2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Explainable Artificial Intelligence for Improving a Session-Based Malware Traffic Classification with Deep Learning

Stefan Machmeier[1], Maximilian Hoecker[1] and Vincent Heuveline[1]

Abstract— **In network security, applying deep learning methods to detect network traffic anomalies has achieved great results with various network traffic representations. A possible representation is the transformation of raw network communication to images to extract valuable information from the unmanageable amount of network traffic by applying representation learning. However, since deep learning models can result in black boxes for users, it is interesting to understand what valuable information is learned from network communication converted into images. This paper elaborates on that question using explainable artificial intelligence (XAI) methods to identify network packets that most influence the prediction and verify that packets in a malware communication containing malicious payloads have high influence on the prediction. We inspect the *Grad-CAM* and visualize the *Integrated Gradients* of Xception and VGG-19 model and investigate the attention heat maps of our Vision Transformer (ViT) model. In addition, we present a novel transformation of sessions to a new image representation to expand the informativeness of network communication. For multiclass classification, our best model Xception achieves an accuracy of 97.95%, whereas, for binary classification, Xception and VGG-19 achieve well above 99.50%. Our ViT model achieves a significantly lower performance with 95.86% for multiclass and 99.36% for binary classification. In particular, computing centers could benefit by examining their inbound and outbound traffic to detect malicious behaviors ahead of time.**

I. Introduction

Security operators deal with network traffic alerts daily, while adversarial actors try to bypass their counterparts, leaving minor traces. Automatically processing network activities could aid operators in unveiling malicious behaviors in the unmanageable amount of packets; however, it is a thin line between flooding them with false-positive alerts and not detecting security threats. Therefore, previous studies proposed representation learning methods to learn features from raw traffic data and classify them for malware detection [1]–[7], traffic identification [5], [6], [8], and traffic categorization [8]. One emerging approach is converting network traffic to images of various sizes, lengths, and dimensions. Recent studies have provided preliminary insights and shown good classification performance using deep learning, where researchers converted each packet into an image and used it as an input feature to train their models [1]–[6], [8]. Thus, new deep learning approaches could help security operation centers (SOCs) to detect malicious actors and minimize potential harm when systems are affected. Unlike inspecting each packet individually like common deep packet inspection (DPI) does, the field of machine learning in network traffic classification is capable of processing complete network communication without limitations such as limited expressiveness of format rules [6]. For specific attacks such as Christmas tree attacks [9], DPI tools such as nDPI [10] can reliably detect such packets. One crucial aspect is labeling network traffic with tools of unknown reliability [11]. For instance, traffic classification solutions that propose high classification results using a private ground truth labeled with techniques of unknown reliability are challenging to compare and validate. Hence, an image representation of raw network traffic could help to overcome these obstacles. The question remains: What does a model learn from raw traffic data? For instance, classification tasks on the CIFAR-10 data set differ from images of network packets. For CIFAR-10 images, a human eye can see clear borders and separate objects from each other, whereas looking closely at raw traffic data converted to an image, a human eye cannot comprehend the traffic behind it. Therefore, it is interesting to interpret trained models to see if they can comprehend the traffic in the image.

In this study, we apply explainable artificial intelligence (XAI) methods to understand deep learning models for network traffic classification and verify that malicious packets within a session correspond to the prediction. We investigate the *Integrated Gradients* [12], and visualize the *Grad-CAM* [13] of VGG-19 and Xception to see the area that most influences the prediction. For the Vision Transformer (ViT) model, we visualize the attention heat maps to identify packets with high attention weights that affect the model's outcome [14], [15]. In addition, we propose a novel method to classify malicious network traffic by converting network session traffic into images.

II. Related Work

Applying deep learning to network traffic detection and classification is a widely used approach [1]. In 2005, Kim and Reddy [2], [3] generated images of network traffic to enable a real-time analysis to detect network anomalies like Denial of Services (DoS) or Worms spreading throughout a network. The authors retrieved the normalized packet counts in the address domain and converted them to pixel intensity. All four bytes of the IP address were converted into a (16×16) square for the source and destination address. They investigated the impact of sampling rate and retained discrete cosine transform (DCT) coefficients for detecting and identifying malicious network behavior [2], [3].

