Compressed Neural Networks for IoT Network Forensics

Neelay Shah
Deparment of ELectrical and Electronics Engineering
Birla Institute of Technology and Science, Pilani
Goa, India
f20180400@goa.bits-pilani.ac.in

Nitin Sharma
Department of Electrical and Electronics Engineering
Birla Institute of Technology and Science, Pilani
Goa, India
nitinn@goa.bits-pilani.ac.in

***Abstract*—In recent years, widespread manufacturing of small scale hardware and increases in working power efficiency have allowed the deployment of intelligent software systems on otherwise ordinary and household hardware devices. Such increased computational capabilities of everyday devices has led to the emergence of the Internet-of-Things (IoT) domain. The IoT is basically a network of interconnected objects possessing some measure of computational capabilities. Being a network, the IoT is vulnerable to malicious attempts and attacks aiming to disrupt the normal working of the system. Recently, a variety of deep learning techniques have been proposed to identify and combat the aforementioned botnet activities. However, as effective as deep learning solutions are, the highly complex neural networks often have large sizes and compute requirements; making them unsuitable for deployment on resource constrained edge devices in IoT networks. In this paper, we introduce compressed neural networks for network intrusion detection. Such compressed neural networks exhibit performance performance similar to larger ones while being much smaller in size and having lower compute requirements.**

***Index Terms*—Internet-of-Things, network intrusion detection, deep learning, model compression.**

I. Introduction

The Internet of things describes the network of physical objects—"things"—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the Internet. With the prevalence of (IoT) systems, inconspicuous everyday household devices are connected to the internet, providing automation and real-time services to their users. In spite of their light-weight design and low power, their vulnerabilities often give rise to cyber risks that harm their operations over network systems. One of the key challenges of securing IoT networks is tracing sources of cyber-attack events, along with obfuscating and encrypting network traffic.

For years, commercial tools have mainly depended on signature-based approaches for detecting the presence of botnets. However, these rule based systems are incapable of detecting constantly evolving and new forms of botnets. People have recently come to appreciate the role of machine learning algorithms for classifying cyber attacks. Lately, deep learning solutions have been proposed for the aforementioned task. However, while designing such deep learning solutions, no heed has been paid to the size of the heavily parameterized models and their feasibility to run on low compute devices used in IoT systems. This project aims to develop compressed deep neural networks to discover and trace abnormal events from IoT network of smart homes.

II. Internet of Things

The Internet of things (IoT) describes the network of physical objects—"things"—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the Internet. A thing in the internet of things can be a person with a heart monitor implant, a farm animal with a biochip transponder, an automobile that has built-in sensors to alert the driver when tire pressure is low or any other natural or man-made object that can be assigned an Internet Protocol (IP) address and is able to transfer data over a network.

A. IoT Security

Security is one of the biggest concerns in adopting Internet of things technology, with concerns that rapid development is happening without appropriate consideration of the profound security challenges involved and the regulatory changes that might be necessary

Most of the technical security concerns are similar to those of conventional servers, workstations and smartphones. These concerns include using weak authentication, forgetting to change default credentials, unencrypted messages sent between devices, SQL injections, and poor handling of security updates. However, many IoT devices have severe operational limitations on the computational power available to them. These constraints often make them unable to directly use basic security measures such as implementing firewalls or using strong cryptosystems to encrypt their communications with other devices - and the low price and consumer focus of many devices makes a robust security patching system uncommon.

