved in the data cleaning process are represented as follows.

\[ \psi = Q_3 - Q_1 \]  

(12)

\[ Q_1 = \text{Median} \left[ 1, \frac{n}{2} \right] \]  

(13)

\[ Q_3 = \text{Median} \left[ \frac{n}{2} + 1, n \right] \]  

(14)

where \( \psi \) is the midspread, called the inter-quartile range, which removes the outliers from the dataset, the first quartile \( Q_1 \) is the median of the lower half of the ordered dataset, the second quartile \( Q_2 \), is the median value, and \( Q_3 \) is the median of the upper half of the ordered dataset.

\[ Z_i = \frac{f_i - \text{max}(f_i)}{\text{max}(f_i) - \text{min}(f_i)} \]  

(15)

where \( z_i \) is the normalized value of the feature \( f_i \), which brings the numeric value of the feature of data to scale between 0 and 1. And \( \text{max}(f_i) \) and \( \text{min}(f_i) \) gives the maximum mad minimum value in \( f_i \) given its range. Then, one hot encoding is used to generate binary values \((B_i)\) for the categorical features in the dataset.
where \( O \) is the set of categorical feature values, \((B_0^1)\) is the binary output values after one-hot encoding, and \( \zeta \) signifies the convert relation.

Once the data cleaning is done, the preprocessed data will be passed to the learning algorithm (Decision tree, KNN, SVM, Random Forest, DBN, Ensemble, DNN, LSTM) for threat prediction and classification. Algorithm 2 presents the steps of preprocessing. If the system is found under threat, an alert will be generated and will be passed to the end-user through email or text message. Algorithm 3 presents the procedure to find the known threats with a learning classifier. Although, subsequent studies with ensemble learning and neural networks have performed well with good accuracy. However, these prevailing approaches result in a high false alarm rate and work under the default assumption that both training and predicting datasets follow the same probability distribution with the same threat variants in both datasets. But, this assumption fails to hold, thus degrading the accuracy of the traditional machine learning approaches. Figure 7 depicts different variants of DDoS attacks that have a different distribution, which when falls under the different datasets like training and testing results in poor performance of ML models. The proposed TL-based threat model allows the trained ML/DL classifiers to retain robust performance even against unknown threats.

D. Detecting unknown threats with Transfer Learning: SALT leverages transductive TL to expand the existing knowledge attained during the learning process of known threats, against unknown threats in the smart home, as a binary classification problem, classifying the events as benign or threat. This phase of the threat model follows the pipeline, executed in four steps. (i) Firstly, collect the labeled source data, i.e., known threats, and the unlabeled target data, i.e., unknown threats from their respective repositories. Then, this data undergoes various phases of pre-processing before moving further as an input to learning models; (ii) The models like GAN, ANN, Autoencoders, and CNN are trained in a manner to accomplish the domain adaptation over both the domains, to provide a new feature representation of the source and target data. For this, certain loss functions like maximum mean discrepancy are calculated, which are backpropagated to adjust the weights of neural networks till the objective function is minimized; (iii) Then, the labeled data with the new feature representation is used to train the classifiers; (iv) After the model is trained with new feature space, then map the new unlabeled target data to the new feature space, and the trained model will predict their class labels. Algorithm 4 explains the detailed procedure of detecting unknown threats with TL. This phase of the proposed threat model covers the problem of detecting unknown threats, which can be represented as follows. Given a source domain \( D_s \), a source task \( T_s \) with labeled traffic data \( L_d \) and a related target domain \( D_t \), a target task \( T_t \) with unlabeled traffic data \( U_d \), the transductive TL is applied to learn the target predictive function \( P_t(\cdot) \) from labeled source data and unlabeled target data to correctly predict the target labels in \( D_t \). subject to constraints

\[
D_t \neq D_s \text{ or } M_d t(f_t) \neq M_d t(f_t)
\]  

(17)

\[
T_s = T_t \text{ and } L_s = L_t
\]  

(18)

With labeled source domain traffic data \( L_d \) and unlabeled target domain traffic data \( U_d \), represented as follows.

