Design and Implementation of a new Visualization Aided Anomaly Detection Framework

by

Ahmed Farag

B.Sc., Helwan University, 2013

A Report Submitted in Partial Fulfillment
of the Requirements for the Degree of

MASTER OF ENGINEERING

in the Department of Electrical and Computer Engineering

University of Victoria

Supervisory Committee

Dr. Issa Traore, Department of Electrical and Computer Engineering

Dr. Waleed Yousef, Department of Electrical and Computer Engineering

Abstract

In today's data-driven world, the identification of unusual patterns or anomalies in data sets has become increasingly vital, particularly in the realm of security data where the detection of these atypical patterns can preempt security threats. This is the juncture where our work, as an extension to UNAVOIDS (Unsupervised and Nonparametric Approach for Visualizing Outliers and Invariant Detection Scoring) **[**1**]**, **[**2**]**, becomes instrumental. UNAVOIDS is a distinctive model that integrates specialized techniques for both detection algorithms and visualization methods, operating within a unique space known as the Neighborhood Cumulative Distribution Function (NCDF) space. In this two-dimensional space, each data point is transformed into a unique 2D curve, facilitating visual identification and examination. A salient feature of UNAVOIDS is its fully unsupervised nature, which requires neither prior training nor specific data inputs, eliminating the need for parameter selection or tuning. Another feature is its assignment of a deviation score to each unusual data point, offering a clear gauge of its abnormality. In this study, we successfully deployed UNAVOIDS across four platforms: the Python Package Index (PyPI) , a Restful API**,** a software named VAAD**]**—which integrates UNAVOIDS with the Data Visualization Platform (DVP) **[**3**],** **[**7**]**—, and a custom Microsoft PowerBivisual.

Two main challenges were tackled in this implementation. First, handling large datasets within the RESTful API posed an ongoing challenge. To address this, we adopted compression over file streaming, enabling the efficient transmission of data within the API constraints. Second, creating an interactive visual representation presented a significant challenge due to the unique nature of the data, where each observation is mapped to a 2D curve. We overcame this challenge

by mapping curve indices and implementing a reflection mechanism for interactivity between selected curves and other visuals.

Our study contributes to the practical implementation and effectiveness of UNAVOIDS, and all these implementations along with their documentations are accessible from the official repository **[14]** of the ISOT lab. These implementations, catering to users from various sectors including research and development, provide the versatility and effectiveness of UNAVOIDS in diverse environments.

Table of Contents

List of Tables

List of Figures

Glossary

AUC: Area Under the Curve pip: Package Installer for Python

CDF: Cumulative Distribution Function

DVP: Data Visualization Platform

DVSW: Data Visualization Software

NCDF: Normalized Cumulative Distribution Function

PowerBI: Power of Business Intelligence

PyPI: Python Package Index

SCEs : Scientific computing Environment

UNAVOIDS: Unsupervised and Nonparametric Approach for Visualizing Outliers and Invariant Detection Scoring

VAAD: Visualization-Aided Anomaly Detection

Acknowledgments

I extend my heartfelt gratitude to my supervisor, **Dr. Waleed A. Yousef**, whose unwavering guidance, encouragement, and support have been instrumental in my project and throughout my academic journey.

My sincere thanks also go to my supervisor, **Dr. Issa Traore**, whose persistent support and guidance have been invaluable throughout the course of my project.

Dedication

Dedicated to my parents for their support, motivation and guidance.

Chapter 1 : Introduction

1.1 Background and Motivation

Anomaly detection plays a pivotal role in modern information systems and information security, offering a means to identify unusual patterns. When deviations from the typical system behavior are detected, this information can prompt immediate alerts for security personnel and analysts. Anomaly detection has found utility across a plethora of domains, including fraud detection, intrusion detection, medical informatics, and fault or damage detection. For example, detecting anomalies in purchases, medical tests, or network traffic can reveal potential problems such as fraudulent activities, security breaches, or health issues. This tool assists specialists in exploring these deviations, thereby determining their causes.

However, traditional anomaly detection methods often face difficulties in accurately distinguishing anomalies from normal patterns. This challenge has underscored the need for innovative solutions such as the Unsupervised and Nonparametric Approach for Visualizing Outliers and Invariant Detection Scoring approach (UNAVOIDS) **[1]**, **[2]**. UNAVOIDS is a unique model that synergizes detection algorithms with visualization techniques to identify and analyze anomalies in tabular data. The objective of this thesis is to implement and deploy UNAVOIDS across various platforms and endpoints, facilitating its application in real-world scenarios and offering its unique visualization approach, the NCDF space, to be integrated to visualization software and platforms interactively with all other plots and figures. This implementation was suggested by the authors of UNAVOIDS in their work **[1]**, **[2]** as a future work to utilize the benefit of their invented NCDF space. To the best of our knowledge, we are the first to pursue this implementation. We provided

all the codes, documentation, and use cases, as a Github repo on the ISOT laboratory Github space **[14].**

This report describes the work conducted in the project, delves into the background of UNAVOIDS , and outlines the project's scope, objectives, and contributions.

