NetSpirit: A Smart Collaborative Learning Framework for DDoS Attack Detection

Ke Xu, Yong Zheng, Su Yao, Bo Wu, and Xiao Xu

Abstract

Facing one of the most common threats to Internet security, the existing traffic-driven distributed denial of service (DDoS) defense schemes mainly focus on establishing more accurate detection models that highly require labeled untrusted traffic flows in the attacked network. Unfortunately, they usually ignore the communication overhead when collecting data through inherently distributed networks, which also introduces nontrivial privacy leakage. In this article, we propose a collaborative learning framework called NetSpirit to achieve effective detection of DDoS attacks. Leveraging parameter interactions instead of traffic data between network elements, its detection model can be efficiently trained and synchronized, with lightweight overhead and packet privacy protection. Meanwhile, semi-supervised machine learning is employed to learn from unlabeled data, and model pruning is used to further reduce the traffic transmission cost. NetSpirit can be implemented by several major machine learning frameworks. In this article, we choose to implement the NetSpirit prototype using MindSpore in our simulated environment and use public datasets to evaluate its effects. The experimental results demonstrate that NetSpirit can reduce by 28.28 percent the average transmission amount compared to traditional collaborative learning and achieve a detection accuracy of 63.80 percent, with a top-3 accuracy of 87.57 percent and a top-5 accuracy of 90.34 percent for the 13-classification problem of DDoS attacks using only 50 percent labeled data. Moreover, by adjusting the hyperparameters, it can make a good trade-off between computing time and transmission amount. We hope the intra- and inter-domain collaboration in NetSpirit can act as a fundamental primitive to build the intelligence layer of a trustworthy network architecture.

Introduction

With the development of the Internet over the years, the threat of network security attacks has become increasingly serious. One trend in network attacks is the use of network traffic. An attacker can endanger a network or host without intruding into it. This kind of attack, represented by the distributed denial of service (DDoS) attack, will cause more damage due to the following reasons. There are many tools for launching DDoS attacks that can be used even by unskilled users, and it is difficult to trace these attacks back to the attacker. A successful DDoS attack will quickly impact the target, and the bandwidth consumption in the process will also affect network performance. According to a report in 2020 from Radware [1], DDoS attacks were the most common attacks on service providers and telecom companies at 64 percent, and the third most common to all respondents at 48 percent. Furthermore, 10 percent of DDoS attacks were above 10 Gb/s, and about 58 percent lasted more than one hour [1]. Although the DDoS attack is by no means new, it still poses a tremendous threat to many systems.

These threats urge us to be well prepared for DDoS attacks. Defense against DDoS attacks has been a significant topic for a long time and can be divided into two types: architecture-based and traffic-driven. The architecture-based defense schemes usually require many new devices in the network to work with when applied in practice, while the existing traffic-driven defense schemes mainly focus on the attacked network and centralized data. However, because of the distributed nature of the network, the traffic data of interest are also inherently distributed, and they are difficult to transmit, and should not be transmitted to the same place, due to the following reasons. The traffic on the Internet is huge, so the transmission would undoubtedly increase the burden on the network. The Internet has developed a hierarchical and domain-specific structure, with multiple stakeholders from all aspects, who have sufficient reasons not to share their data, so the transmission would reveal plenty of privacy and would certainly not be accepted by them. Therefore, multiple network elements must share data and participate in model training in a manner that does not move data and protects privacy, which is called collaborative learning.

In this article, as a traffic-driven defense scheme, a collaborative learning framework called NetSpirit is proposed for the effective detection of DDoS attacks, which is a general framework suitable for both intra-domain and inter-domain collaboration. In NetSpirit, network elements such as switches and routers rely on parameter interactions instead of data sharing or movement to participate in model training in a collaborative manner, which can mitigate the transmission overhead while achieving privacy protection of traffic data. Meanwhile, since labeled expert data for model establishment is limited, our proposed NetSpirit integrates semi-supervised machine learning to learn from unlabeled data. Moreover, NetSpirit also makes use of model pruning to further reduce the network transmission. As a result, NetSpirit achieves good efficiency and prediction accuracy. In our simulated environment, we implement a prototype using a typical artificial

Digital Object Identifier: 10.1109/MNET.001.2100234

Ke Xu is with Tsinghua University, Beijing National Research Center for Information Science and Technology, and Peng Cheng Laboratory; Yong Zheng and Su Yao are with Tsinghua University and BNRist;Bo Wu is with Tencent; Xiao Xu is with Peking University.

