UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus Bauru

Paulo Roberto Galego Hernandes Junior

Application of Explainable Artificial Intelligence (*XAI*) to Detect Anomalies in Computer Networks through the Creation of Images from Data Packets

Bauru
2025

Paulo Roberto Galego Hernandes Junior

Application of Explainable Artificial Intelligence (*XAI*) to Detect Anomalies in Computer Networks through the Creation of Images from Data Packets

Tese apresentada como parte dos requisitos para obtenção do título de Doutor em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de Bauru.

Orientador: Prof. Dr. Kelton Augusto Pontara da Costa

Bauru
2025

H557a

Hernandes Junior, Paulo Roberto Galego
Application of Explainable Artificial Intelligence (XAI) to Detect Anomalies in Computer Networks through the Creation of Images from Data Packets / Paulo Roberto Galego Hernandes Junior. -- Bauru, 2025
81 p. : il., tabs.

Tese (doutorado) - Universidade Estadual Paulista (UNESP), Faculdade de Ciências, Bauru
Orientador: Kelton Augusto Pontara da Costa

1. Network Anomalies. 2. Deep Learning. 3. Grad-CAM. 4. Explainability. 5. Convolutional Neural Networks. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Dados fornecidos pelo autor(a).

ATA DA DEFESA PÚBLICA DA TESE DE DOUTORADO DE PAULO ROBERTO GALEGO HERNANDES JUNIOR, DISCENTE DO PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO, DA FACULDADE DE CIÊNCIAS - CÂMPUS DE BAURU.

Aos 26 de novembro de 2025, às 14h30min, por meio de Videoconferência, realizou-se a defesa de TESE DE DOUTORADO de PAULO ROBERTO GALEGO HERNANDES JUNIOR, intitulada **Application of Explainable Artificial Intelligence (XAI) to Detect Anomalies in Computer Networks through the Creation of Images from Data Packets**. A Comissão Examinadora foi constituída pelos seguintes membros: Professor Doutor KELTON AUGUSTO PONTARA DA COSTA (Orientador(a) - Participação Virtual) do(a) Departamento de Computação / UNESP / Câmpus de Bauru - FC, Prof. Dr. ALEX MARINO GONÇALVES DE ALMEIDA (Participação Virtual) do(a) Computação / Faculdade de Tecnologia de Ourinhos, Prof. Dr. BRUNO BOGAZ ZARPELÃO (Participação Virtual) do(a) Computação / Universidade Estadual de Londrina, Prof. Dr. LEANDRO APARECIDO PASSOS JUNIOR (Participação Virtual) do(a) Departamento de Computação / UNESP / Câmpus de Bauru - FC, Prof. Dr. LUCAS CORREIA RIBAS (Participação Virtual) do(a) Departamento de Ciências de Computação e Estatística / UNESP / Câmpus de São José do Rio Preto - Ibilce, Após a exposição pelo doutorando e arguição pelos membros da Comissão Examinadora que participaram do ato, de forma presencial e/ou virtual, o discente recebeu o conceito final **APROVADO** . Nada mais havendo, foi lavrada a presente ata, que após lida e aprovada, foi assinada pelo(a) Presidente(a) da Comissão Examinadora.

Professor Doutor KELTON AUGUSTO PONTARA DA COSTA

Paulo Roberto Galego Hernandes Junior

Application of Explainable Artificial Intelligence (*XAI*) to Detect Anomalies in Computer Networks through the Creation of Images from Data Packets

Tese apresentada como parte dos requisitos para obtenção do título de Doutor em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de Bauru.

Comissão Examinadora

Prof. Dr. Kelton Augusto Pontara da Costa
UNESP - Câmpus de Bauru
Orientador

Prof. Dr. Alex Marino Gonçalves de Almeida
Faculdade de Tecnologia de Ourinhos

Prof. Dr. Bruno Bogaz Zarpelão
Universidade Estadual de Londrina

Prof. Dr. Leandro Aparecido Passos Junior
UNESP - Câmpus de São José do Rio Preto

Prof. Dr. Lucas Correia Ribas
UNESP - Campus de São José do Rio Preto

Bauru
26 de novembro de 2025

Ao meu pai, que ainda na década de 1990, com o jornal nas mãos, me explicou como se tornar um doutor. Se ele estivesse aqui, estaria orgulhoso dessa conquista.
To my father, who, back in the 1990s, with a newspaper in his hands, explained to me how to become a doctor. If he were here, he would be proud of this achievement.

Acknowledgements

Aos meus filhos, Davi e Antony, e minha esposa Amanda, foi por eles que tive forças em todas dificuldades. À minha mãe e meu irmão, que sempre me apoiaram. Aos amigos da Fatec pelas palavras de incentivo. Ao meu orientador, prof. Kelton, pela paciência, apoio e pelas valorosas recomendações ao longo desta jornada. A todos que direta e indiretamente colaboraram para esta conquista.

To my kids Davi and Antony, and my wife Amanda, it was because of them that I had the strength to overcome all difficulties. To my mother and my brother, who have always supported me. To my friends at Fatec for their words of encouragement. To my advisor, Professor Kelton, for his patience, support, and valuable recommendations throughout this journey. To everyone who directly and indirectly contributed to this achievement.

"We stand on the threshold of a brave new world. Our AI systems must do what we want them to do, for the benefit of humanity."

Stephen Hawking[1]

"I think we're moving into a period when for the first time ever we may have things more intelligent than us."

Geoffrey Hinton[2]

[1] Opening remarks at the Leverhulme Centre for the Future of Intelligence, 2016.

[2] *60 Minutes*, CBS News, 2024.

Abstract

With the increasing complexity of cyber threats, understanding how artificial intelligence models make decisions has become a crucial step in improving trust and transparency. This work presents a methodology that transforms network traffic into images and uses Convolutional Neural Networks (CNNs) to classify flows as benign or malicious. Beyond achieving high predictive accuracy, the study integrates the Explainable AI technique Grad-CAM to visually explain the internal workings of the model. This approach allows analysts to inspect which parts of the image were most influential in the model's decision, facilitating human oversight and system auditing. The findings highlight the potential of combining deep learning with interpretability tools in the field of cybersecurity.

Keywords: Network Anomalies. Deep Learning. Grad-CAM. Explainability. Convolutional Neural Networks.

Resumo

Com a crescente complexidade das ameaças cibernéticas, compreender como os modelos de inteligência artificial tomam decisões tornou-se um passo crucial para melhorar a confiança e a transparência. Este trabalho apresenta uma metodologia que transforma o tráfego de rede em imagens e utiliza Redes Neurais Convolucionais (CNNs) para classificar fluxos como benignos ou maliciosos. Além de alcançar alta precisão preditiva, o estudo integra a técnica de IA Explicável Grad-CAM para explicar visualmente o funcionamento interno do modelo. Essa abordagem permite que os analistas inspecionem quais partes da imagem foram mais influentes na decisão do modelo, facilitando a supervisão humana e a auditoria do sistema. Os resultados destacam o potencial da combinação de aprendizado profundo com ferramentas de interpretabilidade no campo da segurança cibernética.

Palavras-chave: Anomalias de rede. Aprendizado profundo. Grad-CAM. Explicabilidade. Redes neurais convolucionais.

List of Figures

List of Tables

List of abbreviations and acronyms

AI	Artificial Intelligence
API	Application Programming Interface
AUC	Area Under the Curve
BEN	Benign
CAM	Class Activation Mapping
CAPEC	Common Attack Pattern Enumeration and Classification
CNN	Convolutional Neural Network
CSV	Comma-Separated Values
DL	Deep Learning
DNS	Domain Name System
ECE	Expected Calibration Error
EER	Equal Error Rate
EN	English
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GAN	Generative Adversarial Network
GPU	Graphics Processing Unit
Grad-CAM	Gradient-weighted Class Activation Mapping
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
IP	Internet Protocol
KNN	K-Nearest Neighbors

ML	Machine Learning
NB	Naive Bayes
NIDS	Network Intrusion Detection System
NLP	Natural Language Processing
PCA	Principal Component Analysis
PDF	Portable Document Format
ReLU	Rectified Linear Unit
RF	Random Forest
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TP	True Positive
TN	True Negative
TPR	True Positive Rate
UDP	User Datagram Protocol
URL	Uniform Resource Locator
XAI	Explainable Artificial Intelligence
XGBoost	Extreme Gradient Boosting

Contents

1 Introduction

The exponential growth of the Internet each year continues to transform how individuals interact with the world. It has produced an increasingly interconnected lifestyle in which information flows seamlessly everywhere and at all times. People generate digital information continuously through their smart devices and social networks. When processed and analyzed by Artificial Intelligence (AI), this scenario creates an extensive data repository, enabling algorithms to predict personalized recommendations and behaviors (KAUR; GABRIJELčIč; KLOBUčAR, 2023).

However, this massive volume of stored and processed data also amplifies critical cybersecurity challenges. Detecting attacks, especially in large organizations, has become a priority to maintain the integrity, confidentiality, and availability of systems and information. According to ITPro (2025), in 2026 Gartner estimates that global spending on cybersecurity will increase by 12%, totaling $240 billion. To address this growing threat landscape, AI-based solutions such as machine learning have become central to modern defense strategies (SALEM et al., 2024).

Within this context, Machine Learning (ML) plays an important role in detecting attacks on computer networks. The ability to analyze large volumes of data and identify patterns in (near) real time is critical for effectively recognizing malicious activity and adapting to evolving threats (OZKAN-OKAY et al., 2024). Among the different ML approaches, representation learning techniques have emerged as a promising way to capture complex structures present in network traffic.

One line of research that has shown promising results is the conversion of network data into images during the preprocessing phase. This strategy enables the use of powerful image-based models and has been successfully explored in works addressing anomaly detection in networks, such as Wu, Chen e Li (2018) and Wang et al. (2017). By representing network flows as images, these approaches allow spatial patterns and correlations between features to be exploited by computer vision models.

Several techniques have been employed in creating Network Anomaly Detection Systems, always seeking high detection rates with the lowest possible number of false positives. In this landscape, image-based approaches represent an alternative to traditional feature engineering and sequence-based models, providing a different perspective on how network behavior can be modeled.

Among these techniques, Chen et al. (2017) developed a method called *Seq2Img*, which transforms IP traffic into images and classifies application traffic (such as social networks)

using Convolutional Neural Networks. Their method achieved superior performance compared to other approaches, illustrating the potential of image representations for network traffic classification and motivating further exploration of similar ideas in security-oriented scenarios.

Neural Networks (NN) in general have demonstrated strong capability in processing complex data and identifying subtle patterns, making them ideal candidates for dealing with the increasing sophistication and adaptability of cyber attacks. Within this family of models, one of the most interesting and widely adopted architectures for image-like data is the Convolutional Neural Network (CNN).

A Convolutional Neural Network (CNN) is a type of Neural Network architecture specifically designed to process data with an inherent spatial structure, such as images. CNNs comprise convolutional layers that apply filters to extract relevant features from the input, followed by pooling layers to reduce dimensionality. These learned features are then used for tasks such as classification, object detection, and visual pattern recognition. When network traffic is transformed into images, CNNs can be employed to learn visual patterns that correspond to normal and anomalous behaviors.

Nonetheless, as AI and cybersecurity become more intertwined, it is not sufficient to focus solely on detection performance. It is also essential to address the interpretability and transparency of the results, especially in security-sensitive applications. Neural Networks often behave as "black boxes", since their internal decision process is difficult to understand and explain, which complicates the validation and trustworthiness of automated decisions (GUIDOTTI et al., 2018).

In this context, Explainable Artificial Intelligence (XAI) has gained increasing importance, as it seeks to provide clear insights and understandable interpretations of AI models (RIBEIRO; SINGH; GUESTRIN, 2016). XAI techniques allow analysts to inspect which features or regions of the input most influenced a given prediction, thereby bridging the gap between high-performance models and the need for accountability and forensic analysis in cybersecurity.

Gradient-weighted Class Activation Mapping (Grad-CAM) is an XAI technique designed to visualize which regions of an input image most influenced a CNN's decision. In practice, it computes the gradients of the predicted class with respect to the last convolutional layer and uses them to generate a heatmap that highlights the most relevant areas for that prediction. When network traffic is transformed into images, Grad-CAM allows us to see which parts of the encoded packet the model is "looking at" when it classifies a flow as benign or malicious. This makes the model's behavior more transparent, helps validate whether it is focusing on meaningful patterns rather than noise, and provides security analysts with visual evidence that supports both anomaly detection and forensic interpretation.

This thesis addresses the convergence between network attack detection, the transformation of network data into images, and the use of Convolutional Neural Networks combined

with Explainable Artificial Intelligence. Its main and unprecedented contribution lies in uniting XAI with the image-based representation of network packets, enabling both effective anomaly detection and visual interpretability of the model's decisions. The core principles of AI and how CNNs can be applied to detect attacks will be discussed, followed by an analysis of emerging strategies for Explainable Artificial Intelligence in this domain.

Therefore, the objective of this work is to detect anomalies in networks by transforming data packets into images and utilizing XAI techniques to understand the decisions made by the proposed model. More specifically, the general objective is to develop an anomaly detection model that converts network data into images in a way that allows detecting anomalous behaviors through CNNs, while also highlighting the reasons for classifying packets as anomalous or benign using an XAI method.

The specific objectives of this thesis are the following:

- Convert data acquired from open *datasets* into images suitable for CNN processing;
- Submit the images to a CNN and evaluate its accuracy in detecting anomalies; and
- Apply XAI techniques to understand which attributes are truly relevant for anomaly detection.

The central hypothesis of this thesis is that transforming numerical network data into images can enhance anomaly detection in networks through the use of Convolutional Neural Networks. Furthermore, the application of Explainable Artificial Intelligence is expected to bring transparency to the decisions made by the model, making its behavior more understandable and trustworthy for security analysts.

In summary, this thesis emphasizes the advantages of a new approach to cybersecurity by analyzing the interactions between anomaly detection, CNNs, and Explainable Artificial Intelligence. In addition to improving detection capability, it underscores the importance of transparency and interpretability in automated decision-making, thereby supporting the development of more secure and trustworthy networked systems.

This doctoral thesis will consist of this Introduction, followed by Chapter 2, which outlines the works used as references in the application of the techniques employed, including a Systematic Literature Review. Chapter 3 presents the methodology used in the model, as well as the datasets used, the architecture of the proposal, and the proposed scenarios. Chapter 4 documents the use of the presented methodology and the results obtained in each scenario. In Chapter 5, the conclusion is presented.

2 Related Works

This chapter presents the published works related to the research topic, highlighting papers that utilize Convolutional Neural Networks and those that create images from data packets in computer networks. Some fundamental concepts will also be presented to help understand the ideas used in the elaboration of this thesis.

2.1 Background

This section presents examples of successful applications of AI and CNNs in various areas, highlighting the theoretical foundations and significant contributions to technological development.

2.1.1 Artificial Intelligence

For Sternberg (2010), intelligence is the ability to learn from experience, using metacognitive processes to enhance learning, and the ability to adapt to the environment that surrounds us. Metacognition refers to an individual's understanding and control of their own thought processes.

Artificial Intelligence (AI) emerged in the mid-1950s, according to Medeiros (2018), and there was great expectation that, with the advancement of computer technology, machines would reach the same level of human intelligence in a short time.

According to Norvig e Russell (2014), AI not only tries to understand but also to produce intelligent entities. For the author, AI currently encompasses a wide variety of subfields, ranging from the general (learning and perception) to specific tasks, such as playing chess, driving a car on a busy road, and diagnosing diseases, demonstrating its relevance to any intellectual task and highlighting it as a universal field.

Mondal (2020) presents the relationship between Artificial Intelligence, Machine Learning (ML), and Deep Learning (DL), bringing ML as a system built to learn from historical data or an environment. DL is an emerging part of ML, where a neural network is trained on enormous historical data to perform a specific task.

An Artificial Neural Network (ANN) is a massively distributed parallel processor composed of simple processing units with a natural propensity to store knowledge from experiences and make it available. It resembles the brain in two characteristics: (i) the network from its environment acquires knowledge through the learning process, and (ii) interneuron connection

strengths, known as synaptic weights, which are used to store the acquired knowledge (HAYKIN, 2010).

Artificial Neural Networks, according to Abiodun et al. (2018), are systems produced to work the same way the human brain would work with a given task. One of the areas where ANN is widely used is in detecting anomalies in computer networks.

Mohammed e Ali (2014) used a Packet Feature Extractor (PFE) to transform an offline Intrusion Detection System (IDS) into an online one. An online IDS deals with real-time networks, capturing packets, and extracting the needed characteristics. An offline system handles recorded data, representing the extracted features. The proposed work used 12 characteristics of the NSL-KDD database to obtain good results and excellent performance.

Choi et al. (2015) proposes reconstructing malware from network packets to analyze them. The transmitted file reconstruction technique is based on file type signature and network protocol analysis, which can reconstruct various file types from network packets.

Carriquiry et al. (2019) discusses pattern recognition using machine learning, providing practical examples from criminal forensics and demonstrating the improvement in final results achieved when ML techniques are employed. The work presents examples of the use of ML in recognizing projectile patterns, improving the results of false positives and negatives.

A comprehensive survey was conducted by Oladipo et al. (2020), presenting research in the area of computer forensics. It showed that *Deep Learning* techniques, using Convolutional Neural Networks (CNN), are obtaining the best results in classifying images to detect fraudulent ones.

Sikos (2020) presents research on analyzing network packets in forensic investigations, demonstrating methods for acquiring data packets. The study also suggests methods of processing packets, such as Deep Packet Inspection (DPI) or the use of AI, exemplifying tools that can be used.

2.1.2 Convolutional Neural Networks

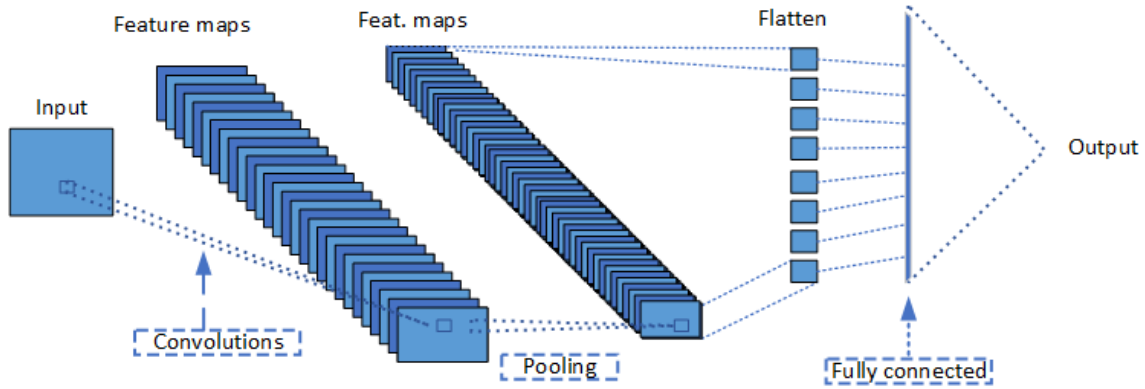
Computer Vision (CV) aims to understand how human beings interpret information through their vision. It is a biological fact that the human visual system perceives light through beams captured by the retina, which decodes this information, transforms it into nerve impulses, carries them through neurons, and interprets them in the brain. One of the most exciting characteristics of our vision is the hierarchical learning of information, because as a person sees a specific object, they try to decompose it into small blocks of characteristics to facilitate its understanding. Such an analogy is employed in a CV algorithm known as Convolutional Neural Networks (CNN) (LECUN; KAVUKCUOGLU; FARABET, 2010).

A CNN, presented by LeCun et al. (1998), is a class of artificial neural networks inspired

by the organization of neurons in the visual cortex and their connectivity patterns. That is, how the brain of living beings processes visual information. A CNN is a multi-layer architecture comprising input, hidden, and output layers. Notably, at least one of the hidden layers must be a convolutional layer.

Figure 1 presents an architecture example composed of an input layer, a convolutional layer, two pool layers, a flatten layer, a fully connected layer, and, finally, the output layer. In a CNN, feature maps are intermediate representations of the input data obtained after applying convolutional filters to the input images. Each feature map captures specific visual patterns or features present in the input.

Figure 1 – Architecture of a Convolutional Neural Network



Source: from the author (2025)

The convolution layer captures local features, such as edges, curves, colors, lines, and other image details. Thus, the number of local features learned by the convolutional layer is proportional to the number of filters.

The flatten layer converts the multidimensional feature maps produced by the previous layers into a one-dimensional vector. This flattened vector can then be fed into a fully connected layer, typically used for making predictions or performing classification.

A pooling layer is a type of layer commonly used to downsample the spatial dimensions of feature maps generated by convolutional layers. The primary purpose of the pooling layer is to reduce the spatial size of the input and extract the most significant features while maintaining their essential properties.

The fully connected layer turns a list of inputs into a list of outputs. The transformation is called fully connected, as any input value can affect any output value. These layers will have many parameters that can be learned even for relatively small inputs, but they have the advantage of not assuming any structure in the inputs. (RAMSUNDAR, 2018)

2.1.3 Explainable Artificial Intelligence (XAI)

With recent advances in Artificial Intelligence, many sectors are leveraging the benefits that can be gained. With the growing complexity of AI algorithms, there is a rising concern about the lack of transparency and interpretability. Traditional AI systems, such as CNNs, often operate as unclear models, making it challenging for humans to understand the underlying reasoning behind their outputs. This need for more explainability restricts their acceptance, trustworthiness, and effective management in real-world scenarios (RIBEIRO; SINGH; GUESTRIN, 2016).

The primary objective of Explainable Artificial Intelligence (which will be referred to as *X*AI) is to develop a collection of novel or adapted machine-learning techniques that generate explainable models. When coupled with effective explanation methodologies, these models empower end-users to comprehend, appropriately trust, and efficiently manage the evolving generation of AI systems. *X*AI specifically aims to cater to end users who rely on the decisions, recommendations, or actions generated by an AI system and, therefore, require a clear understanding of the system's underlying reasoning (GUNNING, 2017).

2.1.3.1 Grad-CAM

Gradient-weighted Class Activation Mapping (Grad-CAM), presented by Selvaraju et al. (2017), uses the gradient of a target class with respect to the final convolutional feature maps to compute channel weights and produce a class-discriminative localization heatmap. Because it only requires gradients and a convolutional layer, it applies broadly across CNN architectures and tasks (classification, captioning) without retraining.

