A Behavioral Trust Model for Internet of Healthcare Things using an Improved FP-Growth Algorithm and Naïve Bayes Classifier

Saiful Azad[1], Amin Salem Saleh[2], Mufti Mahmud[3], M. Shamim Kaiser[4], Md. Saefullah Miah[2]
[1]Department of Computer Science and Engineering, Green University of Bangladesh, Dhaka, Bangladesh
[2]Faculty of Computing, College of Computing & Applied Science, University Malaysia Pahang, Malaysia
[3]Department of Computer Science, Nottingham Trent University, Clifton, NG11 8NS Nottingham, UK
[4]Institute of Information Technology, Jahangirnagar University, Savar, 1342 Dhaka, Bangladesh
Email: sazadm684@gmail.com, mufti.mahmud@ntu.ac.uk, mskaiser@juniv.edu, md.saefullah@gmail.com

2021 3rd International Conference on Sustainable Technologies for Industry 4.0 (STI) | 978-1-6654-0009-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/STI53101.2021.9732588

***Abstract*—Healthcare 4.0 has revolutionized the delivery of healthcare services during the last years. Facilitated by it, many hospitals have migrated to the paradigm of being smart. Smartization of hospitals has reduced healthcare costs while providing improved and reliable healthcare services. Thanks to the Internet of Healthcare Things (IoHT) based healthcare delivery frameworks, integration of many heterogeneous devices with varying computational capabilities has been possible. However, this introduced a number of security concerns as many secure communication protocols for traditional networks can not be verbatim employed on these frameworks. To ensure security, the threats can largely be tackled by employing a Trust Management Model (TMM) which will critically evaluate the behavior or activity pattern of the nodes and block the untrusted ones. Towards securing these frameworks through an intelligent TMM, this work proposes a machine learning based Behavioral Trust Model (BTM), where an improved Frequent Pattern Growth (*i*FP-Growth) algorithm is proposed and applied to extract behavioral signatures of various trust classes. Later, these behavioral signatures are utilized in classifying incoming communication requests to either trustworthy and untrustworthy (trust) class using the Naïve Bayes classifier. The proposed model is tested on a benchmark dataset along with other similar existing models, where the proposed BMT outperforms the existing TMMs.**

***Index Terms*—Internet of things, secure healthcare framework, FP-growth algorithm, Naïve Bayes classifier.**

I. Introduction

Internet of Healthcare Things (IoHT) [1] aims to facilitate the ministration of care services to patients through improved analytics of gathered health data to expedite the healthcare decision making and service delivery process. It also assists healthcare professionals by taking off some workload through (semi-)automated remote monitoring of patient health, their treatment progress, etc. [2]. According to an estimation reported in [3], despite being slow in adopting new technologies, the healthcare sector will observe an incredible growth of 50 million connected devices worldwide by 2021.

Since these devices are connected to the global information superhighway for their ubiquitous access, they are targeted by many bandits. This vulnerability is even exacerbated, in comparison to the conventional network, as most of these IoHT frameworks (IoHTF) are comprised of heterogeneous devices with varying computational and memory capabilities to which many existing secure communication protocols can not be verbatim applied. To the rescue of such a situation, a Trust Management Model (TMM) can be additionally employed alongside the communication protocols to tackle these threats and attacks by exploiting the activity patterns of the connected devices [4], and hence, is the focus of this paper.

Over the last years, several TMMs have been proposed targeting different applications [5], [6], [7]. However, one of the major drawbacks of many of these available models is that, they are unable to adapt to the dynamically evolving threat profiles. An effective TMM is required to learn about the evolving threat profiles and automatically account for those changes and adapt its safeguarding strategy. It is, therefore, imperative to teach the TMM these dynamic threat profiles and set the retaliation strategy on whether or not to authorise specific connection requests. To partially achieve this, several intelligent models have been proposed in the literature.

