Network-based Intrusion Detection Through Image-based CNN and Transfer Learning

Pedro Horchulhack, Eduardo K. Viegas, Altair O. Santin, João A. Simioni
Graduate Program in Computer Science — Pontifical Catholic University of Parana (PUCPR), Brazil
{pedro.horchulhack, eduardo.viegas, santin, joao.asimioni}@ppgia.pucpr.br

***Abstract*—Machine learning (ML) techniques for network intrusion detection is still limited in production environments despite promising results reported in the literature. Network traffic behavior exhibits considerable variability and evolves over time, requiring periodic model updates. This paper proposes a new approach to intrusion detection modeling based on CNN and transfer learning to reduce updating overhead. Its implementation is twofold. First, CNN is implemented using flow-based feature expansion derived from neural flattened hyperdimensional space. This expanded space representation contributes to a longer model lifetime and maintains system accuracy over time. Second, the required training data and computational cost are significantly reduced by performing periodic model updates based on a transfer learning approach. Experiments on a novel dataset with over 2.6 TB of data and one year of real-world network traffic demonstrate the feasibility of the proposal. Our proposal improves the average F1 score by up to 0.19 when no model updates are performed. While improving the system's accuracy, model updates impose only 42.8% of the computational cost.**

***Index Terms*—Intrusion Detection, CNN, Transfer Learning**

I. Introduction

In recent years, there has been a substantial escalation in the incidence of network attacks. For example, a recent security report [1] indicates a 150% surge in the number of Distributed Denial-of-Service (DDoS) attacks over the past three years. Network operators employ Network-based Intrusion Detection Systems (NIDSs) to identify the increasing array of threats. These systems utilize either *rule-based* or *behavior-based* techniques [2]. *Rule-based* approaches searches for well-known attack patterns in their input data, making them capable of detecting only previously known threats. Conversely, *behavior-based* techniques identify threats based on deviations from the modeled behavior, allowing them to detect new attacks provided that these new threats behave similarly to those modeled during the system's behavior modeling process [3].

Consequently, *behavior-based* intrusion detection techniques have been the most extensively researched approach in the literature [2], with authors frequently employing pattern recognition techniques, often in the form of Machine Learning (ML) tasks [4]. To attain this objective, a behavioral ML model is developed based on the environment behavior available in a training dataset [5]. Subsequently, the ML model can be used to identify intrusions in production [6].

In practice, networked environments present numerous challenges to ML-based techniques in contrast to fields where it has been successfully applied over the years [5]. On the one hand, network traffic behavior undergoes significant changes over time, either due to the emergence of new attacks or the introduction of new services [2]. These changes require the ML model to be regularly updated, a process that entails the creation of a new training dataset and the conduction of a computationally intensive model training. As a result, providing an updated ML model can often take several days or even weeks to be conducted. In addition, the behavior of network traffic in production environments can be highly variable, featuring multiple variations that arise from network traffic properties. This attribute demand that the underlying ML model can effectively capture dynamic and variable behavior. Surprisingly, the majority of current ML-based approaches rely on *shallow* classifiers [4]. While these approaches often yield high accuracy rates during testing, they struggle to represent the true complexities of network traffic due to the limitations of their training structure [7].

In recent years, a substantial portion of ML applications has relied on Deep Neural Network (DNN) techniques, notably Convolutional Neural Networks (CNNs), which have showcased their excellence across diverse domains like image recognition and object detection, delivering state-of-the-art results [8]. However, the effectiveness of CNN-based approaches typically hinges on having substantial features in the input data (e.g., pixels in image applications) and access to significant training data. This presents a challenge when applying CNN to NIDS, as the network traffic behavior is usually represented by a constrained number of flow-based features, typically numbering in the dozens derived from summarizing network traffic over time windows [9]. Conversely, NIDS model training and update task requirement for minimal data is a must, given the challenges in acquiring labeled network traffic, a task that frequently relies on human intervention [5].