According to (WANG; ZHANG, 2023), in studies, Grad-CAM has helped users better understand areas of use where this type of visualization is helpful: object localization and image classification in general, medical diagnosis (to inform clinical predictions), autonomous driving (to understand environmental perception), and video analysis (to identify relevant frames/segments). In short, Grad-CAM serves as a practical bridge between CNN performance and the transparency required in real-world scenarios by making visible "where" the model looked to make decisions.

Saliency maps can look plausible yet be unfaithful; sanity checks that randomize model weights or labels reveal that some methods can be insensitive to the learned parameters. Grad-CAM's reliance on the last convolutional layer also means that maps can be coarse when the spatial resolution is low (ADEBAYO et al., 2018).

2.1.4 Converting text to images

CNNs are among the most effective learning algorithms for understanding image content and performing tasks related to image segmentation, classification, detection, and retrieval (KHAN et al., 2020).

In their work, Wang et al. (2017) converted traffic into images to preprocess the raw network traffic (PCAP file) into input data for a CNN. The work performs malware detection through a method that does not require manually designing traffic resources. The raw traffic data is directly input to the traffic classifier, which can automatically learn features.

In the work of Chen et al. (2017), network traffic was classified by creating images without using the packet's payload, instead relying on other data such as size, direction, and time between packet arrivals. Two *datasets* were used: the first with traffic from the FTP, HTTP, SSH, TFTP, and TLSV protocols, and the second with traffic from five applications, namely Instagram, Skype, Facebook, WeChat, and YouTube. In the first dataset, the identification method outperformed those of Support Vector Machine (SVM), Multi-Layer Perception (MLP), Naïve Bayes Classifier (NB), and Decision Tree (DT), although not by a considerable margin. However, in the second set of data, the identification and classification of network traffic using images was superior to the previously mentioned methods.

Wu, Chen e Li (2018) also converted network data into images in the preprocessing phase of the NSL-KDD dataset. Detecting anomalies in the analyzed data in real time could prove that the image format can reduce the computational parameters used.

Liu (2021) used a neural network to process the images created from the USTC-TFC2016 dataset. In their work, authors used only packages with a fixed size of 784 bytes, which were converted into figures with a size of 28x28. The algorithm, based on Mobilenet [3], reduces the number of calculations and model parameters, making the proposed model lightweight. The authors obtained excellent results compared with other classical networks, highlighting the low time used for training.

Kaur, Lashkari e Rahali (2020) utilized two well-known datasets, CICIDS2017 and CSE-CICIDS2018, to develop an image-based model that employs deep learning to classify attacks. The authors created a robust set of features, which were used to generate grayscale images that fit the model, resulting in better classification and characterization outcomes.

2.2 Systematic Literature Review

In this section, the Systematic Literature Review will be presented, which aims to survey works related to the research theme in a manner that allows other researchers to replicate the findings.

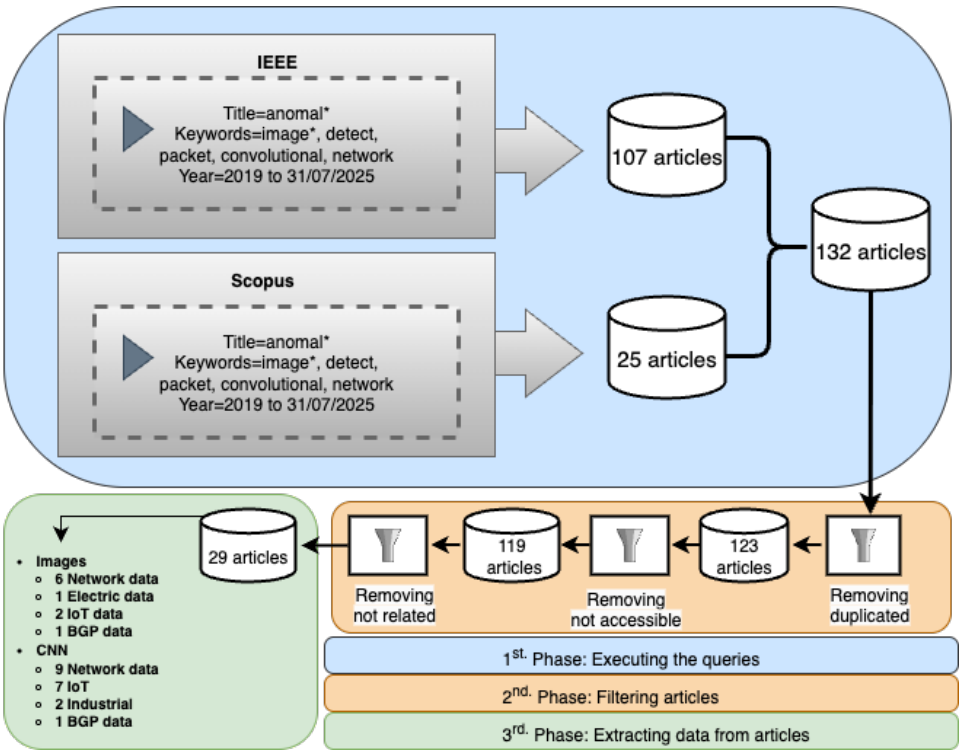
Keele et al. (2007) wrote a guideline emphasizing the importance of Systematic Literature Review as an objective approach to summarize existing research. According to the authors, systematic reviews aim to present a fair evaluation of a research topic using

[3] MobileNets are neural networks based on a simplified architecture that use depth-separable convolutions to build lightweight deep neural networks (HOWARD et al., 2017)

a trustworthy, rigorous, and auditable methodology. The guidelines cover three phases of a systematic literature review: planning, conducting, and reporting the review.

Based on these phases, the development of the procedure adopted in this research is illustrated in Figure 2.

Figure 2 – Systematic Literature Review Diagram



Source: from the author (2025)

In the following, we will see all the phases covered during this Systematic Literature Review:

1. The choice of keywords is crucial for the preparation of the Systematic Literature Review, as it is from this choice that the articles will be searched. For this thesis, the search used the following terms: Title: anomal*; Full text: image*, detect, packet, convolutional, network. The survey searched for articles published between 2019 and July 31, 2025, at the time of the search. The choice of these terms aimed to conduct a more refined search, attempting to exclude works unrelated to the theme of this thesis.

2. In this phase, research is carried out on two scientific bases. Due to their importance in the field of Computer Science, particularly in Computer Networks, IEEE and Scopus were selected. The search for IEEE yielded 107 articles, while the search for Scopus yielded 25 results, totaling 132.

3. In this step, articles were filtered to identify and extract the relevant information. The remaining 132 articles were analyzed to identify information about their contents, to

establish a relationship with the theme proposed in this thesis, particularly regarding the use of CNN and the creation of images from data packets to detect network anomalies.

4. In this stage, relevant information is extracted from the articles and organized better to understand the panorama of the network anomaly detection area using CNN and image creation. Nineteen studies were found that used CNN at some stage of the process. Nine works created images using data obtained from the network. Still, 90 studies were found that were not related to using CNN during the process, nor did they use images at any stage of anomaly detection; therefore, they will not be addressed in this section. Four articles could not be accessed; one was by the author of this thesis, and eight were duplicates. A summary of this phase is illustrated in Figure 3.

5. Finally, we separate the papers according to the research content. The works were organized as follows:

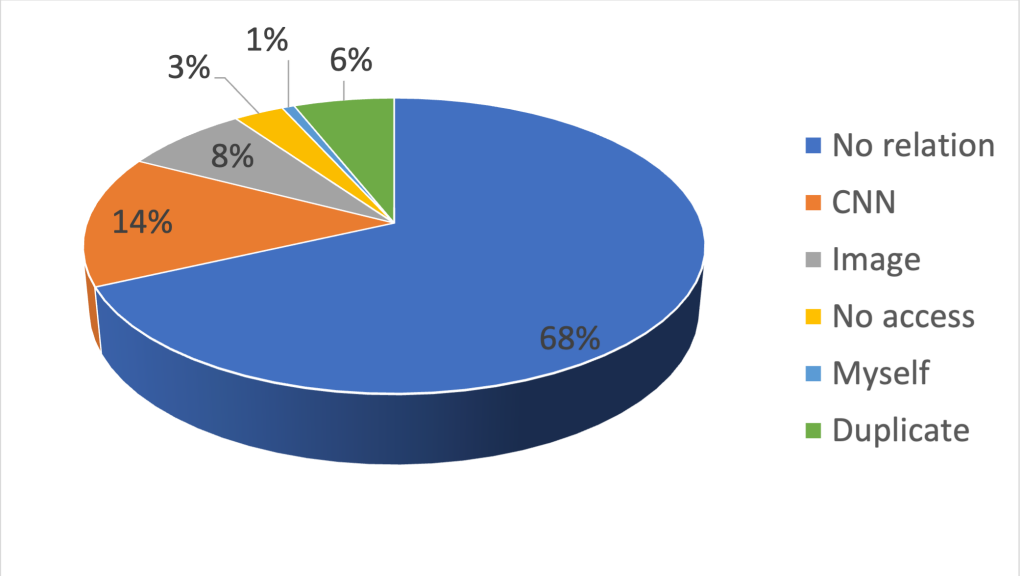
    - Ten works using images to detect anomalies, being:
        - Six using data networks;
        - One using power grid data;
        - Two, using data from IoT networks, and
        - One using BGP network data
    - Regarding the works that used CNN, we had a total of 19 articles, including:
        - Nine using data from computer networks
        - Seven using IoT (including vehicular networks);
        - Two using industrial networks, and
        - One using BGP network data

In the following sections, we will describe the articles related to the proposed topic, as they were separated in the previous phase.

2.2.1 Articles that use Images in Anomaly Detection

In this study, Sana et al. (2024) presents a highly accurate Intrusion Detection System tailored for IoT environments, using Vision Transformers (ViT) in conjunction with classical ML methods and LSTMs, by creating images from network data to detect anomalies. Given the complexity and multivariate nature of the IoT traffic, the authors optimize model performance using Bayesian hyperparameter tuning. The ViT-based architecture achieved near-perfect results, including 100% training accuracy and excellent anomaly detection metrics in terms of precision, recall, AUC, and MCC. The system significantly outperforms other traditional models, demonstrating its potential for real-time, interpretable, and scalable IoT anomaly detection.

Figure 3 – Systematic Literature Review Distribution



Source: from the author (2025)

Shaffi et al. (2024) evaluates the effectiveness of supervised deep learning methods for anomaly-based intrusion detection. Using benchmark datasets such as NSL-KDD and KDDCup99, the study compares two deep learning architectures in terms of precision and accuracy: CNN and Recurrent Neural Networks (RNN). The results demonstrate that both supervised deep learning models significantly improve the detection of hidden and complex intrusions that traditional IDS models often miss. The findings highlight the potential of deep learning to enhance cybersecurity in increasingly interconnected environments.

Ola-Obaado e Suleiman (2023) introduces a novel approach for intrusion detection by converting network traffic data into grayscale images and using the ResNet50 deep learning model with transfer learning to classify both binary and multiclass network anomalies. To overcome data imbalance in the CSE-CIC-IDS2018 dataset, the authors integrate the Synthetic Minority Oversampling Technique (SMOTE), resulting in improved generalization and classification performance. The proposed method achieved an accuracy of 92% in both binary and multiclass intrusion detection tasks, demonstrating the effectiveness of combining visual data representation and transfer learning for robust anomaly detection.

In their work, Xu et al. (2023) introduces a few-shot learning approach for network traffic anomaly detection using a Siamese Neural Network. This paper proposes a novel method for encoding raw traffic data from the CICIDS2017 dataset into 3D images. By comparing feature embeddings of network traffic flows, the model identifies anomalies with limited labeled data.

In the work of Hairab et al. (2022), the Bio-IoT dataset is utilized, which comprises real and simulated traffic data on IoT device networks, as well as various types of attacks.

Nine features are used, which are converted into a $3x3$ matrix to generate a grayscale image. The proposed model demonstrated high accuracy in detecting zero-day attacks, i.e., attacks that had not been previously reported.

Ahmad et al. (2022) presents a work that introduces an anomaly detection system for IoT networks, based on spectrogram images created from the Bot-IoT dataset, which organizes all the characteristics of the records into a Discrete-time signal. The proposed system uses a CNN to extract information from the spectrograms and detect anomalous behavior with an excellent accuracy rate of 99.97%.

Fejza e Lambert (2022) 's proposal uses BGP (Border Gateway Protocol) data to create images that will generate videos from the sequence of images. These videos will be used by Convolutional Autoencoders (CAE), which are neural network architectures that combine the convolutional layers of CNNs with the encoding and decoding components of autoencoders. The work achieved excellent detection rates for specific anomalies in BGP networks.

Khan et al. (2021), use the CICIDS-2017 dataset to generate spectrograms by extracting frequency information from network flows, using the Short-Time Fourier Transform (STFT) technique for performing the time-frequency analysis. The spectrograms are trained on a CNN, which achieves a success rate of 99.35% in detecting anomalies.

In Ullah e Mahmoud (2021), the authors also utilize data from IoT networks to generate images that a CNN can use for feature extraction. The proposed model is then implemented using 1D, 2D, and 3D CNNs. The proposed convolutional neural network model is validated using the BoT-IoT, IoT Network Intrusion, MQTT-IoT-IDS2020, and IoT-23 intrusion detection datasets. The authors also utilized transfer learning to enhance the classification of images into legitimate or anomalous traffic. The proposed model achieved both a high detection rate and a low false alarm rate.

Moore e Vann (2019) utilizes data acquired from vehicular networks to generate 2D images, which can be of fixed dimensions or dynamically adjusted according to the nature and size of the data. Each pixel in the image represents a specific feature value or combination of values. A CNN extracts the characteristics of the images, making it possible to detect with reasonable accuracy if a vehicle has had its electronic control unit (ECU) compromised.

2.2.2 Articles that use CNN in the anomaly detection process

In their research Lee e Nadeem (2025) explore anomaly detection in IoT by applying spectral analysis to acoustic signals, employing a lightweight model suited for resource-constrained environments. Although primarily centered on sound-based spectral features, the methodological principles can be extended to network traffic streams, where CNNs are conventionally applied for pattern recognition in spectral representations. The study underscores the importance of compact and efficient models, a key consideration when implementing CNNs

for anomaly detection in IoT network traffic.

Shaffi et al. (2024) introduces the HIAD (Hardware Introspection for Anomaly Detection) framework, which leverages low-level hardware debugging interfaces, such as JTAG, to extract memory traces from embedded devices and generate image-like data for anomaly classification using machine learning. By monitoring execution at the processor level, HIAD enables real-time anomaly detection in bare-metal programs without significantly affecting system performance. The CNN model outperformed the basic machine learning models by a significant margin, demonstrating that detecting malicious data through images is possible and applicable. The approach enables the proactive detection of deviations from expected behavior, thereby enhancing the security and reliability of embedded systems in resource-constrained environments.

In this work, Ngo et al. (2024) presents a scalable security framework for IoT networks that combines smart contracts with anomaly-based Intrusion Detection Systems (IDS), optimized through the use of hardware accelerators. The approach aims to deliver fast and secure responses to anomalous events at IoT gateways. While the focus is on system-level integration, the use of CNNs for efficient network traffic analysis is highly relevant, particularly when combined with hardware acceleration to meet the latency and computational constraints inherent in IoT ecosystems.

In their paper, Tang, Ye e Hu (2024) presents a network anomaly detection algorithm that leverages deep learning techniques to enhance the identification of abnormal patterns in information communication systems. The authors propose a network anomaly detection algorithm based on CNN and Gated Recurrent Units (GRU) for the spatial and temporal characteristics of network traffic data, aiming to improve detection accuracy over traditional methods. The work emphasizes the effectiveness of deep learning in managing high-dimensional network traffic data and mitigating false alarm rates in anomaly detection scenarios.

In their work, Verma, Keserwani e Jain (2024) introduce a method that uses Convolutional Neural Networks (CNNs) to detect anomalies in Border Gateway Protocol (BGP) traffic, which is crucial for maintaining the stability of the internet routing. Traditional detection methods often fail due to the highly dynamic and complex nature of BGP networks. Utilizing the ability of CNNs to capture spatial patterns in data, the authors trained and tested their model on real-world datasets from RIPE and BCNET. The results show that their CNN-based approach significantly outperforms previous techniques in terms of accuracy, demonstrating its effectiveness in identifying abnormal BGP behaviors and enhancing the internet routing infrastructure.

Fox e Boppana (2023) investigates the application of machine learning, including two-dimensional Convolutional Neural Networks, for the early detection of network-based attacks with minimal preprocessing. The study transforms raw network data into formats suitable for

various machine learning and deep learning models. It evaluates their performance on five open-source datasets that mostly contain DoS and botnet attacks. The results show that many models, including CNNs, achieve high accuracy (over 95%) in detecting anomalies, with Random Forest models consistently outperforming others in both accuracy and computational efficiency. The paper also provides a comparative analysis of features, limitations, complexity, and accuracy in relation to prior works, emphasizing the potential for real-time, in-progress attack detection rather than post-mortem analysis. It highlights the importance of balancing detection accuracy with processing speed for practical deployment in network security systems.

In this paper, Zhou et al. (2023) introduces P^3 AD, a privacy-preserved payload anomaly detection model tailored for the Industrial Internet of Things (IIoT). The approach combines a 2D-CNN-based autoencoder with a federated GAN [4] architecture to learn bidirectional network payload behavior without requiring direct data sharing, thus preserving privacy. It detects anomalies by comparing the original payload with its autoencoded reconstruction and leverages federated learning to enable collaborative training across different IIoT environments. Evaluation in four public industrial datasets demonstrated high effectiveness, achieving F1 scores above 0.966 globally and at least 0.753 in federated settings, making it a practical and secure anomaly detection solution for IIoT deployments.

BP, SM e LV (2023) propose a dual-model framework using Random Forest (RF) and Convolutional Neural Networks to classify applications and usage patterns in network traffic for anomaly detection. The system leverages both payload size sequence (PSS) and IP port-based techniques to capture distinct traffic features and flow signatures. A dataset of over 2.7 million packet flows was collected from a real-world enterprise network, spanning both wired and wireless environments. The CNN-based classifier demonstrated high accuracy in identifying traffic related to popular applications and detecting anomalous patterns in real time, validating the utility of deep learning for adaptive, data-driven network monitoring.

Liu e Wang (2023) proposes a real-time anomaly detection framework for software-defined networks (SDNs) using a CNN to classify network traffic flows based on raw features. The system operates at the edge of the network, enabling fast detection and response to threats such as DoS attacks and replay attacks. The CNN architecture enables efficient feature extraction without manual preprocessing, supporting low-latency deployment in real-world environments.

Yaqoob et al. (2023) also uses vehicular networks. However, instead of CNN, they use Convolutional Auto-encoders (CAEs). This neural network architecture combines the principles of convolutional neural networks with autoencoders. These unsupervised learning models aim

[4] A Generative Adversarial Network (GAN) is a machine learning model formed by two neural networks: the generator and the discriminator. The generator is responsible for creating new samples, such as images, sounds, or text, from a random latent space, and the discriminator is trained to distinguish between authentic samples and those generated by the generator. The main idea is that the GAN can learn to classify samples of networks as real or fake

to learn the compressed representation or encoding of the input data.

Yoshimura et al. (2022) 's work presents DOC-IDS, an intrusion detection system based on the deep classification of Perera's class. This machine learning method utilizes deep neural networks to model and detect anomalies in sets of unlabeled data, learning only the standard class of data and disregarding anomalies, thereby classifying them as anomalies that are not recognized as usual. DOC-IDS comprises 1D CNN and autoencoder components for feature extraction and anomaly detection. In the experiments, the model processed from feature extraction to anomaly detection without requiring labeling by inserting the pre-labeled traffic from an open dataset and the destination network traffic into the model.

In his work, Wu et al. (2022) presents a method for detecting anomalies in sounds acquired from IoT equipment in industrial networks. The method called You Only Hear Once (Yoho) utilizes a CNN to extract the characteristics of the analyzed sounds, thereby decreasing the completion time of the task and increasing the detection accuracy compared to another detection model, called Set-Det.

In Bertalanic et al. (2021), a dataset containing Wi-Fi network data is used, where anomaly data is synthetically injected into the dataset. The anomaly detection process utilizes a seven-layer CNN, comprising five convolutions and two dense layers. The final result is compared with state-of-the-art models, such as Logistic Regression, Random Forest, and Support Vector Machine, obtaining better results than these.

In work by Mezina, Burget e Travieso-González (2021), the KDD99 and CSE-CIC-IDS2018 datasets were used for validation and testing two methods based on convolutional neural networks: U-Net-based and Temporal Convolutional Networks (TCN). Additionally, the second method was combined with the LSTM model, achieving an increase in anomaly detection efficiency.

Sinha et al. (2021) uses two CNNs to determine the overall health of devices and components in an industrial environment. One CNN analyzes data packets, and the other analyzes sensor data. If the two predictive models differ in their state ranking (consensus break), the system has likely been compromised due to faulty equipment, communication errors, or some other source of malfunction. The system achieved reasonable detection rates for artificially introduced anomalies in the data.

In work by Wyk et al. (2020), a CNN combined with a Kalman filter utilizes data from vehicle sensor networks to detect anomalous behavior in real-time. The paper demonstrated that combining the two techniques yields superior results compared to using them alone.

In his work, Zhou (2020) uses a CNN within a GAN. The payloads generated by the GAN were then used in anomaly detection, which occurs in conditions where the reconstructed payloads have deviated significantly from the true ones, as the encoder-decoder architecture is trained to rebuild only typical payload structures.

In Karam, Salomon e Couturier (2020) 's work, a CNN was used before using a Long Short-Term Memory (LSTM) to extract features from the input data. LSTMs are a recurrent neural network architecture that captures long-term dependencies in data sequences. They can remember and forget relevant information over time, making them effective in tasks that involve sequences, such as time series analysis. The data used were audio files of communication between aircraft and controllers. The combined use of both models obtained excellent rates in detecting changes in aircraft altitude values.

Albasir et al. (2020) use non-intrusive techniques to acquire data from wireless devices, including network traffic and energy consumption data. These data are analyzed using two methods developed by the authors, one of which employs a CNN to detect anomalies. In both methods, accuracy rates of over 90% were achieved in malware detection.

Chapter conclusion

The systematic literature review revealed a growing interest in applying deep learning to network traffic analysis, particularly in the context of anomaly and intrusion detection. Several works demonstrated promising results using CNNs and visual representations of flow data; however, most of them focused solely on predictive performance, with limited concern for interpretability or model transparency. The integration of *X*AI techniques remains scarce, often limited to post-hoc visualizations or basic feature attributions.