Among the existing ones, an Adaptive Neuro-Fuzzy Inference System based trust model targeting Neuroscience applications has been proposed in [4] which suffers from low accuracy in a generalized scenario. Another trust model for pervasive computing based on Apriori association rules learning (AARL) and Bayesian classification has been proposed in [8]. Limitations of the AARL methods is well-known (see [9]) and scalability is a major one. That is, when the size of a database becomes large, the AARL algorithm fails to fit it into the memory, warranting a large number of memory reads in each iteration of the algorithm. In such scenarios, the Frequent Pattern Growth (FP-Growth) algorithm performs better as it scans a database only twice in comparison to the AARL algorithm which scans the transactions in each iteration. Moreover, the FP-Growth algorithm extracts more relevant frequent patterns (also referred as behavioral signatures) and their respective association rules over AARL [10]. This work improved the standard FP-Growth algorithm to reduce the execution time in exploring the behavioral signatures of the devices and classify them to appropriate trust classes using the Naïve Bayes classifier.

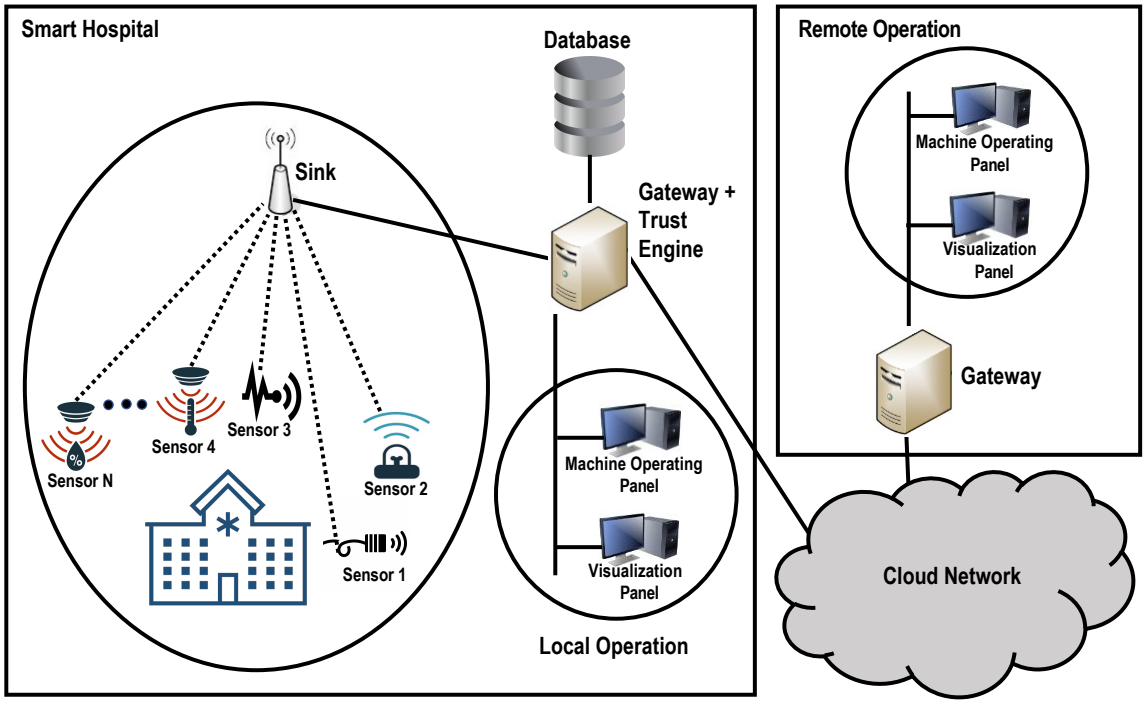

Fig. 1. A IoHTF architecture that consists of two networks, namely an hospital network and a cloud network, is demonstrated in this figure. Here, a trust engine (a machine that is running the proposed BMT model) is placed in the hospital network.