III. Machine Learning

Machine learning (ML) is the study of computer algorithms that improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as "training data", in order to make predictions

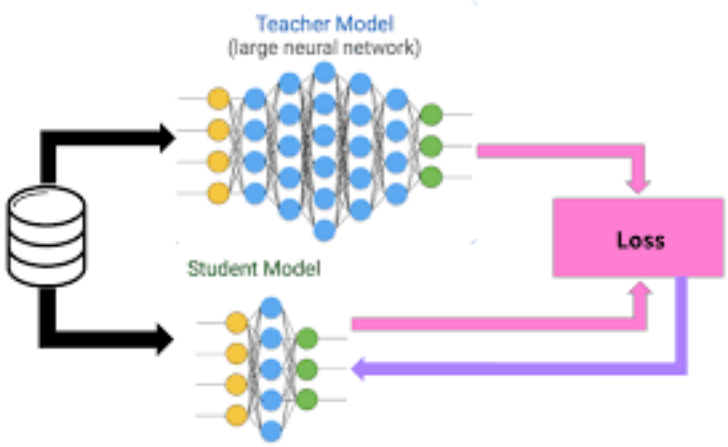

Fig. 1. Knowledge distillation process. A smaller 'student' network is made to mimic a larger 'teacher' in order to achieve similar level of performance

or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

A. Deep Learning

Deep learning (DL) is a subset of machine learning that uses heavily parameterized hierarchical layers of networks to fit models to data and learn good function approximations. It comprises of processing data through a series of non-linear transforms in order to learn function approximators capable of modelling complex input-output relationships.

IV. Related Work

Research on network security has existed ever since the advent of modern computer architectures. Intrusion detection, as applied to IoT networks has been a prominent research topic for a considerable amount of time. Recently, machine learning techniques have found widespread use for the purpose of developing network intrusion systems. [1] used autoencoders, a category of unsupervised deep learning models for network intrusion detection on the NSL-KDD dataset [8]. [2] made use of tradidtional machine learning algorithms such as decision trees and naive bayes for detecting botnet activities on the UNSW-NB15 [7] dataset. The performance of shallow and deep neural networks, as well as a variety of machine learning models such as support vector machines (SVMs) was evaluated by [4]. [6] used hyperparameter optimization techniques such as particle swarm optimization to enhance the performance of deep neural networks for network intrusion detection.

As successful as the previous works have been for IoT network forensics tasks, they have not focused on the feasibility of deploying deep neural networks (DNNs) on edge devices used in IoT systems. DNNs required significantly large space to store their function parameters, also called weights, and thus require heavy compute access. This might make them unsuitable for resource-constrained edge devices which could be present in IoT systems. We propose to use model compression techniques to develop highly resource efficient neural networks for network intrusion detection which are able to match the performance of deep models.

V. Compressed Neural Networks for Intrusion Detection

A. Model Compression in Deep Learning

Model compression is an area of research in deep learning which aims to make deep neural networks fit for deployment in low-power and resource limited devices.The goal of model compression methods is to reduce the size of networks without causing significant drop in their accuracies. Currently, there are 3 major families of model compression algorithms - Knowledge Distillation, Pruning and Quantization.

1) Knowledge Distillation: Knowledge Distillation (KD) [11] is a compression paradigm that leverages the capability of large neural networks (called teacher networks) to transfer knowledge to smaller networks (called student networks). While large models (such as very deep neural networks or ensembles of many models) have higher knowledge capacity than small models, this capacity might not be fully utilized. Knowledge distillation aims to transfers knowledge from a large model to a smaller model without loss of validity.

2) Pruning: While knowledge distillation attempts to train an equally competent smaller network, network pruning attempts to reduce the size of the existing network by removing unimportant weights. Different pruning techniques differ in the choice of weights to eliminate and the methods used to do the same. Pruning can help in reducing the size of the network up to 90% with minimal loss in performance. Some approaches have also been empirically shown to result in faster training of the pruned network along with a higher test accuracy.