![DDoS Variants Distribution](image_url)
The target predictive function $P_t(\cdot)$ with least expected error on $D_t$ given by

$$P_t(\cdot) : f_i \rightarrow l_i$$

And the target predictive function $P_t(\cdot)$ with least expected error on $D_t$ given by

$$D_t = \{(f_1, l_1), (f_2, l_2), \ldots (f_n, l_n)\}$$

$$T_t = \{f_{n+1}, f_{n+2}, \ldots, f_{n+m}\}$$

**Algorithm 2** Data Preprocessing.

**Input:** Dataset $D$.

**Output:** Preprocessed and cleaned dataset.

```
1: procedure PREPROCESSOR(D) ✔️ Data Cleaning
2: df ← FetchDataset(SmartHomeNetwork)
3: df ← RemoveOutliers()
4: ψ ← Q3 − Q1
5: Q1 ← Median $[1, \frac{n}{2}]$
6: Q3 ← Median $[\frac{n}{2} + 1, n]$
7: df ← OnehotEncoding() ✔️ On String column of data
8: df ← MinmaxScaler() ✔️ Bring the values between 0 and 1
9: $Z_i = \frac{f_i - \text{max}(f_i)}{\text{max}(f_i) - \text{min}(f_i)}$
10: df ← ShuffleData()
11: end procedure
```

**Algorithm 3** Process to detect Known Threats.

**Input:** Entities $S_{H_m}$, Preprocessed Source Domain Data $(PD_t)$.

**Output:** Successful prediction of Known Threats.

```
1: procedure KNOWNTD(D_t)
2: $D_t$ ← PD.tLabelData($S_{H_m}$)
3: Train MLClassifier on Source data i.e., $D_t$
4: Model ← ML = Classifier
5: Trained ← Model fit $(f_n, l_n)$
6: $Y_{pred}$ ← Model.predict $(f_{n+m})$
7: ModelACC ← Accuracy($Y_{test}, Y_{pred}$)
8: print(“Classification Report.”)
9: return $Y_{pred}$
10: end procedure
```
**SALT: the theoretical proof**

**Problem statement** Smart home is the most adaptable IoT structure in the coming years. But, this transformation of homes into smart homes has widened the attack surface for the adversaries to remotely move into the smart home. Mainly, a smart home involves two different tasks one is data processing and the other is security. In this problem statement, we majorly focus on the security aspect. So, our focus in this paper is to detect both known and unknown threats in the smart home.

**Assumption** We are considering the unknown active attacks targeting the smart home. These attacks are visible/sensible to users as attackers take some action affecting system resources like services being denied and the creation of dumb files that the system administrator can sense the system is being attacked or under threat.

**Null hypothesis** \((H_0)\) An AI-assisted threat model to detect the unknown threats.

**Alternate hypothesis** \((H_1)\) An AI-assisted threat model can not detect unknown threats.

**Hypothesis testing** Let \(M_1\) be the model to detect known threats, whose signatures have been stored in the database of known threats represented as follows.

\[
M_1 \leftarrow E_k \text{ and } \tag{22}
\]

\[
E_k \leftarrow \{K_1, K_2, \ldots K_3\} \tag{23}
\]

Taking a sample of known threats \(S_k\) from the population \(E_k\), such that a threat is represented by the set of ‘t’ features.

\[
S_k \subseteq E_k \tag{24}
\]

\[
S_k \leftarrow \{F_1^1, F_2^1, F_3^1, \ldots F_t^1\} \tag{25}
\]

Let \(M_2\) be the model to detect unknown threats, after transferring the knowledge of known threats from the model \(M_1\) represented as follows.

\[
M_2 \leftarrow M_1 + \delta \text{ and } \tag{26}
\]

\[
\delta \in \{Z_1, Z_2, \ldots Z_2\} \in E_z \tag{27}
\]

Taking a sample of unknown threats \(U_k\) from the population \(E_z\), such that a threat is represented by the set of ‘t’ features.

\[
U_k \in E_z \text{ and } U_k \notin E_k \tag{28}
\]

\[
U_k \leftarrow \{F_1^2, F_2^2, F_3^2, \ldots F_t^2\} \tag{29}
\]
Then, we apply the Pearson correlation coefficient, to find the correlation of new threat \( U_k \) with existing threats \( E_k \).

