Deep Learning for Secure Mobile Edge Computing in Cyber-Physical Transportation Systems

Yuanfang Chen, Yan Zhang, Sabita Maharjan, Muhammad Alam, and Ting Wu

Abstract

MEC is able to be used to execute the compute-intensive applications on the edge of transportation networks directly. As a result, the communications traffic is substantially increased among the connected edge devices. Therewith, communications security is emerging as a serious problem, and as an important research issue of the communications security, active feature learning is studied in this article for actively detecting unknown attacks. A model based on deep learning is designed to learn attack features. This model uses unsupervised learning to accomplish the active learning process. In the evaluation, 10 datasets are used to conduct experiments. We compare our model with four other machine-learning-based algorithms, and the comparative results illustrate that our model has a 6 percent gain in accuracy.

Introduction

A cyber-physical transportation system (CPTS) uses huge heterogeneous sensors and mobile wireless devices to achieve the capabilities of sensing, communications, and traffic control. Assisted by these capabilities, in the CPTS, mobile edge computing (MEC) can be used to enable heterogeneous things such as connected moving vehicles and traffic sensors to collectively execute real-time compute-intensive applications on the network edge directly [1, 2] (an application scenario is displayed in Fig. 1). In this kind of MEC, the communications use the wireless mode between sensors and traffic controllers, and the open property of the wireless process is appealing to attackers because it makes the communications easy to threaten by security attacks; as an example, eavesdropping and jamming attacks are illustrated in Fig. 2. Moreover, according to the power of information transmission, eavesdroppers and jammers can self-adjust power strategies to maximally affect the wireless communications. For detecting eavesdropping and jamming attacks, active feature learning is necessary to pursue the change in power adjustment. Motivated by active feature learning, this article proposes a model based on deep learning, and unsupervised learning is utilized to implement the active process of feature learning on security attacks.

In the past 10 years, security in CPTS has attracted attention from industry and academia. Encryption-based methods have been proposed. However, such methods need much computing power, so their applications in the MEC environment are limited due to huge sensors that are limited in computing power. To consider this limitation, physical-layer security methods have been studied [3, 4]. Moreover, in security studies, learning the feature of a security attack is an important research issue, and some research has been carried out in this direction. However, the existing work has not considered conducting the learning in an active manner, that is, without labeling the data used for the learning process in a manual way.

In this article, a model is proposed using the deep learning framework. This model uses a deep belief network to extract attack features actively. As a study case, we use this model to learn the attack features on MEC devices with the Android system[1] installed, and eavesdropping and jamming attacks are included in the open MEC environment.

The article's contributions are described as follows:
- The model using the deep learning framework is designed to learn the attack features for the MEC environment, in which huge heterogeneous wireless devices are connected together.
- The learning process of the proposed model is active for the features of malicious attacks, that is, without labeling the data used for the learning process in a manual way. Unsupervised learning is used to achieve the active process for actively detecting unknown attacks.
- This article visualizes the MEC's communications security in the CPTS application scenario.

Feature-Learning-Based Detection: State of the Art

Depending on the method of feature learning, we can classify the existing methods of attack detection, and there are two categories: static-learning-based attack detection and dynamic-learning-based attack detection.

Static-Learning-Based Attack Detection

In this kind of detection, the methods examine the source code of attacks without running the code. Moreover, they analyze the static features of security attacks to learn the patterns of these attacks. Through this learning, the malicious

[1] Many MEC devices have the Android operating system installed. In this article, we use the attack data from these Android devices to evaluate the proposed model.

Digital Object Identifier: 10.1109/MNET.2019.1800458

Yuanfang Chen, Muhammad Alam, and Ting Wu are with Hangzhou Dianzi University; Yan Zhang and Sabita Maharjan are with University of Oslo. Yuanfang Chen and Muhammad Alam are the corresponding authors.