3) Quantization: Quantization is another way to compress neural networks by reducing the number of bits used to store the weights. As the weights of a network are usually

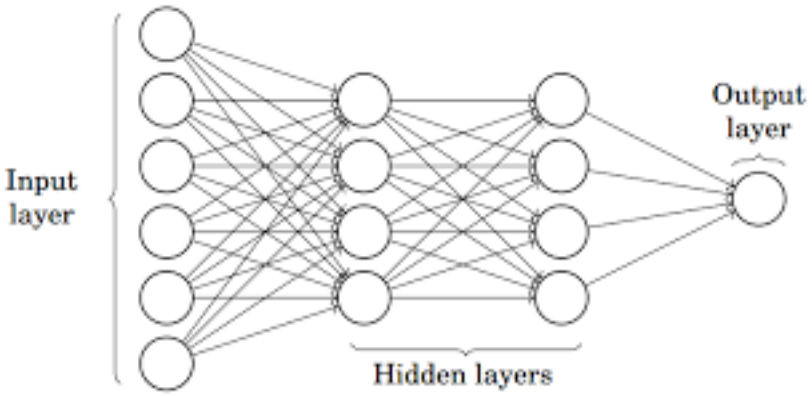

Fig. 2. A multi-layer perceptron (MLP) neural network. It has one input layer, one output layer and a number of hidden layers. Each layer applies non-linear transformations to the data

stored as 32-bit floating values (FP32), reducing the precision to 8-bit integer values (INT8) will reduce the size of the network by 4 times. Several approaches have been developed to quantize networks with minimal loss in performance.

We propose to use various kinds of knowledge distillation techniques to develop compact deep neural networks for network intrusion detection. More concretely, we train deep neural networks for intrusion classification and then 'distill' the knowledge stored in these networks into much smaller neural networks with fewer number of parameters (and hence size). In our experiments, we demonstrate that such a distillation process can help smaller networks achieve performance similar to larger networks, and much better performance than that if the smaller networks are trained from scratch.

VI. Experiments

We carry out our experiments on the UNSW-NB15 dataset [7]. We use the open-source deep learning framework PyTorch and Nvidia RTX 2080 GPU compute available on the online Google Colab service.

A. UNSW-NB15 Dataset

UNSW-NB15 is a network intrusion dataset. it contains nine types of attacks, namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. In our experiments, we try to predict whether a data sample is normal or an attack vector.

B. Data Pre-processing

The data was first split into a train set for training the neural networks and a test set for evaluating their performance. A 80-20 split was used. All categorical features in the dataset were transformed into numerical ones using one-hot encoding. Duplicate records were removed from both the train and test sets. The train set data was then normalized by removing the mean of the data and scaling to have unit variance. The test set was also normalized using the calculated statistics of the training data.

C. Deep neural networks

We use multi-layer perceptron (MLP) neural networks for both the teacher and student models, with student models having fewer parameters (and hence size) than the teacher model. We train all models using stochastic gradient descent with a learning rate of 0.1. We train all models for 50 epochs and then evaluate them on the test set.

D. Training distilled neural networks

While pruning and quantization are direct method of reducing a model's size, knowledge distillation builds on the fact that it is possible for a smaller neural network to match a larger one's representational power through proper optimization. More often than not direct methods of model compression lead to significant drop in model performance. We demonstrate that knowledge distillation can be used to reduce a model's size without compromising on its performance.

We first train teacher multi-layer perceptron neural networks having 3 hidden layers containing 32, 16 and 8 neurons in order. After a teacher network is trained, its parameters are frozen. Then we use knowledge distillation to train smaller student networks from the teacher network. The smaller student networks either have fewer hidden layers or have lesser number of neurons in their hidden layers, or both. consider 3 such student models for our experiments. For each of the student models, we also train them from scratch to observe difference in performance from the teacher-distilled student models. We report accuracies achieved by and sizes of all models in our results.

We also carry out experiments using different knowledge distillation algorithms to develop student networks and report their results.