Then, we apply the Pearson correlation coefficient, to find the correlation of new threat \( U_k \) with existing threats \( E_k \).

\[
    r(F_1^i, F_2^i) = \frac{N \sum F_1^i F_2^i - (\sum F_1^i)(\sum F_2^i)}{\sqrt{N \sum x^2 - (\sum x)^2} \sqrt{N \sum y^2 - (\sum y)^2}}
\]

where \( N \) is the number of sample rows and \( x \in F_1^i, y \in F_2^i \)

\[
    r_i(S_k, U_k) = \sum_{i=1}^{t} r(F_1^i, F_2^i)
\]

\[
    r'_s(S_k, U_k) = \text{Maxminscaler}(r_s)
\]

where \( r'_s(S_k, U_k) \) is the normalized value of the correlation coefficient. The final state i.e., \( R(SH_m) \) of the system being under the threat or not is given as follows.

\[
    R(SH_m) = \begin{cases} 
    S_k, & \text{for } r'_s \geq \theta \\
    U_k, & \text{for } r'_s < \theta 
    \end{cases}
\]

where the threshold, i.e., \( \theta \), is decided by finding the correlation coefficient between samples of known and unknown threats. If the correlation coefficient, i.e., \( r'_s(S_k, U_k) \) is higher than \( \theta \), then the system is found to be under the known threat, and for values, \( r'_s(S_k, U_k) \) lesser than \( \theta \), the system is found to be under the unknown threat. For deciding the threshold, we have considered the TON-IoT dataset. In this dataset, the subset containing the network traffic of the IoT network of devices like fridges, thermostats, garage doors, GPS trackers, Modbus, weather, and motion light is used in the experimental analysis. The experiment is run for critical analysis for six different attack categories, including backdoor, the man in the middle attack, DDoS, DoS, injection, and password. Different samples are taken with relevant features to find the correlation between the known threats from each category. Then, the Pearson correlation matrix is computed using the \texttt{corrcoef} function of the NumPy library in Python. From the results, it is observed that for known threats, the correlation factor is found to be greater than 65%. And for unknown threats, the correlation factor for samples from different threat categories is observed to be less than 63%, which is also depicted in Fig. 8. Thus, from the experimental analysis, the threshold value is decided as 0.63.

**Results and discussion**

In this section, we evaluated the performance of general ML-based solutions to detect known and unknown threat variants. We addressed the following question: Does an ML-based classifier work well in detecting threats with unknown/different distributions? We utilized a benchmark IoT dataset-the ToN-IoT dataset, collected over distributed data sources. We carried out eight experiments to check the performance over unknown threat variants. To evaluate the performance of general ML-classifiers, we divided the network dataset of the ToN-IoT dataset into eight subsets, as presented in Table 1 for the first testing scenario. The ToN-IoT dataset consists of nine attack families, including backdoor, injection, ransomware, DoS, DDoS, password, scanning, cross-site scripting, and man-in-the-middle attack. Figure 9 shows the differences in the distribution of individual threats represented by the normalized vector of corresponding features. This dataset has the telemetry data collected from IoT nodes, with different operating System logs and network data obtained from a practical depiction of a
medium-scale network at the UNSW Canberra Cyber Range and IoT labs (Australia). This is a publicly accessible dataset available at the ToN-IoT repository. These datasets were collected from the three-tier (edge, fog, and cloud) testbed comprising different IoT sensors like weather, Modbus, etc., designed to cover traces of network, process, memory, and hard disk in Windows and Linux-based IoT systems.

To analyze the behavior of multiple classifiers on different threat distributions between the source threat domain and target unseen threat variants, we have used the network traces of the ToN-IoT dataset. Each data sample in the network dataset has 44 features covering activities such as HTTP, violation, SSL, DNS, statistical, and connection, presented in Table 2. Before training the classifier, data undergoes various preprocessing phases to remove outliers, handle missing values, and normalize the data. Then the ML model is trained on the dataset with normal data samples and two attack types and tested on the dataset with different attack types. To follow the pattern, eight experiments are conducted. For example, the classifier is trained on the dataset IoT 1 with 20,000 normal samples and 20,000 samples for each category of DoS and DDoS attacks for the first testing scenario. In the first scenario, the learning model is trained on two threats i.e., dos and ddos, and tested for the remaining seven attack categories on all datasets from IoT 2 to IoT 8, comprising different attack categories of backdoor, injection, ransomware, scanning, cross-site scripting, man-in-the-middle attack, and password, with around 20,000 samples in each category. The 30 relevant features are taken in the final dataset, presented in Table 3.