1.2 What are anomalies?

Consider, for illustrative purposes, the example depicted in **Figure 1.1** (as appears in **[6]**). This figure presents a time series chart that represents temperature readings collected over a period of time from a temperature sensor. Imagine a scenario wherein the sensor experiences a malfunction in reading or temperature regulation, leading to an unusual temperature reading. Even without any quantitative analysis, this data point can be intuitively identified as an outlier, as its value deviates significantly from the range of values exhibited by the rest of the dataset.

However, this is not the sole metric for identifying outliers, or anomalies. A data point could fall within the value range of the entire dataset across all dimensions, yet still be classified as an outlier.

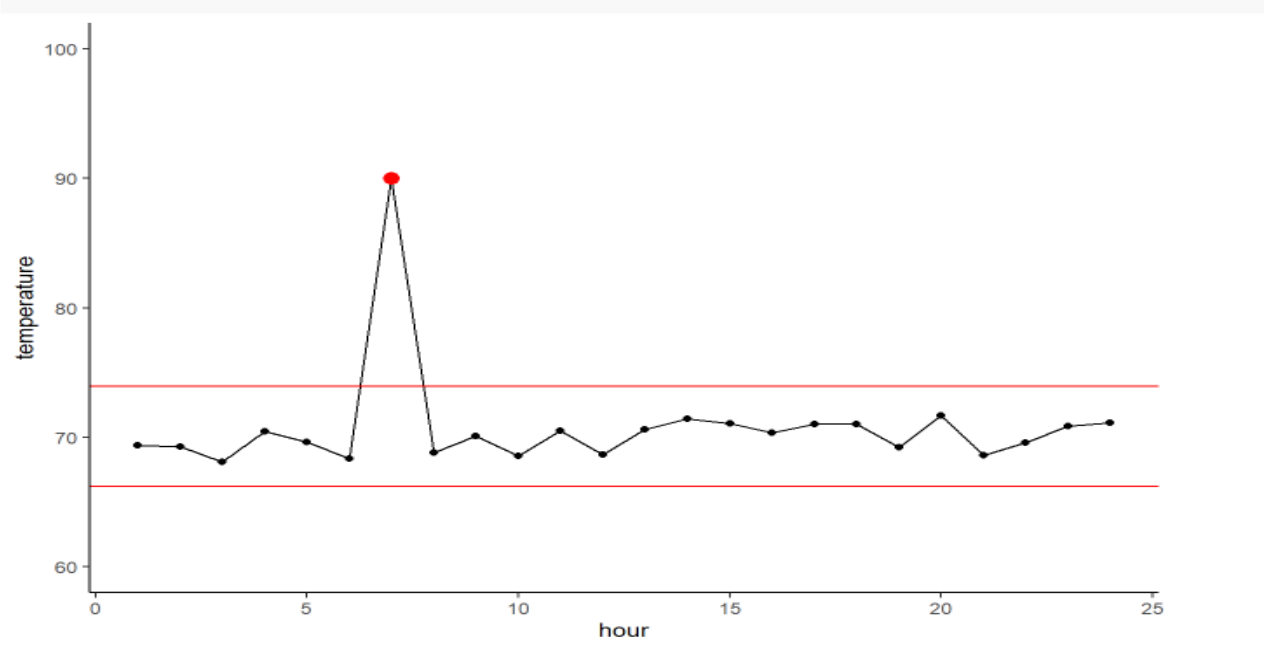

Figure 1.1 : Identifying Temperature Anomalies: A Time-Series Analysis.

Consider another illustrative example in **Figure 1.2** (as appears in **[6]**), which depicts the correlation between age and income. This correlation is represented as a scatter plot, derived from a simulated dataset **[6]**. Data points have been color-coded according to their classification as anomalies, enabling us to discern points that deviate from the established patterns of specific clusters. Despite some of these data points exhibiting coordinate values within the range of the entire dataset, their joint distribution diverges from the joint distribution of the dataset as a whole. Visually, this divergence manifests as data points straying from the primary clusters within the dataset.