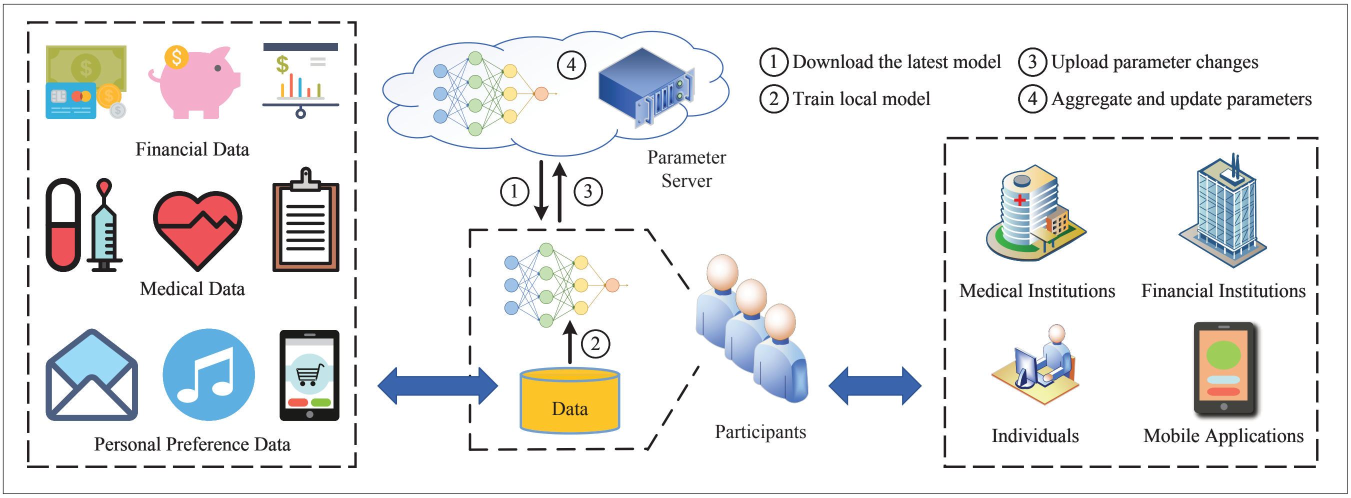

FIGURE 1. The typical workflow of collaborative learning systems.

neural network (ANN) model to verify the feasibility of NetSpirit. The experimental results show that by employing only 50 percent labeled data, it achieves a detection accuracy of 63.80 percent, with a top-3 accuracy of 87.57 percent and a top-5 accuracy of 90.34 percent for the 13-classification problem, reducing by 28.28 percent the transmission overhead compared to the traditional collaborative learning scheme.

Background

DDoS Attack and Defense

The DDoS attack is a type of attack designed to cause computer or network failure to provide normal services. DDoS attacks mostly occur on the network layer and application layer, such as UDP flood, TCP SYN flood, and HTTP flood. DDoS attacks usually use a botnet, which is a network of several compromised hosts (called zombies) on the Internet. The attacker can launch an attack by sending remote commands to each zombie, and it will send requests to the victim server or network, which may cause the server or network to be overwhelmed, resulting in denial of service to normal requests. Since each zombie is a legitimate Internet device, it may be difficult to distinguish between attack traffic and normal traffic.

The purpose of DDoS defense is to detect DDoS attacks and filter attack data packets to make it difficult to reach the attack target, thereby protecting the target. DDoS defense methods can be divided into two types: architecture-based and traffic-driven. As for architecture-based defense schemes, Gong *et al.* [2] identified a subtle but important security risk in the existing in-network filtering recommendations, and they proposed a verifiable in-network filtering (VIF) system for DDoS defense that offers filtering verification to DDoS victims and neighboring networks. As for traffic-driven defense schemes, Gulisano *et al.* [3] characterized regular network traffic of a service by aggregating it into common prefixes of IP addresses, determining attacks when the aggregated traffic deviates from regular traffic. They then proposed STONE, a framework with expert system functionality that provides effective and joint DDoS detection and mitigation.

However, since DDoS attacks come from outside networks instead of the target one, it is unrealistic to completely prevent DDoS attacks in the victim network. To successfully defend against DDoS attacks, the response method should not rely on traffic in the attacked network, but meet the requirements of accurate detection, effective response, and less impact on legitimate users.

Collaborative Learning

Data, algorithms, and computing power are three basic elements supporting the development of artificial intelligence (AI). In recent years, the growing problem of data islands and the increasing attention to data privacy protection have made the acquisition, exchange, and aggregation of data important factors limiting the development of AI. Researchers have been exploring how to break the data island barrier by allowing joint training under privacy-preserving constraints. Inspired by distributed machine learning, Shokri and Shmatikov [4] designed, implemented, and evaluated the PPDL system, a practical collaborative learning system that trades off between utility and privacy. As shown in the typical workflow in Fig. 1, this system allows participants to use their private data for local training, and it maintains a common global model in the parameter server. The training process does not require data interaction between participants, and at the same time, differential privacy is applied to protect shared parameters, ensuring that each participant does not leak their private data, and the model achieves accuracy close to that of centralized training. Google [5] implemented a scalable instance of a collaborative learning system (i.e., federated learning), which can be carried on tens of millions of mobile phones to jointly train an ANN model.