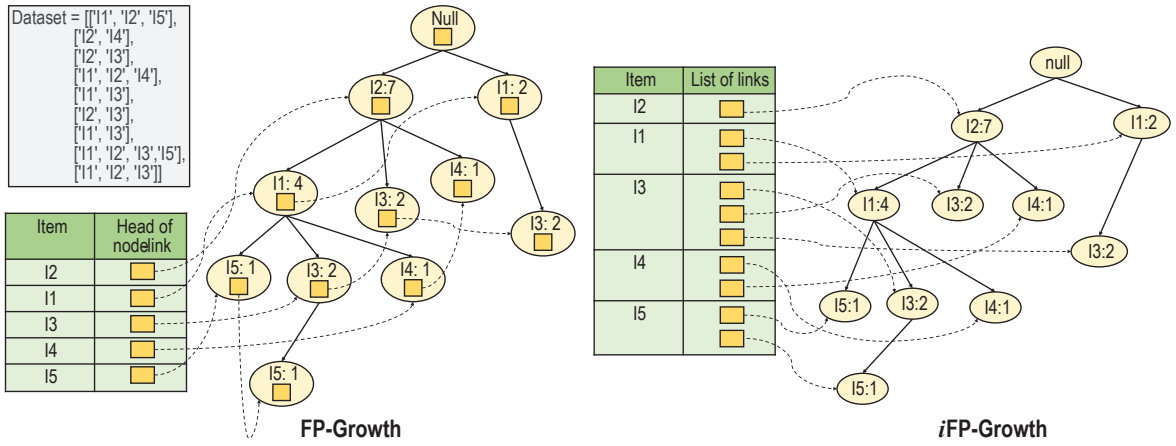

Fig. 2. This figure demonstrates the proposed improvement over the original FP-Growth algorithm. As could be seen in the *i*FP-Growth, a link table is introduced instead of the header table, which offers a suffix list for an item from the link table; and thus, eliminates the necessity of generating the list using a singly linked list involving the header table.

II. Network Framework

Although, the proposed BTM is incorporatable to any IoHT based network frameworks; however, for guiding principle, it is incorporated in a framework that is proposed in [4], which is an attempt by the authors towards supporting heterogeneous devices as demonstrated in Fig. 1. This cloud-based IoT framework comprises of three main components: the user end (providing analyzed and processed data to users, e.g., doctors, caregivers, and researchers), the IoT end (composed of data generating devices), and the cloud component (providing access and connectivity, and processing and analysis of data). For detail descriptions of these components, which is out of the scope of this paper, please read [4].

At the perception layer or IoT end, various data acquisition devices are connected to their respective transceivers which forward acquired data to the cloud through the IoT gateway via the local server, whether for data analytics or simply for storage. Since most of the necessary data are injected to the

cloud through the local server from this layer, we opt to install our proposed BTM model here in this local server as a service, and named it as trust engine. The key responsibilities of the trust engine are: *i*) acquiring data from various sensors and actuators using the sink, *ii*) delivering operational directives to the respective nodes, *iii*) identifying the behavioral pattern of various nodes, *iv*) classifying the behaviors of various nodes between trustworthy and untrustworthy, and *v*) delivering the data of the trustworthy node to the various panels and the gateway. The latter is connected to the cloud network, which is necessary for remote operations.

In this network architecture, the users can visualize as well as access the data with adequate permission from the web server using respective Application Programming Interfaces (APIs). A database is also connected to the web server to store relevant data and activities, which are later shared using APIs. Here, it is noteworthy to mention that there are several secure protocols like IPSec, HTTPS, SSL, and others are proposed to secure the communication between the gateways and the end devices at the cloud. Hence, the most vulnerable area in this framework is the IoT end, which is the focus of this work. It is a known fact that the internal attacks on a network have significantly more adverse effects than the external attacks [11].