Contribution. In light of this, this paper introduces a new network-based intrusion detection model that utilizes image-based CNNs and employs transfer learning for model updates. The objective of this model is to extend the model's lifespan while reducing the cost of model updates. The implementation is divided into two phases. First, the CNN is implemented by expanding flow-based features into a hyperdimensional space, resulting from a neural network's hidden layer. This expanded space representation contributes to an extended model lifespan and the maintenance of system accuracies over time. Second, our approach performs periodic updates through a transfer learning method leveraging the outdated CNN model

to facilitate model updates. The key insight is that model updates can be carried out with decreased computational costs and a reduced need for training data by leveraging the prior knowledge of the outdated CNN model.

The main contributions of this paper are:

- An evaluation of conventional *shallow*-based intrusion detection approaches concerning their accuracy in intrusion detection over time. The experiments on a novel dataset comprising over 2.6TB of real network traffic revealed that these approaches experience a notable decline in accuracy as time passes.
- A new network-based intrusion detection model that performs the classification task through an image-based CNN and model updates through transfer learning. The proposed model can improve the system F1-score by up to 0.19 while demanding an average computational training cost of only $42,8\%$.

II. Preliminaries

Network operators frequently rely on applying NIDSs tools to identify network-related attacks [10]. In general, *behavior-based* NIDSs are built by implementing four sequential modules, namely *Data Acquisition*, *Feature Extraction*, *Classification*, and *Alert* [11]. First, the *Data Acquisition* module collects network events from the monitored environment, such as network packets from a Network Interface Card (NIC). Second, the behavior of the collected events is extracted by a *Feature Extraction* module, compounding an associated feature vector. In such a case, in general, the behavior of network events is represented by network flows that summarize the network communication between hosts and services in a given time window through dozens of features. The extracted feature vector is classified by a *Classification* module as either *normal* or *attack*, e.g., by applying a ML model. If the event is classified as *attack*, the *Alert* module properly reports it.

Over the past few decades, various approaches have been proposed for the classification task, with authors often turning to the application of ML techniques [4]. In such scenarios, an ML model is constructed using a training dataset comprising multiple samples expected to represent real-world production deployment behavior. In practice, network environments exhibit dynamic behavior, entail ML models that can handle these complexities, often only achievable by having large labeled training datasets [12]. Notwithstanding, even if the underlying ML model can address such complexities, it will need to be regularly updated, given the behavior changes of network traffic as time passes [5].

III. Related Works

Most ML-based NIDS approaches in the literature predominantly focus on achieving improved intrusion detection accuracy [4]. Albeit the promising reported results, they tend to overlook challenges associated with model updates arising from new network traffic [5] and the potential influence of network traffic variation on their proposed schemes. For instance, A. E. Kamali *et al.* [13] proposed the application of CNN coped with a Recurrent Neural Network (RNN) to conduct intrusion detection. Their scheme yielded exceptionally high detection accuracies on static and outdated datasets, largely overlooking the dynamics of network traffic behavior. Similarly, L. Yang and A. Shami [14] proposed the utilization of an ensemble of CNNs for intrusion detection, which led to improved detection accuracy compared to traditional techniques. However, their approach did not address the methodology for conducting model updates. To enhance classification reliability, authors also turn to the fine-tuning of model hyperparameters. As an example, A. N. Calugar *et al.* [15] optimizes CNN hyperparameters to increase classification accuracy on static datasets. Their approach overlooks network traffic behavior variations and how model updates can be conducted.

