Discover Artificial Intelligence

Research

Transfer learning with ResNet50 for malicious domains classification using image visualization

Fikirte Ayalke Demmese[1] · **Shaghayegh Shajarian**[1] · **Sajad Khorsandroo**[1]

Received: 12 March 2024 / Accepted: 22 July 2024
Published online: 26 July 2024

Abstract
The Internet has become a vital part of our daily lives, serving as a hub for global connectivity and a facilitator for seamless communication and information exchange. However, the rise of malicious domains presents a serious challenge, undermining the reliability of the Internet and posing risks to user safety. These malicious activities exploit the Domain Name System (DNS) to deceive users, leading to harmful activities such as spreading drive-by-download malware, operating botnets, creating phishing sites, and sending spam. In response to this growing threat, the application of Machine Learning (ML) techniques has proven to be highly effective. These methods excel in quickly and accurately detecting, classifying, and analyzing such threats. This paper explores the latest developments in using transfer learning for the classification of malicious domains, with a focus on image visualization as a key methodological approach. Our proposed solution has achieved a remarkable testing accuracy rate of 98.67%, demonstrating its effectiveness in detecting and classifying malicious domains.

Keywords Traffic classification · Domain name system (DNS) · Image visualization · Transfer learning · Malicious domains

1 Introduction

The proliferation of internet usage has led to a significant increase in cyber-attacks. These attacks are particularly concerning due to their exploitation of legitimate services and protocols for malicious purposes. In the face of these challenges, establishing a robust defense strategy becomes critical. Cyber attackers often leverage reliable services to spread malware, establish covert networks, and create fraudulent websites for scams and phishing.

Hence, accurately identifying and locating the sources of these malevolent activities is an imperative task for security experts and organizations toward protecting internet users and preventing the misuse of legitimate internet services and protocols.

A range of advanced techniques is utilized to accomplish this objective. This includes conducting in-depth analyses of Uniform Resource Locators (URLs) associated with malicious content, the diligent identification of suspicious domain names used for unlawful purposes, and the meticulous tracing of Internet Protocol (IP) addresses linked to these malicious activities. The emphasis of this paper is on the identification of malicious domain names. Traditionally, the primary method for this task involved filtering domains against known blacklists [1]. However, this method encountered significant limitations, particularly in identifying newly generated domains that had not yet been added to these blacklists [2].

✉ Shaghayegh Shajarian, sshajarian@aggies.ncat.edu; Fikirte Ayalke Demmese, fademmese@aggies.ncat.edu; Sajad Khorsandroo, skhorsandroo@ncat.edu | [1]Department of Computer Science, College of Engineering, North Carolina Agricultural and Technical State University, Greensboro, NC, USA.

To address these limitations, utilizing Machine Learning (ML) techniques has marked a significant advancement. These techniques have proven to be invaluable assets in improving the effectiveness of domain vetting systems, offering a more dynamic and comprehensive approach to identifying potential cyber threats.

Continuing from the integration of ML techniques in domain vetting systems, these systems, equipped with ML algorithms, are capable of analyzing extensive data sets. This analysis allows for the identification of distinct patterns and features that differentiate malicious domains from their legitimate counterparts. ML models, when trained on historical datasets that contain both known malicious and benign domains, acquire the ability to discern and recognize the typical characteristics and behaviors associated with malicious domains. This training equips these systems to more effectively identify potential threats, enhancing the overall efficacy of cybersecurity measures.

In parallel with the advancements in ML, image-based network traffic approaches present a complementary and innovative technique in cybersecurity [3]. This approach involves transforming network traffic data into visual representations, enabling the application of computer vision techniques for enhanced analysis. The conversion of network data into images paves the way for ML algorithms to classify and analyze network behavior effectively. To be more specific, by merging computer vision and ML, image-based network traffic analysis offers a novel perspective in understanding network dynamics, identifying anomalies, and improving network security and performance. Image-based visualization allows us to transform complex DNS traffic data into visual representations, revealing intricate patterns and characteristics that are otherwise challenging to discern through traditional data analysis methods.

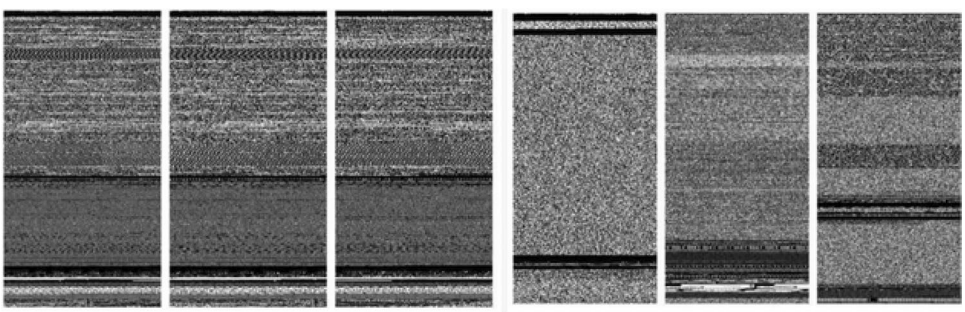
The use of image-based techniques in representing malware significantly enhances visual tools, facilitating the rapid identification of suspicious or unfamiliar malware variants. This is coupled with the ability to recognize abnormal behavioral patterns in a timely manner. These techniques are effective at detecting minor alterations in malware yet maintain the overall structure of samples from the same family [4, 5].

For instance, as illustrated in Fig. 1a, malware classes that are closely related tend to display more pronounced visual similarities among themselves compared to those from unrelated classes, which is clearly distinguishable in Fig. 1b.

There is a significant gap in the current research regarding the integration of image-based visualization with advanced ML-based techniques specifically for the classification of malicious domains, as discussed in detail in the related work section. Our study addresses this critical gap by presenting a novel approach that leverages transfer learning, a technique where knowledge gained from solving one problem is applied to a different but related problem, and image visualization, thereby enhancing the accuracy and effectiveness of threat detection. The contributions of this paper are outlined as follows:

- We introduce an advanced approach for accurately detecting and categorizing malicious domains into specific types, including malware, spam, phishing, and benign domains, by leveraging transfer learning techniques.
- Central to our methodology is the utilization of image visualization, where labeled CSV data is transformed into RGB images to visually represent domain name information, enhancing the analytical capability of the model.
- We employ the pre-trained ResNet-50 [6] model as the foundation for our transfer learning approach, achieving a testing accuracy rate of 98.67%, which outperforms traditional methods and other machine learning models.