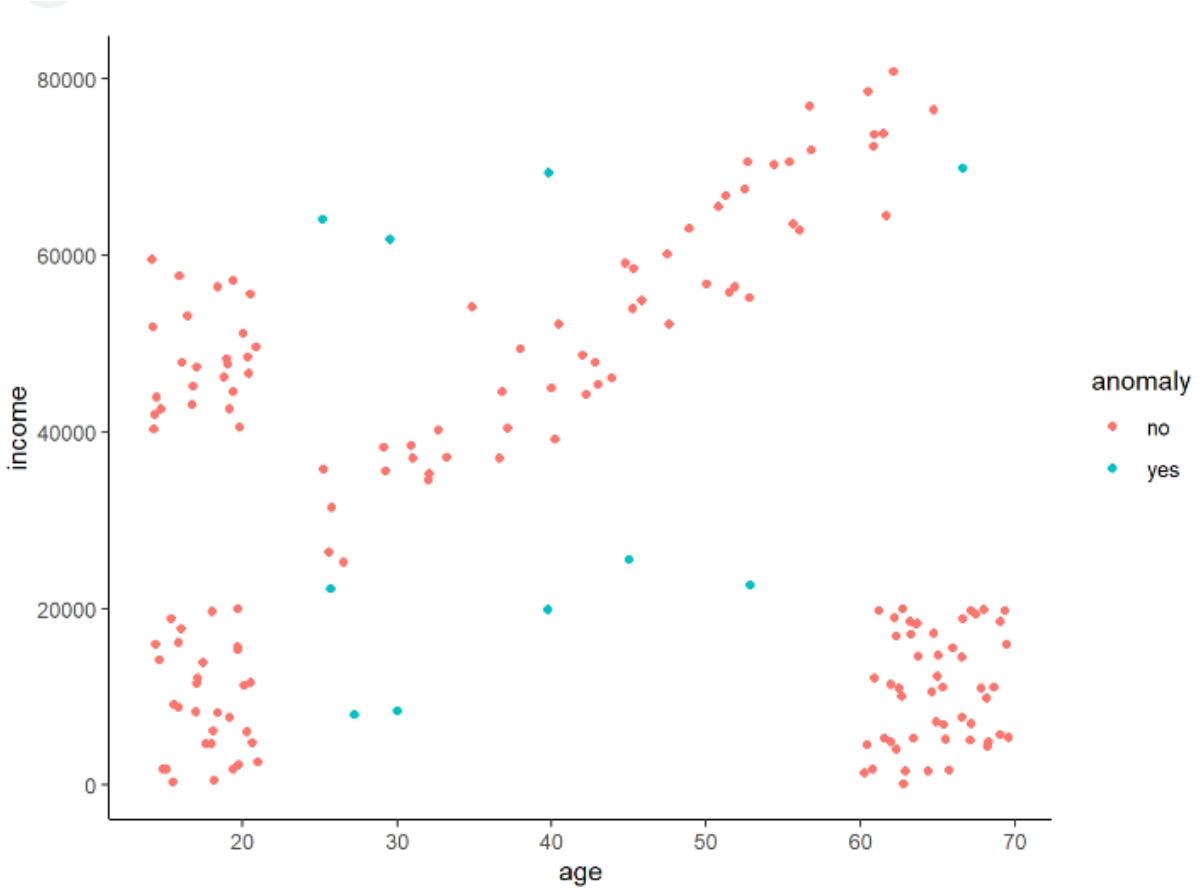

Figure 1.2 : Anomaly Detection in Age-Income Relationship: A Scatter Plot Analysis.

1.3 UNAVOIDS for Anomaly Detection, and Objectives of the Project

The primary objective of this project is the implementation and deployment of UNAVOIDS **[1]**, **[2]**—a unified algorithm that synergizes detection and visualization—across diverse platforms and endpoints, thus enabling its application in real-world scenarios. This endeavor involves four distinct development strands: (1) deployment as a PyPI package, (2) delivery as a RESTful API, (3) the implementation of our visualization approach integrated into the publicly available version of the Data Visualization Platform (DVP) **[3]**, **[7]** to provide a standalone platform that we call VAAD (Visualization Aided Anomaly Detection), and (4) development as a custom Microsoft

Power BI visual [10].