As for network security, especially intrusion detection and attack detection, collaborative learning is also considered to be useful and effective. Zhang and Zhu [6] proposed a collaborative learning-based intrusion detection system in the vehicular ad hoc network. By local training and collaborative communication with neighboring vehicles and roadside units, the vehicle updates the local detection engine to detect intrusion. Khoa *et al.* [7] proposed a collaborative learning-based attack detection system in the Internet of Things (IoT) and Industry 4.0. Each smart “filter” deployed at the IoT gateway uses

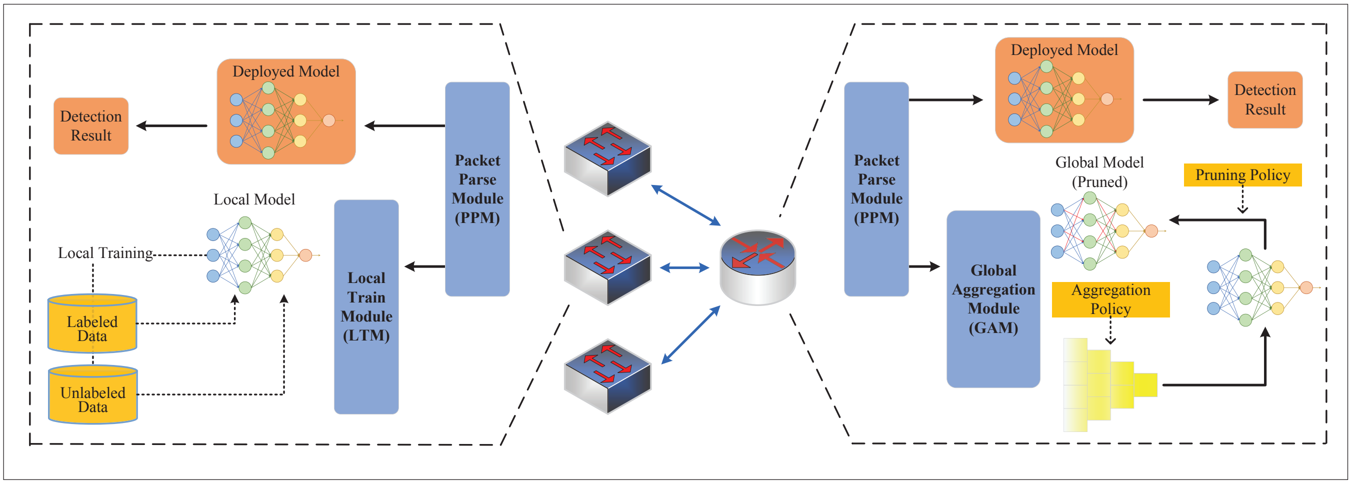

FIGURE 2. The steps between the participants and the server.

the collected data in its network to train its attack detection model, which will be shared with other IoT gateways to improve accuracy.

However, the existing collaborative learning systems cannot be simply applied to the detection of DDoS attacks on the Internet. The diversity of DDoS attacks, the difficulty of judging traffic, and the expectation of low latency all require a collaborative learning framework more suitable for this scenario.

NetSpirit Design

Framework Overview

To enable the use of our proposed framework on the Internet for effective detection of DDoS attacks, we analyzed the characteristics of the network and designed a collaborative learning framework called NetSpirit, which consists of three key components: the local train module (LTM), global aggregation module (GAM), and packet parse module (PPM). As shown in Fig. 2, the network elements as the participants and the server perform the following steps during each round:

1. LTM requests the latest global model.
2. PPM filters out the model information.
3. LTM trains the ANN model with domain local data (both labeled and unlabeled data) to get local gradients.
4. LTM uploads local gradients to the server, which is also served by a network element.
5. PPM (on the server) filters out the gradient information.
6. GAM aggregates received gradients following aggregation policy and applies them to the global model.
7. GAM prunes the global model following pruning policy.

Although currently shown in Fig. 2 as a single parameter server and a centralized training method, it can also be expanded to tree-shaped or ring-shaped parameter servers and a decentralized training method without changing the overall composition of the framework.

The training method implicit in the framework could be described as a semi-supervised and halfway model pruning collaborative learning method, denoted by SSHFMP. In this method, model pruning is triggered only in the first λ part of the rounds to ensure the convergence of the model.

Local Train Module

LTM runs on the participants and is responsible for pulling the latest parameters of the global model, updating the local model with the latest parameters, training the model with local data to get local gradients, and uploading local gradients to the server.

The traffic flows of DDoS attacks are the most similar to legitimate traffic flows among the various cyberattacks, but the intention is to disrupt the normal operation of services, which makes it difficult to determine the presence of malicious behaviors from network traffic data. In this case, the speed of judging and labeling traffic data is limited even for experienced experts, and also prone to label the data incorrectly. Therefore, we claim that the limited labeled data is not enough to produce a good model. To overcome this difficulty, LTM leverages semi-supervised learning, which is an approach that combines labeled data with unlabeled data during training, so it falls between unsupervised learning and supervised learning. The effectiveness of semi-supervised learning is based on several different assumptions, among which the manifold assumption claims that the data lie approximately on a manifold of much lower dimension than the input space. In this case, learning the manifold using both labeled and unlabeled data can avoid the curse of dimensionality.