The rest of the paper is organized as follows. Section 2 provides a thorough review of existing literature. Section 3 offers the necessary background on Domain Name System (DNS), network traffic image conversion techniques, and transfer learning. Also, it details the proposed methodology we used to visualize domain names and automatically classify them



(a) Related Malware Family (b) Unrelated Malware Family

Fig. 1 Images of malware. (a) The comparison of a related malware family. (b) The comparison of an unrelated malware family

using images as input. Section 4 discusses our experimental setup and the evaluation of the proposed approach. Finally, Sect. 5 concludes the paper and discusses limitations.

2 Related work

In the field of cybersecurity, significant advancements have been made in the visualization and classification of network threats and malware, as evidenced by pioneering studies such as those by Kim et al. [7] and Ren et al. [8], and others. Kim et al. [7] developed a method to visualize DNS traffic, specifically to detect patterns unique to botnet attacks. They created a visualization technique called BotXrayer, which displays DNS traffic using parallel coordinates. This method uses four key parameters to show the structure and attack strategies of botnets accurately. With BotXrayer, they successfully analyzed and understood DNS traffic linked to botnet actions. Ren et al. [8] introduced a new visualization method to address DNS security issues, including Distributed Denial of Service (DDoS) and cache poisoning attacks. They presented Flying Term, a unique visualization tool that captures the evolving nature of DNS queries. This tool integrates various visual metaphors like Stacking Graphs, Two Tone Pseudo Color, and Chernoff Face Glyph into a single framework. This combination improves the ability to monitor and understand DNS queries.

In a study presented by Nataraj et al. [5], they implemented an innovative technique for malware classification by converting binary data of malware into 2D grayscale images. This approach revealed that variants within the same malware family displayed consistent textural patterns in the visual representation. The authors proceeded to extract GIST features from these images, utilizing them as distinctive markers for different malware types. Subsequently, they applied the KNearest Neighbors (KNN) algorithm, leveraging the Euclidean distance metric, to categorize the malware based on the identified features. Their methodology, applied to a comprehensive dataset comprising 9,458 malware samples across 25 distinct families, demonstrated a notable classification accuracy of 97.18.

Another study by Dai et al. [9] introduced a novel method for malware analysis involving the extraction of data from memory dumps and their transformation into grayscale images. The study utilized three distinct classification algorithms, KNN, Random Forest (RF), and Multi-Layer Perceptron (MLP), for the categorization of malware. The classification process was executed in three stages. Initially, malware memory dumps were extracted in a controlled sandbox environment, allowing for an assessment of their behavior. These dump files were then converted into grayscale images. Subsequently, the team extracted feature vectors through bi-cubic interpolation to define key image attributes. The efficacy of their methodology was tested and validated using actual malware samples.

In another research conducted by Gibert et al. [4], the researchers expanded upon the approach of Nataraj et al. [5] by integrating a Convolutional Neural Network (CNN) for classifying assembly language in portable executable (PE) files. They adopted the technique of transforming malware's binary data into grayscale images, aiding in the recognition of patterns and the classification of malware into distinct families. Employing the CNN architecture, they extracted a variety of features from these images, focusing on both local and invariant characteristics. These features provided valuable insights into the malware's structure and behavior, enabling the identification of specific patterns associated with different malware families. The effectiveness of this method in detecting malware patterns was confirmed through extensive experimentation and analysis.

In an innovative study, Ni et al. [10] proposed a technique for classifying malware families that merges image-based malware visualization with CNN technology. Their approach involved extracting similar hash values from akin malware codes using locality-sensitive hashing, resulting in comparable hash values yielding similar grayscale images. They enhanced the performance of their algorithm by incorporating techniques like multi-hash, major block selection, and bilinear interpolation. The final step involved employing CNNs to determine the specific family to which each malware sample belonged.

Also, in a related development, Zhang et al. [11] transformed network flow data into grayscale images for the classification of encrypted network traffic. Their methodology combined Long Short-Term Memory (LSTM) with CNN to extract and analyze statistical features. Specifically, they utilized the LSTM model to derive representation vectors from the images, while CNN was employed to extract key features and facilitate network classification.

Prasad and Chandra [25] introduced the PhiUSIIL framework, which leverages a diverse security profile and employs similarity index and incremental learning techniques to detect phishing URLs. This framework's ability to continuously update and improve its detection model highlights the potential of advanced ML techniques in cybersecurity. The PhiUSIIL model achieved high accuracy and robustness by employing a combination of pre-training and incremental

learning, which significantly improved its performance over time. The comparison of methodologies and focus areas with key findings from related studies is summarized in Table 1.

In light of the extensive research reviewed, it becomes clear that while significant advancements have been achieved in malware visualization and classification, a comprehensive integration of image-based visualization with advanced ML techniques for the specific purpose of classifying malicious domains remains unexplored. Previous research primarily focused on malware classification through various visualization techniques, such as transforming binary data into grayscale images and employing neural networks for pattern recognition. However, these studies largely concentrated on the identification and categorization of malware and encrypted network traffic without extending their methodologies to the broader context of malicious domain classification.

Our paper addresses this gap by introducing a refined, image-based transfer learning approach for the specific classification of malicious domains. This method builds upon the foundational concepts of visual analysis used in malware identification, enriched with the advanced use of transfer learning techniques. By adapting these established techniques to domain classification, our aim is to offer a more precise and comprehensive tool for the identification of various types of malicious domains, such as benign, phishing, malware, and spam. This strategy not only utilizes the strengths of existing research but also expands their applicability, potentially enhancing the effectiveness and scope of cybersecurity analysis. Our approach uniquely combines image-based visualization and transfer learning specifically tailored for malicious domain.

classification. Unlike previous studies that focused primarily on malware and phishing URLs, we extend these visualization techniques to DNS data, converting structured domain information into RGB images. This conversion allows us to leverage the powerful feature extraction capabilities of pre-trained models like ResNet-50. By fine-tuning these models for the specific task of domain classification, we achieve a high testing accuracy rate of 98.67%, significantly improving detection and classification performance over traditional methods. This integration and application can address the identified research gap and push the boundaries of how ML and image-processing techniques can be applied in cybersecurity.