*This work was not supported by any organization

[1]Engineering Mathematics and Computing Lab, Heidelberg University, 69120 Heidelberg, Germany `<firstname.lastname>@uni-heidelberg.de`

Wang et al. [4] gave the first insights into convolutional neural network (CNN) performances by classifying malware traffic with raw traffic data. They created a data set that contains network traffic of ten malware and ten benign classes, where the malware traffic is provided by the Stratosphere Lab at Czech Technical University in 2016 [16]. The benign traffic was created using a network traffic simulation. They suggested using a session representation and splitting traffic into discrete units, where each unit serves as one data point to be classified by the CNN [4]. To assure same input size, the authors trimmed each flow to 728 bytes with an image dimension of (28×28) pixel.

Lim et al. [5], [6] suggested using only payload data of network traffic. The authors hypothesized that header data could bias the model and hinder generalization if the data set was from controlled environments. Instead of cutting flows directly to a specific size, they cut each packet to a specific size and then use a fixed number N of the first packets of each flow to ensure an equal image size when flows are used as input data. They applied deep neural networks to the problem and achieved the best results using the largest packet size and the highest number of packets in a flow.

Based on this image representation, Xu et al. [8] classified Android malware and presented Falcon, a framework to detect and categorize malware. The authors used the Android Malware CICMal2017 data set [17], trimmed each flow to the same length of 728 bytes, and represented them in a 2D grayscale image. Unlike Lim et al. [5], [6], they used a bi-directional Long Short-Term Memory (LSTM) model for feature extraction. They passed them to a classifier to detect or categorize the flow. The authors evaluated various classifiers and claimed that Random Forest (RF) yields the best performance.

Cao et al. [1] proposed a new traffic feature representation method in which each session is transformed into a unified image where all headers, all source, and destination payload form one channel in the image, respectively. In order to achieve a unified representation, they used a Markov transition matrix. To cope with data imbalance, the authors used the feature images to train a Generative Adversarial Network (GAN) model with a sample filter only to add generated samples close to the distribution of the real samples. The authors trained their model on different Malware Capture Facility data sets [16] and yielded a detection accuracy of up to 99%. For imbalanced classes, they improved up to 6% in classification accuracy.

Interpreting deep learning models for malware classification based on bi-directional network communication, also referred to as a session, transformed into an image was not investigated until now. Therefore, we apply XAI methods to understand the influence of packets of our new image representation for network sessions on the model's outcome.

III. Methodology

A sequence of packets at an observation point summarizes to a flow with the condition that each packet is sent from a particular source IP address to a particular destination IP address without a statically bound transport connection [18]–[21]. The 5-tuple consisting of source IP address, destination IP address, source port, destination port, and transport protocol defines the network frame [21], [22]. A set of raw packets $P = \{p_1, p_2, \ldots, p_n\}$ at an observation point is defined by the 5-tuple x_i, byte size $b_i \in [0, \infty]$, and the transmission time $t_i \in [0, \infty]$ bounded by the maximum packet transmission time, where $p_i = \{x_i, b_i, t_i\}$ defines a packet [8], [21], [22]. All traced packets $P = \{p_1 = \{x_1, b_1, t_1\}, \ldots, p_n = \{x_n, b_n, t_n\}\}$ define a set of flows due to established connections, and thus a flow $f = \{x, b, d_t, t\}$ is a subset of raw packets P with $f \subseteq P$, where packets are order by time $t_1 < t_2 < \cdots < t_n$, with flow duration $d_t = t_n - t_1$, and starting transmission $t = t_1$ [4], [21]. A session is a bi-directional flow including packets with the same IP addresses and ports for either source or destination [22]. Fig. 1 depicts the 5-tuple definition.

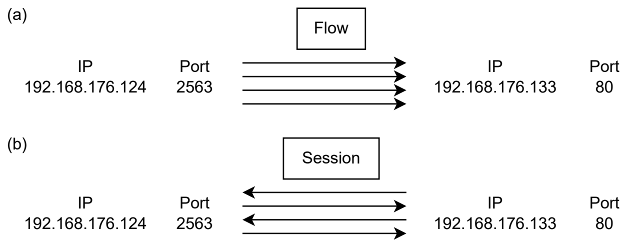

Fig. 1. Comparison of the 5-tuple definition of a flow (a) and session (b). As depicted, a flow contains only unidirectional traffic, whereas a session is a bidirectional flow.