Furthermore, many studies rely on outdated or synthetic datasets, lack reproducible protocols, or fail to explore the inner workings of models when faced with ambiguous or borderline cases. These gaps highlight the need for research that not only classifies traffic accurately, but also helps analysts understand why a given flow is considered malicious.

Therefore, this thesis aims to address this gap by combining high-performing CNN-based classification with visual explanations derived from Grad-CAM. By doing so, it contributes to the development of more transparent and trustworthy anomaly detection systems in the cybersecurity domain.

3 Methodology

In this section, the methodology adopted will be described, along with the datasets used to validate the experimental phase.

The transformation of network data into images is an approach that aims to extract features from images generated through a CNN, which would likely not be perceived if the data were used in its raw form, without extraction using convolutional filters.

3.1 Database

Both databases used in this work are publicly available to researchers for improving their models and will be presented below.

3.1.1 NSL-KDD

The NSL-KDD, described by Mahbod et al. (2009), is a dataset of Internet traffic composed of records generated by an Intrusion Detection System (IDS). The training samples from the data set represent 125,973 records, while the test set contains 22,544 records.

Based on the work of Bajaj e Arora (2013), who discovered that some attributes have no role in attack detection, it was decided to remove attribute number 9 so that the *dataset* has 40 characteristics, which express the traffic of the Internet and two classes (attack or non-attack). The number of 40 features will be significant because, when generating the images, it will be possible to utilize all the features when developing a figure with the proportions of an 8×5 matrix.

3.1.2 CSE-CIC-IDS2018

The CSE-CIC-IDS2018, presented by Sharafaldin et al. (2018), is a widely used benchmark dataset for evaluating intrusion detection systems (IDS), especially those based on machine learning. It was created through a collaboration between the Communications Security Establishment (CSE) and the Canadian Institute for Cybersecurity (CIC). The dataset simulates realistic network traffic across 10 days, including both benign and malicious behaviors in a controlled environment. It encompasses a diverse range of contemporary cyberattacks, including brute-force SSH, Heartbleed, DDoS, web attacks, botnets, and infiltration, thereby enabling the robust training and testing of IDS models.

One of the standout features of this dataset is the inclusion of flow-based features, which are explicitly crafted for intrusion detection. These about 80 features are extracted using

CICFlowMeter and represent both statistical and behavioral attributes of network connections. Examples include Flow Duration, Total Fwd Packets, Average Packet Size, Fwd Header Length, and Subflow Fwd Bytes. These are crucial for identifying patterns in network behavior, enabling models to distinguish between normal and anomalous traffic. The dataset also includes labeled data, specifying the exact type of attack or whether a flow is benign, which is essential for supervised learning approaches in IDS.

The CSE-CIC-IDS2018 dataset is designed with scalability and real-world applicability. It supports both binary classification (benign vs. attack) and multiclass classification (specific attack types), making it highly adaptable for various IDS strategies. Its comprehensive feature set, realistic traffic generation, and detailed labeling have made it one of the gold standards for evaluating machine learning-based IDS models. For researchers developing deep learning or anomaly-based IDS frameworks, this dataset provides a robust foundation for training, validation, and benchmarking.

3.2 Scenario 1

In this scenario, the NSL-KDD dataset was utilized to create grayscale bitmaps, and anomaly detection was performed without the application of *X*AI.

3.2.1 Architecture

The feature vector of each record in the NSL-KDD dataset was transformed into a matrix, treating it as an image. The resource arrangement within this new image is described in Table 1. It is worth noting that the order of features in the vector corresponds to the pixel sequence in the image: the first feature corresponds to the first pixel, the second feature to the second pixel, and so on.

Table 1 – Arrangement of features in the image.

1	2	3	4
5	6	7	8
10	11	12	13
14	15	16	17
18	19	20	21
22	23	24	25
26	27	28	29
30	31	32	33
34	35	36	37
38	39	40	41

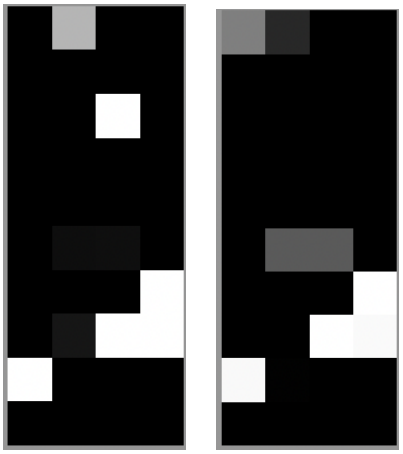
Source: from the author (2025)

In the creation of the images, each character was used in a pixel of the image for a grayscale, varying between the values of 0 and 255, according to the formula:

$$x = \sum_{j=1}^{4} \sum_{k=1}^{10} \frac{x_{jk} - 0}{255 - 0} \tag{1}$$

Consequently, each image pixel corresponds to a specific feature, with its brightness determined by the value of that feature. Examples of two converted records as images can be seen in Figures 4a and 4b.

Figure 4 – 40 features represented in two images.



(a) (b)

Source: from the author (2025)

A Convolutional Neural Network (CNN) was constructed using three convolutional layers. The aim is to classify the images and detect potential attacks. A maxpool layer was applied between each convolution layer, which downsamples the output of the previous convolutional layer by taking the maximum value from each region of the feature map. The maxpool helps reduce the feature map's size while preserving the most important features.

The activation function chosen for the network was the Rectified Linear Unit (ReLU), a non-linear function often used in neural networks. The ReLU function is defined as follows:

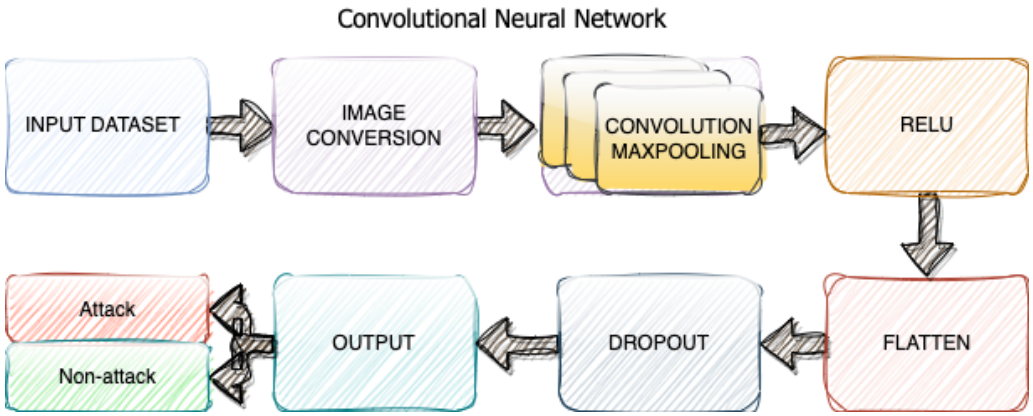
$$f(x) = max(0, x) \tag{2}$$

The ReLU function outputs the input value if it is positive and zero if it is negative, making ReLU a very efficient activation function, as it does not require any multiplication or division operations.

The feature map underwent a flattening operation, converting it into a new feature vector, which will be utilized by the fully connected layer in subsequent stages. Additionally, a dropout layer was incorporated to mitigate overfitting.

Finally, a classification was conducted using a fully connected layer with a sigmoid activation function. Since preprocessing the features, the complete process is illustrated in Figure 5 until the attack or non-attack classification is performed.

Figure 5 – Proposed methodology flowchart.

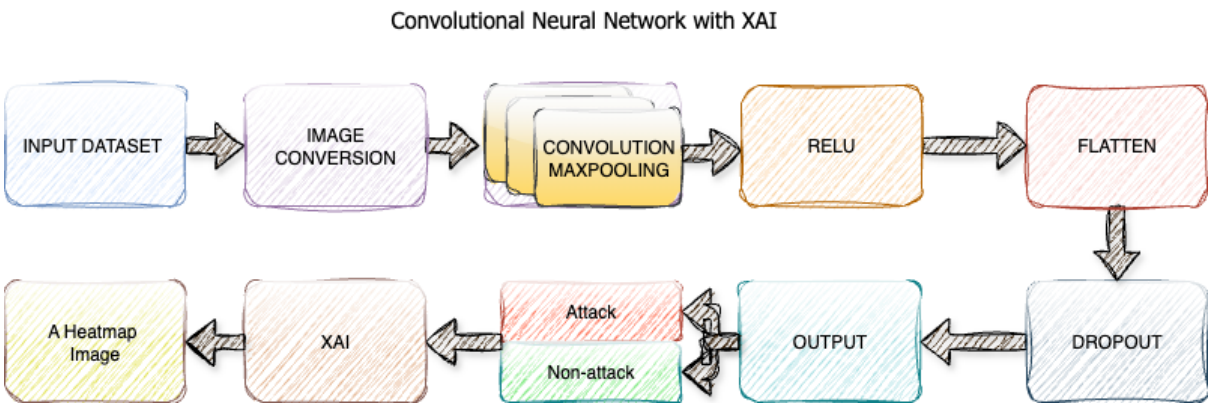


Source: from the author (2025)

3.3 Scenario 2

In this scenario, a methodology for detecting attacks in network traffic is proposed by transforming tabular features from the CSE-CIC-IDS2018 dataset into RGB image representations, extracting spatial patterns using a Convolutional Neural Network, and applying the Gradient-weighted Class Activation Mapping (Grad-CAM) technique to produce interpretable visual explanations of model predictions. The pipeline consists of four main stages: (1) dataset preparation, (2) data-to-image transformation, (3) CNN-based classification, and (4) explainable AI analysis via Grad-CAM, as illustrated in Figure 6.

Figure 6 – Proposed methodology flowchart using *X*AI.

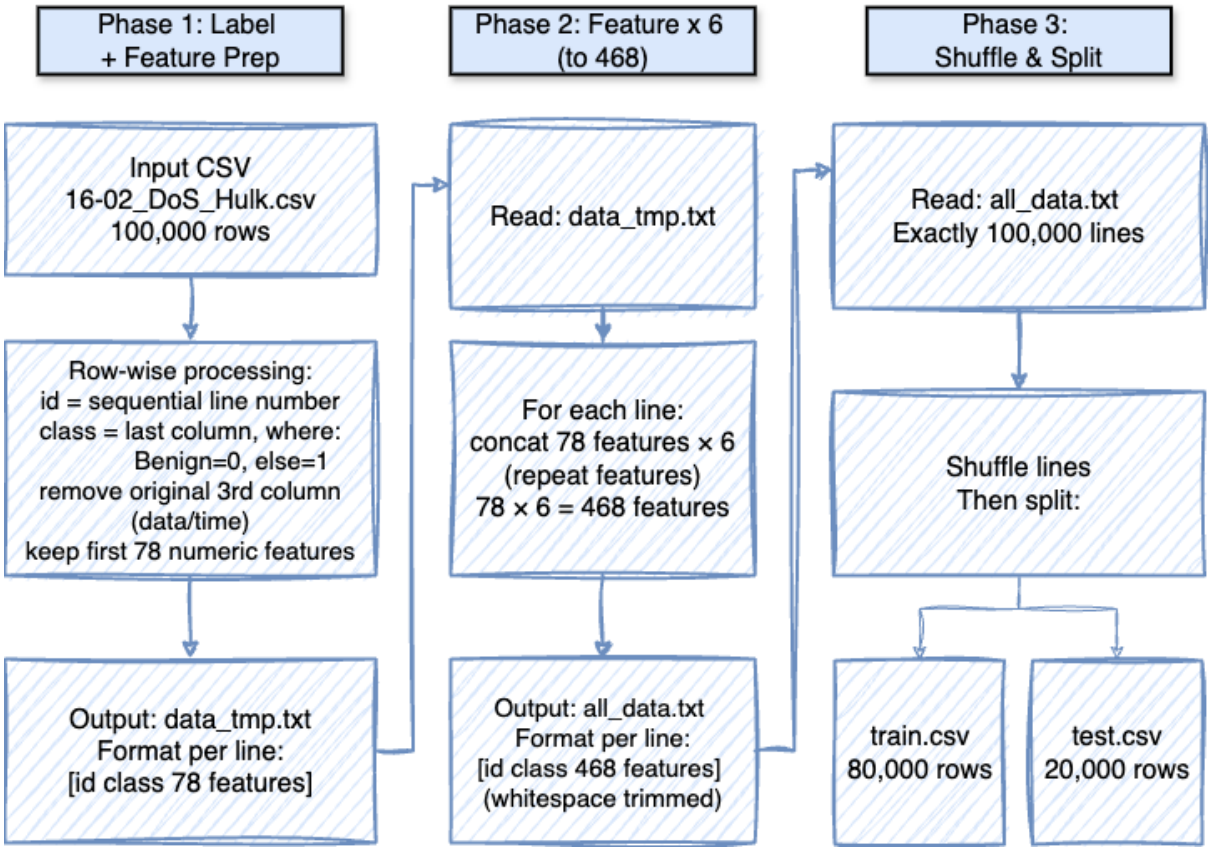


Source: from the author (2025)

3.3.1 Preparation

At this stage, the data are prepared for processing and then divided into training and test datasets. The images will be created using data from Denial of Service (DoS) attacks, specifically those conducted using the Hulk tool on February 16, 2018. In this instance, 100,000 rows were randomly selected to create these bitmaps. Figure 7 illustrates the three phases of this processing.

Figure 7 – Three–phase data preparation pipeline for CSE-CIC-IDS2018 prior to image generation.



Source: from the author (2025)

The original dataset contains 80 columns, where the third column indicates the date and time, and the last column denotes the flow type (benign or attack-type). The date and time column was removed because, when randomizing the data, this information is no longer relevant.

The last column will be used to indicate the flow class (benign or non-benign), changing it to 0 to identify benign traffic and 1 to identify any other type of traffic.

To ensure that the resulting image from this process would be of an adequate size for the convolutional filters to extract the best features from the images, we chose to multiply the number of columns. Tests were performed to determine the optimal accuracy, precision, recall,

and AUC rates, ranging from no multiplication to eightfold multiplication. These are shown in Figure 8, where the rounded feature multiplication rates by 5 to 8 show the best rates.

Upon closer examination of the absolute results, the best multiplication rate found was six times, which will be utilized in image creation.

Then, the file containing the remaining 468 features is divided between the training and testing files in an 80/20 proportion, resulting in 80,000 lines in the training file and 20,000 lines in the testing file.

3.3.2 Data-to-Image Transformation

The generated RGB images will be divided into training and testing sets according to the original dataset split. An `ImageDataGenerator` is used to normalize pixel values to the range $[0, 1]$ and to feed images into the network in batches. The dataset is balanced across benign and attack classes to ensure unbiased classification.

The CIC-IDS2018 dataset contains labeled records of both benign and malicious network flows. Each record is transformed into a fixed-size RGB image by mapping consecutive features to the three color channels (Red, Green, Blue) of each pixel. Before mapping, features are normalized to the range $[0, 255]$ using min–max scaling:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \times 255 \tag{3}$$

Where x is the original feature value, $\min(x)$ and $\max(x)$ are the minimum and maximum feature values in the dataset, and x' is the normalized value. The normalized vector is reshaped to a $(13 \times 12 \times 3)$ array, producing an image with 13 rows, 12 columns, and three RGB channels. This encoding preserves local correlations between features, enabling the CNN to capture spatial dependencies in network traffic patterns. A mapping between feature indexes and pixel positions is stored to support later interpretation.

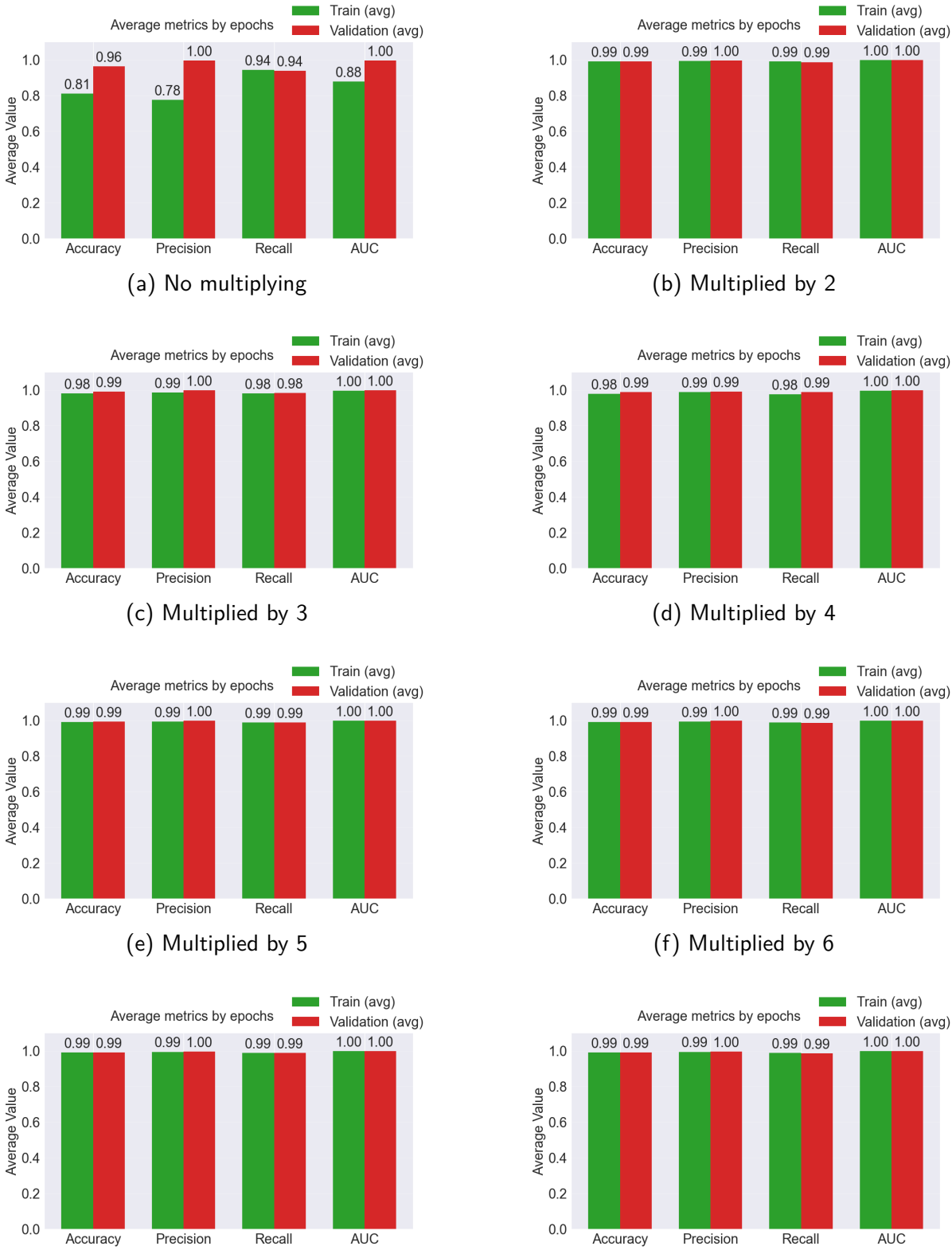
For each line of each file, `train.csv` and `test.csv`, a bitmap image was created using the RGB standard, where each pixel corresponds to one of the feature values in the dataset. Figure 9 shows an example of a generated image.

3.3.3 CNN Architecture

The proposed classifier is a sequential CNN designed to capture local spatial correlations in image-encoded network traffic. The architecture consists of:

1. **Conv2D Layer 1:** 32 filters, 3×3 kernel, ReLU activation, `same` padding.
2. **MaxPooling2D Layer 1:** Pool size 1×2.

Figure 8 – Average metrics by epochs



(a) No multiplying

(b) Multiplied by 2

(c) Multiplied by 3

(d) Multiplied by 4

(e) Multiplied by 5

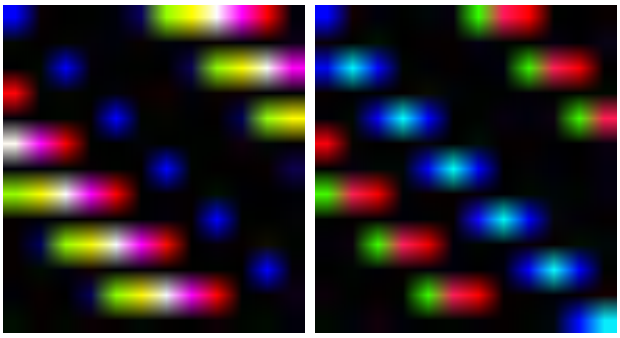
(f) Multiplied by 6

(g) Multiplied by 7

(h) Multiplied by 8

Source: from the author (2025)

Figure 9 – Features into colorized image



(a) Image 1 (b) Image 2

Source: from the author (2025)

3. **Conv2D Layer 2:** 64 filters, 3×3 kernel, ReLU activation, `same` padding.
4. **MaxPooling2D Layer 2:** Pool size 2×1.
5. **Conv2D Layer 3:** 128 filters, 2×2 kernel, ReLU activation, `valid` padding.
6. **MaxPooling2D Layer 3:** Pool size 2×2.
7. **Flatten:** Converts feature maps to a one-dimensional vector.
8. **Dropout:** Dropout rate of 0.5 to mitigate overfitting.
9. **Dense:** Single neuron with sigmoid activation for binary classification.

The model is compiled using the Adam optimizer, binary cross-entropy loss, and evaluated using accuracy, precision, recall, AUC, and metrics derived from the confusion matrix. To improve generalization and avoid overfitting, the training procedure includes two regularization callbacks:

- EarlyStopping with patience=10, monitoring val_loss.
- ModelCheckpoint monitoring val_accuracy to save the best model as best_model.h5.

3.3.4 Explainable AI via Grad-CAM

To interpret the CNN's predictions, Grad-CAM is applied to generate class-discriminative heatmaps. Grad-CAM computes the gradient of the predicted class score with respect to the feature maps of the final convolutional layer, averages the gradients to obtain channel weights α_k^c, and computes a coarse localization map:

$$L^c_{\text{Grad-CAM}} = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right) \tag{4}$$

Where A^k is the k-th feature map of the last convolutional layer, and the channel weights α_k^c are computed as:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k} \tag{5}$$

Here, Z is the number of pixels in A^k, y^c is the class score, and ReLU retains only positive contributions. The resulting heatmap is upsampled to the input image resolution and overlaid on the original RGB image, highlighting the regions most influential in the model's decision. This visual explanation supports human analysts in validating the CNN's focus and improving trust in the detection system.