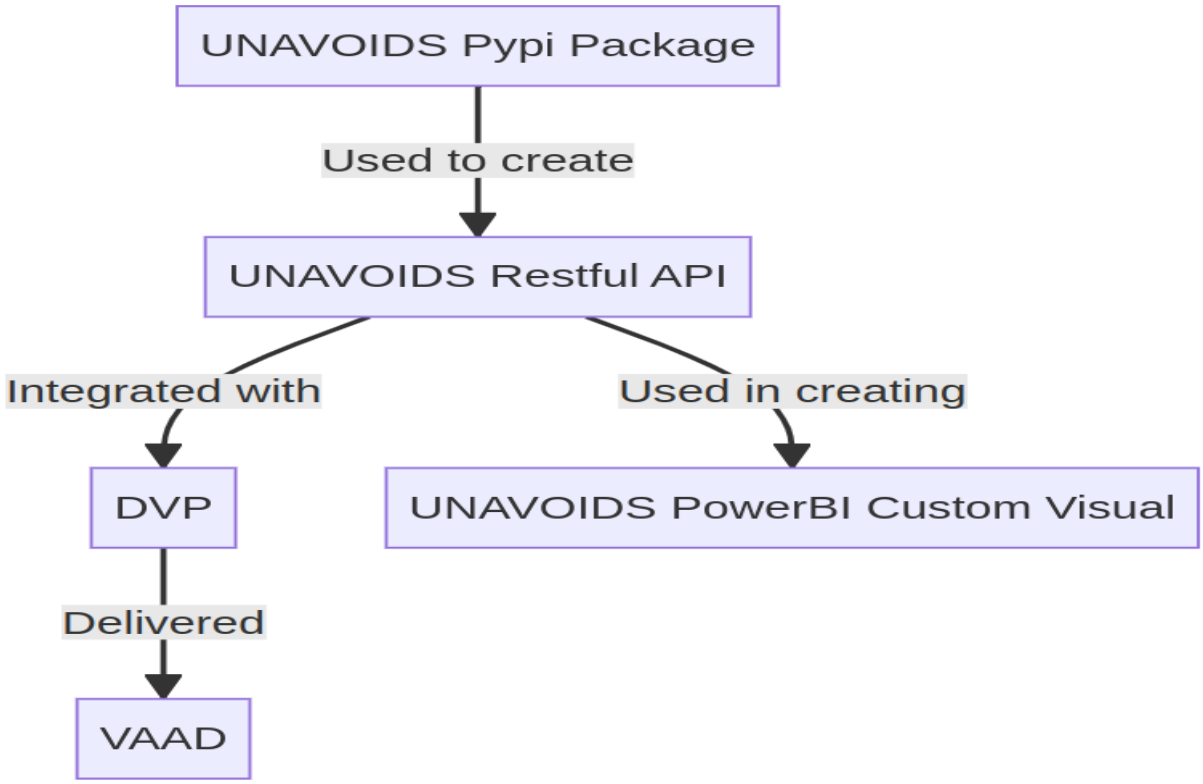

Figure 1.3 : Dependency Flow of UNAVOIDS Deliveries.

Figure 1.3 illustrates the interdependencies among the four implementations. The PyPI package encapsulates and deploys the open-sourced code of UNAVOIDS [1], [2] that was made publicly available by UNAVOIDS's authors as a GitHub repository [4] and a Code Ocean capsule [5]. Because UNAVOIDS's algorithm needs to run to generate the NCDF visualization space, the Restful API, by providing higher-level function calls, enables web-based access to these functionalities—an essential component for developing the VAAD and Power BI implementations.

The integration with the DVP facilitates visualization of the UNAVOIDS NCDF space, allowing it to be integrated with other plots within a unified system that supports interactive visualization features such as brushing and selection. Similarly, the Power BI implementation depicts the NCDF space as a Microsoft Power BI visual, allowing seamless integration with other visualization tools.

1.4 Report Structure

The remainder of this report is structured as follows. **Chapter 2** provides an overview of UNAVOIDS, summarizing its key components and contrasting it with traditional methods. **Chapter 3** discusses the first two forms of implementing UNAVOIDS, specifically on PyPI and as an API. **Chapter 4** discusses the implementation of the latter two forms, namely integration with DVP and delivery as a Power BI Custom Visual. This chapter features real examples and case studies, demonstrating the significant utility of our implementation of UNAVOIDS in anomaly detection. **Chapter 5** concludes the report and summarizes the project's contributions, along with a discussion of its limitations and potential avenues for future contribution.

Chapter 2 : UNAVOIDS

2.1 Introduction

UNAVOIDS [1], [2] represents an unsupervised, nonparametric method for the visualization of outliers and invariant detection scoring. The source code for UNAVOIDS is readily accessible through a GitHub repository [4] and a Code Ocean capsule [5]. These resources provide users with access to the implementation details, thereby fostering further exploration and application of the approach. However, the chosen mode of dissemination by the authors primarily caters to the needs of researchers, potentially overlooking the requirements of practitioners or system analysts. In addition, the NCDF space generated by this code cannot be integrated with other plots and figures generated by other visualization software to work interactively, as the authors suggested as a future work. This serves as our starting point and motivates our aim to implement and deploy UNAVOIDS to a broader array of user communities.