A. Packet Transformation

Unlike Machmeier et al. [21], we transform network session into images with a fixed size. Each packet p_i of a network trace with $P = \{p_1, p_2, \ldots, p_n\}$ packets has a byte stream of $b_i = \{b_1^i, b_2^i, \ldots, b_d^i\}$ bytes that is transformed into an image patch $t_i \in \mathbb{R}^{H \times W \times C}$, where (H, W) is the height and width of the image patch, $u_i = \{b_1^i, b_2^i, \ldots, b_{H \times W}^i\}$ is the total number of bytes that are converted to an image representation, C is the number of image channels, and d is the total byte length [23]. As the length of packets varies, we have to add $0x00$ bytes until the condition $H \times W > d$ is satisfied, whereas for $d > H \times W$, we cut off packets to match the condition. In order to obtain a quadratic representation an image patch satisfies $(H, W) \in \mathbb{N}$ and $H = W$. A grayscale pixel value in the image patch represents a byte with 8 bits [21]. Next, we arrange all patches $t = \{t_1, t_2, \ldots, t_B\}$ into the final image resolution $x \in \mathbb{R}^{H \times L \times W \times L \times C}$, where (H, W) is the resolution of each image patch, L is the number of columns and rows of the final image, C is the number of channels. Like each image patch, the final image resolution satisfies the condition $(H, W, L) \in \mathbb{N}$ and $H = W$. Fig. 2 depicts the transformation process. After transforming each packet's byte stream to an array of pixel values, we append each array to the maximum column. If the number of image patches is less than $B < L \times L$, we add empty packets $p = \{0x00_1, 0x00_2, 0x00_{H \times W}\}$ and convert

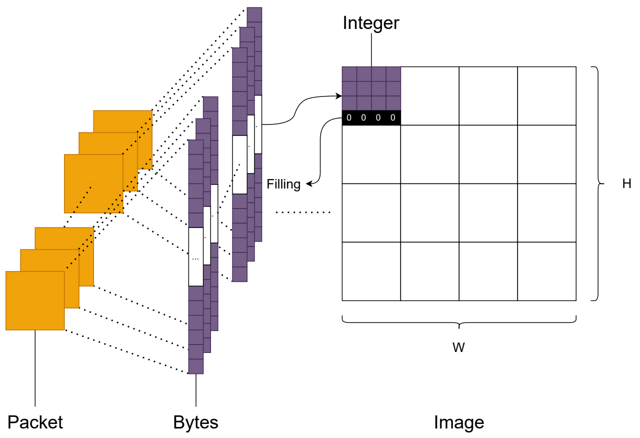

Fig. 2. Overview of the transformation process of network traces. Each packet's byte stream is transformed into an image patch and tiled along the final image. If the packet contains less than the minimum number of bytes, we pad it with black pixels. The same principle applies to the final image to enforce a quadratic representation.

them to image patches until the remaining space is filled. This pads the remaining parts of the final image with black pixels. Note that this is only necessary if a session contains less than $(H \times W)$ packets. Similar to previous studies, we randomize all IP and MAC addresses in our data sets to remove bias during training [4]–[6], [21]. For reproducible images, our network conversion is publicly available[1] [21].

B. Data set

We use the USTC-TFC2016 data set created and used by Wang et al. [4]. It consists of ten benign classes and ten malware classes. Benign refers to normal non-malicious network traffic. The majority of the malware classes use botnets. As we look into the network traces, we see that malicious classes contain UDP traffic used for peer-to-peer bot communication, whereas TCP is used for tunneling. The HTTP protocol on the application layer is primarily used to fetch suspicious data from servers outside of the network. It is important to mention that we filtered ARP, DHCP, and DHCPv6 packets of both classes to remove packets that occur in the local network of the test environment and are thus not relevant for benign and malware behavior. Since most network traces have large PCAP files with many sessions, we split all data into our predefined 5-tuple sessions. Next, we create our data set with final image size (128×128), where (8×8) packets have a dimension of $(16 \times 16) = 256$ bytes. The minimum session size is two packets; hence, we filter any session with less than two packets, losing the FaceTime class due to unidirectional UDP traffic. We use a $70/20/10$ random split of our data set for the train, validation, and test set. In order to deal with the imbalance of data, we add class weights to the loss function to penalize wrong predictions of minority classes [24].