For a training dataset containing unlabeled data, we use a denoising autoencoder (DAE) to learn the identity function of the original data distribution, which can easily be stacked to initialize ANNs. To learn more meaningful relationships between features, it is usually necessary to set some constraints. One of the most common constraints is that the number of neurons at the inner layer should be small, and when that constraint is satisfied, we often claim that DAE learns a compressed representation of the data. It aims to make the learned representations robust to partial corruption of the input pattern. We give pseudo labels to each unlabeled sample by selecting the class that has the maximum predicted probability. Therefore, our classification model of DDoS attack is trained in a supervised fashion with labeled and unlabeled data simultaneously, and with the same batch size. The reasons this method could work are similar to those in [8].

Global Aggregation Module

GAM runs on the server and is responsible for aggregating received gradients and applying them to the global model.

We believe that data packets need to be streamed and processed in near real time on switches and routers, and the model training should not impose a significant transmission overhead that could lead to network congestion. Therefore, GAM prunes the global model following the pruning policy.

Model pruning is a common approach to reducing computation and communication costs in the training stage, which is evaluated to be effective in collaborative learning on edge devices. Model pruning can be classified into two categories: fine-grained and coarse-grained.

Fine-grained pruning usually refers to weight pruning, which forces a portion of each layer's weights to zero according to the L1 or L2 norm, without affecting the structure of the model. In collaborative learning, before the server sends the parameters to the participant, a certain compression algorithm (e.g., gzip) will be applied to the pruned weights. The higher the pruning ratio, the more zeros are set, and the smaller the size of the compressed parameters. Coarse-grained pruning explicitly changes the model structure; specifically, it reduces the number of neurons based on the weight or activation value. The explicit change of the model structure leads to the creation of a new model and copy of all parameters, and the expenses outweigh the benefits while training small models. Therefore, we only use weight pruning to keep the transmission amount small between participants and server.

Packet Parse Module

As the network environment on the Internet changes often, and the model trained last month is not necessarily suitable for this month, we need to train the model continuously.

To better explain why we need PPM, taking a switch as an example, it only knows that it currently receives a packet, but without PPM, it does not know whether this packet is control information or traffic information. By control information, we refer to the model parameter and gradient information that need to be exchanged between the participants and server in the process of collaborative learning. Traffic information is the traffic data that needs to be detected. Therefore, PPM runs on both participants and the server, and is responsible for distinguishing between control information and traffic information.

We can design a protocol, and only data packets that meet the protocol format will be treated as control information. Also, we can speed up the transmission of model parameters and gradients through protocols. For example, in the later stage of training, we only transmit the parameter information that has changed in the entire model.

Trustworthy Network Architecture

For the vision of a more secure and trusted network, our framework can be embedded into the intelligence layer of the trustworthy network architecture. Figure 3 describes this trustworthy network architecture.

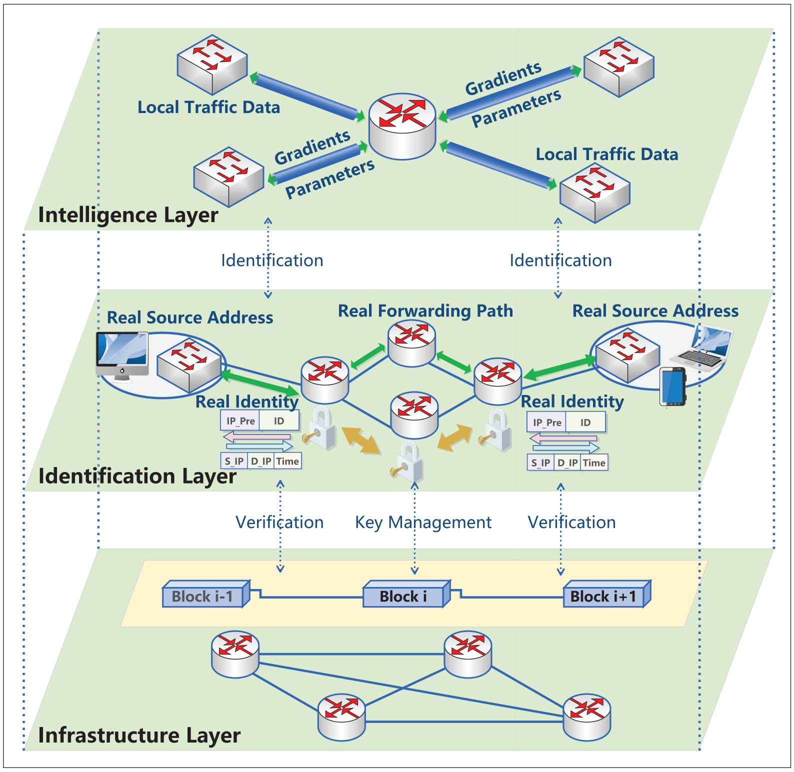

FIGURE 3. The trustworthy network architecture.

The intelligence layer is responsible for the analysis of network traffic and network behaviors, including intrusion detection, anomaly detection, as well as DDoS attack detection mentioned above. From the perspective of machine learning, this problem is to obtain a classifier model through training, usually composed of ANNs, which can classify network traffic data into different attack types or benign. It is for the purpose to obtain better classifier models through intra- and inter-domain collaboration that we propose NetSpirit.