2.2 Anomaly Detection and Visualization Techniques

UNAVOIDS offers an innovative method for detecting and visualizing anomalies in security data, merging detection algorithms and exploratory visualization to facilitate the identification of abnormal patterns and threats. UNAVOIDS uniquely employs a two-dimensional space termed the Neighborhood Cumulative Distribution Function (NCDF) space. This space is not simply a projection of the original data onto two dimensions, but a distinctive transformation capturing the characteristics of all observations relative to different neighborhoods. It derives its name from the cumulative distribution function (CDF) of a random variable, which inspired its creation. Within this space, an observation's classification as anomalous or normal depends on its local

neighborhood. Each data point is represented as a unique curve, allowing anomalies to be easily differentiated from normal observations. The NCDF space preserves the original data's information without loss, rendering it a valuable tool for precise anomaly detection.

This model is entirely unsupervised, providing a normalized score for each observation (or curve) to quantify the degree of its anomalousness. It has been tested on both simulated and real datasets, where it has demonstrated superior detection accuracy compared to other methods.

Figure 2.1 (as appears in [1]) shows a pretend 2D group of two clusters with potentially three outliers shown on the left. The sample NCDF (middle) has a setting of the norm p equals infinity, while on the right, it's set at p equals 2^{-4}. The blue and red outliers can be told apart in the NCDF, while the green outlier can only be distinguished when p is set to 2^{-4}.

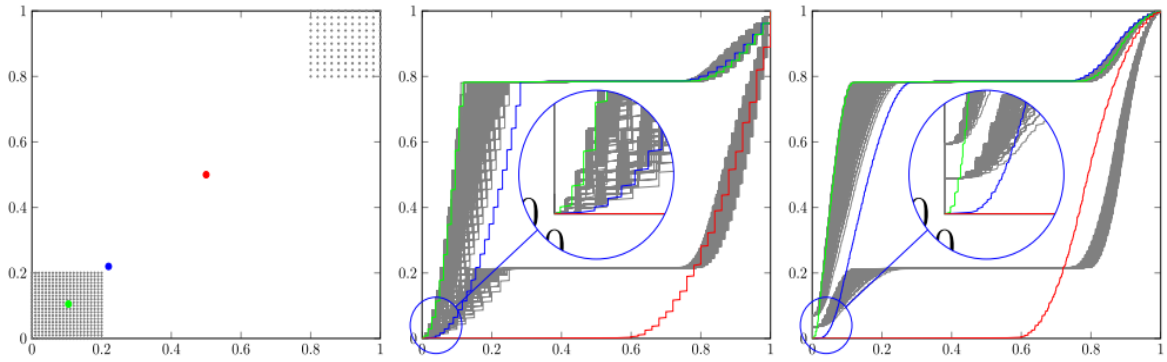

Figure 2.1 : Exploring a Simulated Dataset: Clusters, Outliers, and NCDF Analysis.

2.3 Scoring Anomalies

UNAVOIDS utilizes an invariant detection scoring system where a score is allocated to each observation by the detection algorithm. This score, normalized within the range of [0, 1], signifies the degree of an observation's anomalousness or outlierness. Unlike strict threshold-based decision

systems, this scoring system offers a quantitative measurement of the intensity of an anomaly.

2.4 Comparison with Traditional Anomaly Detection Methods

A comprehensive comparison between UNAVOIDS and three established anomaly detection algorithms (LOF, IF, and FABOD), previously recognized for their superior performance in prior studies, was conducted in [1]. The comparative evaluation employed two real-world datasets with varying configuration parameters. Additionally, the performance of UNAVOIDS was benchmarked against baseline methods, namely K-NN and K-means, using the same datasets.

The results highlighted UNAVOIDS' superior performance on the CIRA-CIC-DoHBrw-2020 [12] dataset across varying K values. However, on the CICIDS-2017 [13] dataset, K-NN outperformed UNAVOIDS at lower K values, while K-means demonstrated better performance at specific K values. Importantly, no single method consistently outperformed others across all scenarios, underlining the importance of context-specific performance evaluation based on the particular dataset and task at hand. **Figure 2.2** (as appears in [1]) summarizes this comparative study. For more details, the reader may refer to the original manuscript [1].