3 Methodology

This section elucidates the fundamental principles and technological foundations critical to our methodology for classifying malicious domains. Subsequently, we will provide a detailed explanation of the proposed model.

3.1 Domain name system (DNS)

The DNS is an essential protocol on the internet that translates domain names into IP addresses. This decentralized, distributed system allows the association of user-friendly domain names, like www.example.com, with their numerical IP counterparts, like 192.0.2.1, facilitating communication between computers and servers. The DNS is vital for accessing websites, sending emails, and other internet services, eliminating the need to remember numerical IP addresses.

However, DNS has been subject to misuse and abuse in numerous widespread attacks, resulting in substantial consequences for millions of internet users [12]. These attacks target weaknesses in the DNS infrastructure, jeopardizing the security, availability, and reliability of the domain name resolution process. When a domain is registered and utilized with malicious intent, such as carrying out phishing attacks, spreading malware, hosting illicit content, or participating in cybercrime, it is categorized as malicious [13]. They are specifically designed to deceive users, take advantage of vulnerabilities, and undermine their privacy and security.

3.2 DNS traffic to image generation

By leveraging data preprocessing to transform raw data into a format that is more suitable for analysis, a particularly effective technique in network analysis is the conversion of network traffic data into image format, a method that has proven to be promising and effective across various network analysis tasks [3]. In this section, we will explore the process of converting network traffic stored in the form of CommaSeparated Values (CSV) into image representations. Our

Table 1 Summary of key studies in cybersecurity visualization and classification

Study	Methodology	Focus area	Key findings
Kim et al.	Visualization of DNS traffic using parallel coordinates	Botnet detection	Analyzed and understood DNS traffic linked to botnet actions
Ren et al.	Flying term visualization tool integrating multiple visual metaphors	DNS security issues (DDoS, cache poisoning)	Improved ability to monitor and understand DNS queries
Nataraj et al.	Binary data of malware transformed into 2D grayscale images, KNN algorithm	Malware classification	Achieved classification accuracy of 97.18%
Dai et al.	Memory dump grayscale images, KNN, RF, MLP algorithms	Malware classification	Effective malware categorization using image features
Gibert et al.	CNN for classifying assembly language in PE files	Malware family classification	Enhanced pattern recognition in malware Classification
Ni et al.	Image-based malware visualization, CNN technology	Malware family classification	Improved classification accuracy
Zhang et al.	Network flow data transformed into grayscale images, LSTM, CNN	Encrypted network traffic classification	Effective classification of encrypted traffic
Prasad and Chandra	Diverse security profile, similarity index, incremental learning	Phishing URL detection	High accuracy and robustness, continuous model improvement
Our Study	Transfer learning with ResNet50 using the image visualization	Malicious domain domain classification	Achieved a remarkable accuracy rate of 98.67%, demonstrating the effectiveness in detecting and classifying malicious domain

primary objective is to examine how structured data from CSV files can be transformed into visual formats, effectively illustrating the diverse patterns and characteristics inherent in network traffic.

The DNS traffic dataset, which is used to train our proposed model, has 32 fields for each DNS transaction, including key attributes such as TLD (top-level domain) and the length of the domain and subdomain. This data is converted into a grayscale image through the following process. Initially, relevant fields are extracted from the DNS traffic dataset, which is stored in a CSV file format. During the grayscale representation phase, each field is assigned a specific grayscale intensity value based on its characteristics. For instance, higher intensity values (e.g., 255) are assigned to certain query types like"A" or"AAAA," while lower intensities (e.g., 0) are allocated to other types. Numeric fields, such as the length of the domain and subdomain, are directly translated into grayscale intensity values corresponding to their actual value. Subsequently, a blank grayscale image is created with dimensions that represent the DNS traffic data, where each pixel in the image corresponds to a specific field or a combination of fields from the DNS traffic data. Due to the varying sizes of DNS traffic data, a resizing step is incorporated to standardize the image dimensions, thereby simplifying the conversion process and enhancing the model's performance.

The next stage involves mapping the data to intensity values. This is achieved by iterating through each DNS transaction in the dataset and identifying the relevant pixels in the image based on the field values, adjusting the pixel intensity according to the predefined grayscale scheme.

Finally, once the mapping is complete, the resulting grayscale image is saved to a file. Each image is appropriately labeled to ensure it is accurately associated with its corresponding DNS traffic data, thus completing the conversion process. These steps are outlined in Algorithm 1.

map_f1_intensity() and map_f2_intensity() in Algorithm 1 are functions that define the mapping of Feature 1 and Feature 2 to grayscale intensity values, respectively. We transform the grayscale images into RGB format by replicating the grayscale channel across the red, green, and blue channels. This transformation provides additional information by representing the grayscale values into three distinct color channels [14]. Although we did not employ traditional data augmentation techniques, we transformed the grayscale images into RGB format by replicating the grayscale channel across the red, green, and blue channels. While grayscale images only contain intensity values, RGB images distinguish this intensity across red, green, and blue channels. This allows for the representation of color variations and can reveal subtle details or patterns that may not be apparent in grayscale.

3.3 Transfer learning

Transfer learning has recently emerged as a significant trend, greatly benefiting the research community. This sophisticated deep learning technique involves initially training a model on a comprehensive dataset and subsequently applying the knowledge thus acquired to a smaller or distinct dataset [15]. This approach is particularly advantageous when dealing with smaller datasets, as it enables the model to utilize the extensive insights gathered from the larger dataset.

In the context of image-processing deep learning networks, a critical aspect is the stratified acquisition of fundamental features. The network's initial layers excel in identifying essential elements such as edges, textures, and basic shapes. These components are universal, representing fundamental visual patterns that are consistent across a variety of image datasets. The core of transfer learning lies in its ability to harness these universal features from an already trained model. Rather than starting from scratch with a new dataset, the model is fine-tuned using its pre-existing knowledge base, tailored to the new dataset. This method ensures the preservation of the model's core understanding while customizing it to the specific requirements of the new dataset. This approach has been consistently effective in image classification, establishing it as a preferred method for achieving superior results in image data-related tasks.