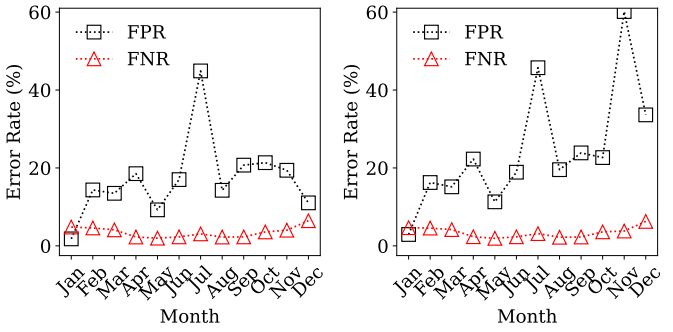
In recent years, several works have challenged the applicability of reported results in intrusion detection. G. C. Bertoli *et al.* [16] pursued better model generalization in spite of accuracy. The authors showed that CNN can improve detection generalization on multiple intrusion datasets. Their work did not address how model updates can be conducted. O. D. Okey *et al.* [17] aimed better model generalization through CNN transfer learning. The authors showed that pre-trained models can improve accuracy, but no discussion was made on how it can alleviate model updates. A similar approach was introduced by Sk. T. Mehedi *et al.* [18], employing transfer learning to enhance detection accuracy on a heterogeneous testbed. Despite improving detection accuracy and generalization, they did not address model updates. As a result, there is still a gap in the literature on understanding how network traffic behavior changes can be addressed in ML-based intrusion detection.

IV. Problem Statement

In this section, we delve deeper into the challenges posed by changes in network traffic behavior within production environments for traditional ML-based NIDS. To be more specific, we first introduce the dataset we have utilized, which encompasses a year's worth of real network traffic. Subsequently, we assess the performance of ML-based intrusion detection techniques.

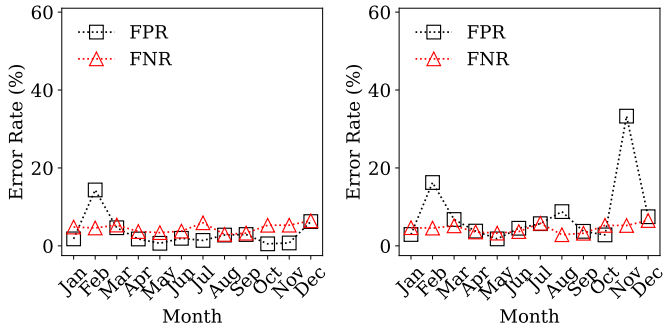
A. MAWIFlow

To enable the evaluation of intrusion detection schemes in the context of changing network traffic behavior, our work relies on the MAWIFlow dataset [5]. This dataset was created using Samplepoint-F from the MAWI archive, which comprises real network traffic collected daily for 15-minute intervals from a transit link between Japan and the USA. For the purposes of our evaluation we consider the entire network traffic collected during the year of 2014, which is subsequently used to assess widely used ML-based NIDS. The dataset contains over 2.6 TB of data, consisting of $\approx$ 300 billion network packets. To label events, we employ an unsupervised ML technique from MAWILab [19], which automatically classifies input records as either normal or attack. MAWILab utilizes several unsupervised ML algorithms to detect anomalies in MAWI data without the need for individual or human



(a) Random Forest (RF) (b) Extra Trees (EXTRA)

Fig. 1: Accuracy trend over a year of commonly used classifiers without periodic model updates. Classifier is trained in January and evaluated in subsequent months without updates.



(a) Random Forest (RF) (b) Extra Trees (ExT)

Fig. 2: Accuracy trend over a year with periodic model updates using commonly used classifiers, where the model is updated on a monthly-basis.

intervention in the event labeling task. The detected anomalies are labeled as attack, while the remaining data is assumed to be normal events. For the feature extraction task, we use BigFlow [5], which groups events in 15-second intervals and extracts 60 flow-based features from Nigel feature set [9].

B. On Dealing With Real Network Intrusions Over Time

The evaluation aims to answer two main Research Questions (RQs):

- (***RQ1***) *What is the accuracy performance of selected techniques over time without model updates?*
- (***RQ2***) *How do periodic model updates improve the accuracy performance of selected techniques?*

Two commonly used classifiers were evaluated, namely Random Forest (RF) and Extra Tree (ExT). Both use a decision tree as their base learner, implemented through a C4.5 algorithm, with a confidence factor of 0.25 and *gini* as the node split quality metric. Both classifiers use 100 decision trees as their base-learner and *gini* as quality measure split. In the training procedure, random undersampling without replacement was applied. The classifiers were implemented using *scikit-learn* API 1.3.1. The classifiers were evaluated through the False Positive (FP) and False Negative (FN) rates. The FP denotes the ratio of *normal* samples incorrectly classified as *attack*, and the FN denotes the ratio of *attack* instances incorrectly classified as *normal*.