[1]`https://github.com/stefanDeveloper/heiFIP`

C. Deep Learning Models

We rely on the well-established models VGG-19 [25] and Xception [26]. In addition, due to our new image representation consisting of patches tiled into quadratic images, we investigate the performance of a ViT [23] model for binary and multiclass classification. All models have an input size of $(128 \times 128 \times 1)$. The patch size of our ViT model overlays the packet size of (16×16) with 6 layers and 4 heads with a projection dimension of 64.

D. Explainable AI for Interpretability

Aiding the prediction with a visual explanation helps users to make deep learning models more transparent and explainable. The Gradient-weighted Class Activation Mapping (*Grad-CAM*) [13] utilizes the gradient to produce a coarse localization map on the last convolutional layer that highlights the important regions of an image for the prediction. The highlighted areas indicate a high score for classes. An advantage to previous approaches, such as CAM [27], is the application to various CNN models such as Xception, and VGG-19.

Integrated Gradients [12] attribute the prediction to its input features by adding the two fundamental axioms *Implementation Invariance* and *Sensitivity*. *Sensitivity* refers to the influence of an input feature on the prediction. If an input feature changes the outcome of a classification, the attribution should not be equal to 0. The fundamental of *Implementation Invariance* means that the attribution of input features should not depend on the design and structure of the network. Attributing input features based on their influence on the classification outcome helps users understand their model.

Inspecting the attention weights of ViT models helps users to gain insights into the prediction [14], [15]. The attention weights are computed using a self-attention mechanism, where all tokens of each image patch are appended to get a weighted sum of their representation. Visualizing the attention weights in a heat map shows the input feature with the highest influence on the prediction.

E. XAI for Malware Traffic Classification

Applying DL approaches to detect malicious activities in networks has been incorporated in Intrusion Detection Systems (IDSs) to automate the process of finding intrusions [28]. Zebin et al. [29] use SHapley Additive exPlanations (SHAP) to display the influence of features of the model's classification, whereas Andresini et al. [30] apply *Grad-CAM* as post-hoc explanations to inspect DNS over HTTPS attacks on flow features. Malware detection of executable files draws similarities to our approach. Iadarola et al. [31] use images of malware executables to detect malicious program executions of Android applications. They use *Grad-CAM* for explanation to disclose malicious programs. In contrast, the proposed framework MalConv exploits weights and gradients of the architecture to understand what the model learns from raw bytes [28]. For a detailed survey on XAI for traffic classification, we refer to Capuano et al. [28],

TABLE I

MULTICLASS CLASSIFICATION ON THE TEST SET PER CLASS.

Traffic Class Names	VGG-19			Xception			ViT		
	F1	Rec	Pr	F1	Rec	Pr	F1	Rec	Pr
BitTorrent	99.87	99.87	99.87	99.80	100	100	95.29	100	91.01
FTP	99.89	99.97	99.89	100	99.99	100	99.61	99.75	99.47
Gmail	97.95	97.66	98.24	99.65	99.88	99.42	91.81	89.02	94.78
MySQL	99.97	99.94	99.99	99.95	100	99.90	98.15	100	96.37
Outlook	98.86	98.39	99.32	99.13	98.93	99.33	96.14	98.39	93.98
Skype	98.57	98.98	98.85	99.55	99.36	99.74	98.92	99.49	98.36
SMB	92.33	99.47	86.15	94.67	99.82	90.02	79.36	100	65.78
Weibo	99.95	99.90	100	99.89	99.77	100	99.58	99.17	100
WorldOfWarcraft	99.81	99.62	100	99.75	99.62	99.87	99.68	99.37	100
Cridex	100	100	100	100	100	100	99.99	99.97	100
Geodo	99.45	99.29	99.62	99.59	99.64	99.54	97.84	98.01	97.66
Htbot	94.45	95.27	93.64	97.72	97.95	97.49	91.34	94.79	88.12
Miuref	99.51	99.40	99.62	99.47	99.40	99.55	96.15	93.70	98.74
Neris	87.26	96.95	79.34	88.23	96.02	81.61	86.11	94.44	79.13
Nsis-ay	97.13	97.38	96.88	96.81	97.91	95.73	89.36	98.95	81.47
Shifu	98.57	97.58	99.58	98.96	98.15	99.79	90.49	82.69	99.91
Tinba	99.68	99.87	99.49	99.94	99.87	100	99.87	99.87	99.87
Virut	92.85	87.75	98.57	93.41	89.26	97.96	91.78	86.69	97.50
Zeus	99.59	99.57	99.61	99.90	99.95	99.86	99.73	99.71	99.76
Accuracy	97.63			97.95			95.86		
Macro Avg. F1	97.69			98.23			94.80		

TABLE II

BINARY CLASSIFICATION ON THE TEST SET.