Algorithm 1 DNS Traffic data to image conversion

```
procedure ConvertToImage(dns_traffic_data)                    ▷ Covert dns_traffic to Image
    from PIL import Image
    for dns_transaction in dns_traffic_data do
        f1_data = dns_transaction ['f1']
        f2_data = dns_transaction ['f2']
        f1_intensity = map_f1_intensity(f1_data)              ▷ Convert Feature 1 to equivalent intensity
        f2_intensity = map_f2_ intensity(f2_data)             ▷ Convert Feature 2 to equivalent intensity
        x1, y1 = calculate_ pixel_ position(f1_data)          ▷ Determine the coordinates (x1, y1)
        x2, y2 = calculate_ pixel_ position(f2_data)          ▷ Determine the coordinates (x2, y2)
        image.putpixel((x1, y1), f1_intensity )               ▷ Set the corresponding pixels in the image
        image.putpixel((x2, y2), f2_intensity )               ▷ Set the corresponding pixels in the image
    end for
    image_label = "example_label"                             ▷ Replace with corresponding label
    image.save("dns_traffic_image_< image_label > .png")      ▷ Save Image in PNG format
end procedure
```

3.3.1 ResNet

ResNet, short for Residual Network, has transformed how deep CNNs are used in image recognition and classification [16]. Its innovative design is tailored to address the increasing complexity of tasks in this field, enhancing the precision of such systems. ResNet's main innovation is its use of residual learning, a technique that effectively overcomes challenges like the vanishing gradient problem and the degradation issue. This is achieved through residual blocks, which help the network focus on learning the differences between the layer's input and the expected output, thereby improving the training of deep neural networks.

In our study, we will be employing the ResNet-50 model, which consists of 50 layers. This model exemplifies the principles of residual learning, employing residual blocks to train deep neural networks efficiently. ResNet-50 is known for its high performance and accuracy in complex image recognition tasks. Furthermore, we plan to utilize transfer learning with the pre-trained ResNet-50 model for our specific classification task. This approach allows us to benefit from the model's pre-established knowledge base, fine-tuning it to suit our specific requirements. This method is not only efficient but also ensures enhanced accuracy in our task, leveraging the advanced architecture of ResNet-50.

3.3.2 VGG-19

The VGG-19 architecture, a well-regarded CNN model, features a deep structure composed of 19 layers, predominantly convolutional and fully connected [17]. To train our model, we used images with a size of 224×224 pixels. This model, foundational to our study, was trained using images of 224×224 pixel resolution. The VGG-19's layers facilitate the extraction and learning of complex features from these images, contributing significantly to its analytical capabilities. To augment the VGG-19's proficiency in classification tasks, we implemented specific modifications. Notably, a global average pooling layer was introduced, which effectively compresses the spatial dimensions of outputs from preceding layers. This compression allows for a more focused analysis of crucial features pertinent to classification.

Moreover, our enhancements include the integration of two dense layers into the base architecture. Dense layers, characterized by their fully connected structure, receive inputs from all neurons in the preceding layer, thus enabling the model to discern intricate patterns and associations within the data. The incorporation of these dense layers was strategically aimed at improving the model's capacity to classify DNS samples into four distinct groups categorically.

3.3.3 Inception-V3

The Inception-V3 model, part of the renowned Inception series, stands as a notable CNN architecture that is designed for image classification tasks [18]. It distinguishes itself through a deep structural design and an efficient deployment of computational resources. This architecture contains various layers, including convolutional, pooling, and fully connected

layers. Inception-V3 represents an evolution of the original Inception model by introducing enhancements that elevate both its performance and efficiency. In our research, we utilized the standard Inception-V3 model, meticulously configured to classify images into four distinct DNS categories.

An integral part of our methodology was the employment of images sized at 299×299 pixels for training the model. This resolution was specifically chosen to enable the Inception-V3 to process and utilize the spatial information inherent in the images effectively. Consequently, this approach facilitates the model's ability to discern and leverage complex, distinctive features crucial for accurate classification results.

3.4 Proposed model

The proposed approach comprises the following fundamental steps:

- Data preparation: Our initial focus is on the preparation of the dataset. We utilize a specifically labeled DNS CSV dataset, where each data entry represents a distinct DNS sample. Each of these samples undergoes a transformation process, converting it from its original tabular form into a grayscale image.
- Image processing: Post-conversion, the grayscale images are further processed. Each image is converted into a three-channel RGB (Red, Green, Blue) format, aligning with the input requirements of standard CNN models. Following this, the images are resized to a uniform resolution of 224×224 pixels. This resizing is necessary to ensure consistency in the input data and to make it compatible with the neural network architectures we employ.
- Selection of transfer learning architectures: Our methodology involves exploring three distinct architectures in the realm of transfer learning: ResNet 50, VGG19, and Inception-V3. Each of these architectures offers unique features and benefits, making them suitable for our analysis. ResNet50 is known for its deep network capabilities, VGG-19 for its simplicity and depth, and Inception-V3 for its efficiency and lower computational demands.
- Model training and adaptation:
    - Parameter freezing: In each chosen architecture, we begin by freezing the parameters of the convolutional layers. This step involves retaining the pre-trained weights in these layers, ensuring that the rich feature detection capabilities learned from large-scale datasets like ImageNet are preserved.
    - Network customization: For the CNN model built on the ResNet-50 architecture, we implement a specific customization. The network's architecture is modified at its final stage, where the original 1000 fully connected softmax layers are replaced with four fully connected softmax layers. This alteration tailors the model to our specific classification task of identifying distinct categories of DNS data.
    - Parameter transfer: Additionally, we transfer the parameters from the convolutional layers of the original ResNet-50 model to the corresponding layers in our customized CNN model. This transfer is a critical component of our approach, leveraging the pre-trained capabilities of the ResNet-50 model to enhance our model's performance without the need for extensive training from scratch.