To address *RQ1*, we assess the performance of the chosen classifiers without conducting any model updates as time progresses. The objective of this evaluation is to determine how the changes in network traffic behavior over time affect traditional ML-based intrusion detection techniques. To accomplish this objective, we conduct the model training process for the selected classifiers using data from January. Subsequently, we evaluate the accuracy performance of these classifiers over the year. This evaluation will help us understand the impact of evolving network traffic behavior on the performance of traditional ML-based intrusion detection techniques.

Figure 1 illustrates the accuracy performance of the selected classifiers when no model updates are conducted. The results show that the error rates are notably low in January, which is the time of the initial model training. However, as time progresses, these error rates gradually increases. For instance, when examining the RF classifier (Fig. 1a), we observe a FP rate of 44% in July, indicating an 25.4x increase compared to its performance in January. This trend is consistent with the ExT and highlights that current ML-based intrusion detection techniques, as used in the literature, struggle to adapt to the changes in real network traffic. These techniques experience a significant degradation in accuracy as time passes, indicating their limitations in handling evolving network traffic behavior.

To address $RQ2$, we assess the accuracy performance of the selected classifiers when model updates are conducted monthly. To accomplish this objective, model updates are performed at the beginning of each month using data from the previous month. For example, on March 1st, the models are updated using data collected from February 1st to February 28th. This approach allows us to analyze how periodic model updates may improve the performance of traditional ML-based intrusion detection techniques over time.

Figure 2 illustrates the accuracy performance of the selected classifiers when model updates are conducted every month. A notable improvement in accuracy can be observed when comparing these updated models to their counterparts without updates (Fig.1 vs. Fig.2). In practice, model updates make the evaluated ML classifiers capable of maintaining their accuracy over time. For example, the monthly updated RF classifier (Fig. 2a) presented a FP rate of 1.3% in July, a 33.5x decrease when compared to its no-updated counterpart. This indicates that periodic model updates significantly improve the performance and reliability of traditional ML-based techniques.

C. Discussion

The evaluation in this section highlights the limitations of current ML-based intrusion detection techniques in real production environments. These techniques struggle to adapt to evolving network traffic, needing frequent model updates to maintain accuracy. Developing schemes that capture network traffic complexities and reduce update frequency is essential.

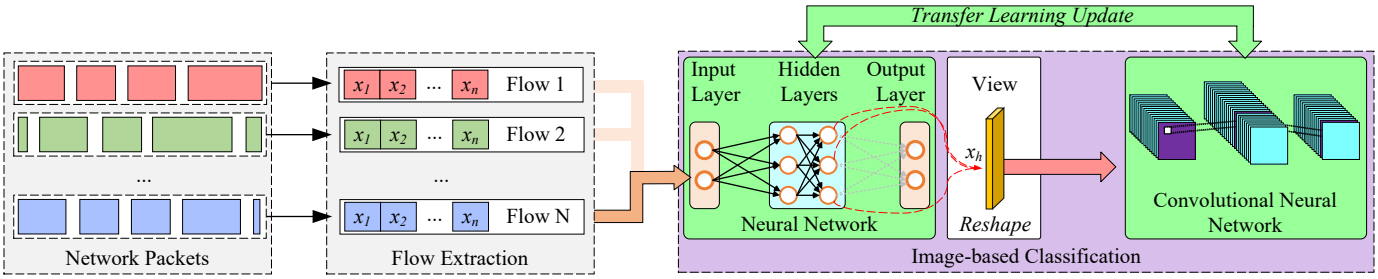


Fig. 3: Proposed image-based CNN intrusion detection scheme through transfer learning for model updates.

Efficient execution of updates, given labeling and computational challenges, is vital for reliable and cost-effective ML-based solutions in real-world scenarios.