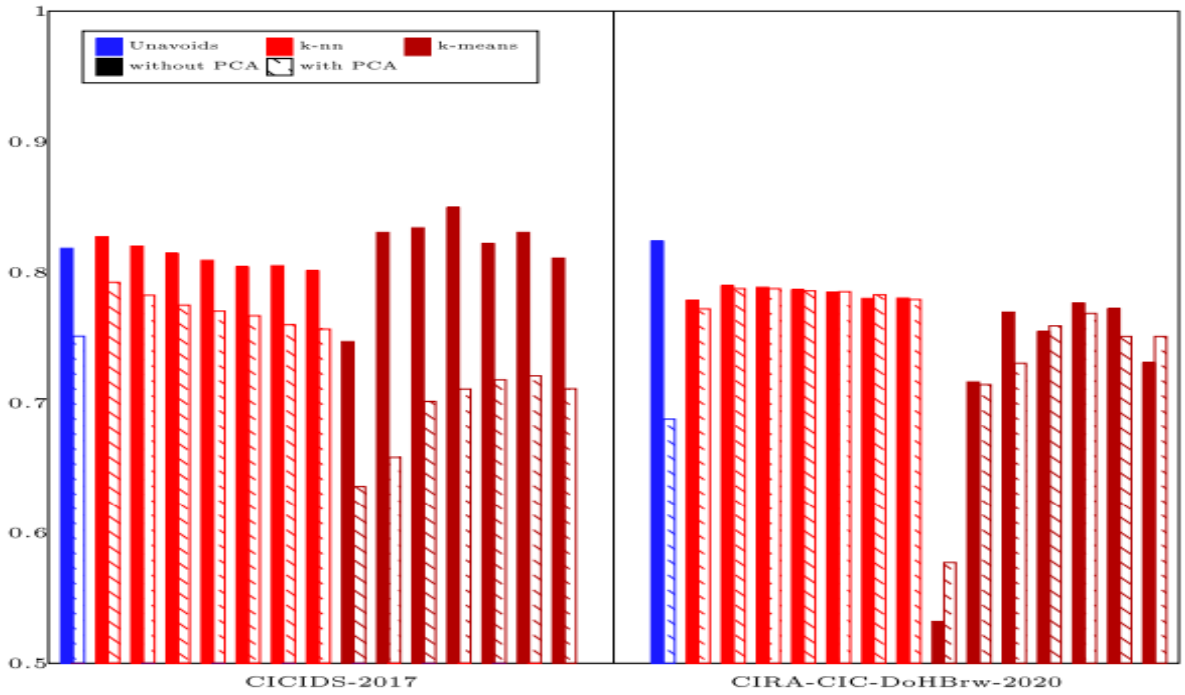

Figure 2.2 : AUC comparison of UN-AVOIDS, KNN, and K-means. K-NN and K-means outperform UN-AVOIDS on CICIDS2017 for some K values.

2.5 Leveraging NCDF Visualization for Comprehensive Anomaly Detection Systems

The authors of UNAVOIDS proposed the visualization of the NCDF space as a potent instrument, advocating for its integration with specialized data visualization software to create an all-inclusive visualization system. Figure 2.3 (as appears in [1]) serves as an illustrative example, presenting sample NCDFs (generated by the authors' code) alongside the scatter plot matrix and the parallel coordinates plot (generated by the DVP [3], [7]) of the first 10 features derived from 400 observations within the same window of CICIDS2017 dataset [12]. The 2D NCDF space, despite not functioning as a lower subspace for data representation, effectively distinguishes the outlier. It is important to mention that the NCDF in this figure is not linked or dynamically interacting with

the other two figures. This is what the authors proposed as a future work to augment anomaly interpretation. Upon selecting and brushing the outlier detected in the NCDF space, corresponding plots in the DVP are automatically highlighted, thereby delivering additional insights into the anomalies' underlying causation. The authors named that suggested integration as VAAD (Visualization Aided Anomaly Detection), whose implementation is one of our contributions in this project.

This integration enables effective interaction between the NCDF visualizations and other relevant graphical representations, thus empowering analysts to explore and interpret anomalies with greater accuracy.

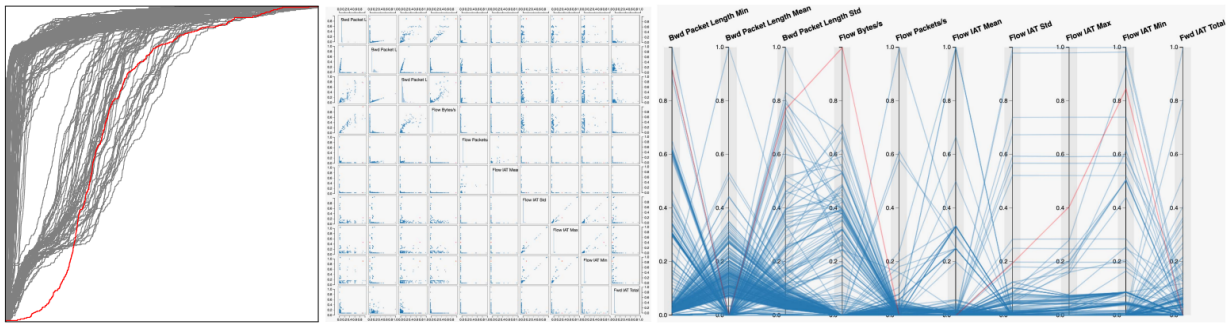

Figure 2.3 : Brushing and Linking the NCDF, Scatter Plot Matrix, and Parallel Coordinates for Outlier Distinction