Following the selection of ResNet 50, VGG-19, and Inception-V3 for our transfer learning framework, we adapted these architectures to meet the specific requirements of our research problem. In particular, we employed the ResNet-50 model, originally trained on the extensive ImageNet dataset that includes 1000 different classes[19]. Utilizing transfer learning techniques [20], we adapted the ResNet-50 architecture as our primary input. This adaptation involved the addition of global average pooling and two successive dense layers at the output stage for efficient image classification. Furthermore, we modified the architecture by replacing the original 1000 fully-connected softmax layer with a customized version consisting of four fully-connected softmax layers, a modification clearly illustrated in Fig. 2.

To mitigate the risk of overfitting, a dropout strategy was implemented, applying a rate of 0.5 prior to the flattening of the pooled feature map. For the purpose of fine-tuning our model, we employed the Adam optimizer [21], setting a conservative learning rate of 0.0001. This same methodology was consistently applied to both the VGG-19 and Inception-V3 models. In these instances, the output layer was configured to classify data into four distinct categories of malicious domains. The complete set of parameters used in our experiments is detailed in Table 2.

We selected ResNet-50, VGG-19, and Inception-V3 as the primary architectures for transfer learning based on their proven effectiveness and unique strengths in image classification tasks. ResNet-50 is known for its deep residual learning framework. It addresses the vanishing gradient problem and allows for the training of very deep networks. Its use of residual blocks helps the model to learn more effectively and achieve high accuracy in complex image classification

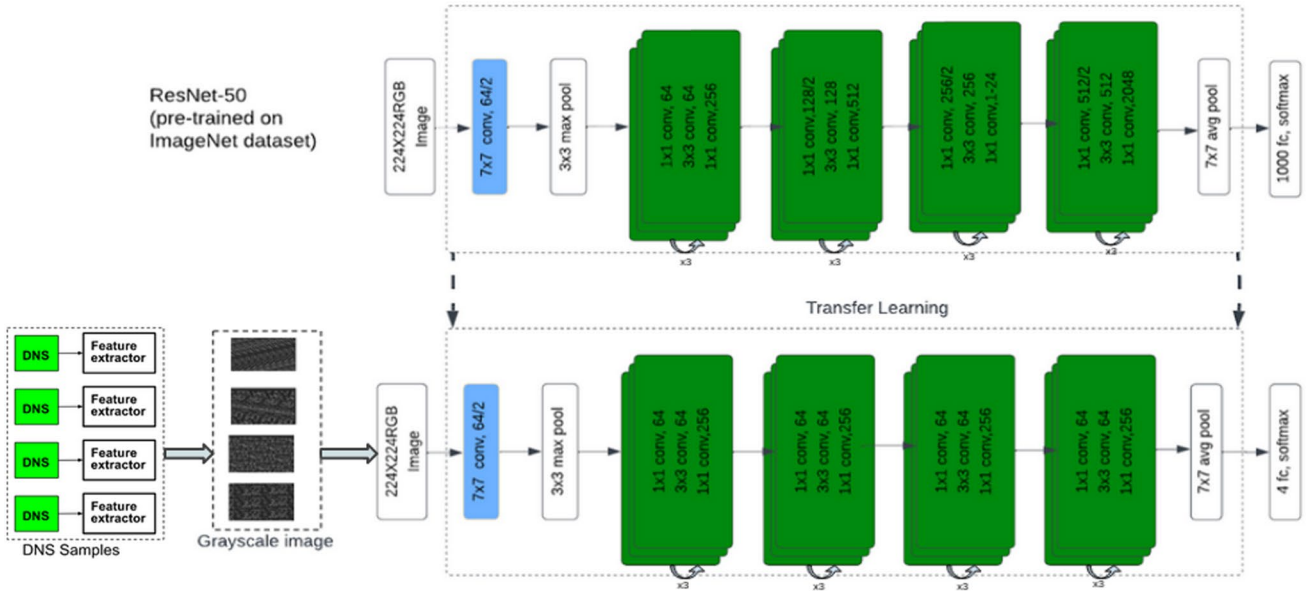


Fig. 2 Transfer learning using Resnet50

Table 2 Model's hyperparameters and characteristics

Parameter	Resnet50	VGG19	Inception-V3
image size	224×224	224×224	299×299
Batch size	32	32	32
Dropout rate	0.5	0.5	0.5
Epochs	25	25	25
Learning rate	0.0001	0.0001	0.0001
Trainable parameters	525,572	132,356	262,788
Non-trainable parameters	23,587,712	20,024,384	21,802,784
Loss function	Categorical Cross entropy	Categorical Cross entropy	Categorical Cross entropy
Optimizer	Adam	Adam	Adam

tasks. The VGG-19 model is chosen for its simplicity and depth. VGG-19 uses a straightforward architecture with small convolution filters, making it easy to implement and modify. The Inception-V3 is selected for its efficient use of computational resources and its capability to capture diverse features through its inception modules. It balances depth and computational efficiency, making it suitable for tasks requiring detailed feature extraction and high performance.

4 Experimental setup and evaluation

4.1 Dataset and experimental settings

In this experiment, we utilize the CIC-Bell-S2021 dataset [22, 23] sourced from the Canadian Institute for Cybersecurity. This dataset includes DNS traffic data labeled into four categories: benign, phishing, malware, and spam. Each entry in the dataset comprises 32 features, such as TLD, length of the domain and subdomain, and various DNS query types. We performed several preprocessing steps on the dataset before converting it to images. First, we removed outliers and unnecessary features, including the 'page_rank' feature, which had no significant impact on our task. To address the class imbalance, where spam files were underrepresented compared to malicious and phishing files, we employed random sampling. We set the sampling size to 4337 (the number of spam files) and randomly selected 4337 instances from the malicious and phishing categories, ensuring each class had an equal number of instances.

After converting the raw DNS traffic data to grayscale images, we found that the images varied in size. To standardize them, we resized all images to a consistent size of 224×224 pixels, simplifying the conversion process and enhancing model performance. The collection of this DNS dataset involves two primary stages. In the first stage, a comprehensive set of domain names is gathered from diverse sources, including both benign and malicious domains. These domains are categorized into types such as benign, phishing, malware, and spam.