III. Improved FP-Growth or *i*FP-Growth

In FP-Growth algorithm, a frequent pattern is generated by representing a database in the form of a tree — called FP-tree — without the need for candidate generation. Here, an item of an itemset in the database is utilized in generating a node in this tree. Once an itemset, r is read from a database, the algorithm checks whether the prefix of r maps to a path in the tree or not. If it maps, the support counts of the corresponding nodes are increased; conversely, new nodes are created with a support count of 1 and added to the tree. Again, a FP-Tree uses references for connecting nodes that have the identical item and thus, creates a singly linked list (as demonstrated in Fig. 2 in FP-Growth). These singly linked lists speed up the discovery process of individual items from the tree by eliminating the requirements of tree traversal. All these nodes that are discovered using a singly linked list create a suffix list, which is later utilized in extracting frequent patterns. The heads of these node-links or singly linked lists are saved in a table, called header table. To obtain this advantage of FP-Growth, several other tree-based algorithms also incorporated this technique including Associated sensor pattern mining of data stream (ASPMS) [12] and Enhanced FP-Growth (EFP) [13]. Hence, any improvement here implies similar improvement opportunity on those analogous algorithms.

Our improvement relies on the intuition that when a tree is considerably large and an item becomes nodes of many subtrees due to the high variability of the itemsets, generating a suffix list for that item using a singly linked list is time consuming. Conversely, a link table that accumulates references of all the nodes belongs to every item (as demonstrated in Fig. 2 in *i*FP-Growth) would reduce the aggregate executing duration for extracting frequent patterns; and thus, for respective association rules mining without compromising the quality of the rules. Therefore, in *i*FP-Growth algorithm, the header table is replaced with a link table that eliminates the necessity of generating a suffix list for an item using a singly linked list. In addition, due to inclusion of the link table, space complexity of the algorithm will not increase since a node is no more keeping the reference of its adjacent node.

IV. Proposed System

A. Generating Threat Profiles and Identifying Attribute Vectors

The first task of constructing a behavioral trust model for a system is to identify the threat profile of that system, which is IoHTF in our case. It is essential in developing such a model because threat profiles and risk analyses are intrinsically related. It identifies the specific threats that are most likely to put a system at risk. Here, it is necessary to take into account that threats can come from external sources as well as from internal sources. Therefore, calculating the level of trust of each node (which includes devices, humans, techniques and relevant aspects) is mandatory.

For that, following attributes are listed in [8], namely Node Identification (i), Counting Trust (CT), Counting Untrust (CU), Transactions Context (TC), Direct Knowledge (DK), Source Entity (SE), Time (T), Hierarchical Level (HL), and Trust Score (TS). For more details, readers are requested to read [8]. These attributes constitute a tuple, called Trust Tuple (TT), which are saved in a database to form a set, called Trust Tuple Set (TTS), for all the interested node in the network for future use.

B. Finding Behavioral Signatures using iFP-growth

To identify the behavioral signature of a node, i, a subset, TTS_i is extracted from the TTS and categorized it based on a specific TS class (trustworthy or untrustworthy). Afterwards, the *i*FP-growth algorithm is run on the categorized subset to identify the associations or frequent patterns among the items for that specific TS class. These patterns represent the behavioral signatures of that node for that TS class. Afterwards, they are saved in a database as demonstrated in Fig. 3. Note that such signatures may change over time; and hence, need to be revised after a fixed epoch.