2.6 Potential Applications of UNAVOIDS

Anomaly detection, in general, and UNAVOIDS, in particular, offer a broad spectrum of potential applications, particularly in the field of security data analysis. Some of these applications include:

- Fraud Detection: UNAVOIDS has substantial potential for uncovering fraudulent activities across various sectors, such as insurance and banking. By identifying anomalous patterns in transactions, it aids in revealing potential fraud attempts.
- Intrusion Detection: In the realm of computer network monitoring, UNAVOIDS can be used to detect unauthorized access or suspicious activities. It can analyze network traffic and pinpoint anomalies potentially indicating a security breach or intrusion.
- Medical Informatics: Within medical settings, UNAVOIDS can be leveraged for diagnosis and disorder detection, especially when there is a scarcity of data and only unsupervised methods are potential.
- Fault/Damage Detection: In industrial sectors, such as commerce and manufacturing, UNAVOIDS can be utilized to detect faults or damages in machinery or production processes. By analyzing sensor data, it identifies anomalies that could signal potential malfunctions or defects.

Chapter 3 : Deployment of UNAVOIDS as PyPI and RESTful API

3.1 Packaging as PyPI

3.1.1 Implementation

Publishing UNAVOIDS on the Python Package Index (PyPI) simplifies the process for users wishing to access and utilize this tool on their local machines. With our published PyPI package **[14]**, users can seamlessly install UNAVOIDS using the widely adopted package manager, pip, with the straightforward command "pip install unavoids". This implementation represents our first extension beyond the authors' original distribution of UNAVOIDS, which was solely available as raw Python code in a public repository **[4]** and a Code Ocean capsule **[5] as mentioned earlier.**

For successful installation and operation of the PyPI package, the following packages are necessary:

- asyncpg
- fastapi
- Matplotlib
- Numpy
- Scikit-learn
- uvicorn

3.1.2 Usage Example

Figure 3.1 provides a simple step-by-step, yet practical, example of how users can utilize UNAVOIDS via our PyPI package [14] seamlessly. This packaging encapsulates functions like generating Normalized Cumulative Distribution Functions (NCDFs) and obtaining anomaly scores, enabling data analysts to integrate this powerful tool into their workflows effortlessly. However, this does not provide integrating the NCDF plots to other visualization systems, the topic of the next chapter.