3.3.4.1 Threshold Selection Methodology

To systematically determine the optimal threshold for binarizing Grad-CAM activation maps, we developed a dedicated script named `select_gradcam_threshold.py`. This script combines quantitative metrics of map coherence with a principled prior distribution in order to identify the threshold value that produces the most interpretable saliency maps.

3.3.4.1.1 Activated Proportion

For a given threshold $t \in [0, 1]$, the activation map $M(x, y)$ is first binarized:

$$B_t(x, y) = \begin{cases} 1 & \text{if } M(x, y) \geq t, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

The activated proportion is then computed as the fraction of pixels activated:

$$p(t) = \frac{1}{N} \sum_{x,y} B_t(x, y), \tag{7}$$

where N denotes the total number of pixels in the image.

3.3.4.1.2 Normalized Entropy

To assess the spatial coherence of the binary mask, we compute the normalized entropy:

$$H(t) = -\frac{1}{\log K} \sum_{i=1}^{K} q_i(t) \log q_i(t), \tag{8}$$

where $q_i(t)$ denotes the relative frequency of activated pixels in the i-th connected component, and K is the number of components. This normalization ensures $H(t) \in [0, 1]$, with lower values indicating more concentrated and interpretable activation.

3.3.4.1.3 Gaussian Prior on Activated Proportion

To avoid extreme thresholds that either saturate or nullify the activation map, we introduce a Gaussian prior over the activated proportion:

$$\pi(p(t)) = \exp\left(-\frac{(p(t)-\mu)^2}{2\sigma^2}\right), \tag{9}$$

with μ denoting the desired central activation proportion (here $\mu = 0.15$) and σ controlling its spread (here $\sigma = 0.1$).

3.3.4.2 Final Scoring Function

The final score for each threshold is defined as the product of the prior and the inverse of the entropy:

$$S(t) = \pi(p(t)) \cdot \frac{1}{H(t)+\epsilon}, \tag{10}$$

where ϵ is a small constant to avoid division by zero. The optimal threshold t^* is then chosen as:

$$t^* = \arg\max_{t \in T} S(t), \tag{11}$$

with $T = \{0.1, 0.2, \ldots, 0.9\}$ being the set of candidate thresholds.

3.3.5 Results of Threshold Optimization

Following the steps described previously, we applied the `select_gradcam_threshold.py` script to perform a systematic threshold sweep over the range $t \in \{0.1, 0.2, \ldots, 0.9\}$. For each candidate threshold, three metrics were computed: the activated proportion $p(t)$, the normalized entropy $H(t)$, and the final score $S(t)$ defined in Equation 10. The results are summarized in Table 2.

Table 2 – Threshold sweep results with activated proportion, normalized entropy, and final score.

Threshold t	Activated Proportion $p(t)$	Normalized Entropy $H(t)$	Score $S(t)$
0.1	0.6090	0.9655	0.0346
0.2	0.4808	0.9989	0.0053
0.3	0.4295	0.9856	0.0345
0.4	0.3333	0.9183	0.2680
0.5	0.2885	0.8667	0.5167
0.6	0.2244	0.7680	0.9904
0.7	0.1987	0.7194	1.1687
0.8	0.0769	0.3912	**1.3744**
0.9	0.0000	0.0000	1.3247

Source: from the author (2025)

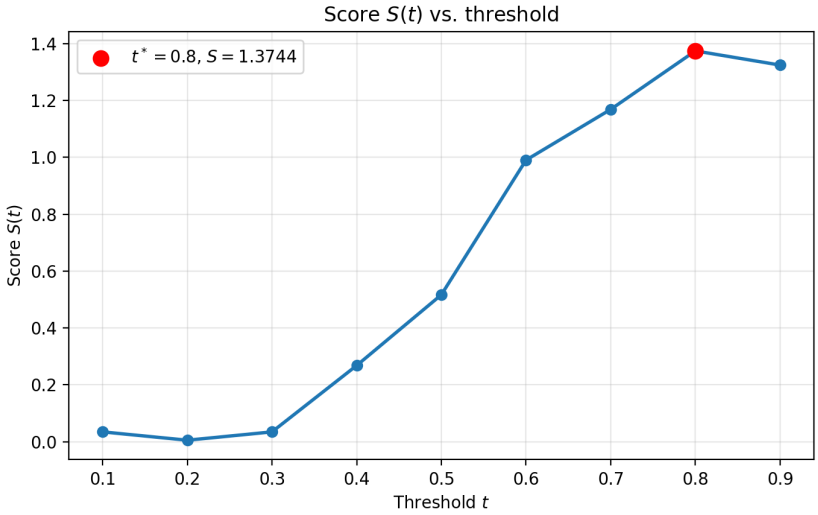
As shown in Table 2, low thresholds ($t = 0.1$–0.3) led to high activation proportions ($p(t) \geq 0.42$), but their normalized entropy values approached 1.0, indicating highly scattered and less interpretable activation maps. Intermediate thresholds ($t = 0.4$–0.6) gradually reduced the entropy, resulting in improved scores and suggesting more localized regions of interest.

The best performance was observed at $t^* = 0.8$, which achieved the maximum score $S(0.8) = 1.3744$. This value corresponds to a low activation proportion ($p(0.8) = 0.0769$) combined with a reduced normalized entropy ($H(0.8) = 0.3912$), resulting in a sparse but highly concentrated activation map. This balance between compactness and focus aligns with the expected behavior of Grad-CAM explanations, where salient regions should be both minimal and semantically meaningful.

It is also noteworthy that $t = 0.9$ produced a comparable score ($S(0.9) = 1.3247$), but with zero activated proportion, effectively removing the explanation. Therefore, the chosen threshold $t^* = 0.8$ provides the most interpretable tradeoff, corroborating the Gaussian prior preference for sparse yet non-empty activation patterns, as discussed previously.

As illustrated in Figure 10, the score function $S(t)$ increases steadily with higher thresholds, peaking at $t^* = 0.8$. This value maximizes the tradeoff between low activation proportion and reduced entropy, yielding the most interpretable Grad-CAM explanations.

Figure 10 – Score $S(t)$ as a function of threshold $t \in [0.1, 0.9]$. The curve shows that the maximum score was reached at $t^* = 0.8$, indicated by the red marker.

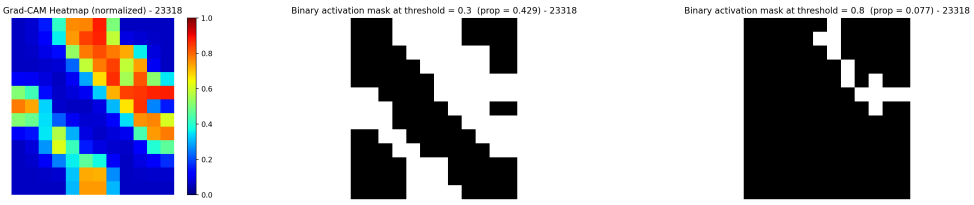


Source: from the author (2025)

Figure 11 further illustrates the effect of thresholding on Grad-CAM explanations in an aleatory image. At low threshold values, such as $t = 0.3$, a high activation proportion ($p = 0.4295$) results in broad highlighted areas with high entropy. The activated region is large and dispersed, attenuating the interpretability of the explanation. In contrast, in the optimal threshold $t = 0.8$, the activation proportion drops to $p = 0.0769$, producing a sparse but

highly focused explanation, consistent with the optimal score $S(0.8) = 1.3744$, which produces compact and semantically meaningful regions, consistent with the quantitative findings in Table 2.

Figure 11 – Visualization of Grad-CAM thresholding. (a) The normalized heatmap, (b) At threshold $t = 0.3$, (c) At threshold $t = 0.8$.



(a) Heatmap (normalized) (b) Mask at $t = 0.3$ (c) Mask at $t = 0.8$

Source: from the author (2025)

3.3.6 Mann–Whitney *U* Test

The Mann–Whitney *U* test, is a non-parametric statistical method introduced by Mann e Whitney (1947). It is used to determine whether two independent samples originate from the same distribution, without assuming that the underlying data is usually distributed. This characteristic makes it especially suitable for evaluating deep learning activation metrics, such as the Grad-CAM intensities analyzed in this study, which may not follow a Gaussian distribution.

Let two independent samples be denoted as $X_1, X_2, \ldots, X_{n_1}$ and $Y_1, Y_2, \ldots, Y_{n_2}$, representing the values of a given metric for the benign and attack classes, respectively. Both samples are combined and ranked in ascending order. The sum of ranks for the first group, denoted as R_X, is calculated as:

$$R_X = \sum_{i=1}^{n_1} r(X_i) \tag{12}$$

where $r(X_i)$ is the rank assigned to the observation X_i in the combined dataset. The Mann–Whitney statistic U_X for the first group is then given by:

$$U_X = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R_X \tag{13}$$

Analogously, the statistic for the second group is $U_Y = n_1 n_2 - U_X$. The smaller of the two values, $\min(U_X, U_Y)$, is used as the final test statistic, referred to simply as U. Under the null hypothesis H_0, which assumes that both groups come from the same population, the expected value and variance of U are given by:

$$E[U] = \frac{n_1 n_2}{2}, \quad Var(U) = \frac{n_1 n_2 (n_1 + n_2 + 1)}{12} \tag{14}$$

For sufficiently large samples ($n_1, n_2 > 20$), the distribution of U can be approximated by a normal distribution, allowing the computation of a standardized z-score:

$$z = \frac{U - E[U]}{\sqrt{Var(U)}} \tag{15}$$

From this value, a two-tailed *p*-value is derived, which represents the probability of observing such an extreme difference in ranks if the null hypothesis were true. A small p-value (typically $p < 0.05$) provides statistical evidence that the two distributions differ significantly.

In this thesis, the Mann–Whitney *U* test was employed to compare the distributions of Grad-CAM activation metrics between benign and attack images. Specifically, it was applied to two quantitative measures: the proportion of activated pixels above a fixed threshold and the mean activation intensity of each image. By doing so, it was possible to verify whether the activation patterns produced by the CNN were statistically distinguishable across classes, supporting the interpretability analysis of the model.

3.3.7 Loss Function

During the training process of machine learning models, it is necessary to have a mechanism to quantify how far the model's prediction is from the expected (true) value. This measure is provided by the *loss function*.

The loss value is computed at each iteration (or epoch) and reflects the average error made by the model on a given dataset. In general terms:

- High loss indicates that the predictions are far from the true values, suggesting that the model has not yet adequately learned the patterns in the data.
- Low loss indicates that the predictions are closer to the true values, suggesting better performance of the model on that dataset.

The goal of the training process is to minimize the loss value by adjusting the model's parameters through optimization algorithms such as Stochastic Gradient Descent (SGD) or its variants.

In this work, the Binary Cross-Entropy Loss function was employed, as it is suitable for binary classification problems, enabling the training process to adjust the model parameters to minimize the discrepancy between predictions and true values.

The monitoring of the loss value was carried out for both the training set (*training loss*) and the validation set (*validation loss*) in order to evaluate the model's learning and detect possible cases of overfitting.

3.3.8 Regularization Strategy

To enhance generalization and prevent overfitting during CNN training, two regularization strategies were incorporated: EarlyStopping and ModelCheckpoint. These techniques are widely adopted in deep learning experiments to dynamically monitor performance metrics during training and retain the optimal model.

The EarlyStopping callback was configured to monitor the validation loss (val_loss) and to interrupt training if no improvement was observed in a window of 10 consecutive epochs. This strategy ensures that the model does not overfit the training data by continuing to train unnecessarily after reaching a performance plateau.

Simultaneously, the ModelCheckpoint callback was set to monitor the validation accuracy (val_accuracy) and to persist the model weights that yielded the best validation performance. The model was saved to the file best_model.h5, which was subsequently used for evaluation and Grad-CAM-based visual explanation.

By combining both mechanisms, the training pipeline effectively balances performance and generalization. All interpretability analyses and final evaluations were conducted using this best-performing model.

Algorithm 1: Proposed Anomaly Detection and Explainability Pipeline

Input: CIC-IDS2018 dataset (tabular features)

Output: Classification label (Benign/Attack) and Grad-CAM heatmap

Stage 1: Data-to-Image Transformation

for each *network flow record* r *in dataset* **do**

- Normalize features to $[0, 255]$ using Eq. 3 ;
- Map consecutive features to RGB pixel channels ;
- Reshape to $(13 \times 12 \times 3)$ RGB image ;
- Save image and corresponding class label ;

end

Stage 2: Dataset Preparation

Split images into *training* and *testing* sets ;

Normalize pixel values to $[0, 1]$;

Configure batch generators for model input ;

Stage 3: CNN Model Training with Regularization

Initialize CNN with three convolutional layers, pooling layers, dropout, and dense output layer ;

Compile model with Adam optimizer and binary cross-entropy loss ;

Define `EarlyStopping` callback monitoring `val_loss` with patience $= 10$;

Define `ModelCheckpoint` callback monitoring `val_accuracy` and saving best model to `best_model.h5` ;

Train model using validation data and callbacks ;

Store best model and training metrics ;

Stage 4: Explainable AI via Grad-CAM

for each *test image* **do**

- Load best model from `best_model.h5` ;
- Forward pass image through CNN ;
- Compute gradients of predicted class score w.r.t. final convolutional feature maps ;
- Compute channel weights α_k^c using Eq. 5 ;
- Generate localization map $L^c_{\text{Grad-CAM}}$ using Eq. 4 ;
- Upsample heatmap to original image resolution ;
- Overlay heatmap on original RGB image ;

end

return classification results and visual explanations ;

3.4 Evaluation Metrics

We used some performance metrics described by Witten, Frank e Hall (2011) to measure the model's efficiency in detecting anomalies. The performance evaluation metrics will be accuracy, recall, precision, and area under the curve (AUC).

A confusion matrix was created to calculate these metrics, identifying all four possibilities to compare the tests performed and the training data. In the confusion matrix, the quantities of True Positive (TP) rates are identified, that is, all the times the traffic was classified as anomalous and presented an anomaly; the True Negative (TN) rates, when the traffic was classified as usual, and it was so, without anomalies; the False Positive Rates (FP), when the traffic was identified as anomalous and was normal; and the False Negative (FN) rates, when traffic was identified as usual and was anomalous. The confusion matrix is presented according to Table 3.

Table 3 – Confusion Matrix

		Actual	
		Anomaly	**Normal**
Predicted	**Anomaly**	TP	FN
	Normal	FP	TN

Adapted from Witten, Frank e Hall (2011)

All evaluation and performance metrics used in this work will be described below.

3.4.1 Accuracy

Accuracy measures the percentage of positive and negative samples (TP and TN) that are correctly classified, considering the sum of all negative and positive samples (TP, TN, FP, FN). It is expressed in accordance with equation 16:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{16}$$

This metric aims to calculate the probability of the system generating correct results.

3.4.2 Recall

Recall evaluates the percentage of positive samples (TP) classified correctly over the total number of truly positive samples, that is, the number of samples that were an anomaly (TP, FN). The calculation is given by equation 17:

$$Recall = \frac{TP}{TP + FN} \tag{17}$$

3.4.3 Precision

Precision calculates the percentage of positive samples (TP) correctly classified over the total number of samples that were classified as positive. It is given by the equation 18:

$$Precision = \frac{TP}{TP + FP} \tag{18}$$

3.4.4 AUC

The receiver operating characteristic (ROC) is a term used in signal detection to characterize the tradeoff between hit rate and false alarm rate over a noisy channel. ROC curves depict the performance of a classifier without regard to class distribution or error costs. They plot the "true positive" rate on the vertical axis against the "false positive" rate on the horizontal axis.

The area under the curve (AUC) is a single scalar value that summarizes the classifier's overall performance across various threshold values. It represents the area under the ROC curve. Specifically, it quantifies the probability that a randomly chosen positive example (e.g., an anomaly detection) will be ranked higher by the model than a randomly chosen negative example (e.g., a regular traffic).

AUC values range from 0 to 1. A higher AUC indicates better model performance in distinguishing between positive and negative classes.

The AUC is represented in equation 19:

$$AUC = \sum_{i=0}^{n-1} \frac{(FPR[i+1] - FPR[i]) \cdot (TPR[i] + TPR[i+1])}{2} \tag{19}$$

In the next chapter, the results and the experiments conducted will be found.

4 Results and experiments

As explained in the Methodology section, two test scenarios were employed: the first involved creating grayscale images using the NSL-KDD dataset, and the second involved creating RGB images using the CIC-IDS2018 dataset.

The results for both proposed scenarios are presented below.

4.1 Scenario 1

For the experiments, we used 50% of the NSL-KDD dataset, comprising 62,986 images for training and 11,271 images for validation and testing purposes. All examples were randomly selected from the dataset, which is already separated between training and testing data, so the data is totally different.

For the creation of CNN, the libraries *Python TensorFlow*, described by Abadi et al. (2016) and *Keras*, described by Keras et al. (2015), were adopted. The metrics accuracy, precision, *recall, f-measure*, and Area Under the Curve (AUC) were used to validate the results.

Table 4 presents a summary of the results, including the average of 100 epochs for CNN training and the results in the test set.

Table 4 – Results

Metric	Test Result	Train Result
Accuracy	78.25%	99.79%
Precision	96.73%	99.78%
Recall	64.03%	99.77%
F-Measure	77.05%	99.37%
AUC	0.823	0.999

Source: from the author (2025)

It can be seen that the accuracy reached 78% in the test samples, which, despite not being equal to the training accuracy, shows that the new method achieves acceptable use in attack detection.

The values found in the precision metric are excellent, indicating that 96.73% of the samples classified as anomalies were anomalous traffic. This result showed a low false positive rate, indicating that the model correctly classified the attacks.

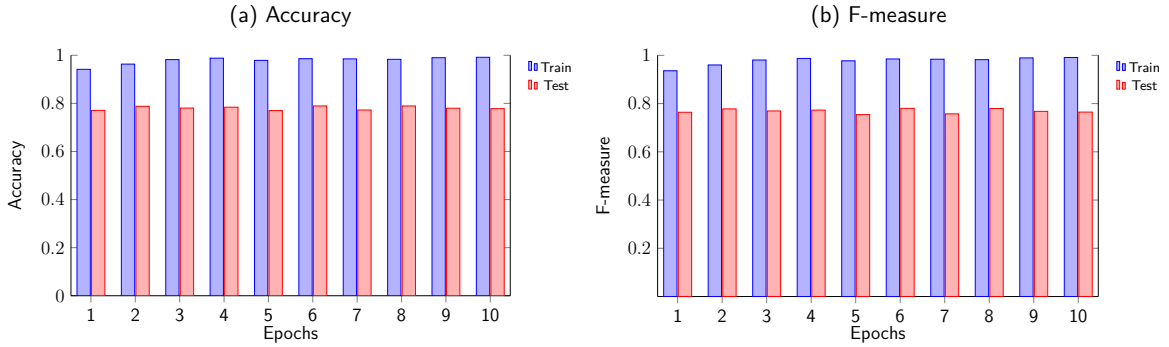
However, the value perceived in recall indicates that there is still considerable value in classifying the reported samples as False Negatives, i.e., the model did not classify an attack

when it should have. This low value indicates that the model needs improvement to reduce False Positives.

The *F-measure* indicates a 77.05% success rate for general forecasts, indicating that the model may have a fair accuracy rate and a low recall rate, resulting in overall good performance.

The AUC value of 0.823 indicates a reasonable identification rate of class hits, meaning the model effectively separates attacks from legitimate traffic.

Figure 12 – Results of the experiments for 10 Epochs.



Source: from the author (2025)

In Figure 12a, the values of 10 epochs for accuracy in each epoch are observed, with training and test values relatively close. Furthermore, in Figure 12b, one can see the values of the *F-measure* in each of the ten epochs.

Regarding the values obtained in the test phase, the model does not exhibit as high efficiency in the test set as it does in training. However, the successes in detecting the attacks, as indicated by the accuracy values, show promising perspectives.

4.2 Scenario 2

For this scenario, the CIC-IDS2018 dataset was used, and new images were created in RGB channels (colorized). The images were submitted to CNN for attack detection classification. Ultimately, a heat map was generated using Grad-CAM, highlighting the areas of the image that were most important in detecting the anomaly.

From the DoS-attack subset from February 16, 2018, the data was separated into training and testing data. The total of 100,000 images was then generated, which were separated into 80,000 images for training and 20,000 for testing. None of the training data was used for validation, so the final result would not be affected.

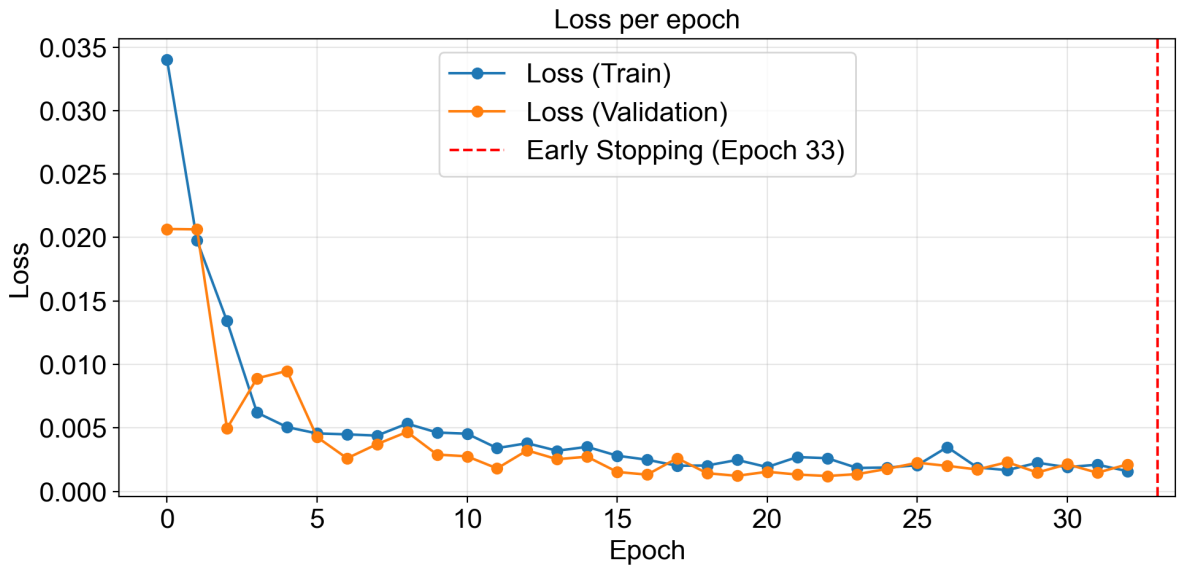
4.2.1 Early Stopping Behavior

The 100 epochs option was used, but to prevent overfitting and enhance generalization, we incorporated two regularization strategies: EarlyStopping and ModelCheckpoint.