The identification layer is under the intelligence layer. In this layer, we can use the source address verification technique (e.g., SAVI [9]) to guarantee the authenticity of the source address of the endpoint. Based on the real source address, we can also ensure the authenticity of the forwarding path and identity by tagging the packets and designing different security levels of tagging verification methods. The model obtained by collaboration in the intelligent layer can also work for the identification layer, and [10] has examined the connections between the identifier and collaborative network elements.

Since the network was not created with consideration of the trust issue, the foundation for trust in the Internet is lacking. In the infrastructure layer, a distributed consensus infrastructure has to be built to guarantee truthful storage and computation in an untrustworthy network environment. This layer is also responsible for the functions of node management, node information query, and node information maintenance. In our opinion, choosing permissioned blockchain to be the trust anchor for the entire architecture is one of the most practical options at the moment. The security of collaborative learning in the intelligent layer partly depends on the infrastructure layer, and [11] has studied how to make collaborative learning inherently more secure using blockchain.

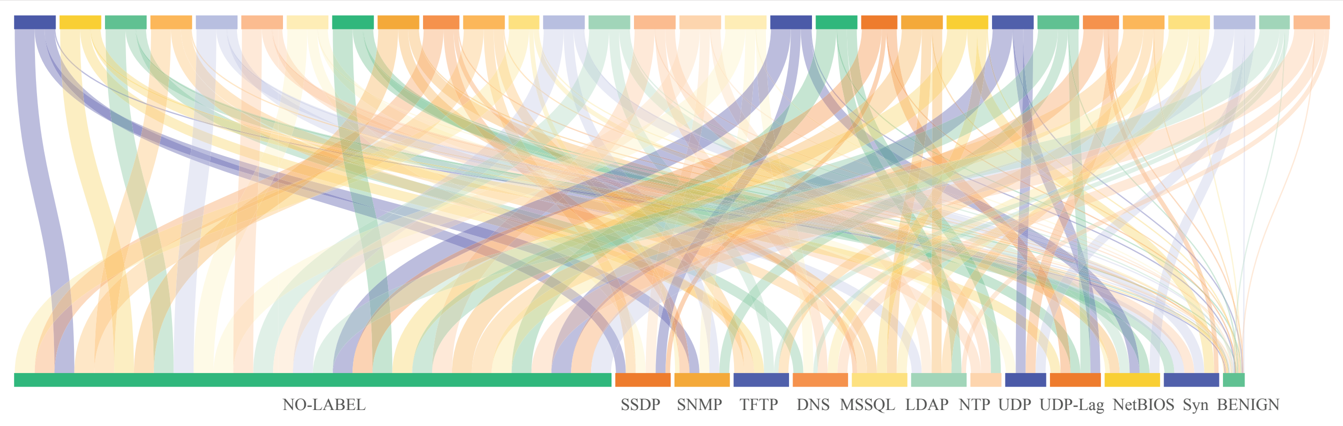

FIGURE 4. The participant-label distribution of our training dataset.

Evaluation

We use the CICDDoS2019 dataset [12] as the data source, which resembles real-world data (PCAPs), and contains benign and the 12 most up-to-date common DDoS attacks. We use the CSV version of this dataset, in which data packets have been parsed and 88 traffic features have been extracted. This dataset consists of two parts: one part was captured on January 12, 2019, called the training day; the other part was captured on March 11, 2019, called the testing day.

Our training dataset: We sample 300,000 attack samples and 300,000 benign samples from each file of the training day before randomly deleting 10 percent of the feature values. Then we distribute these samples to 30 participants in such a way that each participant can get at most two types of attack samples.

Our testing dataset: We sample 100,000 attack samples and 100,000 benign samples from each file of the testing day, and we also sample 100,000 attack samples and 100,000 benign samples from each of the five files (SSDP, DNS, NTP, SNMP, and TFTP) of the training day to make the testing dataset more comprehensive and more balanced, since these types are not included on the testing day.

We use an ANN model that has one fully connected hidden layer of 256 neurons, and similar model structures are very commonly used in the detection of network attacks. In this article, we choose this model structure for the following reasons:

1. The number of parameters is not large but sufficient, and it can be seen afterward that this model structure could be overfitted without any countermeasures.
2. The fully connected layer is suitable for model pruning and compression, which is compatible with the final purpose of this framework being used in the network.
3. This is the model with the best accuracy among the models we have tested.

Here we compare the following six methods:

1. Local: This method means that one device owns all training data, which is impossible in reality, so it should provide an optimal value of training loss.
2. Collaborative: This method stands for traditional collaborative learning, and the following are all collaborative.
3. Semi-supervised (SS): In this method, we marked U percent samples of each participant as unlabeled data to evaluate whether the semi-supervised learning is effective.
4. Model pruning (MP): We perform model pruning by forcing the smallest P part of the weights to be zero every *K* rounds to evaluate whether it is effective.
5. Semi-supervised and model pruning (SSMP): In this method, semi-supervised machine learning and model weight pruning are used together.
6. Semi-supervised and halfway model pruning (SSHFMP): In this method, they are also used together, but model pruning is only triggered in the first λ part of the rounds.