Traffic Class Names	VGG-19			Xception			ViT		
	F1	Rec	Pr	F1	Rec	Pr	F1	Rec	Pr
Benign	99.72	99.88	99.56	99.92	99.98	99.86	99.72	100	99.44
Malware	99.67	99.47	99.86	99.91	99.84	99.98	99.66	99.33	100
Accuracy	99.69			99.92			99.69		
Macro Avg. F1	99.69			99.92			99.69		

however, we are the first to apply XAI methods on network communication converted to images.

IV. RESULTS

We use an Nvidia GeForce RTX 3090 Ti and four Nvidia A 100 Tensor Core GPUs for training with the deep learning platform *Torch*, deep learning backend *Torch-gpu*, Cuda version 12.0, and CuDNN version 11.7. We perform random search for hyperparameter optimization that results in 0.0001 for the learning rate, cross-entropy for our loss function, a batch size of 256, and stochastic gradient descent (SGD) as the optimizer for all models.

A. Multiclass Classification

For multiclass classification, our data set consists of 19 classes, with nine benign and ten malware classes. The imbalance of classes leads back to the requirement of at least two packets per session. VGG-19 and Xception converge at similar performances, whereas our ViT model results in significantly lower accuracy, F1, recall, and precision scores for all classes, Table I. Xception achieves a maximum accuracy of 97.95% with a macro average F1 score of 98.23%. The average accuracy for our VGG-19 and ViT models is lower, with values ranging between 95.86% and 97.69%. We observe high misclassifications for the malware classes Neris and Virut on all three models, while for all other classes, we notice few misclassifications. For Neris, the F1 score ranges between 86.11% to 88.23%, whereas Virut achieves a minimum F1 score of 91.78%. Inspecting the network traces of Virut and Neris shows that both malware classes request similar HTTP resources to fetch scripts for execution, request the same DNS servers, and contain mail- and ad-related traffic. Comparing the network traffic reflects our assumption that Neris and Virut are similar in behavior.

B. Binary Classification

The binary classification on our test set shows overall good classification performance for the Xception model with an accuracy of 99.92% and macro average F1 score of 99.92%, Table II. The VGG-19 and ViT model achieve a similar accuracy of 99.69%, whereas VGG-19 yields a slightly better F1 score for benign and malware classes. Xception achieves overall the best results with the highest accuracy and F1 score. We notice only a few misclassifications on benign and malware classes that could lead back to common network traffic, such as normal TCP handshakes.

C. Grad-CAM

We randomly choose a malware sample from the Virut and Htbot class and compute the *Grad-CAM* of VGG-19 and Xception to visualize the important packets of the session, Fig. 3. Both sessions consist of more than 64 packets, fetch a malware executable, and communicate with its C&C server outside of the network to receive the next commands. For the Htbot sample in Fig. 3(a) and Fig. 3(c), the highlighted areas in red of the heat map indicates that the subset of packets $p_j = \{p_{21}, \ldots, p_{23}, p_{28} \ldots, p_{30}, p_{35}, \ldots, p_{37}\}$ are of interest for the prediction. While inspecting the packets of the session, we can see that most of the highlighted packets (p_{21} - p_{23}, p_{28} - p_{30}) are suspicious HTTP POST requests to an IP address outside of the network whereas p_{35} - p_{37} are non-suspicious TCP connections. Nonetheless, both models correctly predicted the Htbot class as either malware or Htbot itself. This indicates strong relation between the content of packets transformed to the new byte representation and the actual model. For the Virut sample in Fig. 3(b) and Fig. 3(d), the highlighted areas in red show an offset of important packets between the two models. For Xception, we notice a similar subset of packets $p_j = \{p_{20}, p_{21}, p_{27} \ldots, p_{30}, p_{36}, p_{37}\}$, whereas for VGG-19 the subset of packets $p_j = \{p_{25}, p_{26}, p_{33} \ldots, p_{34}, p_{41}, p_{43}\}$ is shifted. However, we notice that all highlighted packets of the two models are requests to fetch data from servers outside of the network, indicating a malicious activity, thus, having a higher influence on the prediction. Hence, using the *Grad-CAM* to aid the explainability of a deep learning model gives promising results in the context of network sessions transformed into images.