The authors have divided the network dataset of the ToN-IoT dataset into nine subsets comprising 1 normal subset and 8 different attack categories of dos, ddos, backdoor, injection, ransomware, scanning, cross-site scripting, man-in-the-middle attack, and password, with around 20,000 samples in each category, except man-in-the-middle attack with 1043 samples. The threats for which the model is trained for are categorized under known threats. The threats for which the model is not trained for becomes unknown to the model. The experiments are conducted for three different scenarios. In the first scenario, the learning model is trained on two threats i.e., dos and ddos, and tested for the remaining seven attack categories. For the first scenario, in terms of known and unknown threats, the training and testing ratio is 2:7. In the second scenario, the learning model is trained on four threats i.e., dos, ddos, mitm, backdoor, and tested for the remaining five attack categories. For the second scenario, the training and testing ratio is 4:5. In the third scenario, the learning model is trained on six threats i.e., dos, ddos, mitm, backdoor, password, injection, and tested for the remaining three attack categories. For the third scenario, the training and testing ratio is 6:3.

To evaluate the performance of basic ML classifiers like Artificial Neural Networks (ANN), Support Vector Machine(SVM), etc., over both the domains of known and unknown threats, we use confusion matrix and several other performance criterion like accuracy, precision, recall, and f1-score.

### Table 1. Description of IoT datasets.

| Dataset     | Attacks      | Size of dataset |
|-------------|--------------|-----------------|
| ToN-IoT 1   | dos, ddos    | 60,000          |
| ToN-IoT 2   | Backdoor     | 20,000          |
| ToN-IoT 3   | Injection    | 20,000          |
| ToN-IoT 4   | Ransomware   | 20,000          |
| ToN-IoT 5   | Scanning     | 20,000          |
| ToN-IoT 6   | Cross-site scripting | 20,000 |
| ToN-IoT 7   | Man-in-the-middle | 1043          |
| ToN-IoT 8   | Password     | 20,000          |

### Figure 9. Distribution of different threats.
Achieving 100% precision in all experiments in the first scenario for detecting known and unseen threat variants reflects that the classifier correctly predicts all the non-malicious/normal data samples. This indicates that the model does not have any false positives. On the contrary, lower f1-scores falling in the range of 56 to 71% show the model’s inability to classify the unseen malicious data samples correctly. Thus, extensive experiments on the network dataset of Ton-IoT validate that learning classifiers (SVM) can achieve over 100% of f1-score and recall for detecting known threats. However, it significantly underperforms for the first testing scenario, with a 56.6% of f1-score and 39.4% of accuracy for detecting unseen threat variants (cross-site scripting), presented in Table 4. Figures 10, 11 shows the roc curve for known and unseen threats. The orange dot depicts the curve for known threats, indicating zero false positives and a 100% true positive rate. The green dot depicts the curve for unseen threat variants, indicating the high false-positive rate of 100%, though with a 100% true positive rate.

| Feature     | Service profile          | Feature     | Service profile          |
|-------------|--------------------------|-------------|--------------------------|
| Ts (timestamp) | Connection activity      | Ssl_version | SSL activity             |
| Src-ip      | Connection activity      | Ssl_cipher  | SSL activity             |
| Src-port    | Connection activity      | Ssl_resumed | SSL activity             |
| Dst-ip      | Connection activity      | Ssl_established | SSL activity           |
| Dst-port    | Connection activity      | Ssl_subject | SSL activity             |
| Proto       | Connection activity      | Ssl_issuer  | SSL activity             |
| Service     | Connection activity      | Weird_name  | Violation activity       |
| Duration    | Connection activity      | Weird_addl  | Violation activity       |
| Src-bytes   | Connection activity      | Weird_notice| Violation activity       |
| Dst-bytes   | Connection activity      | http_trans_depth | HTTP activity |
| Conn-state  | Connection activity      | http_method | HTTP activity             |
| Missed-bytes| Connection activity      | http_uri    | HTTP activity             |
| src_pkt     | Statistical activity     | http_version| HTTP activity             |
| rcv_ip_bytes| Statistical activity     | http_request_body_len | HTTP activity |
| dst_pkt     | Statistical activity     | http_response_body_len | HTTP activity |
| dst_ip_bytes| Statistical activity     | http_status_code | HTTP activity         |
| Dns-query   | DNS activity             | http_user_agent | HTTP activity         |
| dns_qclass  | DNS activity             | http_org_mime_types | HTTP activity       |
| dns_qtype   | DNS activity             | http_resp_mime_types | HTTP activity      |
| dns_rcode   | DNS activity             | label       | Data labelling           |
| dns_AA      | DNS activity             | type        | Data labelling           |
| dns_RD      | DNS activity             | dns_rejected | DNS activity           |
| dns_RA      | DNS activity             |            |                          |