After we set U = 50 and perform marking, the distribution of the training dataset can be seen in Fig. 4, where the upper segments indicate the different participants, and the lower segments indicate the different labels, and the curves in the middle indicate how much of each class of data is available in a certain participant.

To conduct our experiment, we build a simulated environment using MindSpore [13] (on Intel® Core™ i7-9700K), in which *K* is fixed to 20, λ is set to 0.5, and P can be 0.1 or 0.2.

Figure 5a shows a line chart of the loss function value with respect to the number of rounds. The training loss of SSMP keeps oscillating until the end, which means that the model cannot converge. Furthermore, we can learn from Fig. 5a that the training loss of SSHFMP is generally on a downward trend until the end, which means the model has converged, and it is even smaller than MP at the end.

Figure 5b shows a line chart of the testing accuracy with respect to the number of rounds. The testing accuracy of the normal collaborative method is unstable, demonstrating the largest fluctuations among all the methods. When semi-supervised learning is used to leverage unlabeled data, the testing accuracy stays stable. While model pruning is added, the testing accuracy starts to fluctuate significantly again. More importantly, we can learn from Fig. 5b that SSHFMP effectively suppresses the fluctuation of the model's performance.

Table 1 shows the evaluation results after 400 rounds. We use the top-3 accuracy and top-5 accuracy as additional metrics in this table. The top-*N* accuracy is the accuracy where true class

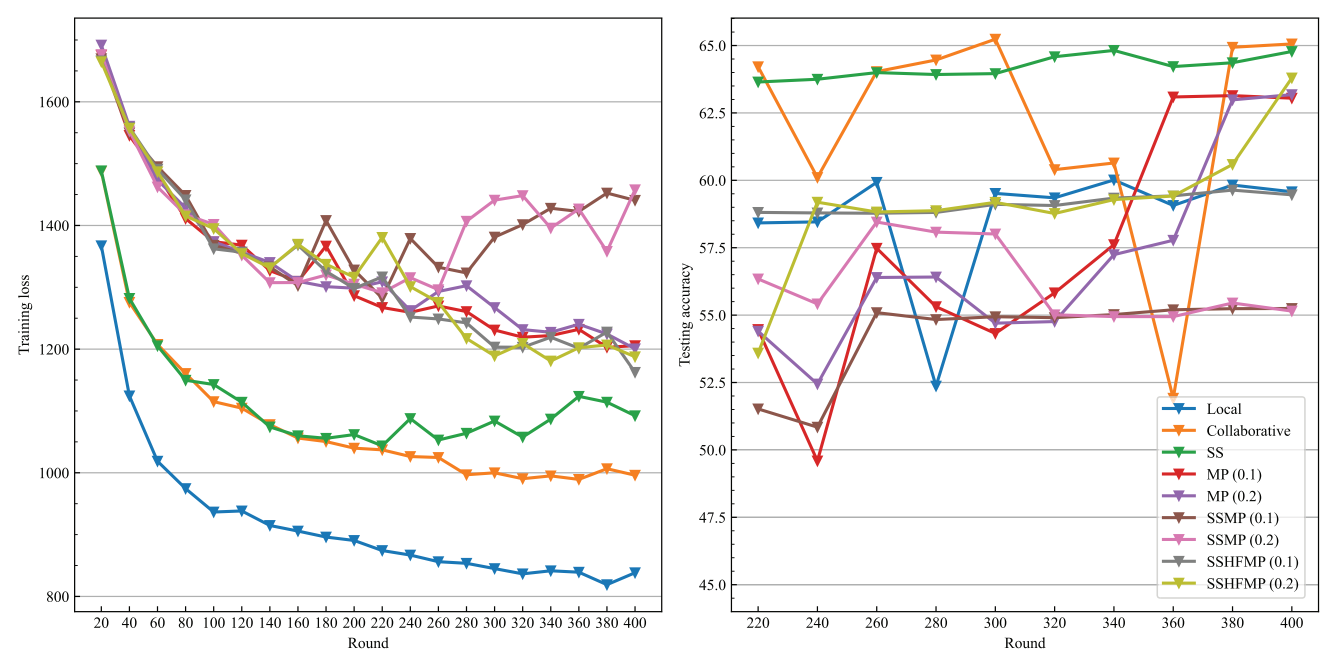

FIGURE 5. Training loss and testing accuracy during training.

matches any one of the *N* most probable classes predicted by the model. We can see that the local method produces the lowest training loss and the highest training accuracy, but not the highest testing accuracy, which indicates that the model could be overfitting if we do not take any measures. As for the SS method, unlabeled data accuracy is significantly higher than testing accuracy, which means this method is effective for learning from unlabeled data. The average transmission amount of the MP method is much smaller than traditional collaboration, which means this method is able to reduce the transmission amount while collaborating. Furthermore, we can learn from Table 1 that SSHFMP achieves much lower training loss than SSMP, and it outperforms SSMP in terms of labeled and unlabeled data accuracy, as well as all testing accuracies. Although an increase in average transmission amount can be seen when we switch from SSMP to it, SSHFMP can still reduce by 28.28 percent the transmission amount compared to the normal collaborative method. Moreover, if we compare SSHFMP to SS, not only is the average transmission amount reduced, but also the total computing time; if we compare SSHFMP to MP with the same P equal to 0.2, with only 50 percent labeled data, the training loss is lower and the testing accuracy is higher.