V. An Image-based Intrusion Detection Model for Improved Model Lifespan and System Updates

To tackle the challenge of network traffic behavior changes, we propose a novel intrusion detection model using a CNN implemented through transfer learning. Our key insight in the proposed scheme is that leveraging the CNN can enhance system reliability over extended periods, improving its lifespan, while transfer learning can be used to facilitate model updates. As a result, our scheme can reduce the computational costs associated with model updates and require fewer labeled training examples of network traffic. The proposed model is shown in Figure 3, and is composed of two main steps, namely *Image-based Classification* and *Transfer Learning Update*.

The *Image-based Classification* module aligns with traditional NIDS settings, wherein network events are classified as either *normal* or *attack*. The classification procedure commences with the collection of network packets from the monitored environment. The behavior of the collected event is then extracted using a flow-based *Feature Extraction* module, compounding a flow-based feature set. To increase our model generalization, our proposed scheme generates a hyperdimensional feature set derived from a flattened output of a *Neural Network*'s hidden layer (Fig. 3, x_h). Our assumption is that the expanded feature space representation contributes to an extended model lifespan and the maintenance of the system accuracies over time. The resulting feature set is converted into an image representation format, which serves as input for the image-based CNN classification. As a result, the CNN evaluates a higher input dimensionality, increasing its generalization capabilities, resulting in a better model lifespan.

The *Transfer Learning Update* module is designed to address the evolving behavior of network traffic over time while easing the model update costs. In this context, our proposal incorporates model updates through a transfer learning approach, wherein both the outdated *Neural Network* and the CNN model (Fig. 3, *Transfer Learning Update*) are employed during model updates. Our key insight is to utilize the outdated model, resulting in significant reductions in computational expenses by leveraging prior knowledge of network data.

The subsequent subsections provide detailed description of our proposed model's classification and update procedures.

A. Image-based Classification

Current state-of-the-art classification accuracies are achieved by applying CNN architectures for the intrusion detection tasks. This is attributed to the improved CNN's ability to depict the training dataset's characteristics, a valuable feature for better generalization on network traffic classification. In light of this, our proposed model leverages CNNs to perform the network traffic classification. Our main insight is to use CNNs to increase the system lifespan as time passes, even if no model updates are conducted. To achieve such a goal, our proposed *Image-based Classification* module is implemented in two phases, as shown in Figure 3.

It considers a streaming of network packets captured from a given NIC. The behavior of the network packets is extracted, compounding an associated flow vector x of size n (Fig. 3, *Flow Extraction*). Given the flow vector x such that $x = \{x_1, x_2, ..., x_n\}$, our goal is to find the associated label y. To achieve such a goal, we first apply a neural network with a hidden layer composed of m neurons, where $m > n$ (Fig. 3, *Neural Network*). During training, the neural network employed is optimized for network traffic classification, adhering to identical training procedures as the CNN. Consequently, the hidden layer outputs non-linear hyperdimensional features that serve the classification process within a higher-dimensional space. The resulting vector, denoted as x_h such that $x_h = \{x_1, x_2, ..., x_m\}$ is subsequently reshaped into a matrix format, ensuring it adheres to squared-sized image constraints. The matrix is then used as input by our CNN model, which classifies it as either *normal* or *attack*.

Our proposal's core benefit is harnessing image-based CNNs to enhance system reliability and lifespan, enabled by their greater model complexity. We achieve such a goal twofold. First, we increase the feature dimensionality representation by extracting the outputs of a *Neural Network*'s hidden layers (Fig. 3, x_h). The resulting vector improves the subsequent CNN generalization capabilities and, thus, its lifespan. Second, we leverage the resulting hyperdimensional feature vector to compound an image representation, which is then used as input by our deployed CNN. As a result, our proposed scheme can significantly increase the network flow behavior representation before using it for the CNN classification task. Given such an approach, our model can further increase the CNN generalization and lifespan.