**Table 2.** Features of TON datasets.

| Feature     | Feature          |
|-------------|------------------|
| Ts (timestamp) | Ssl_version      |
| Conn-state   | Ssl_cipher       |
| Src-port     | Dst-port         |
| Dst-bytes    | Src-bytes        |
| Missed-bytes | http_uri         |
| src_pkt      | dns_RD           |
| rcv_ip_bytes| http_request_body_len |
| dst_pkt     | http_response_body_len |
| dst_ip_bytes| http_status_code |
| Dns-query   | http_user_agent  |
| dns_qclass  | dns_rejected     |
| dns_qtype   | http_resp_mime_types |
| dns_rcode   | proto            |
| label       | dns_AA           |
| Ssl_established |            |

**Table 3.** Selected features of TON dataset for training.
Although, the model also underperforms for unknown threats in the other testing scenarios, with the accuracy dropping to 41.95 in the second scenario and 61.4 in the third scenario. However, it is observed that the model
outperforms for even the unknown threats in the third scenario, when the training and testing ratio is 6:3. Table 5 shows the performance of the model under the three different scenarios.

Figure 10a shows the confusion matrix for known threat detection. Out of the normal data points, all are correctly classified as normal except one data sample classified as anomalous. Out of the anomalous data, 8062 are correctly identified as malicious, whereas 3 are classified as normal. Figure 10b shows the confusion matrix for unknown threat detection where 11,514 attack data points are wrongly classified as normal. Figure 12 shows the performance of common ML classifiers, namely, Support Vector Machine (SVM), ANN, decision tree, and k nearest neighbor (Knn), over different threat variants. It is observed that these classifiers perform well when tested on data (ToN IoT-1) having the same distribution as the training data, i.e., known threats. However, for the different distribution, i.e., unknown threats (ToN IoT-2, 3, 4, 5, 6, 7, 8), the accuracy drops from 98 to around 39%. Hence, the basic ML models underperform for threats with different distribution in comparison to the distribution of the training set.

Conclusion
The smart home, empowered with declining costs in distributed computing technology, communications, and sensors, connects users and utilities, providing remote control and management. However, the expedited growth of interconnections among billions of smart nodes brings cyber-threats to homes with disastrous influence. Therefore, it is important to assess and mitigate these threats to avail benefits of smart services. Motivated by the discussion mentioned above, in this work, we presented a threat modeling methodology using DFDs up to level-2 depicting the process of handling known and unknown threats. Then, we have given an intelligent threat model based on a TL scheme with basic ML classifiers for known and unknown threat variants by reducing the differences in their distribution. Finally, the theoretical proof with the threshold decidability factor is given. In addition, the proposal demonstrated three distinctive testing scenarios in which the model outperform for the third scenario with training testing ratio of known unknown threats as 6:3. However, the overall results show that the traditional schemes underperform for unknown threats with different distribution with accuracy dropping to 39% and high false-positive rates for these variants.

In the future, the proposed threat model will be implemented, and domain adaptation will be employed to reduce the distance between data of different distributions for unknown threat detection. This work is confined to single smart home environment, further this work could be extended to inter-domain. Also, we can analyse the behaviour of the model by working upon the imbalanced dataset. The proposed threat model could be extended for mitigation and real-time implementation in heterogeneous scenarios.