In summary, SSHFMP integrates semi-supervised learning and model pruning into collaborative learning in a flexible way. This method can leverage unlabeled data and achieve a detection accuracy of 63.80 percent, with a top-3 accuracy of 87.57 percent and a top-5 accuracy of 90.34 percent, which are not much lower than that of the normal collaborative method using full labeled data. Furthermore, it can also make a good trade-off between computing time and transmission amount by adjusting the hyperparameters P and λ. Therefore, by using SSHFMP, NetSpirit is quite suitable for the scenario of the effective detection of DDoS attacks.

Open Issues and Future Directions

Powerful Control Plane

The ANN model training and pruning require substantial resources on the control plane, which may be hard to achieve in current network devices. To overcome this challenge, the model establishment of our proposed NetSpirit can rely on a powerful controller that can be prepared for each device. Furthermore, the specifically designed AI chip can also be employed to empower the control plane that actually acts as a controller, particularly for float arithmetic resources. NVIDIA is now making steady progress in this direction with the launch of the BlueField-3 data processing unit (DPU).[1]

Intelligent Data Plane

In our proposed NetSpirit, the detection of DDoS attacks requires the computation of model inference on the data plane, which will involve multiplication operations and float arithmetic. This goes against the current data plane, which only supports addition, xor, and bit shifting operations of integers. To address this challenge, there are currently two directions. One is to convert all difficult calculations into table lookup without adding new hardware modules. IIsy [14] is a framework that converts features and different models into match-action tables to do in-network classification. The other is to add new hardware modules to provide new abstract primitives. Taurus [15] creates map and reduce primitives by adding a P4 control block and uses the SIMD parallelism to provide high computing throughput. It is an intelligent data plane that can perform the ANN model inference at line rate.

Automated Machine Learning

To reduce the inference time, it is important to find ANN models that perform well with fewer neurons, filters, and layers. Moreover, DDoS attacks have uncertainty. When the attack pat-

[1] https://www.nvidia.com/en-us/networking/products/data-processing-unit/

Method	Training loss	Labeled data accuracy (%)	Unlabeled data accuracy (%)	Testing accuracy (%)	Testing top-3 accuracy (%)	Testing top-5 accuracy (%)	Average transmission amount (bytes)	Total computing time (s)
Local	838.33	79.54	–	59.58	87.15	90.28	–	522.50
Collaborative	996.06	69.74	–	65.06	88.51	90.30	91568.45	540.74
SS	1092.27	70.14	68.08	64.78	87.58	90.13	91585.40	545.40
MP (P = 0.1)	1205.60	66.14	–	63.05	83.65	87.92	52234.40	548.51
MP (P = 0.2)	1200.51	66.55	–	63.18	83.64	88.08	47727.25	562.26
SSMP (P = 0.1)	1440.67	61.37	59.36	55.25	86.26	90.23	54677.75	538.13
SSMP (P = 0.2)	1457.75	61.01	59.15	55.15	86.43	90.25	50210.55	538.44
SSHFMP (P = 0.1, λ = 0.5)	1162.42	64.84	62.81	59.36	87.56	90.17	68116.10	531.36
SSHFMP (P = 0.2, λ = 0.5)	1187.65	68.26	65.99	63.80	87.57	90.34	65671.05	537.30

TABLE 1. Evaluation results after 400 rounds.

tern changes, the corresponding detection model should be able to update autonomously to adapt to new attack patterns. Having experts design the models is not only time-consuming and labor-intensive, but also makes it difficult to keep up with the rapidly changing attack patterns. Automated machine learning can be used to better deal with the changes of DDoS attacks and achieve efficient detection. While neural architecture search, which directly produces ANN model structures from datasets, has currently become a hot research topic, meta-learning goes a step further and does not even require the user to know the actual model being run.

Conclusion

In this article, as a traffic-driven defense scheme, we propose a collaborative learning framework called NetSpirit to achieve the effective detection of DDoS attacks. Leveraging parameter interactions instead of traffic data between network elements, the DDoS detection model can be efficiently trained and synchronized, with lightweight overhead and packet privacy protection. Unlike traditional collaborative learning, NetSpirit is practical because it integrates semi-supervised learning to learn from unlabeled data, and NetSpirit is efficient because it integrates model pruning to reduce the transmission cost. The experimental results demonstrate that NetSpirit can reduce by 28.28 percent the average transmission amount and achieve a detection accuracy of 63.80 percent, with a top-3 accuracy of 87.57 percent and a top-5 accuracy of 90.34 percent for the 13-classification problem of DDoS attacks using only 50 percent labeled data. By adjusting the hyperparameters, it can also make a good trade-off between computing time and transmission amount. We hope the intra- and inter-domain collaboration in NetSpirit can act as a fundamental primitive to build the intelligence layer of a trustworthy network architecture.