The subsequent stage of dataset collection employed Wireshark, a network protocol analyzer. This phase focused on capturing network packets associated with the previously collected domains through HTTP request transmissions. Once the DNS packets are captured, a preprocessing step is implemented to retain only the packets that contain the IP address information. Further enrichment of the dataset is achieved by extracting 32 distinct DNS features from the preprocessed packets. These features offer a multifaceted view of the DNS data, providing valuable insights for the ensuing analysis and classification tasks [22]. The dataset is formatted as a CSV file. We conducted data cleaning to remove any empty rows and columns, as they contribute no meaningful information.

Our domain image dataset is divided into four classes, which are labeled as Malware, phishing, spam, and benign domains. In total, we generated 4,337 image samples, resulting in 13,011 malicious image samples and 2,337 benign image samples. The experiments were conducted using Google Colab Cloud Pro, utilizing Python Language 3.5, along with libraries such as Tensorflow, Scikit-learn, and Keras. The labels were deduced from the dataset using the image dataset from the directory function provided by TensorFlow Keras. As part of the dataset preparation, the dataset was segmented into three subsets: training (80%), validation (20%), and a separate test set comprising 2,608 samples.

4.2 Comparison with other models

In this section, our performance is benchmarked against other foundational models, including the Multi-Layer Perceptron (MLP), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM).

For this comparative analysis, we utilized the basic features of the DNS dataset in its original form without image conversion. This approach allowed us to assess the relative effectiveness of our model in contrast to various established methodologies.

To guarantee consistency and reproducibility in our multiclass classification task, we adopted a rigorous data preprocessing protocol. Our dataset is categorized into four classes: Benign, Malware, Phishing, and Spam. We eliminated any incomplete or missing DNS traffic records to preserve data integrity and mitigate potential biases or inaccuracies in our analysis. We applied feature scaling to normalize numerical attributes, ensuring that all features contribute equally to the model training and preventing any single attribute from disproportionately influencing the results due to scale differences.

Additionally, categorical variables were transformed using one-hot encoding, converting these attributes into binary vectors for more effective processing by machine learning algorithms. For the evaluation of our models, we divided the dataset into training, validation, and testing subsets. The training set was utilized to train the models, the validation set assisted in hyperparameter tuning and model selection, and the testing set was used for the final model evaluation. To maintain consistent outcomes across multiple executions, we employed a fixed random seed (set to 42) for both the train-test split and machine learning classifiers, ensuring the repeatability of our results. Each algorithm was individually trained on the training set with its respective optimized parameter settings, tailored to leverage its unique strengths and functionalities. The performance of each model was then evaluated based on accuracy, measured on the test set. This accuracy metric provides a comprehensive overview of the model's ability to classify the different categories within the dataset accurately.

For each model in our experiment, we employed different hyperparameters. In the MLP architecture, we used the Adam optimizer. Adam's adaptive learning rates and momentum characteristics facilitate rapid convergence and expedite the training process [24]. We also employed a hidden layer configuration with sizes of (30, 20, 10). These hidden layers are an essential component of the neural network and are responsible for learning complex patterns and representations from the input data. In terms of fine-tuning the training process, we set the learning rate to 0.001. This parameter governs the step size at which the model updates its internal parameters during training. As shown in Table 3, although the MLP model exhibited relatively lower performance in our experiment, it still provided valuable insights, achieving a classifier accuracy of 62.7%, a precision of 72.6%, and an F1 score of 57.2%.

In our implementation of KNN, we configured the model to consider the three nearest neighbors when generating predictions (n neighbors = 3). This precise configuration allowed us to fine-tune the model's reliance on its closest neighbors, enabling more informed and accurate prediction-making. We also incorporated the Ball Tree algorithm into our

Table 3 Comparison of different models on the classification result

Model	Image data	Accuracy	Precision	Recall	f1-score
Resnet-50	Yes	98.67	98.70	98.67	98.66
VGG19	Yes	93.33	93.45	93.33	93.33
Inception-V3	Yes	66.87	66.11	66.87	66.34
MLP	No	62.7	72.6	57.2	63.98
K-NN	No	97.9	98.2	97.8	97.99
SVM	No	97.99	97.8	98.3	98.04

KNN framework. The Ball Tree is a specialized technique designed to facilitate the efficient identification of nearby data points. By implementing this algorithm, we were able to expedite the process of searching for the nearest neighbors, which in turn improved the speed and accuracy of our predictions. It effectively reduced the computational complexity associated with the search process, making our model more efficient in finding the most relevant data points. The KNN model, under this optimization scheme, demonstrated better results, with a classifier accuracy of 97.9%, a precision of 98.2%, and an F1 score of 97.8%.

Finally, in our application of the SVM, we chose the radial basis function (RBF) as the kernel function. Given that we are tackling a multi-class classification problem, we embraced the One-vs-One (OvO) Decision Function approach. With OvO, an individual SVM is trained for each possible pair of classes. The decision process employs a voting mechanism, where each SVM"casts a vote" for one of the classes. The class that accumulates the highest number of votes is ultimately assigned as the prediction. Through this approach, our SVM model achieved remarkable performance, with a classifier accuracy of 97.99%, a precision of 97.8%, and an F1-score of 98.3%.

As shown in Table 2, the most favorable outcome achieved for the dataset, without converting into images, was obtained with the SVM model, reaching a classifier accuracy of 97.99%. Nevertheless, the outcomes from our Resnet-50 surpassed the performance of the traditional base models. Our model achieved higher accuracy rates of 98.67% and better predictive capabilities compared to the base models. These improvements highlight the significance of optimizing image-based ML models to achieve superior results in detecting malicious activities.

4.3 Evaluation

In this experiment, the Resnet-50 model exhibited superior accuracy and minimal loss compared to the other transfer learning models and our base models. One of the key factors contributing to this improved accuracy is that the Resnet-50 model was able to utilize the parameters during training, thereby avoiding any loss of essential features.

The training accuracy of our Resnet-50 model was measured at 95.76%, while the validation accuracy reached 97.17%.

On the other hand, the other transfer learning models achieved a training accuracy of 89.05% for VGG-19 and 76.14% for Inception-V3. We assessed the performance of the model generated through the training pipeline by evaluating it on a separate test set that does not overlap with the data used for training or validation. During the testing phase, out of the 2608 data points used, only 22 were misclassified, resulting in an overall testing accuracy of 98.67%.