0890-8044/19/$25.00 © 2019 IEEE

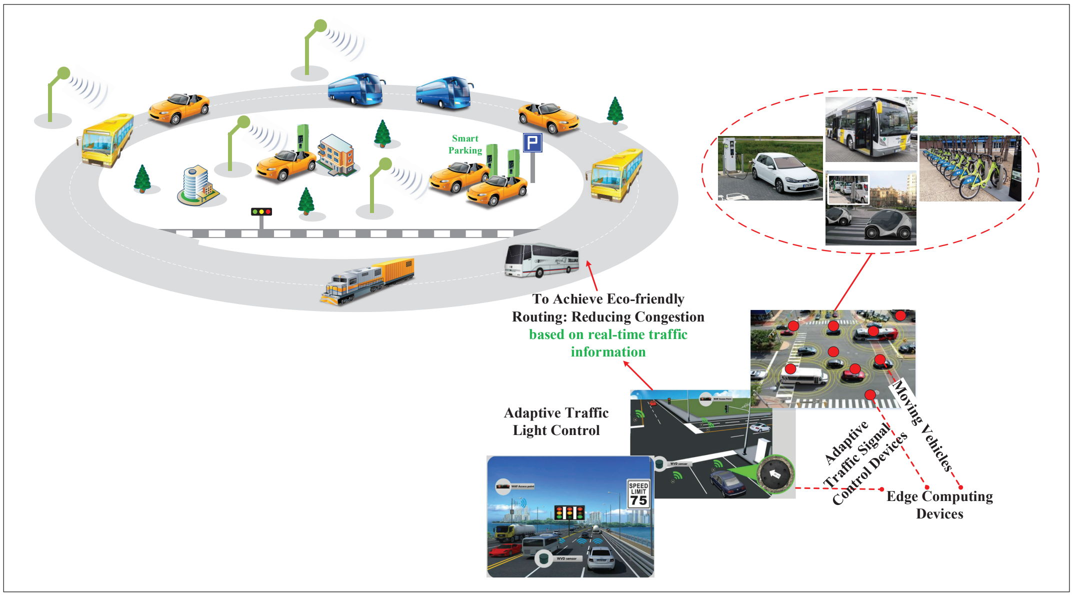
Smart Parking
P
To Achieve Eco-friendly Routing: Reducing Congestion based on real-time traffic information
Adaptive Traffic Light Control
SPEED LIMIT 75
Adaptive Traffic Signal Control Devices
Moving Vehicles
Edge Computing Devices

FIGURE 1. In cyber-physical transportation systems, the adaptive traffic signal control system with sensors and traffic controllers is used to control traffic lights according to traffic conditions. The traffic conditions are well understood with the communications among edge devices (e.g., sensors, controllers, and moving vehicles). The traffic condition information is sent to the controllers by the wireless mode. Then the controllers can conduct the traffic light control.