Acknowledgments

This research was in part supported by the National Key R&D Program of China with No. 2018YFB0803405, the National Science Foundation for Distinguished Young Scholars of China with No. 61825204, the National Natural Science Foundation of China Grants with No. 61932016, Beijing Outstanding Young Scientist Program with No. BJJWZYJH01201910003011, Beijing National Research Center for Information Science and Technology (BNRist) with No. BNR2019RC01011, and the PCL Future Greater-Bay Area Network Facilities for Largescale Experiments and Applications with No. LZC0019; and sponsored by CAAI-Huawei MindSpore Open Fund and CCF-AFSG Research Fund.

References

[1] M. Groskop *et al.*, "Global Application & Network Security Report 2019–2020," *Radware*, Tech. Rep., 2020.

[2] D. Gong *et al.*, "Practical Verifiable In-Network Filtering for DDOS Defense," *2019 IEEE 39th Int'l. Conf. Distributed Computing Systems*, 2019, pp. 1161–74.

[3] V. Gulisano *et al.*, "Stone: A Streaming DDOS Defense Framework," *Expert Systems with Applications*, vol. 42, no. 24, 2015, pp. 9620–33.

[4] R. Shokri and V. Shmatikov, "Privacy-Preserving Deep Learning," Proc. 22nd ACM SIGSAC Conf. on Computer and Commun. Security, 2015, pp. 1310–21.

[5] K. Bonawitz *et al.*, "Towards Federated Learning at Scale: System Design," arXiv preprint arXiv:1902.01046, 2019.

[6] T. Zhang and Q. Zhu, "Distributed Privacy-Preserving Collaborative Intrusion Detection Systems for VANETs," *IEEE Trans. Signal and Info. Processing over Networks*, vol. 4, no. 1, 2018, pp. 148–61.

[7] T. V. Khoa *et al.*, "Collaborative Learning Model for Cyberattack Detection Systems in IoT Industry 4.0," *2020 IEEE Wireless Commun. and Networking Conf.*, 2020.

[8] D.-H. Lee *et al.*, "Pseudo-Label: The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks," *Wksp. Challenges in Representation Learning*, ICML, vol. 3, no. 2, 2013.

[9] J. Wu *et al.*, "Source Address Validation Improvement (SAVI) Framework," IETF RFC 7039, 2013.

[10] S. Yao *et al.*, "Si-Stin: A Smart Identifier Framework for Space and Terrestrial Integrated Network," *IEEE Network*, vol. 33, no. 1, Jan./Feb. 2019, pp. 8–14.

[11] Z. Zhang *et al.*, "Seccl: Securing Collaborative Learning Systems via Trusted Bulletin Boards," *IEEE Commun. Mag.*, vol. 58, no. 1, Jan. 2020, pp. 47–53.

[12] I. Sharafaldin *et al.*, "Developing Realistic Distributed Denial of Service (DDOS) Attack Dataset and Taxonomy," *2019 Int'l. Carnahan Conf. Security Technology*, 2019.

[13] Huawei, "Mindspore," 2021; https://www.mindspore.cn/.

[14] Z. Xiong and N. Zilberman, "Do Switches Dream of Machine Learning? Toward In-Network Classification," *Proc. 18th ACM Wksp. Hot Topics in Networks*, 2019, pp. 25–33.

[15] T. Swamy *et al.*, "Taurus: An Intelligent Data Plane," arXiv preprint arXiv:2002.08987, 2020.

Biographies

KE XU [SM] (xuke@tsinghua.edu.cn) received his Ph.D. from the Department of Computer Science & Technology of Tsinghua University, where he serves as a full professor. He has published more than 100 technical papers and holds 20 patents in the research areas of next generation Internet and network virtualization and optimization.

YONG ZHENG (zheng-y19@mails.tsinghua.edu.cn) received his Bachelor's degree from Hohai University, China, in 2019. He is working toward a Master's degree supervised by Prof. Ke Xu in the Institute for Network Sciences and Cyberspace at Tsinghua University. His research interests include collaborative learning, cyberspace security, and blockchain.

SU YAO (yaosu@tsinghua.edu.cn) received his Ph.D. degree from Beijing Jiaotong University. Since 2017, he has undertaken postdoctoral research in the Department of Computer Science and Technology, Tsinghua University. Currently, he serves as an assistant research fellow at Tsinghua University. His research interests include future network architecture, IoT security, and blockchain systems.

BO WU (wub14@tsinghua.org.cn) received his Ph.D. degree from the Department of Computer Science and Technology at Tsinghua University. From 2019 to 2021, he acted as a research fellow in the Network Technology Laboratory at Huawei Technologies. Currently, he works in the Technology and Engineering Group at Tencent. His research interests include network architecture and security, and network AI.

XIAO XU (1901210544@pku.edu.cn) received his Bachelor's degree from Jilin University, China, in 2018. He is working toward a Master's degree supervised by Prof. Ke Xu in the School of Software and Microelectronics at Peking University. His research interests include collaborative learning and cloud computing.