The training process was monitored using the EarlyStopping callback, configured to observe the validation loss (val_loss) with a patience parameter of 10 epochs. As a result, training was automatically halted at epoch 33, after 10 consecutive epochs without improvement in the validation loss. This early stopping mechanism effectively prevented overfitting and ensured that the model's generalization capability was preserved. The best model, as determined by the validation accuracy, was saved using the ModelCheckpoint mechanism and subsequently utilized for all stages of evaluation and Grad-CAM interpretation.

Figure 13 shows the loss values for each epoch until the stopping criterion was reached at 33 epochs, demonstrating that there was no overfitting and that training time was saved. Loss values close to zero also indicate that the model has few prediction errors, which brings it closer to the ideal.

Figure 13 – Loss per epoch.



Source: from the author (2025)

Table 5 presents the numerical results of the metrics used in the validation. We can see that all values reach about 99%, indicating that using a CNN to extract network data through image creation is highly effective in detecting attacks.

Table 5 – Metrics in the last epoch

Metric	Train	Validation
Accuracy	0.999150	0.999000
Precision	0.999350	0.999913
Recall	0.999176	0.998341
Auc	0.999998	0.999940

Source: from the author (2025)

This stability across epochs reinforces the model's robustness. It reduces the likelihood of overfitting, particularly given the application of early stopping at epoch 33, which prevented unnecessary training iterations and potential performance degradation.

In Table 6, we present the average values of the metrics across the 33 training and testing epochs. Values of 99% efficiency are also observed in all metrics evaluated in both training and test sets.

Table 6 – Average metrics across all epochs

Metric	Train (avg)	Test (avg)
Accuracy	0.998216	0.998508
Precision	0.998852	0.999391
Recall	0.998053	0.998003
Auc	0.999858	0.999929

Source: from the author (2025)

Figures 14 and 15 show the evolution of precision and recall results throughout the training epochs, where the results remained stable, achieving strong classification performance and maintaining a balance between recall and precision, which is essential for reliable anomaly detection in network data.

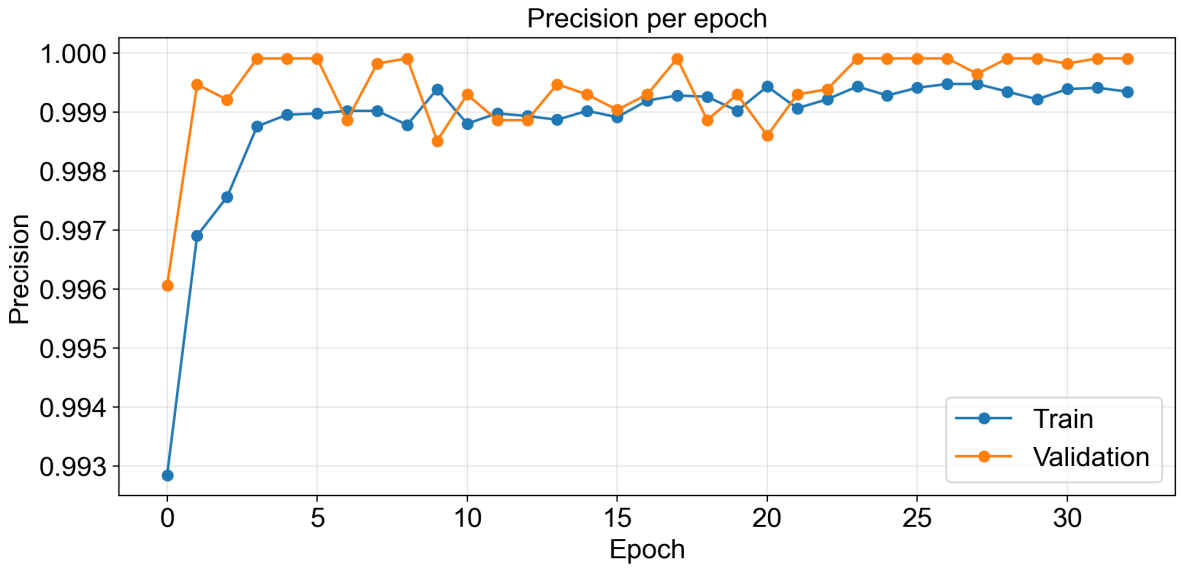
Figure 16 shows the accuracy per epoch, presenting results very close to 1, demonstrating that the model is quite efficient in detecting attacks correctly in samples classified as accurate.

The values obtained demonstrate that creating images from network data extracted from the CIC-IDS2018 dataset is a highly effective method for detecting attacks. After all, the model presented has a low error rate and a high hit rate.

4.2.2 Using Grad-CAM to explain the results

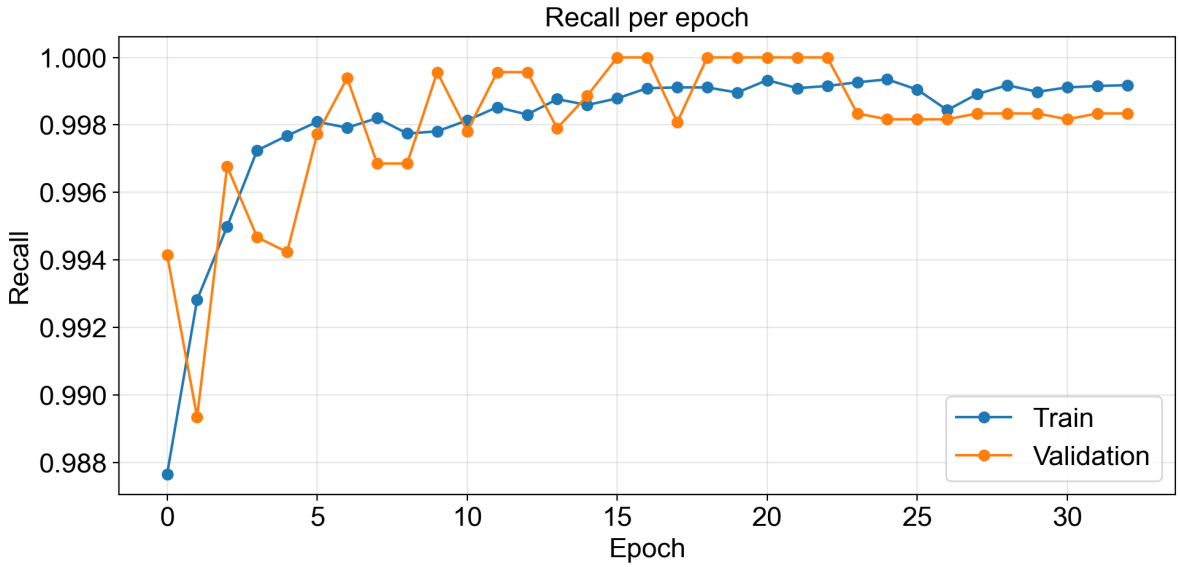
Figure 17 presents an example of the Grad-CAM visual explanations generated for a network traffic sample classified as an attack. Figure 17(a) shows the original RGB representation of the network flow, obtained from the tabular features of the CIC-IDS-2018 dataset. Figure 17(b) displays the Grad-CAM heatmap, where warmer colors indicate regions with a

Figure 14 – Precision per epoch.



Source: from the author (2025)

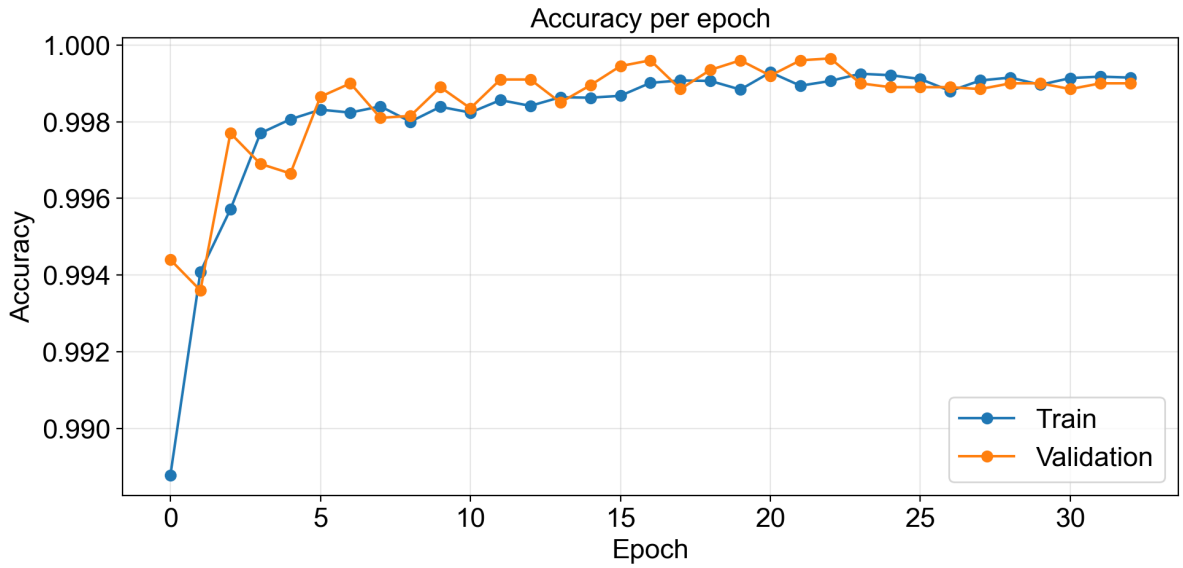
Figure 15 – Recall per epoch.



Source: from the author (2025)

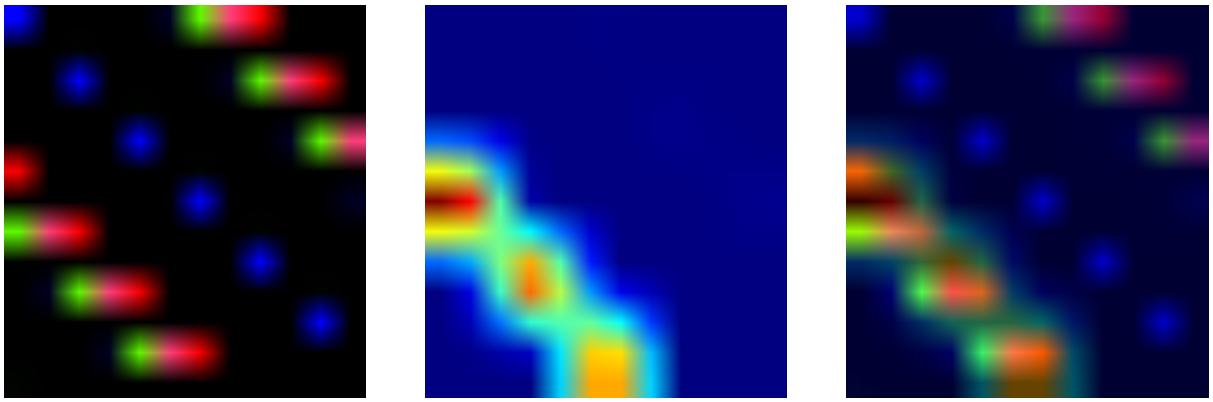
higher contribution to the CNN's classification decision. Finally, Figure 17(c) presents the superimposed image, where the heatmap is blended with the original RGB input to highlight the most influential features in their spatial context.

Figure 16 – Accuracy per epoch.



Source: from the author (2025)

Figure 17 – Example of Grad-CAM visual explanations for a network traffic image classified as an attack.



(a) Original image (b) Grad-CAM heatmap (c) Superimposed image

Source: from the author (2025)

From this visualization, it can be observed that the CNN focuses on specific localized regions of the feature-mapped image when predicting an attack. These high-activation areas correspond to feature clusters in the original network flow data that may represent anomalous patterns, such as abnormal packet sizes, unusual flow durations, or atypical byte distributions. The ability to spatially identify such regions provides a valuable interpretability layer to the model, enabling security analysts to validate detection results and gain insights into the underlying characteristics of malicious traffic.

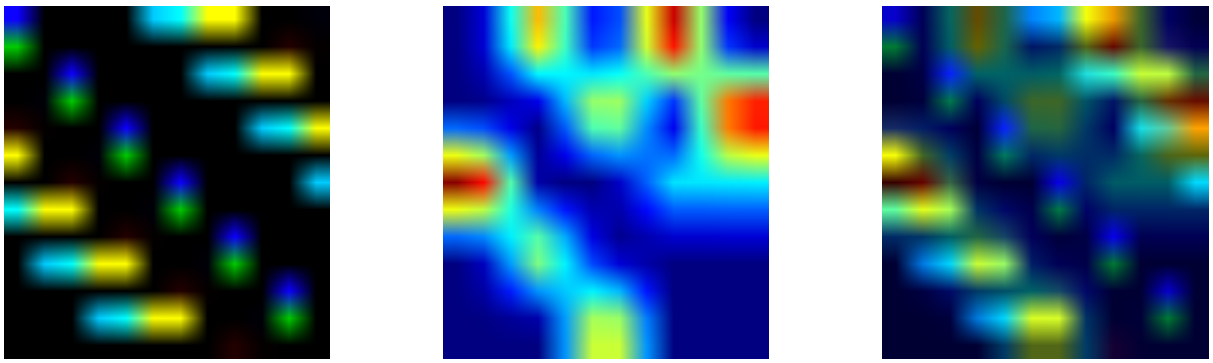
In this specific case, Grad-CAM highlights a diagonal region as the most influential for detecting the attack class, suggesting that the CNN convolutional filters capture structured correlations between consecutive features in the input representation. The resulting visualization

demonstrates the interpretability of the proposed pipeline, allowing us to verify that the network does not rely on spurious noise but rather on feature regions that are semantically relevant for distinguishing normal and anomalous flows. This result strengthens the evidence that the adopted CNN model generalizes to the underlying patterns of malicious traffic.

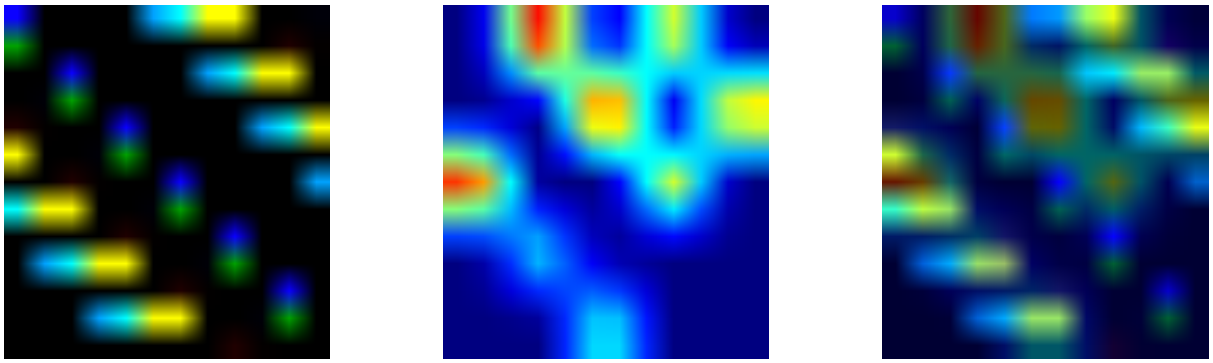
Figure 18 presents four sets of figures that were classified as attacks. These figures display the original images, heatmaps, and overlays, allowing for the visualization of the proportion of the image used in the classification.

Now, Figure 19 shows four sets of images that were classified as regular traffic. Additionally, we can find the original images, the heatmaps, and the superimposed images, which allow us to illustrate the size of the image used to classify it as a normal flow.

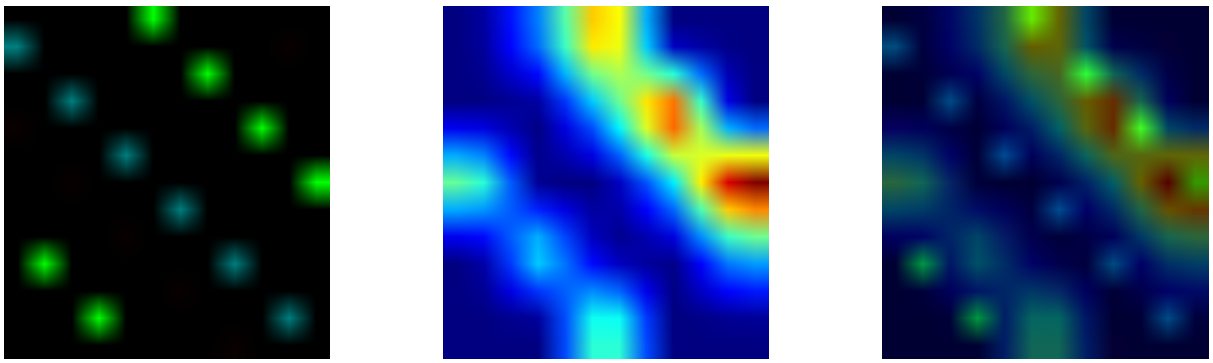
Figure 18 – Grad-CAM visual explanations for network traffic classified as attacks. Each row shows: (a) Original image; (b) Heatmap; (c) Superimposed visualization.



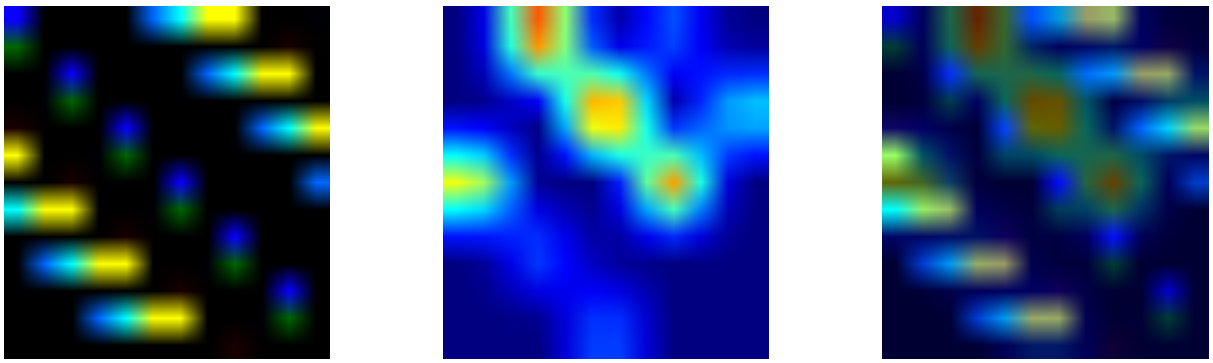
(a) Original image (b) Grad-CAM heatmap (c) Superimposed image



(d) Original image (e) Grad-CAM heatmap (f) Superimposed image



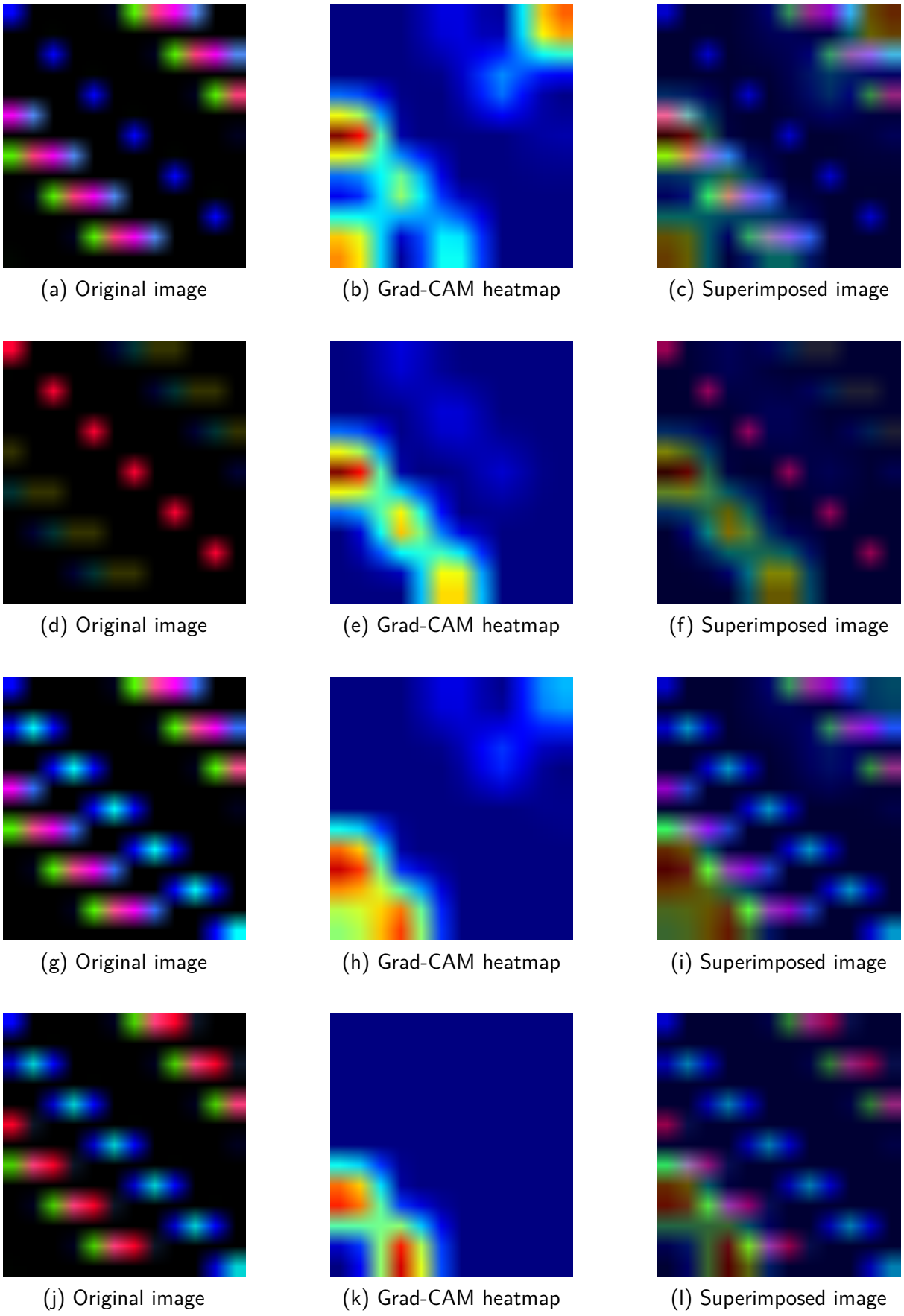
(g) Original image (h) Grad-CAM heatmap (i) Superimposed image



(j) Original image (k) Grad-CAM heatmap (l) Superimposed image

Source: from the author (2025)

Figure 19 – Grad-CAM visual explanations for traffic images classified as benign traffic. Each row shows: (a) original image; (b) Grad-CAM heatmap; (c) superimposed visualization.