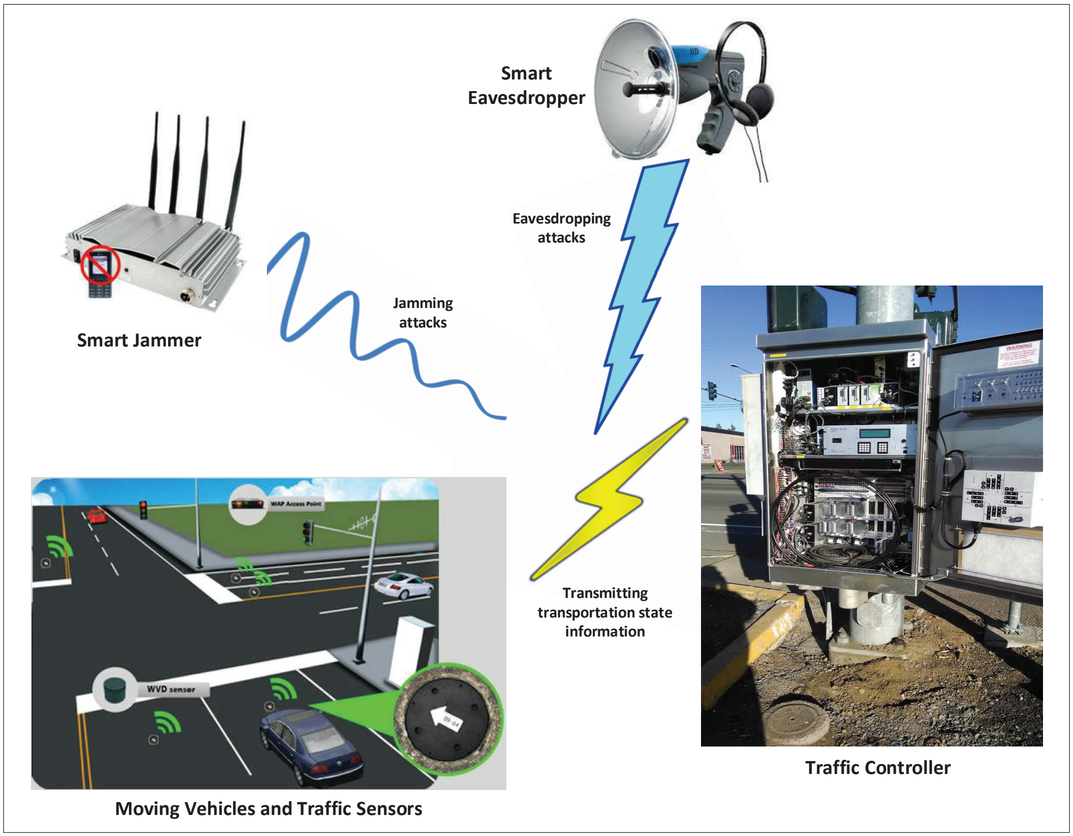

FIGURE 2. For the application scenario displayed in Fig. 1, the communications use the wireless mode between sensors and controllers. The open property of the wireless communications mode easily appeals to the threats from eavesdropping and jamming attacks. Moreover, eavesdroppers and jammers can self-adjust power strategies to maximally affect the wireless communications.

behavior of attacks can be learned, and further can be used to defend against the attacks.

Next, we review the important works on the detection methods based on static learning. In [5], SigPID is proposed. It statically learns the permission usage of the applications on MEC devices. Without analyzing all permissions, SigPID identifies the most significant permissions by three levels of pruning. Further, the classification methods, which are based on machine learning, are used to classify

Feature learning classification	Machine learning methods used in attack detection
Static-learning-based attack detection	Bayesian networks
	Random forest
	Decision tree
	Support vector machine
	K-means
	K-nearest neighbours
Dynamic-learning-based attack detection	Feedforward neural networks
	K-means
	Recurrent neural networks
	Support vector machine

TABLE 1. The machine learning methods for attack detection.

malicious and benign behaviors. In [6], DERBIN is proposed. It is a static analysis framework to extract attack features and disassemble attack code. Moreover, it uses a support vector machine to analyze the extracted features. In [7], DroidMat is proposed to extract different features from the AndroidManifest.xml document. The k-means algorithm is used to model malicious attacks where the extracted features are clustered. In [8], algorithms based on the Bayesian network, decision tree, k-nearest neighbor, random forest, and support vector machine are deployed to get automatic categorizing for detecting malicious attacks. With that, in [9], a support vector machine classifier is trained to construct a system, and this system extracts static features of attacks from android systems.

Dynamic-Learning-Based Attack Detection

This detection examines attacks in the running status of these attacks, and it learns the dynamic behavior features of the attacks, for instance, learning the power strategy of eavesdroppers and jammers.

Next, we review some important works on detection methods that are based on dynamic learning. In [10], an automatic attack detection method is proposed by dynamically analyzing the text semantics of network traffic. Moreover, this work uses the text semantic feature analysis to develop a dynamic attack detection model for the network traffic. In [11], a system based on artificial neural networks is proposed for detecting unknown attacks in the Android system. Moreover, feedforward neural networks and recurrent neural networks are used in this work. Thus, recurrent neural networks train the model with the dynamic feature (e.g., system calls).

In the last 10 years, feature learning has been increasingly used in attack detection schemes. In Table 1, we list some machine learning methods that have been used in existing works for malicious attack detection.

A Model Based on Deep Learning

We design a model based on deep learning to detect attacks by learning attack features. The model is evaluated with attacks aimed at the devices installed the Android system.

There are two components in this model: the feature preprocessing engine and the attack detection engine. The details are introduced as follows.