C. Extracting Feature from Applicant Tuple

When a node requests a connection, the trust level of the requesting node is determined prior sanctioning it. For that, the trust engine constitutes an Applicant Tuple (AT) following the analogous procedure of TT and updates its experience by integrating it in TTS for future reference. Afterwards, all different frequent itemsets are extracted from that AT. For instance, assuming the tuples as 3-dimensional vectors, the applicant tuple is: $< A, B, C >$. The different itemsets that can be extracted from here are: $\{A\}, \{B\}, \{C\}, \{A, B\}, \{A, C\}$, and $\{B, C\}$. These itemsets are joined together along with the

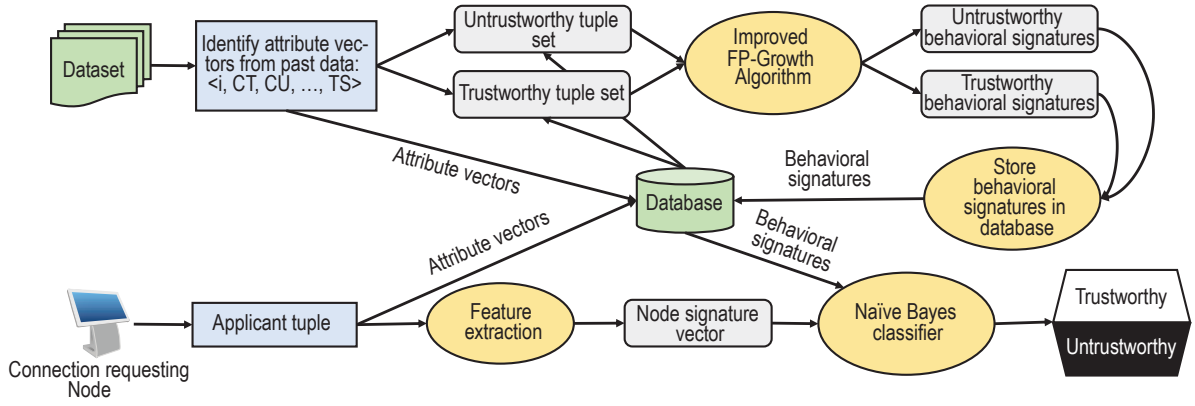

Fig. 3. The proposed ML based behavioral trust model using *i*FP-Growth and Bayesian classifier.

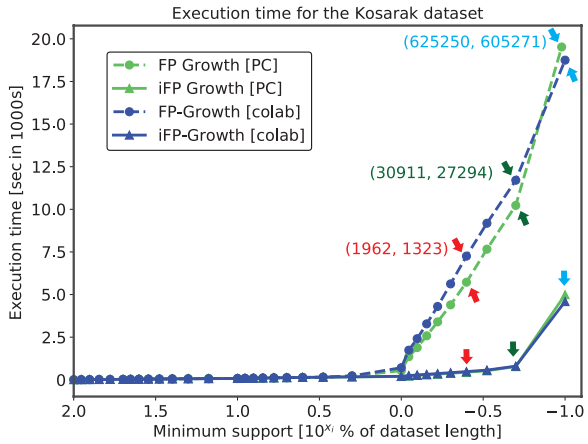

Fig. 4. Performance in terms of execution time of FP-Growth and *i*FP-Growth for two execution environments (namely PC and Google Colab Pro) for the Kosarak dataset is displayed. Here, an oval groups execution times of a certain minimum support; and number of extracted frequent patterns and number of respective association rules for that minimum support are displayed using 2-tuple data structure alongside the oval following the same sequence.

applicant tuple to form a Node Signature Vector (NSV), which is then passed to the Naïve Bayes classifier to determine the trust class.

D. Making Decisions using Bayesian Classifier

After receiving an NSV, the task of the Naïve Bayes classifier is to identify the suitable TS class of it, which is unknown. Let h be the TS class (trustworthy and untrustworthy), t be a specific trust class (either trustworthy or untrustworthy), and the NSV consists of n number of itemsets.

To find out, whether this NSV is belong to a specific t, it is necessary to estimate: the prior probability of t, $P(t)$; the prior probability of the NSV, $P(NSV)$; and the likelihood, which is the probability of NSV given t, $P(NSV|t)$. In this paper, likelihood is estimated as the product of probability of every itemset in NSV given t. Again, the probability of an itemset given t is estimated using the Laplace estimator [14] to avoid the zero probability condition, which is probable in our scenario. Hence, the probability that t holds for the observation NSV can be found as the ratio of the product of $P(t)$ and likelihood, and $P(NSV)$.