D. Integrated Gradients

Fig. 4 shows the *Integrated Gradients* for the same malware samples of Virut and Htbot as in section IV-C for multiclass classification. For the Htbot sample in Fig. 4(a) and Fig. 4(c), we notice a difference of input feature attribution that influence the prediction of the model. For Xception, the area is greater and includes more packets, whereas, for VGG-19, only a small subset of packets is important for the prediction. Overlaying the *Integrated Gradients* from

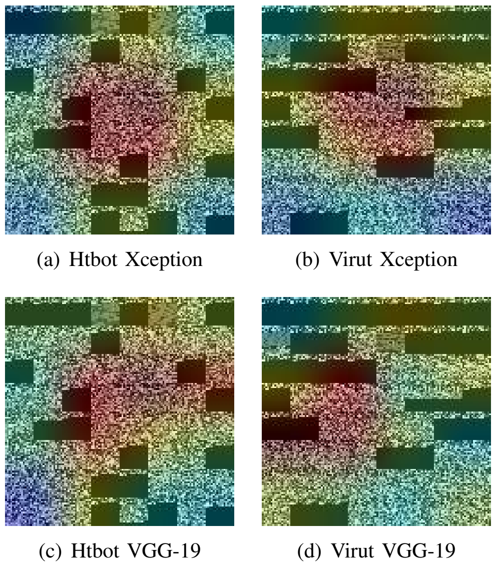

Fig. 3. Grad-CAM of Virut and Htbot malware for Xception (3(a),3(b)) and VGG-19 (3(c),3(d)). The color red indicates a strong influence on the prediction, whereas the color blue has little effect on the prediction. As depicted, the malicious packets of the session that send HTTP POST requests to a server outside of the network are highlighted.

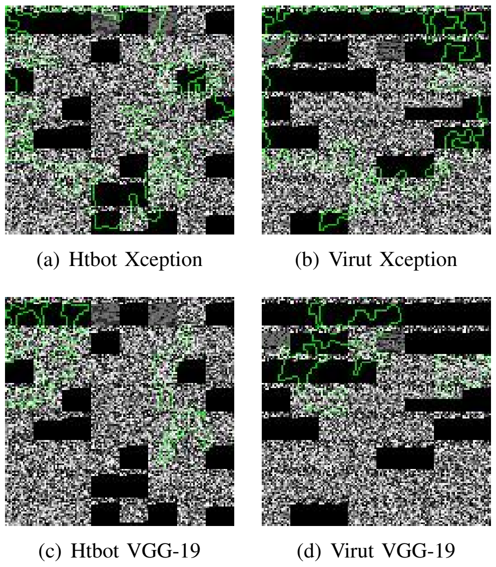

Fig. 4. Integrated gradients of Virut and Htbot malware for Xception (4(a),4(b)) and VGG-19 (4(c),4(d)). The highlighted area in green displays the pixel with the highest influence on the gradient. As depicted, the malicious packets of the session of Xception are similar highlighted as in Fig. 3, whereas VGG-19 attributes non-suspicious TCP handshakes that occur in both classes.

Fig. 4 with the *Grad-CAM* from Fig. 3 shows differences in the input feature importance. For Xception, we identify a similar subset of packets fetching malicious data from servers outside the network. However, investigating the marked area of VGG-19, we see that the first three non-suspicious TCP connections receive a high attribution. For the Virut sample in Fig. 4(b) and Fig. 4(d), we identify a similar behavior for Xception and VGG-19. While Xception attributes a similar subset of packets that fetches malicious data, VGG-19 attributes non-suspicious TCP handshakes that occur in benign and malware traffic interchangeably. Viewing these results could explain the accuracy differences of multiclass classification between the Xception and VGG-19. Thus, showing that *Integrated Gradients* helps to understand the outcome of deep learning models in context network security.

E. Attention heat maps for Vision Transformer

Fig. 4 shows the attention heat maps for the same malware samples of Virut and Htbot as in section IV-C for our ViT model for multiclass classification. For Htbot in Fig. 5(a), we notice a subset of packets $p_j = \{p_{20}, \dots, p_{23}, p_{36} \dots, p_{38}, p_{41} \dots, p_{48}\}$ that fetches malicious data from servers outside of the network, and overlays our previous findings from Xception and VGG-19. We observe that non-suspicious TCP handshakes are not highlighted in yellow in the heat map, Fig. 5(a) and Fig. 5(b). While malware traffic differs between Htbot and Virut, the highlighted attention weights in yellow of Fig. 5 look similar; however, comparing the weights reveals that both heat maps are unequal. Overall, our ViT model correctly predicts both samples as Htbot and Virut classes. Thus, using attention heat maps to get insights into a ViT model is a helpful method to understand its outcome.