B. Transfer Learning Update

Training and updating CNN models pose considerable challenges in NIDS applications, primarily due to the substantial need for labeled data, frequently demanding human intervention. As a result, to make model updates more feasible, techniques must be designed to require fewer computational resources and less labeled training data. To tackle this challenge, our proposed scheme adopts a transfer learning approach for model updates, leveraging the outdated model to yield two key advantages. Firstly, it reduces computational costs by utilizing the CNN weights from the outdated model. Secondly, it minimizes the required training data, thanks to the prior knowledge embedded in the outdated CNN model with respect to network traffic.

Our proposed scheme assumes that model updates are executed periodically by a network operator (Fig. 3, *Transfer Learning Update*). At model update time, the *Neural Network* is updated considering the outdated model, and is optimized towards the classification of the newly updated training dataset. The resulting updated *Neural Network* is then used to generate the hyperdimensional space features to update the deployed CNN. In such a case, the weights of the outdated CNN model are adjusted based on the newly generated training dataset as produced by the *Neural Network* hidden layer outputs. The resulting updated neural network and CNN is then used in the production environment.

VI. Evaluation

The evaluation aims to answer the following RQs:

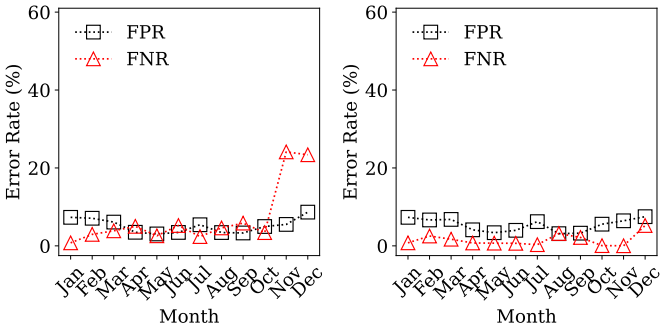
- (***RQ3***) *How does our proposed image-based CNN scheme perform when no model updates are performed?*
- (***RQ4***) *How does model updates affects our model?*
- (***RQ5***) *How does our transfer learning technique improve model training costs?*

The following subsections describe our proposed model construction and its evaluation.

A. Model Training

The proposed model (Fig. 3) was implemented and assessed using the dataset previously discussed in Section IV-A. The proposal *Neural Network* was implemented through a Multilayer Perceptron (MLP) architecture. The MLP comprises an input layer of 60 neurons, a hidden layer with $2,048$ neurons, followed by 2 neurons on the output layer. At training time the network was trained with $1,000$ epochs, with a $relu$ activation function and *adam* optimizer. The MLP was implemented using *scikit-learn* API $v1.3.1$.

In our CNN implementation, we evaluated our scheme using the GoogLeNet architecture. For both the training and update procedures, the inputs for the CNN model are derived from the hidden layer outputs of the MLP (as shown in Fig. 3, labeled as *Neural Network*). In this scenario, the MLP hidden layer output is scaled from a size of $2,048$ to a $48x48$ dimension using the PyTorch *view* function. The CNN was trained and updated using the *adam* optimizer, running for $1,000$ training epochs. We utilized *categorical cross-entropy* as the loss function, with a learning rate of 0.001. The model was implemented using PyTorch version $1.13.1$.



(a) Without model updates (b) With model updates

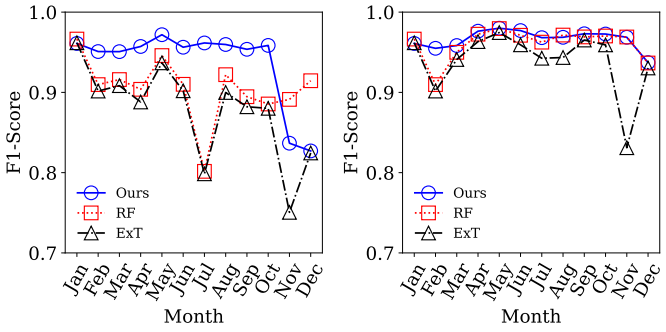
Fig. 4: Performance of our scheme on *MAWIFlow* dataset.