The obtained probability values for various t can be utilized to take the final decision. In this paper, the highest probability value between trustworthy and untrustworthy decides NSV's class; and thus, the trust class of AT. For instance, if the trustworthy score is $3.85E-19$ and the untrustworthy score is $6.40E-19$; since the latter scores the highest probability value, the final decision is untrustworthy.

V. Experimental Results

In this paper, all the relevant codes including algorithms, systems, and experiments are implemented using python and its necessary packages including pyfpgrowth, random, Scikit-learn, pandas, numpy, and matplotlib. All implementation codes and their respective datasets are currently available in [15] to access upon request.

A. Performance of iFP-Growth

1) Experimental Setup: To stress the performance of *i*FP-Growth and FP-Growth, a benchmark dataset, Kosarak [15], is chosen since it contains considerably larger number of itemsets $(990,002)$ and relatively many distinct items $41,270$. For both the algorithms, minimum supports are calculated and passed to their respective functions to extract frequent patterns and to generate respective association rules. Their execution times of running various experiments for various minimum supports are recorded in text files for further use. In addition, all the experiments are performed in two different environments: *i*) a PC with the specification — Intel(R) Core(TM) i7-4600U CPU @ 2.10GHz 2.69GHz, 8 GB RAM, 64-bit Windows 10 Pro operating system and *ii*) Google colab pro or colaboratory pro or colab pro. The justification for choosing two environments is that the execution times on a PC may vary based on other background programs and/or parallel programs running at that time.

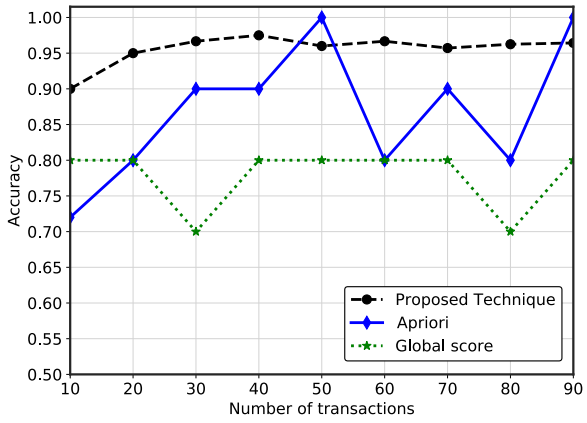

Fig. 5. Accuracy of the compared models are plotted with respect to various numbers of transactions.

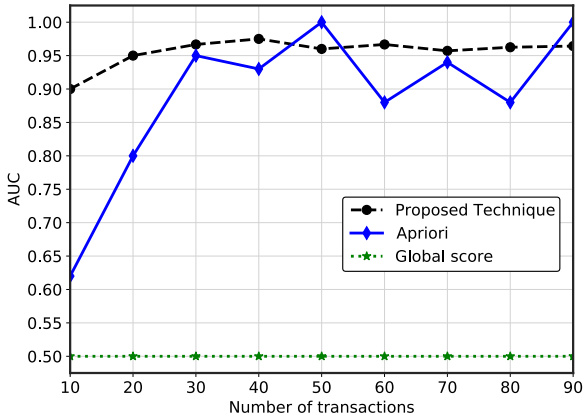

Fig. 6. AUC of the compared models are plotted with respect to various numbers of transactions.