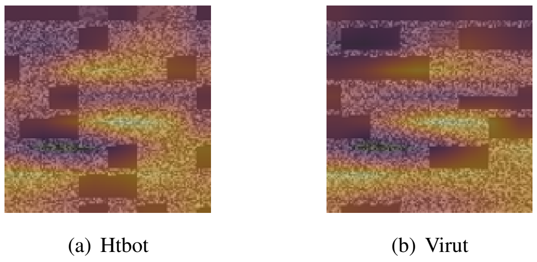

Fig. 5. Attention visualization of the last head of the Vision Transformer model of Virut and Htbot malware sample. The last layer displays the weighted sum of the representation. The highlighted area in yellow displays the pixel with the highest influence on the attention weights.

V. Summary and Discussion

Securing networking infrastructures is a challenging task with newly advancing threats every day. This study showed that inheriting deep learning methods for traffic classification can improve network security by detecting malicious behaviors in various packet streams. In particular, we proposed a new network traffic image representation by converting network sessions into images. Unlike trimming a session to a certain byte length, we created image patches for each packet in a session and tiled them to a quadratic representation. This novel approach helped models train on bi-directional network traffic. We evaluated the performance of the new representation on the well-established deep learning models VGG-19 and Xception. In addition, we conducted an experimental study of ViT models on network images due to their ability to process images in patches. Overall, our best model Xception achieved an accuracy of 99.92% for binary classification and 97.95% for multiclass classification.

Our experimental ViT model achieved a significantly lower performance with 95.86% for multiclass and 99.36% for binary classification. In addition, we applied the XAI techniques *Integrated Gradients* and *Grad-CAM* to interpret the VGG-19 and Xception model to understand the relationship between input features and prediction. For our ViT model, we visualized the attention heat maps. We see that our models can understand the impact of pixel to bytes, and they start to detect information in packets, such as malicious HTTP POST requests or communications to C&C servers. Overall, our approach shows strong classification performance; thus, this study contributes to the network traffic image classification field.

Regarding images, currently, we only use a single channel to represent packets, resulting in a grayscale image; however, recent studies used a three-channel representation. Increasing channels on images allow for storing more bytes and, thus, more information that could help to distinguish benign from malicious traffic. Future research could investigate the impact of input features on the prediction by reverse engineering the highlighted bytes of the XAI method's output.

In summary, deep learning methods on network traffic transformed into images is a promising approach for supporting network infrastructures to detect malicious behavior ahead of time, including XAI methods to interpret results, which could be advantageous to increase network security.

References

[1] X. Cao, Q. Luo, and P. Wu, "Filter-GAN: Imbalanced Malicious Traffic Classification Based on Generative Adversarial Networks with Filter," *Mathematics*, vol. 10, no. 19, p. 3482, Sep. 2022.

[2] S. Kim and A. Reddy, "Modeling network traffic as images," in *IEEE International Conference on Communications, 2005. ICC 2005. 2005*, vol. 1, Seoul, Korea (South), 2005, pp. 168–172 vol. 1.

[3] S. S. Kim and A. Reddy, "A study of analyzing network traffic as images in real-time," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 3, Miami, FL, USA, 2005, pp. 2056–2067 vol. 3.

[4] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking (ICOIN)*, Da Nang, Vietnam, 2017, pp. 712–717.

[5] H.-K. Lim, J.-B. Kim, J.-S. Heo, K. Kim, Y.-G. Hong, and Y.-H. Han, "Packet-based Network Traffic Classification Using Deep Learning," in *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, Okinawa, Japan, 2019, pp. 046–051.

[6] H.-K. Lim, J.-B. Kim, K. Kim, Y.-G. Hong, and Y.-H. Han, "Payload-Based Traffic Classification Using Multi-Layer LSTM in Software Defined Networks," *Applied Sciences*, vol. 9, no. 12, p. 2550, 2019.