Feature Preprocessing Engine: This engine uses static/dynamic learning to preprocess attack features. There are three types of features in our study: required permissions, sensitive application programming interfaces (APIs), and dynamic behavior. Static learning is used to process two kinds of features: required permissions and sensitive APIs. Dynamic learning processes the dynamic behavior.

In this engine, the Android package (APK) files of edge devices are unpacked. Then feature elements are extracted from the files, and they are input into the next attack detection engine.

Attack Detection Engine: There are two modules in this engine:

- The feature learning module, which is based on a deep belief network
- The prediction output module, which is based on a softmax function

Figure 3 (right half) illustrates the deep belief network structure, and this network structure is used to achieve the active feature learning of security attacks.

This deep belief network combines a sequence of unsupervised networks (e.g., restricted Boltzmann machines,RBMs). An RBM is a kind of generative stochastic artificial neural network. In an RBM, there are a layer of visible units, a layer of hidden units, and corresponding weights between the visible units layer and the hidden units layer. Multiple RBMs can be stacked together, and the hidden layer output is the input of the next RBM. In our case, the reasonable number of hidden units, *m*, is 512. This value is obtained by adjusting with training.

Moreover, a set of features from the MEC environment is used as the unlabeled[2] samples. These features are used to analyze attack behaviors. The following features are used: vibration, access coarse updates, access GPS, access norton security, access wimax state, invoke internal handler, install theme, receive broadcasts, read frame buffer, read input state, read profile, write gmail, and wave lock and unlock.

Hidden units and top-level units are set above the RBM layer to construct a back propagation network. They are applied to get the fine tuning in deep learning, and to assist the fine tuning, labeled samples are input into the top-level units.

The left side of Fig. 3 illustrates the training structure for the deep belief network, and the training process includes two phases.

The Pretraining Phase with Unlabeled Samples: The pretraining process is unsupervised, and is shown as follows:

- The initialization of visible and hidden units
- The update of hidden units with the visible units' output
- The reconstruction of visible units with the hidden units' output
- The re-update of hidden units with the reconstructed visible units' output
- The update of the weight values between visible units and hidden units

The Fine Tuning Phase with Labeled Samples: In this phase, back propagation is designed for

[2] A feature is labeled by a label to show whether it is from a malicious attack or not.

finely tuning the pretrained parameters (e.g., the weight of each neuron) that have been trained in the pretraining phase.

Back propagation includes two steps:

- Error calculation. The error describes how far the output of a designed model is from the actual output.
- Error minimization. Back propagation checks whether the error is minimized or not. If the error is not minimized, back propagation will update the parameters (weights). This process is repeated until the error is minimized.

In the aspect of general analysis, with training the proposed model by the sample data from a system, the model will be more suitable to the system. In a real system, there are many interdependent parameters. It is impossible or difficult to accurately model the system only using simple formulations, much less to capture the essence of parameters' interdependency. However, there is enough information in the data from the system, and it is possible to reflect the interdependency of interdependent parameters by analyzing these data. Therefore, it is possible to model a complicated system by using the data to train an artificial neural network.

There are two strengths of the proposed model:

- This model can capture the nonlinear relations between attacks and corresponding features. By such relation capture, the proposed model enhances the performance of attack detection compared to the other four algorithms, which are based on machine learning (softmax regression [12], decision tree [13], support vector machine [14], and random forest [15]). Moreover, our model achieves active learning with unlabeled data, whereas for the other four algorithms, designing a good feature adapter to achieve such active learning requires a considerable amount of improvement.
- There are multi-stacked modules in the proposed model, which are used to calculate the nonlinear mapping between input and output. That is, the intricate function for the input of the proposed model can be implemented by using multiple nonlinear layers.

Experimental Evaluation and Analysis

Experimental Setup

Ten different datasets are used to conduct the evaluation of the proposed model. In the used datasets, there are 500 malicious attack samples that are from the MEC environment, and there are 500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, and 5000 non-attack samples in each dataset, respectively. In each dataset, the ratio of data samples used in training and testing is 0.8:0.2. The proposed model is compared to four other detection algorithms that are based on machine learning methods: softmax regression, decision tree, support vector machine, and random forest. Each algorithm is run 10 times to calculate the average of performance values.