VII. Results

Table I details the performance of a variety of student models trained by a teacher model containing more number of

TABLE I
PERFORMANCE OF VARIOUS DISTILLED STUDENT MODELS

Model type	Number of hidden layers	Hidden layer configuration (number of neurons	Number of parameters	Model size (in bytes)	Base accuracy (in %)	Distilled accuracy (in %)
Teacher	3	[32, 16, 8]	2090	8360	97.66	NA
Student	3	[16, 8, 4]	886	3544	97.19	97.58
Student	1	[16]	738	2952	96.33	97.40
Student	1	[8]	370	1480	95.16	96.51

TABLE II
PERFORMANCE OF VARIOUS KNOWLEDGE DISTILLATION ALGORITHMS

Algorithm/Method	Teacher accuracy	Student base accuracy	Student distilled accuracy
Self Training [10]	NA	97.19	97.61
Vanilla KD [11]	97.66	97.19	97.58
Probability Shift [9]	97.63	97.19	97.40

In the experiments, the teacher model is an MLP with 3 hidden layers containing 2090 parameters. The student model is also an MLP with 3 hidden layers but with a different configuration containing fewer neurons. The number of parameters in the student model is 886.
Note that the Self Training [10] doesn't make use of a separate teacher network. Rather, it trains a student model and then uses it to distill knowledge into another generation of the same student model.

parameters. We train 3 different types of student models, each progressively smaller in size (fewer number of parameters) than the previous and compare their accuracies. Along with the accuracies obrained by the student networks after distillation, we also report the accuracies obtained by the, when trained from scratch (without any knowledge distillation) and the accuracies obtained by the teacher.

As is evident, in all cases the accuracies after distillation for the student models are higher than their base accuracies. We observe that a neural network can be compressed to a fraction of its size without a significant drop in performance. Furthermore, almost all the knowledge stored in an MLP with 3 hidden layers can be transferred to an MLP with only 1 hidden layer containing almost 1/4th the number of neurons as the larger MLP. These results highlight the utility of knowledge distillation techniques for developing neural networks for network forensice in resource constrained IoT devices.

In Table II, we present the results of different knowledge distillation algorithms for model compression. We observe that Self Training [10] - a method where generation of a student are trained which transfer knowledge among themselves perform the best for our network intrusion detection task. Again, we observe that the performance of distilled student model is very near to that of much larger teacher models. This again underscores how knowledge distillation methods can be used to develop compact and resource efficient neural networks for IoT network intrusion detection.

VIII. CONCLUSION

We demonstrate the usefulness of knowledge distillation as a technique to develop compact models fit for deployment on low resource devices for network intrusion detection in IoT networks. From the empirical results presented, we believe that such compressed deep neural networks hold the potential to be a part network intrusion detection systems that run completely on the edge.

ACKNOWLEDGMENT

We thank the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) for providing open access to the UNSW-NB15 dataset. We also extend our gratitude to the Google Colab team for providing free compute resources.

REFERENCES

[1] H. Choi, M. Kim, G. Lee, and W. Kim1, " Unsupervised learning approach for network intrusion detection system using autoencoders "
[2] N. Koroniotis, N. Moustafa, E. Sitnikova, and J. Slay, "Towards Developing Network forensic mechanism for Botnet Activities in the IoT based on Machine Learning Techniques"
[3] N. Shone , T.N. Ngoc, V.D. Phai , and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection"
[4] Rahul Vigneswaran K, Vinayakumar R, Soman KP, and Prabaharan P., "Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security"
[5] C. Ieracitano, A. Adeel, F.C. Morabito, and A. Hussain, "A Novel Statistical Analysis and Autoencoder Driven Intelligent Intrusion Detection Approach"
[6] N. Koroniotisa, N. Moustafaa, and E. Sitnikova, "A New Network Forensic Framework based on Deep Learning for Internet of Things Networks: A Particle Deep Framework"
[7] N. Moustafa, J. Slay, Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set), in: 2015 Military Communications and Information Systems Conference (MilCIS), 2015, pp. 1–6. doi:10.1109/MilCIS.2015.7348942.
[8] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), 2009.
[9] Tiancheng Wen, Shenqi Lai and Xueming Qian, Preparing Lessons: Improve Knowledge Distillation with Better Supervision
[10] Li Yuan, Francis E.H.Tay, Guilin Li, Tao Wang, and Jiashi Feng, "Revisiting Knowledge Distillation via Label Smoothing Regularization"
[11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the Knowledge in a Neural Network"