(a) Original image (b) Grad-CAM heatmap (c) Superimposed image

(d) Original image (e) Grad-CAM heatmap (f) Superimposed image

(g) Original image (h) Grad-CAM heatmap (i) Superimposed image

(j) Original image (k) Grad-CAM heatmap (l) Superimposed image

Source: from the author (2025)

4.2.3 Grad-CAM Quantitative Analysis

To better understand the discriminative behavior of the CNN when applied to network traffic images, a quantitative analysis of the Grad-CAM heatmaps was conducted. For each of the 1000 images (410 benign and 590 attacks), two measures were extracted: (i) the **activated area**, defined as the proportion of pixels with normalized intensity greater than or equal to 0.8, and (ii) the **mean intensity**, defined as the average activation value across the entire heatmap.

Table 7 summarizes the aggregated results by class. The analysis shows that benign flows exhibited an average activated area of 3.83% of the pixels, whereas attack flows exhibited a slightly larger area of 3.98%. Although numerically close, this slight increase suggests that, on average, attack-related flows tend to produce slightly broader regions of activation. However, the Mann–Whitney U test indicated that this difference was not statistically significant ($U = 118182.0$, $p = 0.527$). This result highlights that the size of the activated region alone may not be a reliable marker of malicious traffic when analyzed globally.

Table 7 – Quantitative Grad-CAM statistics by class using threshold $t = 0.8$ (N=1000; 410 benign, 590 attack).

Class	**N**	**Activated area (mean ± SD)**	**Mean intensity (mean ± SD)**
Benign	410	0.0383 ± 0.0039	0.1871 ± 0.0167
Attack	590	0.0398 ± 0.0337	0.1885 ± 0.1037

Source: from the author (2025)

Note. Activated area is the proportion of pixels with normalized activation ≥ 0.8 in each heatmap. Mann–Whitney U tests (two-sided): activated area $U = 118182.0$, $p = 0.527$; mean intensity $U = 109264.5$, $p = 0.009$.

With respect to the mean intensity of activations, benign flows presented an average value of 0.1871, while attack flows reached 0.1885. While the absolute difference is slight, the variability of attack activations was much larger (standard deviation 0.1037 compared to 0.0167 for benign). This pattern suggests that malicious flows tend to generate heatmaps with stronger and more heterogeneous activation patterns, reflecting the diversity of anomalies present in various attack types. Notably, the Mann–Whitney U test revealed this difference to be statistically significant ($U = 109264.5$, $p = 0.009$). Therefore, although the mean values are numerically similar, the greater dispersion among attack samples implies that CNNs are sensitive to a broader spectrum of discriminative features when processing malicious traffic.

Taken together, these results support the hypothesis that Grad-CAM heatmaps capture meaningful differences between benign and malicious flows, even when such differences are subtle in absolute terms. The fact that attack images exhibit both slightly larger activated areas and more intense activations, albeit with high variability, suggests that the CNN is effectively attending to regions of the image that encode attack-specific network behaviors.

This standard reinforces the feasibility of representing network traffic as images and applying visual explainability tools, as the resulting maps provide not only a classification output but also evidence of where the model focuses its attention to distinguish between attacks and benign flows.

5 Conclusion

In today's connected world, accurately identifying attacks on computer networks is a critical task. The more accurate threat detection is, the more secure these networks will be.

In the first scenario, grayscale images were created from data from the NSL-KDD dataset. Good results were obtained in attack detection using a Convolutional Neural Network (CNN).

In the second scenario, this thesis investigated whether network attacks can be effectively detected by transforming network data into images and learning discriminative patterns with a CNN. We proposed a compact $13\times12\times3$ RGB representation that maps salient traffic descriptors into spatially organized pixel neighborhoods, enabling the CNN to exploit local correlations typically overlooked by purely tabular learners. Beyond raw accuracy, we prioritized explainability, generating class-discriminative explanations with Grad-CAM to reveal which image regions most influenced the model's decisions.

A pipeline was developed to convert network data into small RGB images while preserving local semantic relationships among features. Then, a CNN architecture and training setup tailored to this low-resolution regime are achieved, enabling robust detection of attack or benign patterns.

The image formulation consistently enabled the CNN to capture structured, class-specific signatures. Empirically, the approach delivered strong results across held-out data, indicating that spatially arranging features into a 2D manifold can be an effective inductive bias for network security tasks.

Even in this scenario, the Grad-CAM was applied, an explainability technique particularly useful when dealing with images. A heatmap was generated for the image to highlight the areas that had the most significant influence on the detection of attacks or benign traffic.

It is essential to emphasize here that the heatmaps presented are not intended to infer which feature would be most important in detecting an anomaly or not, but more than that, to prove that transforming network data into images allows the CNN to alter the feature map of an image through its convolutional filters and extract information that the human eyes are not able to identify.

In summary, the analysis of Grad-CAM activations demonstrated that, despite similar mean intensities between benign and attack flows, the latter exhibited substantially higher variability, indicating more heterogeneous and expressive activation patterns. The Mann-Whitney U test confirmed this difference, suggesting that the CNN model responds to a broader and more diverse set of discriminative features in malicious traffic. These findings validate that

Grad-CAM heatmaps effectively capture meaningful distinctions between benign and attack behaviors, supporting the representation of network flows as images and reinforcing the role of visual explainability methods in interpreting deep learning models for network anomaly detection.

This thesis provides evidence that *image-based modeling of network telemetry*, combined with a compact CNN and *Explainable AI*, is a viable and interpretable approach to intrusion detection. The association between detection performance and visual explanations enables expert trust, facilitates model governance, and points to a research program in which predictive accuracy and transparency advance together. By treating telemetry as a structured visual signal and opening the model's reasoning with *X*AI, we move one step closer to deployable, auditable, and resilient cyber-defense systems.

Future research will focus on integrating transfer learning strategies into the proposed CNN-based framework to improve anomaly detection performance and generalization further. By leveraging pre-trained convolutional architectures that have learned rich feature hierarchies from large-scale datasets, such as ImageNet, it should be possible to transfer low-level spatial representations and fine-tune higher-level layers for the specific characteristics of network traffic images.

Additionally, combining transfer learning with explainable AI methods, such as Grad-CAM, could provide deeper insights into how pre-trained features adapt to the network domain, helping to identify whether learned filters retain semantic relevance when applied to cybersecurity contexts.

Such extensions are expected to advance both the accuracy and interpretability of image-based models for network intrusion detection.

6 Published and Submitted articles

Here, we can find two published papers related to this thesis from the author. The full papers can be read in the Appendix A.

HERNANDES JUNIOR, PAULO R. GALEGO ; SCHERER, RAFAL ; JANUARIO, LUCAS B. ; RODRIGUES, DOUGLAS ; PAPA, JOAO P. ; COSTA, KELTON A. P. . **From Network Package Flow to Images: How to Accurately Detect Anomalies in Computer Networks**. In: 2022 29th International Conference on Systems, Signals and Image Processing (IWSSIP), 2022, Sofia. 2022 29th International Conference on Systems, Signals and Image Processing (IWSSIP), 2022. p. 1. **doi: http://dx.doi.org/10.1109/IWSSIP55020.2022.9854461**. Qualis CAPES A3

HERNANDES JUNIOR, PAULO R. GALEGO ; FLORET, CAMILA P. ; CARDOZO DE ALMEIDA, KATIA F. ; DA SILVA, VINICIUS CAMARGO ; PAPA, JOAO PAULO ; PONTARA DA COSTA, KELTON A. **Phishing Detection Using URL-based XAI Techniques**. In: 2021 IEEE Symposium Series on Computational Intelligence (SSCI), 2021, Orlando. 2021 IEEE Symposium Series on Computational Intelligence (SSCI), 2021. p. 01. **doi: http://dx.doi.org/10.1109/SSCI50451.2021.9659981**. Qualis CAPES A2

Submitted:

HERNANDES JUNIOR, PAULO R. GALEGO; INAYAT; A.; PONTARA DA COSTA, KELTON A. **From Data Packets to Vision: Explainable Deep Learning for Intrusion Detection in Computer Networks**. Submitted to *IEEE Transactions on Information Forensics and Security (TIFS)*, Nov. 2025. Under review. Qualis CAPES A1

Bibliography

ABADI, M.; BARHAM, P.; CHEN, J.; CHEN, Z.; DAVIS, A.; DEAN, J.; DEVIN, M.; GHEMAWAT, S.; IRVING, G.; ISARD, M. et al. Tensorflow: A system for large-scale machine learning. In: *USENIX symposium on operating systems design and implementation (OSDI 16)*. [S.l.: s.n.], 2016. p. 265–283.

ABIODUN, O. I.; JANTAN, A.; OMOLARA, A. E.; DADA, K. V.; MOHAMED, N. A.; ARSHAD, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, v. 4, n. 11, p. e00938, 2018. ISSN 2405-8440. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2405844018332067>.

ADEBAYO, J.; GILMER, J.; MUELLY, M.; GOODFELLOW, I.; HARDT, M.; KIM, B. Sanity checks for saliency maps. *Advances in neural information processing systems*, v. 31, 2018.

AHMAD, Z.; KHAN, A. S.; AQEEL, S.; JULAIHI, A. A.; TARMIZI, S.; ANNUAR, N.; HABEEB, M. S. S-ads: Spectrogram image-based anomaly detection system for iot networks. In: *2022 Applied Informatics International Conference (AiIC)*. [S.l.: s.n.], 2022. p. 105–110.

ALBASIR, A.; NAIK, K.; MANZANO, R.; SHAH, P.; NAIK, N. A strongly non-intrusive methodology to monitor and detect anomalous behaviour of wireless devices. In: *2020 IEEE International Symposium on Systems Engineering (ISSE)*. [S.l.: s.n.], 2020. p. 1–8.

BAJAJ, K.; ARORA, A. Improving the intrusion detection using discriminative machine learning approach and improve the time complexity by data mining feature selection methods. *International Journal of Computer Applications*, v. 76, p. 5–11, 08 2013.

BERTALANIC, B.; YETGIN, H.; CERAR, G.; FORTUNA, C. A deep learning model for anomalous wireless link detection. In: *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. [S.l.: s.n.], 2021. p. 265–270.

BP, V. K.; SM, K.; LV, P. Deep machine learning based usage pattern and application classifier in network traffic for anomaly detection. In: *2023 International Conference on Advances in Electronics, Communication, Computing and Intelligent Information Systems (ICAECIS)*. [S.l.: s.n.], 2023. p. 50–54.

CARRIQUIRY, A.; HOFMANN, H.; TAI, X. H.; VANDERPLAS, S. Machine learning in forensic applications. *Significance*, v. 16, n. 2, p. 29–35, 2019. Disponível em: <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1740-9713.2019.01252.x>.

CHEN, Z.; HE, K.; LI, J.; GENG, Y. Seq2img: A sequence-to-image based approach towards ip traffic classification using convolutional neural networks. In: *2017 IEEE International Conference on Big Data (Big Data)*. [S.l.: s.n.], 2017. p. 1271–1276.

CHOI, Y.; LEE, J.; CHOI, S.; KIM, J.; KIM, I. Transmitted file extraction and reconstruction from network packets. In: . [S.l.: s.n.], 2015. p. 164–165.

FEJZA, D.; LAMBERT, A. Bgp anomaly detection by the mean of updates projection and spatio-temporal auto-encoding. In: *2022 12th International Workshop on Resilient Networks Design and Modeling (RNDM)*. [S.l.: s.n.], 2022. p. 1–8.

FOX, G. T.; BOPPANA, R. V. On early detection of anomalous network flows. *IEEE Access*, v. 11, p. 68588–68603, 2023.

GUIDOTTI, R.; MONREALE, A.; RUGGIERI, S.; TURINI, F.; GIANNOTTI, F.; PEDRESCHI, D. A survey of methods for explaining black box models. *ACM Computing Surveys*, v. 51, n. 5, 2018. Disponível em: <https://dl.acm.org/doi/10.1145/3236009>.

GUNNING, D. Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, v. 2, p. 2, 2017.

HAIRAB, B. I.; ELSAYED, M. S.; JURCUT, A. D.; AZER, M. A. Anomaly detection based on cnn and regularization techniques against zero-day attacks in iot networks. *IEEE Access*, v. 10, p. 98427–98440, 2022.

HAYKIN, S. *Neural networks and learning machines, 3/E*. [S.l.]: Pearson Education India, 2010.

HOWARD, A. G.; ZHU, M.; CHEN, B.; KALENICHENKO, D.; WANG, W.; WEYAND, T.; ANDREETTO, M.; ADAM, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017. Disponível em: <http://arxiv.org/abs/1704.04861>.

ITPro. *Global cybersecurity spending is going to hit $213 billion in 2025 — here's what's driving investment*. 2025. Disponível em: <https://www.itpro.com/security/global-cybersecurity-spending-is-going-to-hit-usd213-billion-in-2025-heres-whats-driving-investment>.

KARAM, R.; SALOMON, M.; COUTURIER, R. A comparative study of deep learning architectures for detection of anomalous ads-b messages. In: *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*. [S.l.: s.n.], 2020. v. 1, p. 241–246.

KAUR, G.; LASHKARI, A. H.; RAHALI, A. Intrusion traffic detection and characterization using deep image learning. In: *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*. [S.l.: s.n.], 2020. p. 55–62.

KAUR, R.; GABRIJELčIč, D.; KLOBUčAR, T. Artificial intelligence for cybersecurity: Literature review and future research directions. *Information Fusion*, v. 97, p. 101804, 2023. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1566253523001136>.

KEELE, S. et al. *Guidelines for performing systematic literature reviews in software engineering*. [S.l.]: Technical report, ver. 2.3 ebse technical report. ebse, 2007.

KERAS, F. et al. Deep learning for humans. *github*, 2015.

KHAN, A.; SOHAIL, A.; ZAHOORA, U.; QURESHI, A. S. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, Springer, v. 53, n. 8, p. 5455–5516, 2020.

KHAN, A. S.; AHMAD, Z.; ABDULLAH, J.; AHMAD, F. A spectrogram image-based network anomaly detection system using deep convolutional neural network. *IEEE Access*, v. 9, p. 87079–87093, 2021.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, IEEE, v. 86, n. 11, p. 2278–2324, 1998.

LECUN, Y.; KAVUKCUOGLU, K.; FARABET, C. Convolutional networks and applications in vision. In: *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*. [S.l.]: IEEE, 2010. p. 253–256.

LEE, H.; NADEEM, M. Iot anomaly detection with sound-based spectral analysis and lightweight model. In: *2025 IEEE International Conference on Big Data and Smart Computing (BigComp)*. [S.l.: s.n.], 2025. p. 315–322.

LIU, H.; WANG, H. Real-time anomaly detection of network traffic based on cnn. *Symmetry*, v. 15, n. 6, 2023. Cited by: 11; All Open Access, Gold Open Access. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85163720346&doi=10.3390%2fsym15061205&partnerID=40&md5=6993f9963257b2bec6c57c47bfda0d3e>.

LIU, X. An image classification network for network traffic representation learning. In: *2021 6th International Symposium on Computer and Information Processing Technology (ISCIPT)*. [S.l.: s.n.], 2021. p. 358–361.

MAHBOD, T.; EBRAHIM, B.; WEI, L.; GHORBANI, A. A. A detailed analysis of the kdd cup 99 data set. In: IEEE. *IEEE symposium on computational intelligence for security and defense applications*. [S.l.], 2009. p. 1–6.

MANN, H. B.; WHITNEY, D. R. On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics*, Institute of Mathematical Statistics, v. 18, n. 1, p. 50–60, 1947.

MEDEIROS, L. F. de. *Inteligência artificial aplicada: Uma abordagem introdutória*. 1st. ed. Curitiba, PR, Brasil: InterSaberes, 2018.

MEZINA, A.; BURGET, R.; TRAVIESO-GONZáLEZ, C. M. Network anomaly detection with temporal convolutional network and u-net model. *IEEE Access*, v. 9, p. 143608–143622, 2021.

MOHAMMED, A. A.; ALI, D. Packet features extractor for network security systems: Design and implementation. *International Journal of Engineering and Innovative Technology (IJEIT)*, v. 3, p. 225–231, 04 2014.

MONDAL, B. Artificial intelligence: state of the art. *Recent Trends and Advances in Artificial Intelligence and Internet of Things*, Springer, p. 389–425, 2020.

MOORE, M. R.; VANN, J. M. Anomaly detection of cyber physical network data using 2d images. In: *2019 IEEE International Conference on Consumer Electronics (ICCE)*. [S.l.: s.n.], 2019. p. 1–5.

NGO, D.-M.; LIGHTBODY, D.; TEMKO, A.; MURPHY, C. C.; POPOVICI, E. A scalable security approach in iot networks: Smart contracts and anomaly-based ids for gateways using hardware accelerators. *IEEE Access*, v. 12, p. 159519–159533, 2024.

NORVIG, P.; RUSSELL, S. *Inteligência artificial: Tradução da 3a Edição*. Elsevier Brasil, 2014. ISBN 9788535251418. Disponível em: <https://books.google.com.br/books?id=BsNeAwAAQBAJ>.

OLA-OBAADO, S.; SULEIMAN, M. A. Anomaly-based network intrusion detection using transfer learning. In: *2023 2nd International Conference on Multidisciplinary Engineering and Applied Science (ICMEAS)*. [S.l.: s.n.], 2023. v. 1, p. 1–5.

OLADIPO, F.; OGBUJU, E.; ALAYESANMI, F. S.; MUSA, A. E. The state of the art in machine learning-based digital forensics. *Available at SSRN 3668687*, 2020.

OZKAN-OKAY, M.; AKIN, E.; ASLAN,
"O.; KOSUNALP, S. A comprehensive survey: Evaluating the efficiency of artificial intelligence and machine learning techniques on cyber security solutions. *IEEE Access*, v. 12, p. 12229–12256, 2024.

RAMSUNDAR, B. *Tensorflow for Deep Learning: From Linear Regression to Reinforcement Learning*. [S.l.]: O'Reilly Media, 2018. ISBN 9781491980453,1491980451.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. "why should I trust you?": Explaining the predictions of any classifier. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. [S.l.: s.n.], 2016. p. 1135–1144.

SALEM, A. H.; AZZAM, S. M.; EMAM, O. E.; ABOHANY, A. A. Advancing cybersecurity: a comprehensive review of ai-driven detection techniques. *Journal of Big Data*, v. 11, 2024. Article 105. Disponível em: <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-024-00957-y>.

SANA, L.; NAZIR, M. M.; YANG, J.; HUSSAIN, L.; CHEN, Y.-L.; KU, C. S.; ALATIYYAH, M.; ALATEYAH, S. A.; POR, L. Y. Securing the iot cyber environment: Enhancing intrusion anomaly detection with vision transformers. *IEEE Access*, v. 12, p. 82443–82468, 2024.

SELVARAJU, R. R.; COGSWELL, M.; DAS, A.; VEDANTAM, R.; PARIKH, D.; BATRA, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2017. p. 618–626.

SHAFFI, A.; CHACKO, J. V.; ELIYAN, G.; BALAJI, S. A study on anomaly-based intrusion detection systems employing supervised deep learning techniques. In: *2024 8th International Conference on Inventive Systems and Control (ICISC)*. [S.l.: s.n.], 2024. p. 366–370.

SHARAFALDIN, I.; LASHKARI, A. H.; GHORBANI, A. A. et al. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, v. 1, n. 2018, p. 108–116, 2018.

SIKOS, L. F. Packet analysis for network forensics: A comprehensive survey. *Forensic Science International: Digital Investigation*, v. 32, p. 200892, 2020. ISSN 2666-2817. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1742287619302002>.

SINHA, A.; TAYLOR, M.; SRIRAMA, N.; MANIKAS, T.; LARSON, E. C.; THORNTON, M. A. Industrial control system anomaly detection using convolutional neural network consensus. In: *2021 IEEE Conference on Control Technology and Applications (CCTA)*. [S.l.: s.n.], 2021. p. 693–700.

STERNBERG, R. J. *Psicologia Cognitiva*. 5th. ed. São Paulo, SP, Brasil: CENGAGE LEARNING NACIONAL, 2010.

TANG, Y.; YE, L.; HU, L. Application of network anomaly detection algorithm based on deep learning in information communication. In: *2024 International Conference on Power, Electrical Engineering, Electronics and Control (PEEEC)*. [S.l.: s.n.], 2024. p. 538–543.

ULLAH, I.; MAHMOUD, Q. H. Design and development of a deep learning-based model for anomaly detection in iot networks. *IEEE Access*, v. 9, p. 103906–103926, 2021.

VERMA, R. D.; KESERWANI, P. K.; JAIN, V. K. A cnn approach to detect the anomalies in bgp traffic. In: *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. [S.l.: s.n.], 2024. p. 1–6.

WANG, S.; ZHANG, Y. Grad-cam: understanding ai models. *Comput. Mater. Contin*, v. 76, n. 2, p. 1321–1324, 2023.

WANG, W.; ZHU, M.; ZENG, X.; YE, X.; SHENG, Y. Malware traffic classification using convolutional neural network for representation learning. In: *2017 International Conference on Information Networking (ICOIN)*. [S.l.: s.n.], 2017. p. 712–717.

WITTEN, I. H.; FRANK, E.; HALL, M. A. *Data Mining: Practical Machine Learning Tools and Techniques*. 3rd. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN 0123748569, 9780123748560.

WU, H.; HE, J.; TöMöSKöZI, M.; ZHANG, J.; FITZEK, F. H. P. You only hear once: Lightweight in-network ai design for multi-object anomaly detection. In: *2022 IEEE 21st Mediterranean Electrotechnical Conference (MELECON)*. [S.l.: s.n.], 2022. p. 1217–1222.

WU, K.; CHEN, Z.; LI, W. A novel intrusion detection model for a massive network using convolutional neural networks. *IEEE Access*, v. 6, p. 50850–50859, 2018.

WYK, F. van; WANG, Y.; KHOJANDI, A.; MASOUD, N. Real-time sensor anomaly detection and identification in automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, v. 21, n. 3, p. 1264–1276, 2020.