Data availability
There is no data available to carry out this research work so data availability is not applicable for this manuscript.

| Dataset | T-1 (2 known:7 unknown), % | T-2 (4 known:5 unknown) | T-3 (6 known:3 unknown) |
|---------|----------------------------|--------------------------|--------------------------|
| TON IoT-4 | 40.4                      | 43.37                    | 55.02                    |
| TON IoT-5 | 42.4                      | 48.36                    | 57.92                    |
| TON IoT-6 | 39.4                      | 41.95                    | 61.4                     |

Table 5. Performance on three different testing scenarios.
References

1. Jadav, N. et al. Deep learning and onion routing-based collaborative intelligence framework for smart homes underlying 6g networks. IEEE Trans. Netw. Serv. Mang. https://doi.org/10.1109/TNSM.2022.3164715 (2022).
2. Gyarmathy, K. Comprehensive guide to IoT statistics you need to know in 2020. https://www.vxchnge.com/blog/iot-statistics. (Accessed 2021).
3. Networks, P. 2020 unit 42 IoT threat report. https://unit42.paloaltonetworks.com/iot-threat-report-2020/. (Accessed 2021).
4. Doffman, Z. Cyberattacks on IoT devices surge 300 percent in 2019, ‘measured in billions’, report claims. https://www.forbes.com/sites/zakkdoffman/2019/09/14/dangerous-cyberattacks-on-iot-devices-up-300-in-2019-now-rampant-report-claims. (Accessed 2021).
5. Seals, T. IoT attacks skyrocket, doubling in 6 months. https://threatpost.com/iot-attacks-doubling/169224/. (Accessed 2021).
6. Kaakkar, R., Gupta, R., Agrawal, S. & Tanwar, S. Blockchain-based secure and trusted data sharing scheme for autonomous vehicle underlying 5g. J. Inf. Secur. Appl. 67, 1–39. https://doi.org/10.1016/j.isa.2022.103179 (2022).
7. Gupta, R., Patel, M. M., Tanwar, S., Kumar, N. & Zeadally, S. Blockchain-based data dissemination scheme for 5g-enabled software-defined UAV networks. IEEE Trans. Green Commun. Netw. 5, 1712–1721 (2021).
8. Ciso, D. Dark nexus: The evolving IoT botnet targets variety of devices says bitdefender research - dynamics. https://dynamcisco.com/dark-nexus-the-evolving-iot-botnet-targets-variety-of-devices-says-bitdefender-research. (Accessed 2021).
9. Anand, P., Singh, Y., Selwal, A. Internet of things (IoT): Vulnerabilities and remediation strategies. In Expert Syst. 39, e12829. (2021).
10. Anand, P., Singh, Y. & Selwal, A. Internet of things (IoT): Vulnerabilities and remediation strategies. In The International Conference on Recent Innovations in Computing, 265–273 (Springer, 2020).
11. Kaakkar, R., Gupta, R., Tanwar, S. & Rodrigues, J. J. Coalition game and blockchain-based optimal data pricing scheme for ride sharing beyond 5g. IEEE Syst. J. (2021).
12. Anand, P. et al. IoT vulnerability assessment for sustainable computing: Threats, current solutions, and open challenges. IEEE Access 8, 168825–168853. https://doi.org/10.1109/ACCESS.2020.3022842 (2020).
13. Nguyen, C. T. et al. Transfer learning for future wireless networks: A comprehensive survey. CoRR arXiv:2102.07572 (2021).
14. Samee, N. & Shashi, M. Deep transductive learning transfer framework for zero-day attack detection. ICT Express 6, 361–367. https://doi.org/10.1016/j.ictexp.2020.03.003 (2020).
15. Zhang, Y. & Yan, J. Domain-adversarial transfer learning for robust intrusion detection in the smart grid. In 2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), 1–6. https://doi.org/10.1109/SmartGridComm.2019.8909793 (2019).
16. Papjouh, H. H., Javidan, R., Khayami, R. and Dehghan, K. A. A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. IEEE Trans. Emerg. Top. Comput. 7, 314–323. https://doi.org/10.1109/TETC.2019.2907942 (2020).
17. Gao, J., Chai, S., Zhang, R. & Xia, Y. Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis. Energies 12, 1223. https://doi.org/10.3390/en12071223 (2019).
18. Kumar, P., Gupta, G. P. & Tripathi, R. Design of anomaly-based intrusion detection system using fog computing for IoT network. Autom. Control Comput. Sci. 55, 137–147 (2021).