2) Results and Discussions: Acquired results are plotted against various minimum supports in Fig. 4. As could be observed from the figures is that execution times increase with decreasing minimum supports for both the algorithms due to tree expansion by adding more number of nodes. Consequently, more frequent patterns are extracted, and thus, more respective association rules are generated. Nevertheless, FP-Growth spends several magnitude higher execution times for lower minimum supports with respect to its predecessor, *i*FP-Growth, for both the execution environments. The key rationales behind such performance are: *i*) with lower minimum supports, size of the trees expand due to the inclusion of many nodes, and thus, duration of generating suffix lists for various items using singly linked lists increase; and *ii*) the variability of the itemsets also plays an import role here as an item becomes nodes of many different subtrees, and hence, the length of the singly linked list grows, and thus, the duration of extracting a suffix list for an item also extends.

Conversely, as *i*FP-Growth eliminates the necessity of generating a suffix list for an item by offering it from the link table, it takes several magnitude less execution time for lower minimum supports while extracting the same number of frequent patterns and their respective association rules. For Kosarak dataset, average execution time spent by FP-Growth algorithm is $19,268.76$ sec for the minimum support of 990; whereas, it is only $4,790.03$ sec for the *i*FP-Growth algorithm. Due to the improved performance of the *i*FP-Growth algorithm, it is selected for conducting the rest of the experiments.

B. Performance of BTM

1) Experimental Setup: For evaluating the performance of the proposed model, a benchmark dataset, named Dishonest Internet Users Dataset [8], has been utilized in this work. Since the proposed system is a supervised system, 70% of the itemsets in the dataset are utilized during the training phase where it extracts the behavioral signatures for both trustworthy and untrustworthy trust classes employing *i*FP-Growth with the minimum support of 2. The rest of the itemsets (30%) are utilized during the testing phase. Here, two performance metrics, namely Accuracy (Ac) and Area Under ROC curve (AUC) are employed, which are calculated using the standard expressions as in [8]. For the other compared techniques, the results are acquired from [8] since it employs an analogous experimental setup and an identical dataset.

Again, during the experiments, a scenario similar to the malicious attack is designed with two communicating nodes. Here, one is the applicant node ($N1$) who would like to communicate and the other is the receiver ($N2$) or the trust engine (as in the network architecture in Fig. 1), who sanction the connection based on the estimated trust level. To stress the difficulty level of the classification, $N1$ will alternate its behavior by acting as trusted for certain time (ranges between 3 to 5 applicant tuples) and by acting as untrusted for once (i.e., 1 applicant tuple), to maintain its trusted behavior.

C. Results and Discussions

Figure 5 presents Ac of the proposed model with respect to the other existing models, such as TMM Apriori (TMMA) and global score in [8]. Since there are only two TS class levels, Ac is a reliable metric to evaluate the performance of the model. As could be observed from the figure is that the average score of Ac of the proposed BMT is around 0.95; whereas it is below 0.9 for TMMA. It is because *i*FP-Growth generates more relevant frequent patterns over Apriori as reported in [10]. On the other hand, global score receives lesser Ac with an average score below 0.8.

The results of AUC for the proposed model and other compared models are plotted in Fig. 6. This metric is important since it estimates the quality of the classifier that is utilized in the system. A perfect classifier scores 1 and an imperfect classifier scores 0. Note that no realistic classifier should score less that 0.5. Hence, 0.5 is the global score, which confirms that the global-based estimator has a behavior similar to a random classifier. Conversely, the proposed model achieves an an average score of 0.97 and for TMMA it is 0.87. Hence, the proposed technique performs better than the TMMA.

VI. Conclusion

In this paper, a ML based BMT is proposed to tackle various threats and attacks of the IoHT networks. It is designed using the *i*FP-Growth algorithm — an improved FP-Growth algorithm that is proposed in this paper which reduces the execution time for extracting frequent patterns; and thus, respective association rules. Afterwards, the Naïve Bayes classifier is employed to classify any connection request between trustworthy and untrustworthy, and the connection is sanctioned when founds trustworthy. The proposed BMT is tested on a benchmark dataset. From the acquired results, it can be concluded that the proposed BMT overpowers other compared models; and hence, can be considered as a suitable alternative to the existing solutions for the IoHT networks.