XU, S.; HAN, X.; TIAN, T.; JIANG, B.; LU, Z.; ZHANG, C. Few-shot network traffic anomaly detection based on siamese neural network. In: . [s.n.], 2023. v. 2023-May, p. 3012 – 3017. Cited by: 3. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85178283262&doi=10.1109%2fICC45041.2023.10279011&partnerID=40&md5=cf7d25215aa8139e4afce96e6b4f81ae>.

YAQOOB, S.; HUSSAIN, A.; SUBHAN, F.; PAPPALARDO, G.; AWAIS, M. Deep learning based anomaly detection for fog-assisted iovs network. *IEEE Access*, v. 11, p. 19024–19038, 2023.

YOSHIMURA, N.; KUZUNO, H.; SHIRAISHI, Y.; MORII, M. Doc-ids: A deep learning-based method for feature extraction and anomaly detection in network traffic. *Sensors*, v. 22, n. 12, 2022. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/22/12/4405>.

ZHOU, P. Payload-based anomaly detection for industrial internet using encoder assisted gan. In: *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*. [S.l.: s.n.], 2020. p. 669–673.

ZHOU, P.; LI, C.; CHEN, C.; WU, D.; FEI, M. P^3ad: Privacy-preserved payload anomaly detection for industrial internet of things. *IEEE Transactions on Network and Service Management*, p. 1–1, 2023.

Appendix

APPENDIX A – *Published papers*

A.1 From Network Package Flow to Images: How to Accurately Detect Anomalies in Computer Networks

This paper was submitted to the IEEE Symposium Series on Computational Intelligence International Conference on Systems, Signals, and Image Processing (IWSSIP) and demonstrates the whole process from data conversion into images to anomaly detection.

Proc. 29th International Conference on Systems, Signals and Image Processing "IWSSIP 2022", June 01 - 03, 2022, Sofia, Bulgaria

From Network Package Flow to Images: How to Accurately Detect Anomalies in Computer Networks

Paulo R. Galego Hernandes Junior
Department of Computing
São Paulo State College of Technology, Brazil
paulo.galego@fatecourinhos.edu.br

Rafał Scherer
Czestochowa University of Technology, Poland
rafal.scherer@pcz.pl

Lucas B. Januario, Douglas Rodrigues
João P. Papa, Kelton A. P. Costa
Department of Computing
São Paulo State University, Brazil
{lucas.biaggi, joao.papa, d.rodrigues, kelton.costa}@unesp.br

***Abstract*—With the increasing adoption of data networks, attacks will cause more and more damage, including financial losses to organizations and governments. Artificial intelligence, combined with Deep Learning techniques such as Convolutional Neural Networks (CNN), helps organizations detect early attacks on their infrastructures. A CNN is a specialist system in extracting pieces of information about images. This work presents a method that converts data from network connections into images. We use a CNN with three convolutional layers, which will use the generated images as input and extract features based on their weights to perform attack detection. We reached promising results detecting attacks using the NSL-KDD dataset as the input to create the images.**

***Index Terms*—Package Flow, Computer Networks, Security, Anomaly Detection, Deep Learning.**

I. Introduction

Nowadays, technologies that exploit artificial intelligence are present in everyone's daily lives. From navigation and traffic apps [1], [2] to systems that recommend movies according to your profile [3], everyone might adopt artificial intelligence algorithms at some point.

Within the broad spectrum of techniques that make up the field of artificial intelligence, deep learning appears to exhibit excellent results in several applications helping to perform more simply, for example, intelligent assistants that identify diseases and assist doctors in diagnosis. Such assistants can also support and identify unauthorized access to a network or even carry out packet investigations in network forensics. Accordingly, Sikos [4] defines that in packet analysis, the content of the captured packet must be detailed enough to reproduce the network traffic at a given moment to find data violations, unauthorized access, malware, possible intrusion attempts and provide legal evidence against a suspect in a judicial process.

Abdullah [5] developed a new method to design an offline intrusion detection system. It consists of performing pattern matching using Simulink Image Block Matching (IBM) and classifying it as normal or abnormal through Embedded Matlab Function (EMF). Choi et al. [6] propose to reconstruct malwares from network packets to analyze them. The transmitted file reconstruction technique is based on file type signature and network protocol parsing, which can reconstruct various types of files from network packets.

Zhitang et al. [7] introduce a new online IP traffic classification method to get around a limitation where the vast majority of IP traffic classification methods are performed offline. This method named Seq2Img consists of a compact non-parametric kernel-embedding to generate images starting from stream sequences and later classify them with a CNN. Moreover, Basumallik and Eftekharnejad [8] address a class of False Data Injection (FDIA) attacks by extracting and classifying multivariate time-series signals from corrupted phasor measurement units (PMU) with a CNN.

In this context, the present work proposes to compare whether network packets converted into images and later classified using CNN are efficient in detecting network anomalies. It is worth mentioning that, as this is a pioneering work, the carried-out literature review presents only the closest works to the proposal of this article. Therefore, the main contributions of this work are:

- introduce a new method for converting NSL-KDD data into image;
- CNN employment for attack detection.

The remainder of this paper is organized as follows. Section II presents a brief background concerning convolutional

The authors are grateful to the São Paulo Research Foundation (FAPESP) grants 2013/07375-0, 2014/12236-1, 2021/05516-1, and 2019/07665-4, to the Brazilian National Council for Scientific and Technological Development (CNPq) grants 307066/2017-7 and 427968/2018-6, and Petrobras, Brazil grant 2017/00285-6 for their financial support.

978-1-6654-9578-3/22/$31.00

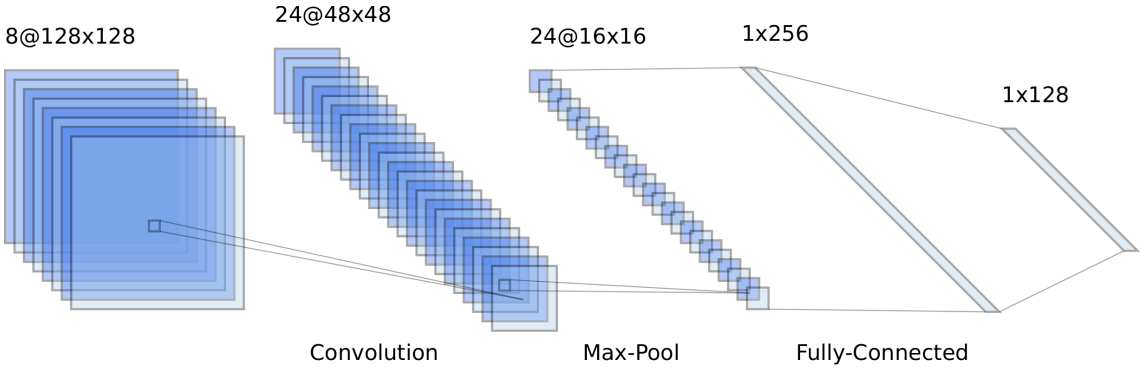


Fig. 1: Convolutional neural network - example architecture.

neural networks, while Section III discusses the methodology employed in this work. Section IV presents the experimental results and Section V states conclusions and future works.

II. Convolutional Neural Network

A Convolutional Neural Network (CNN) [9] is a class of artificial neural networks inspired by the organization of neurons in the visual cortex and their connectivity patterns. That is, how the brain of living beings process visual information. A CNN is a multi-layer architecture composed of an input layer, hidden layers, and an output layer. It is worth mentioning that at least one of the hidden layers must be a convolutional layer. Figure 1 demonstrates an example of an architecture composed of an input layer, a convolutional layer, a pooling layer, a fully connected layer, and finally, the output layer.

A. Convolution Layer

The convolution layer is responsible for capturing local features such as edges, curves, colors, lines, and so on. Thus, the number of local features learned by the convolutional layer is proportional to the number of filters. Given an input image $I \in \mathbb{R}^{W \times H}$ and a convolution filter $F \in \mathbb{R}^{K \times K}$, the convolution process that results in a feature map $A \in \mathbb{R}^{W \times H}$ is described as follows:

$$A = (I \otimes F)[W, H] = \sum_{i=0}^{W} \sum_{j=0}^{H} I[W - i, H - j]F[i, j], \quad (1)$$

where $\otimes$ stands for the convolution operator. Notice that the values of the convolutional filter are determined by the training process.

B. Pooling Layer

The pooling layer reduces the dimensionality of the feature map by partitioning the image into subregions and combining the pixels of each region into a representative value. Among the most used pooling strategies, we can mention the max and the average. The first summarizes the feature map by selecting the highest intensity pixel in the subregion, while the second summarizes the feature map by computing the average of the pixels of each subregion.

The pooling process is similar to the convolution process. Given a feature map $A \in \mathbb{R}^{W \times H}$, the image resulting from the pooling process will have the dimension $B \in \mathbb{R}^{(H-f+1)/s \times (W-f+1)}$, where f is the size of the pooling matrix and s is the stride length.

C. Fully-Connected Layer

In the fully connected layer, we have that all neurons connect to all neurons in the next layer. Each neuron corresponds to a non-linear function. The neurons apply a linear transformation to the data by computing the dot product between the input vector and the weight matrix, and then a non-linear transformation is applied using an activation function σ. The output of neuron j is given by

$$y_j(x) = \sigma(\sum_{i=1}^{m} w_{ji}x_i + w_{j0}) \quad (2)$$

where m is the size of the input vector.

III. Methodology

In this section, we describe the methodology adopted in this work, as well as, the dataset employed to validate the experimental phase.

A. Dataset

The NSL-KDD [10] is an Internet traffic dataset composed of records generated by an Intrusion Detection System (IDS). The train samples of the dataset represent 125,973 records, while the test set contains 22,544 records. Based on [11], which found that some attributes have no role in detection of attack, we decided to remove attribute 9 to have 40 features expressing the Internet traffic and two classes (attack or non-attack).

B. Architecture

For each record of the NSL-KDD dataset, we reshaped the feature vector into a matrix of size 10×4 representing an image. Table I depicts the arrangement of features in the new image. Important to note that the first feature of the vector will represent the first pixel of the image, the second feature will represent the second pixel, and so on.

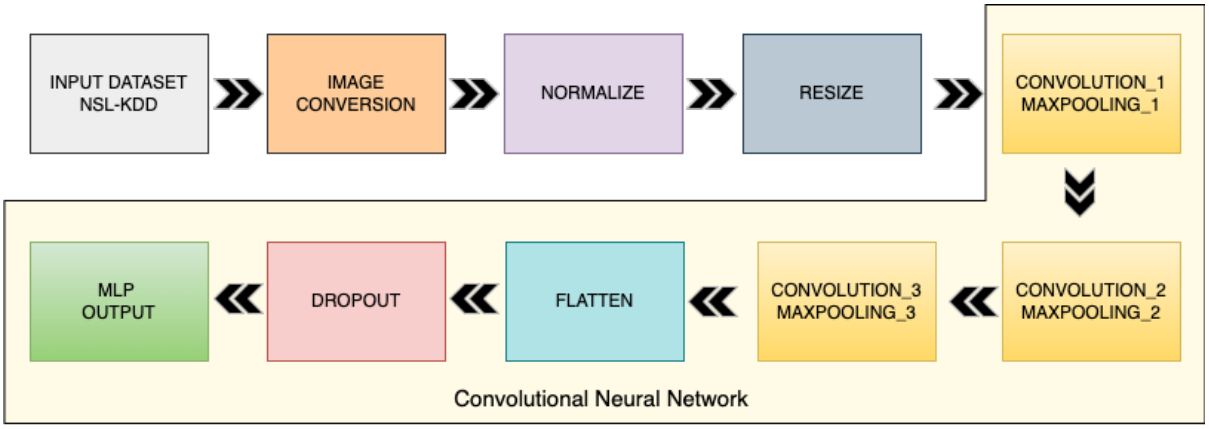


Fig. 2: Pipeline of the proposed methodology.

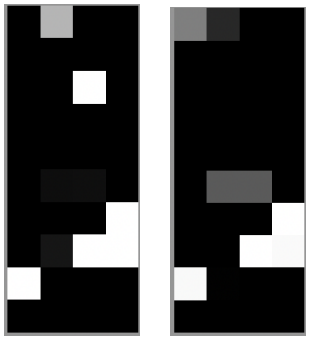
TABLE I: Arrangement of features in the image.

1	2	3	4
5	6	7	8
10	11	12	13
14	15	16	17
18	19	20	21
22	23	24	23
26	27	28	29
30	31	32	33
34	35	36	37
38	39	40	41

Then, this matrix has the range [0, 255] as follows

$$x = \sum_{j=1}^{4} \sum_{k=1}^{10} \frac{x_{jk} - 0}{255 - 0} \tag{3}$$

Therefore, the images have one pixel for each feature where the brightness corresponds to the feature's value. Figures 3a and 3b show examples of two records converted to figures.



(a) (b)

Fig. 3: Two images representing 40 features each one.

To classify the images, that is, to detect possible attacks, we built a CNN with the following architecture: three convolutional layers in which each layer with 32, 64, and 128 filters, respectively, of 3×3 size. For the experiments, we used a batch size of 64 samples, a learning rate of 1×10^{-3} and the Adam optimizer. Notice that to improve the performance of convolution layers, we resized to 28×28. Between each convolutional layer, we use a max-pooling layer. The activation function chosen was ReLU.

We use a flattening operation transforming the feature map into a new feature vector that will be used later by the fully connected layer. To reduce overfitting we employ a dropout layer of 0.5.

Finally, we use a fully connected layer with a sigmoid activation function to perform the classification. Figure 2 demonstrates the pipeline of the entire process from the record pre-processing to the classification into attack or non-attack.

IV. Experimental Results

For the experiments, we used 50% of the NSL-KDD dataset, corresponding to 62,986 images generated for training and 11,271 images for validation and testing. All examples were randomly selected from the dataset, which is already separated between training and testing data, so the data is entirely different.

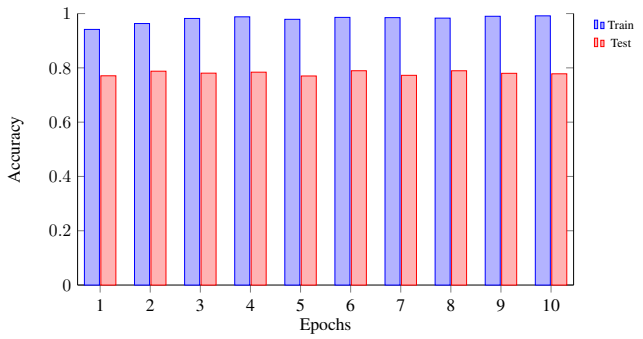
For CNN, we adopted the Python libraries TensorFlow [12], and Keras [13]. We also adopted the metrics accuracy, precision, recall, f-measure, and Area Under the Curve (AUC) to validate the results. Table II report the results concerning the mean of 100 epochs to training the CNN and the results on the test set.

One can observe that the accuracy reached 78% in the test samples, which despite not being the same as the training accuracy, shows that the new method achieves an acceptable use in detecting attacks.

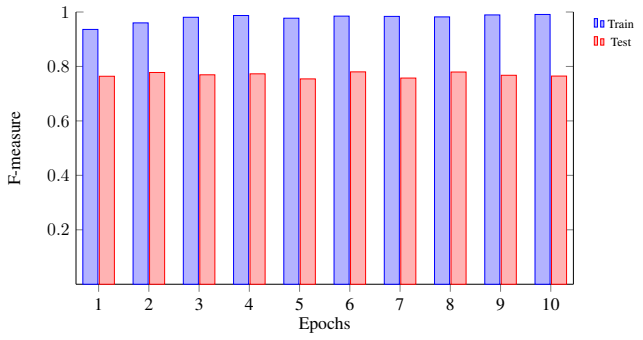
The values found in the precision metric are excellent, indicating that 96.73% of the samples that were classified as anomalies were. It means that we have a low rate of False

TABLE II: Results

Metric	Test Result	Train Result
Accuracy	78.25%	99.79%
Precision	96.73%	99.78%
Recall	64.03%	99.77%
F-Measure	77.05%	99.37%
AUC	0.823	0.999



(a) Accuracy



(b) F-measure

Fig. 4: Experimental results for 10 epochs.

Positives, indicating that the model is correctly predicting attacks.

However, the perceived value in Recall shows us that there is still considerable value in classifying samples reported as False Negative, i.e., the model did not classify that there was an attack, but it should have classified it. This low value shows us that we have to improve the model to reduce False Positives.

Although, the F-measure indicates an index of 77.05% of general predictions. It means that perhaps we have a reasonable rate for Precision and a low rate for Recall, we got good results overall.

The AUC value of 0.823 indicates a reasonable rate of identification of the hits of the classes, which means that the model is managing to separate attacks from legitimate traffic in a satisfactory way.

In Figure 4a, we can observe the values for 10 Epochs for accuracy in each Epoch, with values from train and test relatively close. Also, in Figure 4b, we can see the values from the f-measure in each of the 10 epochs.

Concerning the values obtained in the test phase, one can perceive that the model does not measure as high in the test set as it obtained from training, but the successes in detecting the attacks regarding the accuracy values show good prospects.

V. Conclusion and Future Works

We present a new model for network anomaly detection, which uses intrusion attack data from the NSL-KDD dataset to transform it into images. These images are submitted to a three-layer CNN, which extracts the best features from the images, and passes them on to a dense layer that separates the classes, between positive or negative for an anomaly.

The new technique demonstrated promising results with reasonable accuracy rates. The metrics used also showed the model's deficiencies, such as improving the rate of true positives and reducing false negatives.

For future work, changes are foreseen in the capture of images by the neural network, with better resizing the images and adjustments in the optimizers for better results. Also, we plan to use other datasets to show the model's efficiency.

References

[1] B. D. Renan and O. F. Santos, "Image classification system based on deep learning applied to the recognition of traffic signs for intelligent robotic vehicle navigation purposes," in *Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR)*, 2017, pp. 1–6.

[2] L. Matthias, F. Julian, B. Dennis, S. Florian, S. Roland, and L. P. A. Hendrik, "Robust deep-learning-based road-prediction for augmented reality navigation systems at night," in *IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 2016, pp. 1888–1895.

[3] Z. Shuai, Y. Lina, S. Aixin, and T. Yi, "Deep learning based recommender system: A survey and new perspectives," *ACM Computing Surveys*, vol. 52, no. 1, feb 2019.

[4] L. F. Sikos, "Packet analysis for network forensics: A comprehensive survey," *Forensic Science International: Digital Investigation*, vol. 32, p. 200892, 2020.

[5] A. M. Abdullah, "Designing of intrusion detection system based on image block matching," *International Journal of Computer and Communication Engineering*, vol. 2, pp. 605–607, 03 2013.

[6] Y. Choi, J. Lee, S. Choi, J. Kim, and I. Kim, "Transmitted file extraction and reconstruction from network packets," 10 2015, pp. 164–165.

[7] C. Zhitang, H. Ke, L. Jian, and G. Yanhui, "Seq2img: A sequence-to-image based approach towards ip traffic classification using convolutional neural networks," in *IEEE International Conference on Big Data (Big Data)*, 2017, pp. 1271–1276.

[8] S. Basumallik, R. Ma, and S. Eftekharnejad, "Packet-data anomaly detection in pmu-based state estimator using convolutional neural network," *International Journal of Electrical Power Energy Systems*, vol. 107, pp. 690–702, 12 2018.

[9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[10] T. Mahbod, B. Ebrahim, L. Wei, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *IEEE symposium on computational intelligence for security and defense applications*. Ieee, 2009, pp. 1–6.

[11] K. Bajaj and A. Arora, "Improving the intrusion detection using discriminative machine learning approach and improve the time complexity by data mining feature selection methods," *International Journal of Computer Applications*, vol. 76, pp. 5–11, 08 2013.

[12] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *USENIX symposium on operating systems design and implementation (OSDI 16)*, 2016, pp. 265–283.

[13] F. Keras *et al.*, "Deep learning for humans," *github*, 2015.

A.2 Phishing Detection Using URL-based XAI Techniques

This paper was submitted to the IEEE Symposium Series on Computational Intelligence (SSCI). It demonstrated the use of Explainable Artificial Intelligence to explain why some URLs were classified as legitimate or phishing. The paper focuses on detecting phishing using two explainable techniques, which are Local Interpretable Model-Agnostic Explanations and Explainable Boosting Machines.

Phishing Detection Using URL-based XAI Techniques

Paulo R. Galego Hernandes Jr.
Department of Information Security
Fatec Ourinhos
Ourinhos, Brazil
paulo.galego@fatecourinhos.edu.br

Camila P. Floret, Katia F. Cardozo de Almeida
Vinícius Camargo da Silva, João Paulo Papa
Kelton A. Pontara da Costa
Department of Computing
São Paulo State University
Bauru, Brazil
{camila.pellizon,vinicius.camargo}@unesp.br
k_dcardozo@yahoo.com.br
{joao.papa,kelton.costa}@unesp.br

2021 IEEE Symposium Series on Computational Intelligence (SSCI) | 978-1-7281-9048-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/SSCI50451.2021.9659981

***Abstract*—The Internet has been growing exponentially and expanding facilities, such as payments and online purchases. Likewise, the number of criminals using electronic devices to commit theft or hijacking of information has increased. Many scams still require interaction with the victim, who in many cases is persuaded to access a malicious link sent by email, which is classified as phishing. This technique is one of the biggest threats for users and one of the most efficient for criminals. Several studies show different sorts of protection using Artificial Intelligence, which despite being very efficient, do not describe the reasons for categorizing them or using a URL as phishing. This paper focuses on detecting phishing using explainable techniques, i.e., Local Interpretable Model-Agnostic Explanations and Explainable Boosting Machine, to lighten up new advances and future works in the area.**

***Index Terms*—phishing, machine learning, XAI, explainable, artificial intelligence**

I. Introduction

With the exponential growth of the Internet each year, the facilities that technology brings are expanded, such as a diversity of payment methods, online shopping, all-digital banks, and several other services. Such an increase in the use of the worldwide network for online shopping has brought the adversity of rising the number of victims of crimes committed by electronic means, with a growth proportional to the number of transactions carried out.

Several types of crimes are committed in the electronic environment, such as extortion, theft or hijacking of information, denial of service attacks, and many others. Most scams still require interaction with the victims to convince them, in many cases, to click on a link that will lead the attacker to his goal.

In this scenario, attacks using phishing are pretty efficient from the attacker's point of view. The phishing technique has an electronic message to make the victim access a malicious link, which this user believes to be legitimate. These attacks become more specialized and targeted when the victims click on the link because they believe in the message's veracity.

FAPESP supported this work under the grant #2014/12236-1, and CNPq supported this work under the grant #429003/2018-8.