B. Addressing Network Traffic Behavior Changes

The initial experiment, which addresses $RQ3$, assesses the performance of our image-based CNN pipeline without conducting periodic model updates. In practice, we conduct the same procedure used previously, where we train the model based on January data and evaluate it without conducting model updates. Figure 4a shows the classification accuracy of our model in this context. Our proposed scheme demonstrates the ability to maintain its classification accuracy over extended periods, significantly improving the system lifespan when contrasted with traditional techniques (Fig. 4a $vs.$ Fig. 1). For instance, our proposed model decreases at most its FN rate to 27% throughout the evaluated year, while traditional techniques, such as RF (Fig. 1a), decrease up to 47% its FP rate. As a result, our proposal based on the extracted hyperdimensional features used as input to an image-based CNN can significantly increase the resulting model lifespan.

To address $RQ4$, we assess the performance of our proposed scheme when conducting monthly model updates. In this scenario, the MLP and CNN adopt a transfer learning approach for these updates. Figure 4b illustrates the accuracy performance of our model when periodic model updates are integrated. Our proposed scheme maintains consistent accuracy rates over the entire year while implementing model updates via a transfer learning scheme. In practice, our model delivered an average of 5.5% and 1.8% of FP and FN rates respectively.

We further investigate how our proposal can improve the system's reliability compared to traditional techniques. Figure 5a shows the F1 score of our scheme without periodic model updates $vs.$ traditional techniques. We measure the F1 score as the harmonic mean between precision and recall. It is possible to note a significant improvement in the accuracy performance. On average our scheme presented an F1 score of 0.93, an improvement of up to 0.18 and 0.19 compared to the RF and ExT classifiers respectively (on June). Figure 5b shows the F1 score as time passes when model updates are conducted. Similarly, our proposal reached more stable accuracies while providing high accuracy rates as time passed.



(a) Without model updates (b) With model updates

Fig. 5: Performance comparison on *MAWIFlow* dataset.

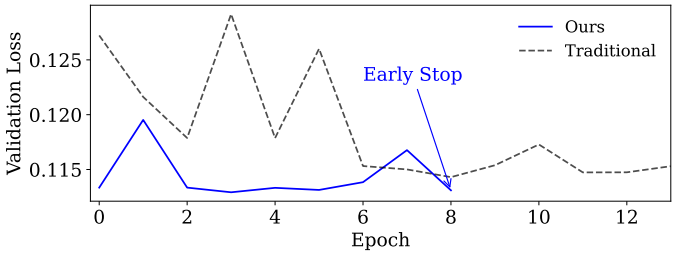


Fig. 6: Model training convergence on March with and without our proposed transfer learning approach.

To address $RQ5$, we investigate our scheme's model update computational costs using our proposed transfer learning approach. Figure 6 shows the model convergence rate of our scheme $vs.$ the traditional approach in March. The traditional approach was measured by training the model from scratch. It is possible to observe that our scheme significantly improves model convergence rates, reducing the required number of training epochs. On average, our model, thanks to the application of transfer learning, demanded only $42,8\%$ of training epochs for the conduction of model updates compared to the traditional scheme. As a result, our proposal can significantly improve the model's lifespan and accuracy while demanding less computational costs for model updates.

VII. Conclusion

In this paper, we presented a new CNN-based NIDS model that realistically captures the network traffic behavior in production environments. This leads to an extended model lifetime while maintaining system accuracy. In addition, our proposal uses a transfer learning approach to facilitate model updating, significantly reducing computational cost. The practical model applicability is demonstrated through experiments on a one-year dataset. The proposed model achieves higher accuracy than traditional techniques while keeping the computational cost of model updates low.

In future work, we aim to extend our proposal to unsupervised identification of new network traffic patterns.

Acknowledgment

This work was partially sponsored by Brazilian National Council for Scientific and Technological Development (CNPq), grants nº 304990/2021-3 and 407879/2023-4.

References

[1] "The state of ddos attacks ddos insights from q1 & q2, 2023," https://go.zayo.com/zayo-ddos-protection-ebook/, Zayo, Tech. Rep., 2023, accessed: 2023-10.