Furthermore, the training and validation accuracies and losses of our custom model were found to be closely aligned, indicating that the model did not suffer from overfitting. This observation suggests that the model generalizes well to unseen data and is not overly specialized to the training set. Figure 3 illustrates the disparity between the achieved training accuracy and validation accuracy.

During the training and validation phases, we divided our dataset into subsets using a variety of random seeds. Notably, this process yielded consistently similar results across the different dataset partitions. To enhance the reproducibility of our results, a specific random seed was selected for the final model configuration. As illustrated in Fig. 4, the optimization curve demonstrates a close alignment between training and validation loss, suggesting an absence of overfitting in the model.

In this experiment, we employ four key evaluation metrics to assess the performance of the model: accuracy, F1-score, precision, and recall. Accuracy serves as a general indicator of the model's prediction correctness. Precision measures the model's capability to accurately identify positive instances, while recall assesses the model's success in identifying all positive instances correctly. The F1-score represents a balanced measure that combines precision and recall, highlighting an effective trade-off between these two metrics.

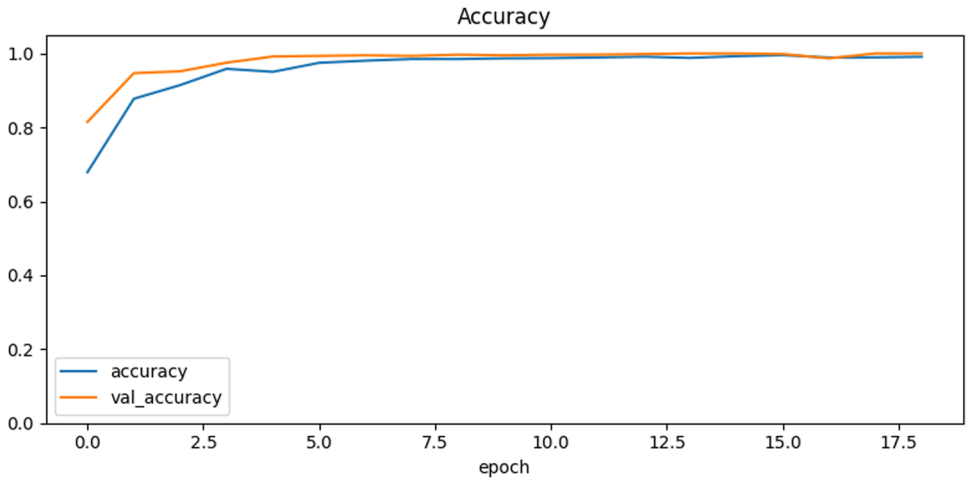


Fig. 3 Validation accuracy vs. training accuracy

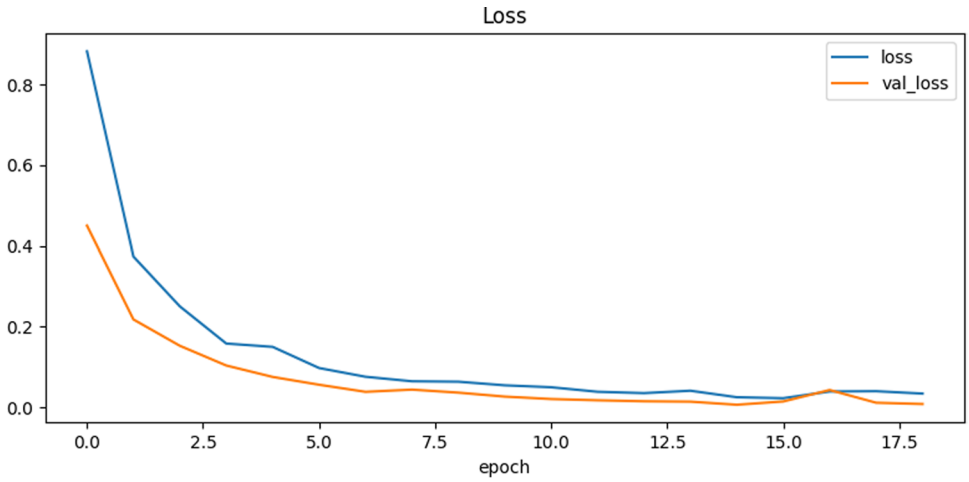


Fig. 4 Validation loss vs. training loss

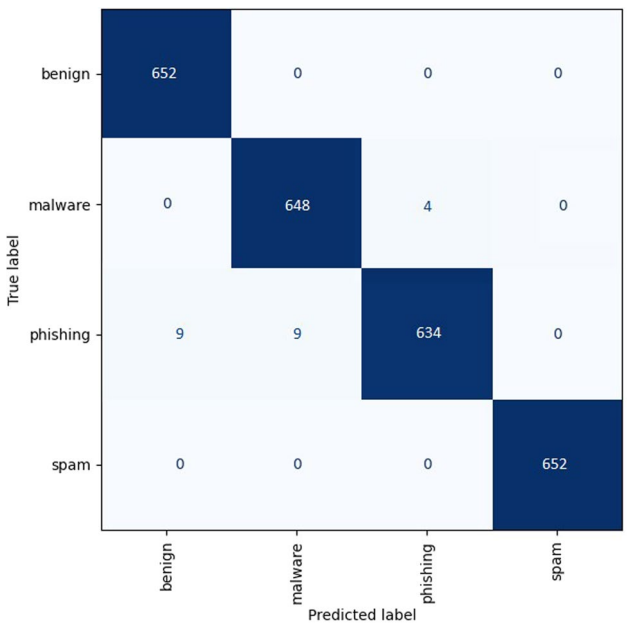


Fig. 5 Confusion matrix for classification of four domains categories

Figure 5 presents the normalized confusion matrix derived from the classification of domain samples into four distinct categories.

5 Conclusion and future works

Traditionally, the identification of malicious domains relied heavily on comparing against existing blacklists, a method that often fell short, especially in recognizing newly emerged domains not yet listed. This study marks a significant advancement in this area, being one of the first to focus specifically on the classification of malicious domains using a novel blend of transfer learning and image-based visualization techniques.