Researchers have sought new forms of protection to reduce these attacks' impact, using Artificial Intelligence (AI) techniques. However, current machine learning models do not bring the transparency that some applications require. On the other hand, Explainable Artificial Intelligence makes it possible to understand why to classify a given data in that way, such as why classify or not a URL as phishing. Although some authors [1] [2] [3] proposed classifying URLs using machine learning, these methods left a lack transparency on how they made the decisions. Therefore, this work's primary contribution is to explain and detect URL Phishing using Local Interpretable Model-Agnostic Explanations (LIME) and Explainable Boosting Machines (EBM).

The article is organized as follows: Section 2 reviews some recent work in detecting phishing using Machine Learning techniques and publications presenting the Explainable AI models. Section 3 shows the experiments conducted by the authors, and Section 4 presents the results obtained and the models' explanations. Section 5 states conclusions and future works.

II. Related Works

Phishing attacks are growing every year. According to Verizon's 2020 report, [4], 88% of organizations have undergone a targeted phishing scam, with 96% of attacks sent via email. This section presents studies to make it possible to minimize the impact of phishing attacks and to use machine learning techniques. We will also discuss the motivation and potential behind Explainable Artificial Intelligence (XAI) techniques in the present work.

A. Machine learning to Phishing detection

Phishing is one of the biggest threats to network and computer security. According to Jain and Gupta [2], a criminal makes a false copy using the original law's contents to use his law enforcement to identify the changes between a trusted website and a phishing website. Figure 1 shows the life cycle of an attack established in the following order:

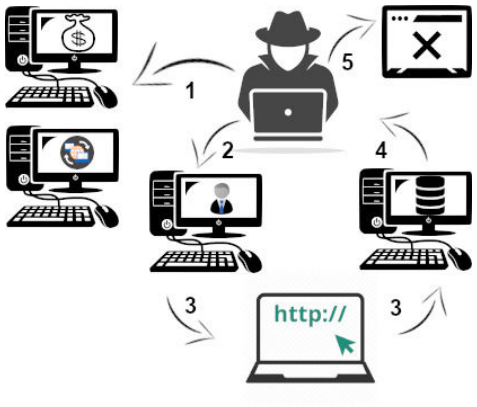


Fig. 1. Phishing attack sequence (adapted from [2]).

1) The cybercriminal makes a copy of the content of a legitimate website and develops the phishing website;
2) The phisher sends a phishing URL link to the user;
3) The user enters the link and fills in the data on the fraudulent website;
4) The infringer steals the user's information, and
5) The phisher deletes the fake web page.

Machine learning is a valuable tool in combating phishing. Unlike other features such as the blacklist, machine learning demonstrates high performance in detecting phishing using both supervised and unsupervised techniques. Abdelhamid et al. [3] presented an investigating of the applicability of machine learning techniques in detecting phishing describing its pros and cons. The experiments demonstrated that the Ridor [5] and eDRI [6] techniques were the most appropriate, with high accuracy rates.

In the work of Peng, Harris, and Sawa [8], the authors use Natural Language Processing (NLP) to process the text contained in the attacks' links, performing an analysis of the text in search of malicious content. The authors determined that the search should find four characteristics: malicious issues/commands, urgent tone, gene compliments, or malicious link, which in its detection uses a commercial tool [9]. If the latter is detected, the message is considered phishing. Otherwise, two of the other three check if the message is valid. His experiments reached 95% accuracy.

Chiew et al. [10] based their studies on Machine Learning, where the authors collected sites in two different years: 2015 and 2017. As a result, 5,000 phishing web pages were selected based on URLs from PhishTank [1], OpenPhish [2], and another 5,000 legitimate pages based on URLs of Alexa [3] and the archive of CommonCrawl [4]. Subsequently, this dataset was processed to remove pages with the "404 error", which failed to load and duplicate instances of web pages. Finally, the authors proposed developing a new resource selecting structure, called Hybrid Ensemble Feature Selection (HEFS), using filtering resources to find a practical subset of resources to detect phishing based on machine learning. The experiments showed a better performance applying the Random Forest classifier, reaching a competitive precision of 94.6% using only 20.8% of the original number of resources compared to the SVM classifiers, Naive Bayes, C4.5, JRip, and PART [10].

In their paper, Jain and Gupta [11] collected 2,544 phishing resources and legitimate websites, including websites in different languages. The resources used hyperlink information provided in the website's source code. Experimental results using the Logistic Regression reached a high accuracy in detecting phishing and legitimate websites, with 98.39% of true positive rate, 98.8% accuracy, and 98.42% accuracy.

Sahingoz et al. [1] proposed a real-time anti-phishing system that uses seven different classification algorithms and NLP-based features. This system used legitimate data and feature-rich classifiers to investigate the URL of the web page with seven machine learning algorithms and different feature sets, achieving excellent accuracy rates between 93% and 98% depending on the algorithm and the features. The authors have built their dataset, which we detail in Section III.

B. Explainable Artificial Intelligence

With the new information technologies, we found that Artificial Intelligence is in multiple sectors of activity. Currently, the ability to learn, reason, and adapt are fundamental characteristics to reach levels of performance of increasingly complex computational tasks, making them fundamental for human society's future development [12]. Machine learning is also among the technological advances. In short, it is a smart multidisciplinary approach used to build predictive models [3].

The unification of AI and machine learning promises to produce autonomous systems to understand, learn, decide, and act independently. However, the effectiveness by the inability of the machines limits to explain their decisions and actions. Thereby, the Explainable Artificial Intelligence has emerged, which intends to produce explainable models, aim to maintain the level of learning performance, and allow human users to understand, trust, and effectively manage artificially intelligent generation [7].

The new learning systems can explain their objectives, characterize the strengths and weaknesses, and transmit how they will behave in the future. These models combine human-computer interface (IHC) techniques to become useful for the end-user and capable of presenting explainable and understandable diary models, as depicted in Figure 2 [7].

In the present work, we used two explanatory techniques to carry out the experiments, LIME and EBM, discussed below. While EBM proposes highly explainable modeling (white box) to construct a prediction model, LIME presents an interpretation of the already trained black box models' predictions. In this way, LIME focuses on generating local explanations, while EBM aims to train an explainable model locally and globally.

[1]https://www.phishtank.com
[2]https://www.openphish.com/
[3]https://www.alexa.com/topsites
[4]https://commoncrawl.org

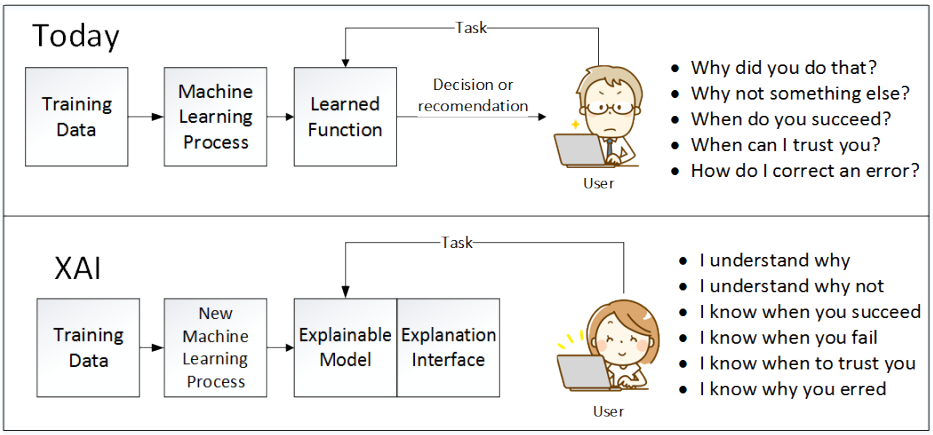


Fig. 2. XAI standard paradigm adapted from [7].

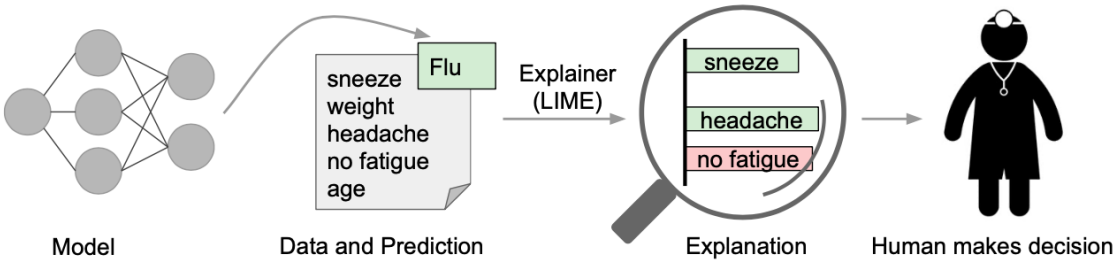


Fig. 3. Explanation flow through LIME [13].

1) Local Interpretable Model-agnostic Explanations (LIME): : it is a technique for explaining models that promotes interpretability through the investigation of prediction sites. The model technique is agnostic, which means that it was designed to work with any classifier or regressor based on machine learning.

The technique's objective is to find a model that can be interpreted by approximating the original model (and less explainable) and is considered in the literature as a post-hoc technique. By allowing identifying the relevance of the input components to the model's predictions, LIME allows obtaining a more transparent analysis of the information being taken into consideration and the decision taken. The technique presents an exciting possibility in generating local explanations of the model, increasing its interpretability and reliability, or even finding possible points of difficulty. The idea is to act on individual requests, promoting explanations about them.

Figure 3 exemplifies a model that predicts that a patient has flu, where LIME highlights the patient's history symptoms, which led him to the decision. Sneezing and headache are represented as contributors to the flu's prediction (green rectangle), while the [lack of] Fatigue (red rectangle) is counter-evidence. With this data, the doctor can decide whether or not he can trust the model [13].

2) Explainable Boosting Machine (EBM): : it is a transparent model developed to have performance comparable to the state of the art models such as Random Forest Boosted Tree, although it is highly interpretable and explainable [14]. It is a Generalized Additive Model with improvements over the more traditional approaches [15]. It uses bagging and boosting techniques, responsible for combining the learning obtained on each input characteristic in N functions (one for each of the N characteristics). EBMs are highly intelligible because each feature's contribution can be visualized and understood by analyzing or plotting these functions.

III. Experiments

We used the Ebbu2017 database, built and publicly available by [1]. The base contains 36,400 legitimate URLs (e.g. http://www.webopedia.com/TERM/L/local_area_network_LAN.html[5]) and 37,175 URLs of *phishing* (e.g. http://lfc.by/fmsk/kate/kate/b30615003c2ebb55e21f3e16cacce4c5[5]), totaling 73,575 URL, collected by the authors. Besides, the pre-processing data followed the same patterns of this work. They used 40 features such as keyword count, random tags and words, special characters, and randomness of the domain name, among others, to create the input vectors proposed to the Machine Learning models.

This system presented in [1] showed some advantages such as independence of the language and may be suitable for dictionaries of different languages. It also highlights the real-time execution. After the attackers quickly create fraudulent

[5] Extracted from [1]

domains, the system can detect phishing in minimal time. Another advantage is the independence of third-party software, as the system does not rely on blacklists or third-party systems for detection. Also, such features extracted via PLN are suitable for developing transparent models since they are readily interpretable.

For the EBM and LIME models to be applied and the visualizations generated, the InterpretML package, a unified framework for interpretability in machine learning models, was used, which provides a series of tools for this purpose [14].

For the LIME approach, we evaluate two traditional algorithms, the SVM by its popularity and Random Forest, that, according to [1] achieve the best results on this feature set. We also separated the data between training and test set in the ratio of 90-10.

IV. Results and Discussions

A. LIME

As described earlier, LIME is a model that interprets the data generated by other black-box models, bringing explainability and transparency about algorithm decision-making. The data were processed using the algorithms Random Forest and SVM, and we obtained excellent results, as presented in Table I with accuracy rates above 97.9% for both algorithms. Figure 4 depicts the ROC curve (*Receiver Operating Characteristic*) of the model *Random Forest* presenting excellent results, with AUC (*Area Under the Curve*) of 0.9955.

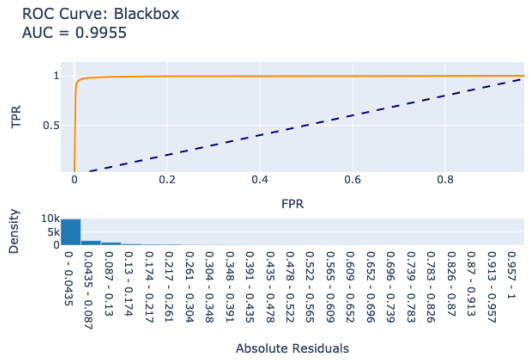


Fig. 4. ROC Curve for LIME using Random Forest.

Figure 5 shows the results obtained for LIME, where they confirm which characteristics had the most influence on decision-making about classifying or not classifying a URL as Phishing. Each characteristic described is assigned a bar in orange or blue colors. The bar size is proportional to the weight that its characteristic influenced the decision of the model, while the color means whether this weight was to indicate a legitimate URL value (orange) or if the influence is for a value of *phishing* (blue).

Still, in Figure 5 one can observe the importance of the characteristic *domain_in_brand_list*, which indicates that the domain belongs to the list of most popular trademarks. Also, characteristic *punny_Code* can be identified as a term that

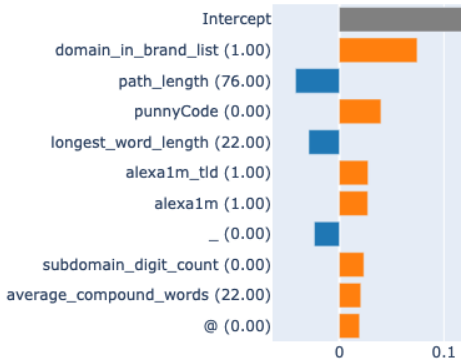


Fig. 5. Explainable results presented by LIME.

TABLE I
Experimental results concerning LIME.

Technique	Prec	Recall	F1-score	Acc	Interpret
EBM	0.9741	0.9536	0.9637	0.9646	Yes
Random Forest	0.9880	0.9571	0.9723	0.9732	No
SVM	0.9790	0.9119	0.9443	0.9469	No

indicates ambiguity or double meaning in the composition of the analyzed URL. It is important to remember that there are no global explanations of the classifications and results due to the LIME model's characteristics, only local explanations.

B. EBM

EBM presents an excellent option, considering that, in general, and performance aside, it is desirable to use naturally transparent techniques to the detriment of techniques *post-hoc*, even if it means loss of performance. Nevertheless, EBM showed very competitive results in performance, as shown in Table I, where the column Interpret shows the native interpretable algorithms. The element that most highlights the use of EBMs are their explanations, which can be both at model (global) levels and the level of individual (local) predictions.

Figure 6 depicts an overview of the most relevant characteristics according to the samples' decision, i.e., *phishing* or legitimate. According to the trained model, the characteristic *alexa1m_tld*, which denotes the presence or absence of the Higher Level Domain (TLD) of URLs in the most accessed sites according to Alexa, was the most important for making predictions. Figure 7 denotes how this characteristic (1 for true and 0 for false) influence the decisions of the algorithm (1 for legitimate and 0 for *phishing*). The explanation demonstrates that when this characteristic becomes true, the algorithm believes that the URL is legitimate and the reverse when the characteristic is false.

Figures 8 and 9 present explanations about correct predictions obtained by the algorithm for legitimate URLs and *phishing*, respectively. There are characteristics bound by relevance, where colors and horizontal sense represent the

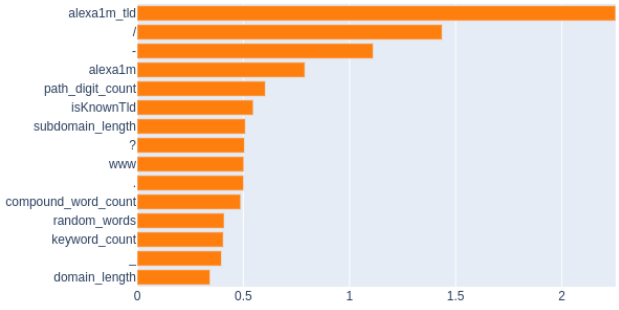


Fig. 6. EBM overall global explanation.

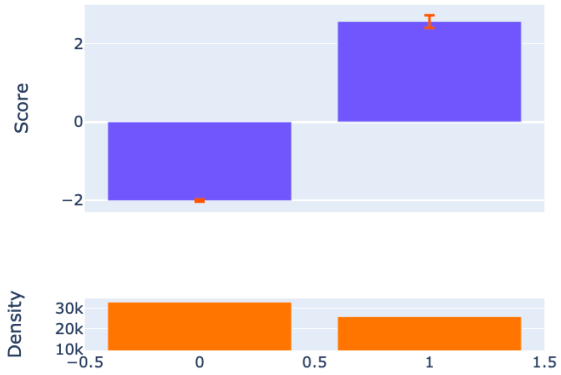


Fig. 7. EBM global explanation for the presence of TLD in Alexa's top 1M sites.

influence according to each class – orange and right represent legitimate, blue and left represent Phishing.

As aforementioned, the algorithm uses the characteristic *alexa1m_tld* in high regard for the prediction's decision of both classes because of the number of bars (/) and dashes (-), which reinforces the coherence between local and global explanations (Figure 6). This type of information could be used both in the making of more accurate models and the generation of knowledge about the problem treated.

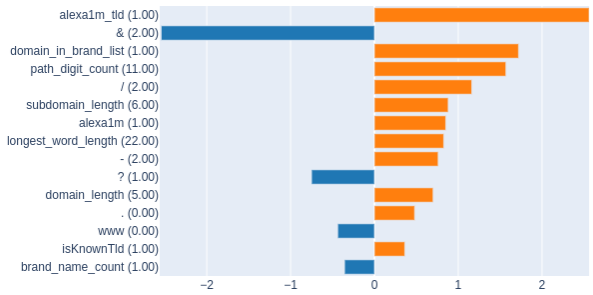


Fig. 8. Local explanation of EBM (correct prediction of legitimate URL).

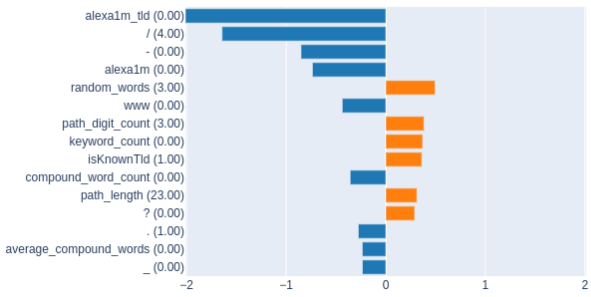


Fig. 9. Local explanation of EBM (correct prediction of a Phishing URL).

V. Conclusion

The Internet has become an essential tool in everyone's life, and there is an impressive growth in its use year after year. Therefore, the ease of buying products or making payments online has become more accessible but brought the adversity of increasing the number of victims of crimes committed in the electronic environment, such as extortion or theft of information. Some scams still require interaction with the user, who is induced to click on a malicious link in many cases.

In this scenario, one of the biggest threats to network and computer security concerns phishing attacks, one of the most efficient virtual attacks from the virtual criminal's perspective. With the increase of attacks, most of which are sent via e-mail, the technique provides a message to make the victim access a malicious link, which this user believes to be legitimate.

Researchers have studied new ways to prevent these attacks' impact, using AI techniques and Machine Learning models. Based on these techniques, experiments use explainable LIME and EBM techniques based on malicious URLs. Explainable models bring light to traditional black-box models, which did not explain why they made their decisions despite excellent data classification results.

The experiments showed that the models presented could classify URLs as phishing correctly or legitimate but also brought explainability to these machine learning models, making the final result more understandable. This facility has the advantage of more information about the problem dealt with, which will lead to more accurate models.

For future works, one possibility is to train new models based on the most relevant characteristics and evaluate their performance in terms of speed and accuracy.

References

[1] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from urls," *Expert Systems with Applications*, vol. 117, pp. 345–357, 2019.

[2] A. K. Jain and B. Gupta, "Phish-safe: Url features-based phishing detection system using machine learning," in *Cyber Security*. Springer, 2018, pp. 467–474.

[3] N. Abdelhamid, F. Thabtah, and H. Abdel-jaber, "Phishing detection: A recent intelligent machine learning comparison based on models content and features," in *2017 IEEE international conference on intelligence and security informatics (ISI)*. IEEE, 2017, pp. 72–77.

[4] Verizon, "2020 data breach investigations report," Verizon, Tech. Rep., 2020.
[5] B. R. Gaines and P. Compton, "Induction of ripple-down rules applied to modeling large databases," *J. Intell. Inf. Syst.*, vol. 5, no. 3, p. 211–228, Nov. 1995. [Online]. Available: https://doi.org/10.1007/BF00962234
[6] F. Thabtah, I. Qabajeh, and F. Chiclana, "Constrained dynamic rule induction learning," *Expert Syst. Appl.*, vol. 63, no. C, p. 74–85, Nov. 2016. [Online]. Available: https://doi.org/10.1016/j.eswa.2016.06.041
[7] D. Gunning, "Explainable artificial intelligence (xai)," *Defense Advanced Research Projects Agency (DARPA), nd Web*, vol. 2, p. 2, 2017.
[8] T. Peng, I. Harris, and Y. Sawa, "Detecting phishing attacks using natural language processing and machine learning," in *2018 ieee 12th international conference on semantic computing (icsc)*. IEEE, 2018, pp. 300–301.
[9] L. F. Cranor, S. Egelman, J. I. Hong, and Y. Zhang, "Phinding phish: An evaluation of anti-phishing toolbars." in *NDSS*, 2007, pp. 1–19.
[10] K. L. Chiew, C. L. Tan, K. Wong, K. S. Yong, and W. K. Tiong, "A new hybrid ensemble feature selection framework for machine learning-based phishing detection system," *Information Sciences*, vol. 484, pp. 153–166, 2019.
[11] A. K. Jain and B. B. Gupta, "A machine learning based approach for phishing detection using hyperlinks information," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 5, pp. 2015–2028, 2019.
[12] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information Fusion*, vol. 58, pp. 82–115, 2020.
[13] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should I trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 1135–1144.
[14] H. Nori, S. Jenkins, P. Koch, and R. Caruana, "Interpretml: A unified framework for machine learning interpretability," *arXiv preprint arXiv:1909.09223*, 2019.
[15] T. Hastie and R. Tibshirani, "Generalized additive models: some applications," *Journal of the American Statistical Association*, vol. 82, no. 398, pp. 371–386, 1987.