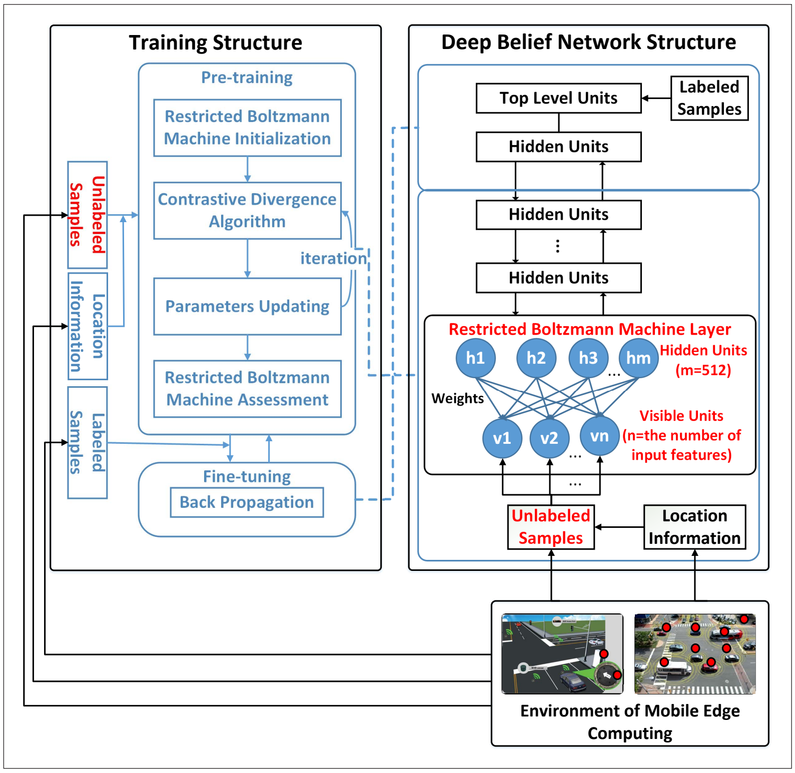

FIGURE 3. The deep belief network structure, which is used to achieve active learning (right), and the training structure with attack samples from the MEC environment (left).

For the proposed model, 512 hidden units are deployed in the RBM. Visible units are the same as input features in terms of the number. In the pre-training phase, we use 200 epochs to train the model, and in the fine-tuning phase, the number of iterations is set to 200.

For the input and output of our model, the softmax function's input is the deep belief network's output. The softmax function's output represents a categorical distribution. For this study, the output of the softmax function is a 0-1 categorical distribution: if the output is 1, it means that the test sample is malicious, and if the output is 0, it means that the test sample is benign.

Experimental Results

We provide the comparison of accuracy for five algorithms on 10 datasets, and the results are shown in Fig. 4. Moreover, we illustrate the pre-training errors and the fine-tuning losses at different training epochs in Fig. 5.

In comparison results, the measurement accuracy measures how well an algorithm detects attacks. It can be visually described as

$$\text{accuracy} = \frac{\text{number of correctly classified samples}}{\text{total number of samples}},$$

which is the proportion of the correctly classified samples.

From the results shown in Fig. 4, in terms of accuracy, the proposed model is 12.61 percent higher compared to the softmax-regression-based detection algorithm, 5.76 percent higher compared to the decision-tree-based algorithm, 3.20 percent higher compared with the support-vector-machine-based algorithm, and 2.61 percent higher compared to the random-forest-based algorithm.

We provide the insights of the performance comparison on the proposed model and the other four algorithms, and give the time complexity of each algorithm.