In our approach, DNS files are converted into RGB images, offering a unique visual perspective of the DNS data. This conversion paves the way for applying sophisticated image analysis methods. By utilizing the ResNet-50 model in our transfer learning framework, we achieved a remarkable classification accuracy of 98.67%. Furthermore, the employment of VGG-19 and Inception-V3 models yielded training accuracies of 89.05% and 76.14%, respectively, underscoring the efficacy of our methodology in accurately identifying various types of malicious domains. Despite these promising results, it is important to note that our model's performance is dependent on the quality and diversity of the dataset used, and the computational intensity of the image conversion process poses scalability challenges.

Future research could explore ways to improve classification accuracy further, especially with the VGG-19 and Inception-V3 models. Investigating more advanced or newer neural network architectures and experimenting with different combinations of layers or tuning parameters could yield better results. Additionally, expanding the dataset to include more diverse and recent samples of malicious domains will enhance the model's robustness and generalizability. Researchers could also focus on improving computational efficiency through techniques such as dimensionality reduction or lightweight architectures and integrating real-time detection capabilities to handle high-velocity data inputs. Developing continuous learning mechanisms that allow the model to update itself with new data periodically will be crucial for adapting to emerging threats and ensuring the model remains effective over time.

Author contributions Fikirte Ayalke Demmese conceived the presented idea, developed the theory, and performed the computations. She also contributed to the manuscript's writing. Shaghayegh Shajarian verified the analytical methods and contributed to the design and implementation of the research as well as the writing of the manuscript. Sajad Khorsandroo was involved in critically revising the manuscript for important intellectual content and approved the final version to be published. All authors discussed the results and contributed to the final manuscript.

Funding This work was supported in part by National Science Foundation (NSF) Grants #2113945 and #2200538. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agency.

Data availability The data utilized in this study is sourced from the "CIC-Bell-DNS 2021 dataset," which is publicly accessible at the following URL: https://www.unb.ca/cic/datasets/dns-2021.html Reference: Canadian Institute of Cybersecurity: CIC-Bell-DNS 2021 Dataset (2021). https://www.unb.ca/cic/datasets/dns-2021.htmlAccessed 05-Jan-2023.

Code availability Not applicable.

Declarations

Competing interests The authors declare no competing interests.

References

1. PaloAltoNetworks: Stop Attackers from using DNS against you. 2023. https://start.paloaltonetworks.com/protect-your-dns-traffic-against-threats? Accessed 18 June 2023.

2. Zhao H, Chang Z, Bao G, Zeng X, et al. Malicious domain names detection algorithm based on n-gram. J Comput Netw Commun. 2019;2019:1.
3. Demmese FA, Neupane A, Khorsandroo S, Wang M, Roy K, Fu Y. Machine learning based fileless malware traffic classification using image visualization. Cybersecurity. 2023;6(1):32.
4. Gibert D, Mateu C, Planes J, Vicens R. Using convolutional neural networks for classification of malware represented as images. J Comput Virol Hacking Tech. 2019;15(1):15–28.
5. Nataraj L, Karthikeyan S, Jacob G, Manjunath BS. Malware images: visualization and automatic classification. In: Proceedings of the 8th International symposium on visualization for cyber security. 2011. ACM: 2011. p. 1–7.
6. TensorFlow v2.12.0. https://tensorflow.org/api.docs/python/tf/keras/applications/resnet50/ResNet50. 2023. Accessed 18-April-2023
7. Kim I, Choi H, Lee H. Botnet visualization using dns traffic. In: Proc. of WISA. 2008.
8. Ren P, Kristoff J, Gooch B. Visualizing dns traffic. In: Proceedings of the 3rd International workshop on visualization for computer security. 2006. p. 23–30.
9. Dai Y, Li H, Qian Y, Lu X. A malware classification method based on memory dump grayscale image. Digit Investig. 2018;27:30–7.
10. Ni S, Qian Q, Zhang R. Malware identification using visualization images and deep learning. Comput Secur. 2018;77:871–85.
11. Zhang Y, Zhao S, Zhang J, Ma X, Huang F. Stnn: a novel tls/ssl encrypted traffic classification system based on stereo transform neural network. In: 2019 IEEE 25th International conference on parallel and distributed systems (ICPADS). IEEE: 2019. p. 907–910
12. Zhauniarovich Y, Khalil I, Yu T, Dacier M. A survey on malicious domains detection through DNS data analysis. ACM Comput Surv (CSUR). 2018;51(4):1–36.
13. Zhang K, Ji W, Li N, Wang Y, Liao S. Detection of malicious domain name based on dns data analysis. J Phys Conf Ser. 2020;1544:012169.
14. Van Dao T, Sato H, Kubo M. An attention mechanism for combination of cnn and vae for image-based malware classification. IEEE Access. 2022;10:85127–36.
15. Niu S, Liu Y, Wang J, Song H. A decade survey of transfer learning (2010–2020). IEEE Trans Artif Intell. 2020;1(2):151–66.
16. Shafiq M, Gu Z. Deep residual learning for image recognition: a survey. Appl Sci. 2022;12(18):8972.
17. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint. 2014. https://arxiv.org/abs/1409.1556.
18. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. p. 2818–2826.
19. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, et al. Imagenet large scale visual recognition challenge. Int J Comput Vision. 2015;115:211–52.
20. Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, Xiong H, He Q. A comprehensive survey on transfer learning. Proc IEEE. 2020;109(1):43–76.
21. Kingma DP, Ba J. Adam: a method for stochastic optimization. arXiv preprint. 2014. https://arxiv.org/abs/1412.6980.
22. Mahdavifar S, Maleki N, Lashkari AH, Broda M, Razavi AH. Classifying malicious domains using dns traffic analysis. In: 2021 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). IEEE: 2021. p. 60–67.
23. Canadian institute of cybersecurity: CIC-Bell-DNS 2021 Dataset. 2021. https://www.unb.ca/cic/datasets/dns-2021.html. Accessed 05 Jan 2023.
24. Rathbun TF, Rogers SK, DeSimio MP, Oxley ME. Mlp iterative construction algorithm. Neurocomputing. 1997;17(3–4):195–216.
25. Prasad A, Chandra S. PhiUSIIL: a diverse security profile empowered phishing URL detection framework based on similarity index and incremental learning. Comput Secur. 2024;136: 103545.