[7] F. Ullah, S. Ullah, M. R. Naeem, L. Mostarda, S. Rho, and X. Cheng, "Cyber-Threat Detection System Using a Hybrid Approach of Transfer Learning and Multi-Model Image Representation," *Sensors*, vol. 22, no. 15, p. 5883, 2022.

[8] P. Xu, C. Eckert, and A. Zarras, "Falcon: Malware Detection and Categorization with Network Traffic Images," in *ICANN - The International Conference on Artificial Neural Networks*, Bratislava, Slovakai, 2021, pp. 14–17.

[9] R. Banu, T. Jyothi, M. Amulya, K. Anju, A. Raju, and S. N. Kashyap, "MONOSEK – A Network Packet Processing System for Analysis & Detection of TCP Xmas attack using Pattern Analysis," in *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, Madurai, India, May 2019, pp. 952–956.

[10] L. Deri, M. Martinelli, T. Bujlow, and A. Cardigliano, "nDPI: Open-source high-speed deep packet inspection," in *2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*, Nicosia, Cyprus, Aug. 2014, pp. 617–622.

[11] V. Carela-Español, T. Bujlow, and P. Barlet-Ros, "Is Our Ground-Truth for Traffic Classification Reliable?" in *Proceedings of the 15th International Conference on Passive and Active Measurement - Volume 8362*, ser. PAM 2014. Berlin, Heidelberg: Springer-Verlag, Mar. 2014, pp. 98–108.

[12] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic Attribution for Deep Networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17. JMLR.org, Jun. 2017, p. 3319–3328.

[13] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017, pp. 618–626.

[14] H. Chefer, S. Gur, and L. Wolf, "Transformer interpretability beyond attention visualization," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 2021, pp. 782–791.

[15] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy, "Do Vision Transformers See Like Convolutional Neural Networks?" Mar. 2022, arXiv:2108.08810 [cs, stat].

[16] Stratosphere, "Stratosphere Laboratory Datasets," 2015.

[17] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization:," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy*. Funchal, Madeira, Portugal: SCITEPRESS - Science and Technology Publications, 2018, pp. 108–116.

[18] G. R. Ruth, N. Brownlee, and C. G. Mills, "Traffic Flow Measurement: Architecture," Internet Engineering Task Force, Request for Comments RFC 2722, Oct. 1999.

[19] B. E. Carpenter, S. E. Deering, J. Rajahalme, and A. Conta, "IPv6 Flow Label Specification," Internet Engineering Task Force, Request for Comments RFC 3697, Mar. 2004.

[20] T. Zseby, B. Claise, J. Quittek, and S. Zander, "Requirements for IP Flow Information Export (IPFIX)," Internet Engineering Task Force, Request for Comments RFC 3917, Oct. 2004.

[21] S. Machmeier, M. Trageser, M. Buchwald, and V. Heuveline, "A generalizable approach for network flow image representation for deep learning," in *2023 7th Cyber Security in Networking Conference (CSNet)*, Montréal, Canada, 2023.

[22] J. Zhao, X. Jing, Z. Yan, and W. Pedrycz, "Network traffic classification for data fusion: A survey," *Information Fusion*, vol. 72, pp. 22–47, Aug. 2021.

[23] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," Jun. 2021.

[24] Z. Xu, C. Dan, J. Khim, and P. Ravikumar, "Class-Weighted Classification: Trade-offs and Robust Approaches," May 2020.

[25] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Apr. 2015.

[26] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," Apr. 2017.

[27] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Is object localization for free? - Weakly-supervised learning with convolutional neural networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 685–694, iSSN: 1063-6919.

[28] N. Capuano, G. Fenza, V. Loia, and C. Stanzione, "Explainable Artificial Intelligence in CyberSecurity: A Survey," *IEEE Access*, vol. 10, pp. 93 575–93 600, 2022.

[29] T. Zebin, S. Rezvy, and Y. Luo, "An explainable ai-based intrusion detection system for dns over https (doh) attacks," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2339–2349, 2022.

[30] G. Andresini, A. Appice, F. P. Caforio, D. Malerba, and G. Vessio, "ROULETTE: A neural attention multi-output model for explainable Network Intrusion Detection," *Expert Systems with Applications*, vol. 201, p. 117144, Sep. 2022.

[31] G. Iadarola, F. Martinelli, F. Mercaldo, and A. Santone, "Towards an interpretable deep learning model for mobile malware detection and family identification," *Computers & Security*, vol. 105, p. 102198, Jun. 2021.