[2] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *2010 IEEE Symposium on Security and Privacy*, 2010, pp. 305–316.

[3] P. Horchulhack, E. K. Viegas, A. O. Santin, F. V. Ramos, and P. Tedeschi, "Detection of quality of service degradation on multi-tenant containerized services," *Journal of Network and Computer Applications*, vol. 224, p. 103839, Apr. 2024.

[4] B. Molina-Coronado, U. Mori, A. Mendiburu, and J. Miguel-Alonso, "Survey of network intrusion detection methods from the perspective of the knowledge discovery in databases process," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2451–2479, 2020.

[5] E. Viegas, A. Santin, A. Bessani, and N. Neves, "BigFlow: Real-time and reliable anomaly-based intrusion detection for high-speed networks," *Future Generation Computer Systems*, vol. 93, pp. 473–485, Apr. 2019.

[6] R. R. dos Santos, E. K. Viegas, and A. O. Santin, "A reminiscent intrusion detection model based on deep autoencoders and transfer learning," in *IEEE Global Communications Conference*, 2021.

[7] R. R. dos Santos, E. K. Viegas, A. O. Santin, and P. Tedeschi, "Federated learning for reliable model updates in network-based intrusion detection," *Computers amp; Security*, vol. 133, p. 103413, Oct. 2023.

[8] X. Wu, D. Sahoo, and S. C. Hoi, "Recent advances in deep learning for object detection," *Neurocomputing*, vol. 396, pp. 39–64, 2020.

[9] N. Williams, S. Zander, and G. Armitage, "A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 5, p. 5–16, Oct 2006.

[10] J. Geremias, E. K. Viegas, A. O. Santin, A. Britto, and P. Horchulhack, "Towards a reliable hierarchical android malware detection through image-based cnn," in *2023 IEEE 20th Consumer Communications amp; Networking Conference (CCNC)*. IEEE, Jan. 2023.

[11] P. Horchulhack, E. K. Viegas, and A. O. Santin, "Detection of service provider hardware over-commitment in container orchestration environments," in *IEEE Global Communications Conference*, 2022.

[12] R. R. d. Santos, E. K. Viegas, A. O. Santin, and V. V. Cogo, "Reinforcement learning for intrusion detection: More model longness and fewer updates," *IEEE Transactions on Network and Service Management*, vol. 20, no. 2, p. 2040–2055, Jun. 2023.

[13] A. E. Kamali, K. Chougdali, and K. Abdellatif, "A new intrusion detection system based on convolutional neural network," in *ICC 2023 - IEEE International Conference on Communications*. IEEE, May 2023.

[14] L. Yang and A. Shami, "A transfer learning and optimized CNN based intrusion detection system for internet of vehicles," in *ICC 2022 - IEEE International Conference on Communications*. IEEE, May 2022.

[15] A. N. Calugar, W. Meng, and H. Zhang, "Towards artificial neural network based intrusion detection with enhanced hyperparameter tuning," in *IEEE GLOBECOM*. IEEE, Dec. 2022.

[16] G. de Carvalho Bertoli, L. A. P. Junior, O. Saotome, and A. L. dos Santos, "Generalizing intrusion detection for heterogeneous networks: A stacked-unsupervised federated learning approach," *Computers & Security*, vol. 127, p. 103106, Apr. 2023.

[17] O. D. Okey, D. C. Melgarejo, M. Saadi, R. L. Rosa, J. H. Kleinschmidt, and D. Z. Rodriguez, "Transfer learning approach to IDS on cloud IoT devices using optimized CNN," *IEEE Access*, pp. 1023–1038, 2023.

[18] S. T. Mehedi, A. Anwar, Z. Rahman, K. Ahmed, and R. Islam, "Dependable intrusion detection system for IoT: A deep transfer learning based approach," *IEEE Transactions on Industrial Informatics*, no. 1, pp. 1006–1017, Jan. 2023.

[19] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, "MAWILab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking," in *Proc. of the 6th Int. Conf. on emerging Networking EXperiments and Technologies (CoNEXT)*, 2010.