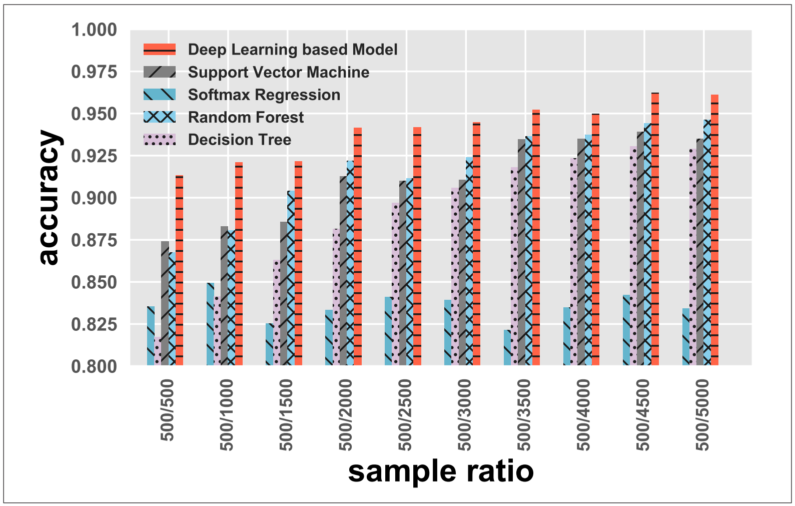

FIGURE 4. The comparison of accuracy for five algorithms. We run each algorithm 10 times to calculate the average of accuracy values on each dataset. The sample ratio of malicious and benign in a dataset is represented as, for example, 500/1000, which means 500 malicious samples and 1000 benign samples.

Softmax-Regression-Based Detection Algorithm: (Time Complexity: $O(n + 1)m$, where the size of training data is indicated as m, and the number of features is indicated as n. In this algorithm, to minimize a cost function, we obtain the optimum values of the softmax regression parameters. Softmax regression is used for linearly separable datasets. However, the datasets used in this article are not linearly separable.

Decision-Tree-Based Detection Algorithm: (Time Complexity: $O(mn)$). By breaking down a dataset into smaller subsets and increasingly recombining the subsets, an associated decision tree can be incrementally constructed. However, such an incremental scheme cannot always help enhance the performance of the decision tree by only increasing the number of subsets.

Support-Vector-Machine-Based Detection Algorithm: (Time Complexity: $O(m^3)$). The kernel function of the support vector machine decides the performance of the algorithm, not data-based training.

Random-Forest-Based Detection Algorithm: (Time Complexity: $O(nmlog(m))$). A multitude of decision trees construct a random forest, and the forest's output is the classification or the prediction for each tree. This forest construction is similar to the construction of a decision tree. Therefore, by only increasing the size of data, it cannot always help enhance the random forest performance.

Deep-Learning-Based Detection Model: (Time Complexity: $O(m^2)$). The proposed model is based on the deep learning framework, and benefits from data-based model training. To be specific, attack data include enough information and features to reflect the complicated behavior of attacks. By training the data, mining information and features from the data are helpful to improve the detection accuracy for the attacks. Moreover, this model has an advantage in flexibility:

- Using new input features is easy in this model to satisfy the requirements of complex attack scenarios; for example, the location features of attacks are added as input parameters.
- New modules are easily added to achieve new functions; for example, the unsupervised learning module is added to develop the active feature learning function.

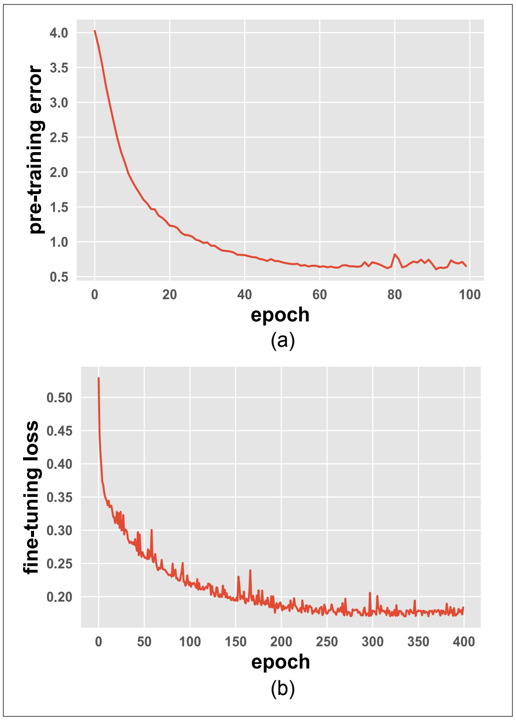

FIGURE 5. Pre-training errors and fine-tuning losses at different epochs: a) pre-training errors. At the point epoch = 50, the error is minimum; b) fine-tuning losses. After the point epoch = 350, the fluctuation of the loss value trends to be steady.