Acknowledgment

This work was supported in part by the FRGS project under Grant FRGS/1/2019/ICT04/UMP/02/1 (or RDU1901109), by the RDU project under Grant RDU182201-3, and by the Center of Research, Innovation, and Transformation of Green University of Bangladesh.

References

[1] M. Asif-Ur-Rahman, F. Afsana, M. Mahmud, M. S. Kaiser, M. R. Ahmed, O. Kaiwartya, , and A. James-Taylor, "ehealth: Survey on research projects, comparative study of telemonitoring architectures and main issues," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4049–4062, 2019.

[2] O. Hamdi, M. A. Chalouf, D. Ouattara, and F. Krief, "ehealth: Survey on research projects, comparative study of telemonitoring architectures and main issues," *J. Netw. Comput. Appl.*, vol. 46, no. 1, pp. 100–112, 2014.

[3] Statista, "Number of connected things/devices worldwide by vertical from 2015 to 2021," 2017. [Online]. Available: https://www.statista.com/statistics/626256/connected-thingsˆdevices-worldwide-by-vertical (Accessed on May 5, 2020).

[4] M. Mahmud, M. S. Kaiser, M. M. Rahman, M. A. Rahman, A. Shabut, S. Al-Mamun, and A. Hussain, "A brain-inspired trust management model to assure security in a cloud based iot framework for neuroscience applications," *Cognit Comput*, vol. 10, no. 5, pp. 864–873, 2019.

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

[6] J. Pawlick and Q. Zhu, "Strategic trust in cloud-enabled cyber-physical systems with an application to glucose control," *IEEE Trans. Inf. Forensics Secur.*, vol. 12, no. 12, pp. 2906–2919, 2017.

[7] H. Zhao, D. Sun, H. Yue, M. Zhao, S. Cheng, J. Hopkins, and G. Burnie, "Dynamic trust model for vehicular cyber-physical systems," *Int. J. Netw. Secur.*, vol. 20, no. 1, pp. 157–167, 2018.

[8] G. D'Angelo, S. Rampone, and F. Palmieri, "Developing a trust model for pervasive computing based on apriori association rules learning and bayesian classification," *Soft Comput.*, vol. 21, no. 1, pp. 6297—-6315, 2017.

[9] M. Al-Maolegi and B. Arkok, "An improved apriori algorithm for association rules," *IJNLC*, vol. 3, no. 1, pp. 21–29, February 2014.

[10] P. Fournier-Viger, J. Lin, B. Vo, T. T. Chi, J. Zhang, and H. B. Le, "A survey of itemset mining," *Wires Data Min. Knowl.*, vol. 7, no. 4, July 2017.

[11] N. Giandomenico and J. de Groot, "Insider vs. outsider data security threats: What's the greater risk?" Digital Guardian's Blog, April 2018. [Online]. Available: https://digitalguardian.com/blog/insider-outsider-data-security-threats (Accessed on April 21, 2020).

[12] S. Rewatkar and A. Pimpalkar, "Aspms based approach to mine frequent pattern from wsn dataset," *IJARCSSE*, vol. 5, no. 11, pp. 151–154, 2015.

[13] A. A. G. Al-Hamodi, S. Lu, and Y. E. A. Al-Salhi, "An enhanced frequent pattern growth based on mapreduced for mining assocuatoin rules," *IJDKP*, vol. 6, no. 2, pp. 19–28, 2016.

[14] I. H. Sarkar, "A machine learning based robust prediction model for real-life mobile phone data," *Internet Things*, vol. 5, no. 1, pp. 180–193, 2019.

[15] A. S. S. Bllagdham and S. Azad, "Behavioral trust model," 2020, bTM 1.0. [Online]. Available: https://drive.google.com/drive/u/0/folders/1bD0OAPgfi9J8RPIqOCjJ5ph3qagUbaTx (Accessed on April 21, 2020).