19. Almiani, M., AbsGhadleh, A., Al-Rabyeh, A., Atiehi, S. & Razaque, A. Deep recurrent neural network for IoT intrusion detection system. Simul. Model. Pract. Theory 101, 102031. https://doi.org/10.1016/j.simpat.2019.102031 (2020).
20. Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J. & Alazab, A. Hybrid intrusion detection system based on the stacking ensemble of c5 decision tree classifier and one class support vector machine. Electronics 9, 173. https://doi.org/10.3390/electronics9040173 (2020).
21. Al-Hawwreh, M., Mostafa, N., Garg, S. & Hossain, M. S. Deep learning-enabled threat intelligence scheme in the internet of things networks. IEEE Trans. Netw. Sci. Eng. https://doi.org/10.1109/TNSE.2020.3032415 (2020).
22. Li, H. et al. Dart: Detecting unseen malware variants using adaptation regularization transfer learning. In ICC 2019–2019 IEEE International Conference on Communications (ICC), 1–6. https://doi.org/10.1109/ICC.2019.8761958 (2019).
23. Vu, L., Nguyen, Q. U., Nguyen, D. N., Hoang, D. T. & Dutkiewicz, E. Deep transfer learning for IoT attack detection. IEEE Access 8, 107335–107344. https://doi.org/10.1109/ACCESS.2020.3000476 (2020).
24. Iaved, D., Gao, T. & Khan, M. T. Sdn-enabled hybrid dl-driven framework for the detection of emerging cyber threats in IoT. Electronics 10, 918. https://doi.org/10.3390/electronics10080918 (2021).
25. Zhao, J., Shetty, S. & Pan, J. W. Feature-based transfer learning for network security. In MILCOM 2017–2017 IEEE Military Communications Conference (MILCOM), 17–22. https://doi.org/10.1109/MILCOM2017.8170749 (2017).
26. Zhao, J., Shetty, S., Pan, J. W., Kamhous, C. & Kwiat, K. Transfer learning for detecting unknown network attacks. EURASIP J. Inf. Secur. 2019, 1–13 (2019).
27. Singla, A., Bertino, E. & Verma, D. Preparing network intrusion detection deep learning models with minimal data using adversarial domain adaptation. In Proceedings of the 15th ACM Asia Conference on Computer and Communications Security, ASIA CCS ’20, 127–140. https://doi.org/10.1145/3320269.3384718 (Association for Computing Machinery, 2020).
28. Shevchenko, N., Chick, T. A., O’Riordan, P., Scanlon, T. P. & Woody, C. Threat modeling: a summary of available methods. Tech. Rep., Carnegie Mellon University Software Engineering Institute Pittsburgh United . . . (2018).
29. Ciso, D. An automated threat modeling solution that secures and scales the enterprise software development life cycle. https://threatmodeler.com/ (2021).
30. Team, M. Microsoft threat modeling tool. https://docs.microsoft.com/en-us/azure/security/develop/threat-modeling-tool. (Accessed 2021).
31. Shotack, A. Experiences threat modeling at microsoft. MODSEC@ MoDELS 2008 (2008).
32. Agarwal, A. et al. Vast methodology: Visual, agile, and simple threat modeling. Var. Interviews. Transform. Oppor. Prescott Val. (2016).
33. Kim, J., Sim, A., Kim, J., Wu, K. & Hahm, J. Improving botnet detection with recurrent neural network and transfer learning (2021).
34. Anand, P. et al. IoV: Internet of vulnerable things? threat architecture, attack surfaces, and vulnerabilities in internet of things and its applications towards smart grids. Energies 13, 4813 (2020).
35. Robles, R. I. & Kim, T.-H. A review on security in smart home development (2010).
36. NoorMoustafa. Ton-IoT datasets. https://ieeexplore.ieee.org/documents/toniot-datasets. (Accessed 2021).

Acknowledgements
Taf University Researchers Supporting Project number (TURSP-2020/126), Taif University, Taif, Saudi Arabia.
Author contributions
Conceptualization, P.A., Y.S., H.S. and S.T.; Data curation, M.D.A.; Formal analysis, S.T., P.A., and Y.S.; Methodology, P.A., H.S., and Y.S.; Domain knowledge, Y.S., and S.T.; Resources, P.A.; Software, M.D.A., H.S., and Y.S.; Validation, S.T., H.S., and Y.S.; Writing—original draft, P.A., S.T., and H.S.; Writing—review and editing, S.T., Y.S., P.A., and M.D.A.

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to H.S. or S.T.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

© The Author(s) 2022