Moreover, the proposed model is sensitive to the dataset size in performance. With the increase of the training dataset size, the performance of the model is increased, but it does not simply depend on the size of the dataset. There is a relationship between the "performance" and the "difference of malicious samples and benign samples in a dataset." If the difference is larger, the trained model with the dataset is better in the performance of detecting attacks.

The model errors after pre-training are shown in Fig. 5a. With the increase of the training epoch, the error is decreased gradually, and at a certain epoch point, the error tends to be stable, which indicates that the minimum training error is reached at a certain epoch. More concretely, with the epoch increase, the model is fitted with the training data, and more features of the data are learned by the model. The model's detection accuracy is enhanced with this learning. However, the model will be overfitted into the training data once the training is longer than a certain duration. Because of the overfitting, the output of an analysis from the model is too related to the training data, and therefore the model fails to work in the data that are not used in model training.

In the process of fine-tuning, the loss decreases, with some oscillations, as the epoch increases, and the result is shown in Fig. 5b.

In MEC, the most critical challenge is to handle streaming data, and further use the data to conduct model training. It is time-consuming to train the proposed model by streaming data. Another challenge is to achieve online training for the model, and then to use the trained model in real-time applications.

Conclusions

This article proposes a model based on the deep learning framework to detect attacks in MEC. The model's active feature learning has an advantage in the accuracy improvement compared to four other machine-learning-based detection algorithms. Moreover, the deep belief network with 512 hidden units is used to ceaselessly learn attack features, and the number 512 is set by data-training-based adjustment. Furthermore, the contrastive divergence algorithm is used for iteratively updating the parameter values of the model. In addition, by observing the results, in terms of improving accuracy, the training dataset size has an important impact. On average, our model achieves a good gain in detection accuracy. It is 12.61 percent higher compared to the softmax-regression-based algorithm, 5.76 percent higher compared to the decision-tree-based algorithm, 3.20 percent higher compared to the support-vector machine-based algorithm, and 2.61 percent higher compared to the random-forest-based algorithm. However, the proposed model still has limitations in streaming data handling, and online training to serve applications in real time.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 61802097), and the Project of Qianjiang Talent (Grant No. QJD1802020).

References

[1] E. Ahmed and M. H. Rehmani, "Mobile Edge Computing: Opportunities, Solutions, and Challenges," *Future Generation Computer Systems*, vol. 70, 2017, pp. 59–63.

[2] M. Ali *et al.*, "Joint User Association and Power Allocation for Licensed and Unlicensed Spectrum in 5G Networks," *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, IEEE, 2017, pp. 1–6.

[3] N. Yang *et al.*, "Safeguarding 5G Wireless Communication Networks Using Physical Layer Security," *IEEE Commun. Mag.*, vol. 53, no. 4, 2015, pp. 20–27.

[4] Y. He *et al.*, "Cross-Layer Resource Allocation for Multihop V2x Communications," *Wireless Communications and Mobile Computing*, vol. 2019, 2019, pp. 1–16.

[5] J. Li *et al.*, "Significant Permission Identification for Machine Learning Based Android Malware Detection," *IEEE Trans. Industrial Informatics*, 2018.

[6] D. Arp *et al.*, "Drebin: Effective and Explainable Detection of Android Malware in Your Pocket," *Proc. NDSS*, 2014.

[7] D.-J. Wu *et al.*, "Droidmat: Android Malware Detection Through Manifest and API Calls Tracing," *Proc. 7th IEEE Asia Joint Conf. Info. Security (Asia JCIS)*, 2012, pp. 62–69.

[8] B. Sanz *et al.*, "On the Automatic Categorisation of Android Applications," *Proc. 9th IEEE Conf. Consumer Commun. and Networking (CCNC)*, 2012, pp. 149–53.

[9] J. Sahs and L. Khan, "A Machine Learning Approach to Android Malware Detection," *Proc. 2nd IEEE Conf. Euro. Intelligence and Security Informatics (EISIC)* 2012, pp. 141–47.

[10] S. Wang *et al.*, "Detecting Android Malware Leveraging Text Semantics of Network Flows," *IEEE Trans. Info, Forensics and Security*, vol. 13, no. 5, 2018, pp. 1096–1109.

> In MEC, the most critical challenge is to handle streaming data, and further use the data to conduct model training. It is time-consuming to train the proposed model by streaming data. Another challenge is to achieve online training for the model, and then to use the trained model in real-time applications.

[11] A. Shabtai *et al.*, "Andromaly: A Behavioral Malware Detection Framework for Android Devices," *J. Intelligent Info. Systems*, vol. 38, no. 1, 2012, pp. 161–90.

[12] M. Jiang *et al.*, "Text Classification Based on Deep Belief Network and Softmax Regression," *Neural Computing and Applications*, vol. 29, no. 1, 2018, pp. 61–70.

[13] S. S. S. Sindhu, S. Geetha, and A. Kannan, "Decision Tree Based Light Weight Intrusion Detection Using a Wrapper Approach," *Expert Systems with Applications*, vol. 39, no. 1, 2012, pp. 129–41.

[14] E. A. Shams and A. Rizaner, "A Novel Support Vector Machine Based Intrusion Detection System for Mobile Ad Hoc Networks," *Wireless Networks*, vol. 24, no. 5, 2018, pp. 1821–29.

[15] P. A. A. Resende and A. C. Drummond, "A Survey of Random Forest Based Methods for Intrusion Detection Systems," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, 2018, p. 48.

Biographies

Yuanfang Chen [S'09-M'15] (chenyuanfang@hdu.edu.cn) received her Ph.D. and M.S. degrees from Dalian University of Technology, China, and a second Ph.D. degree from the Université Pierre et Marie Curie, France. She currently works at Hangzhou Dianzi University as a professor. She was an assistant researcher at the Illinois Institute of Technology with Prof. Xiangyang Li. She has been invited as the Session Chair of several conferences, is an Associate Editor of *Industrial Networks and Intelligent Systems*, and was a Guest Editor of *MONET*.

Yan Zhang [SM'10] (yanzhang@ieee.org) is a full professor at the University of Oslo. He is an Editor of several IEEE publications, including *IEEE Communications Magazine*, *IEEE Network*, *IEEE Transactions on Green Communications and Networking*, *IEEE Communications Surveys & Tutorials*, and the *IEEE Internet of Things Journal*. His current research interests include next generation wireless networks leading to 5G and cyber physical systems. He is an IEEE VTS Distinguished Lecturer and a Fellow of IET. He received the "Highly Cited Researcher" award (Web of Science top 1 percent most cited worldwide) according to Clarivate Analytics.

Sabita Maharjan [M'09] (sabita@simula.no) received her Ph.D. degree in networks and distributed systems from the University of Oslo and Simula Research Laboratory, Norway, in 2013. She is currently a senior research scientist at the Simula Metropolitan Center for Digital Engineering, Norway, and an associate professor at the University of Oslo. Her current research interests include wireless networks, network security and resilience, smart grid communications, the Internet of Things, machine-to-machine communications, software defined wireless networking, and the Internet of Vehicles.

Muhammad Alam [S'10, M'14, SM'17] (alam@ua.pt) holds a Ph.D. degree in computer science from the University of Aveiro, Portugal (2014), and an M.S. degree in computer science from the International Islamic University Islamabad, Pakistan (2008). He has participated in several EU funded projects such as C2POWER, ICSI, and PEACE. Currently, he is working as an assistant professor at Xi'an Jiaotong-Liverpool University, Suzhou, China. His research interests include IoT, real-time wireless, and vehicular communications.

Ting Wu is currently a professor at Hangzhou Dianzi University. He is the Dean of the School of Cyberspace, Hangzhou Dianzi University, a committee member of the Secrecy and Management Department at the Ministry of Education, and the director of the Chinese Confidential Association. He has been invited as the principal investigator in the 863 Project and a member of the 973 Program. His research interests include computers and network security.