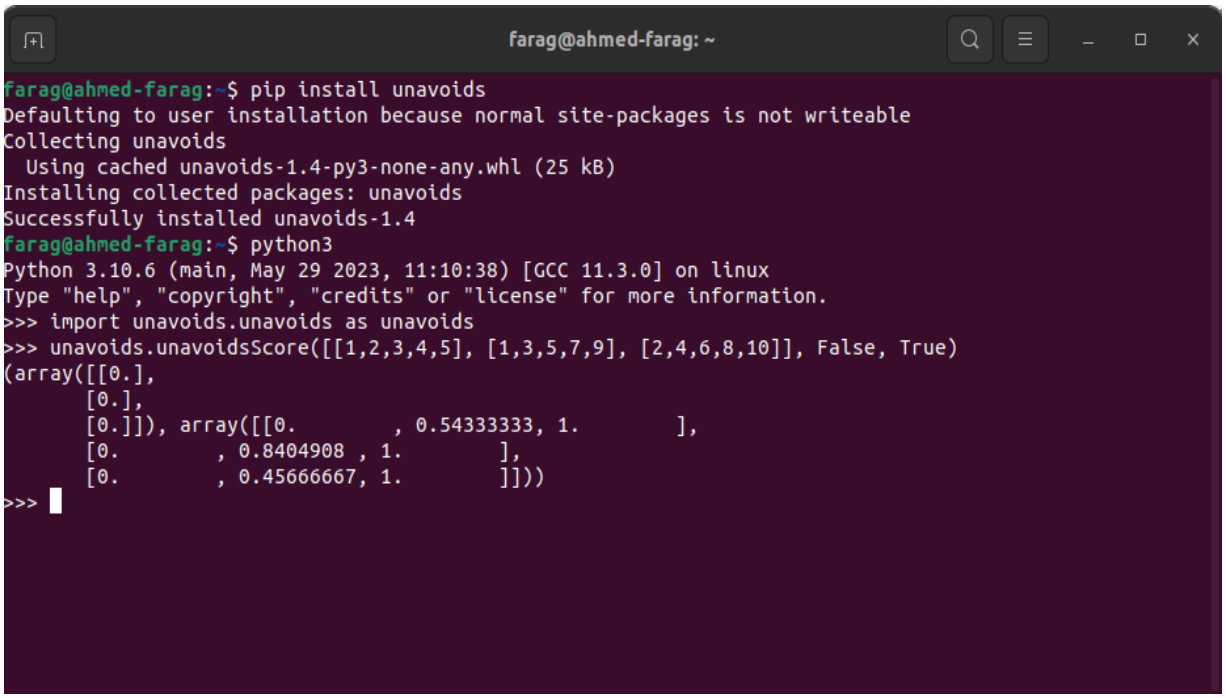

Figure 3.1 : UNAVOIDS Usage Example from PyPI.

3.2 Deploying as an API

3.2.1 Implementation

The API has been developed using Fast API, a modern, high-performance web framework for building APIs with Python. One of its priorities is data security, ensuring that all transmitted information is encrypted—effectively transforming it into a secret code to guard against unauthorized access. By employing an HTTPS connection— a secure variant of the internet protocol—data exchange between the API and its users is safeguarded. This robust security measure engenders a safe and reliable environment where users can conduct anomaly detection and visualization tasks with the confidence that their data remains confidential and secure. Both the API and its documentation are made available at the ISOT repository **[14] .**

Remarks

There are two unique versions of this API **[14]** . The first version implements data compression, ensuring all incoming request data and responses are compacted to enhance performance and minimize bandwidth consumption. Conversely, the second version operates without this compression functionality, offering an alternative in circumstances where data compression may not be necessary or desired.

Figure 3.2 outlines the code required to establish a FastAPI application with a router for UNAVOIDS. It defines various endpoints to manage different types of requests related to UNAVOIDS:

- The root endpoint ("/") returns a welcome message.
- The "/unavoids" endpoint accepts a list of data, processes it using the get_unavoids

function, and returns the result.

- The "/unavoids_compressed" endpoint does the same as the "/unavoids" endpoint, but it also compresses the result using the compress_as_gzip function before returning it.
- The "/unavoids_streaming_response" endpoint also processes a list of data with get_unavoids, but it returns the result as a streaming response in JSON format.
- The "/unavoids_streaming_compressed_response" endpoint combines the functionalities of "/unavoids_compressed" and "/unavoids_streaming_response". It processes a list of data, compresses the result, and returns it as a streaming response.

```python
# Importing necessary modules
from fastapi import APIRouter
from app.api.helpers import compress_as_gzip, get_unavoids, get_compressed_data_as_gzip
from fastapi.responses import StreamingResponse
import json
import io

# Define router
unavoids_router = APIRouter()

# Route for home page
@unavoids_router.get("/")
async def root():
    # Returns a welcome message
    return {"message": "Welcome to UnAvoid APIs"}

# Route for handling unavoidable data
@unavoids_router.post("/unavoid_handler")
async def unavoid_handler(dataSet: list):
    # Returns unavoidable data
    return get_unavoids(dataSet)

# Route for handling unavoidable data in compressed format
@unavoids_router.post("/unavoid_compressed")
async def unavoid_compressed(dataSet: list):
    # Returns unavoidable data compressed as gzip
    return compress_as_gzip(get_unavoids(dataSet))

# Route for handling unavoidable data streaming
@unavoids_router.post("/unavoid_streaming_response")
async def unavoid_streaming_response(dataSet: list):
    # Process unavoidable data
    data = get_unavoids(dataSet)

    # Generator function to stream json data
    def generate():
        # Yielding json data
        yield json.dumps(data).encode("utf-8")

    # Returns a streaming response of unavoidable data in json format
    return StreamingResponse(generate(), media_type="application/json")

# Route for handling unavoidable data streaming in compressed format
@unavoids_router.post("/unavoid_streaming_compressed_response")
async def unavoid_streaming_compressed_response(dataSet: list):
    # Process unavoidable data and compress it as gzip
    d = get_compressed_data_as_gzip(get_unavoids(dataSet))

    # Returns a streaming response of unavoidable data in compressed format
    return StreamingResponse(io.BytesIO(d['data'])
                             , media_type="application/octet-stream"
                             , headers=d['headers'])
```

Figure 3.2 : Code Snapshot of UNAVOIDS API Implementation

3.2.2 Usage Example

Refer to **Figure 3.3**, which presents an example of utilizing the UNAVOIDS API. In this scenario, the API is tasked with processing a subset of 750 observations from the Iris dataset **[8]**. The resulting output encompasses the coordinates of the Neighborhood Cumulative Distribution Functions (NCDFs) and the scores.

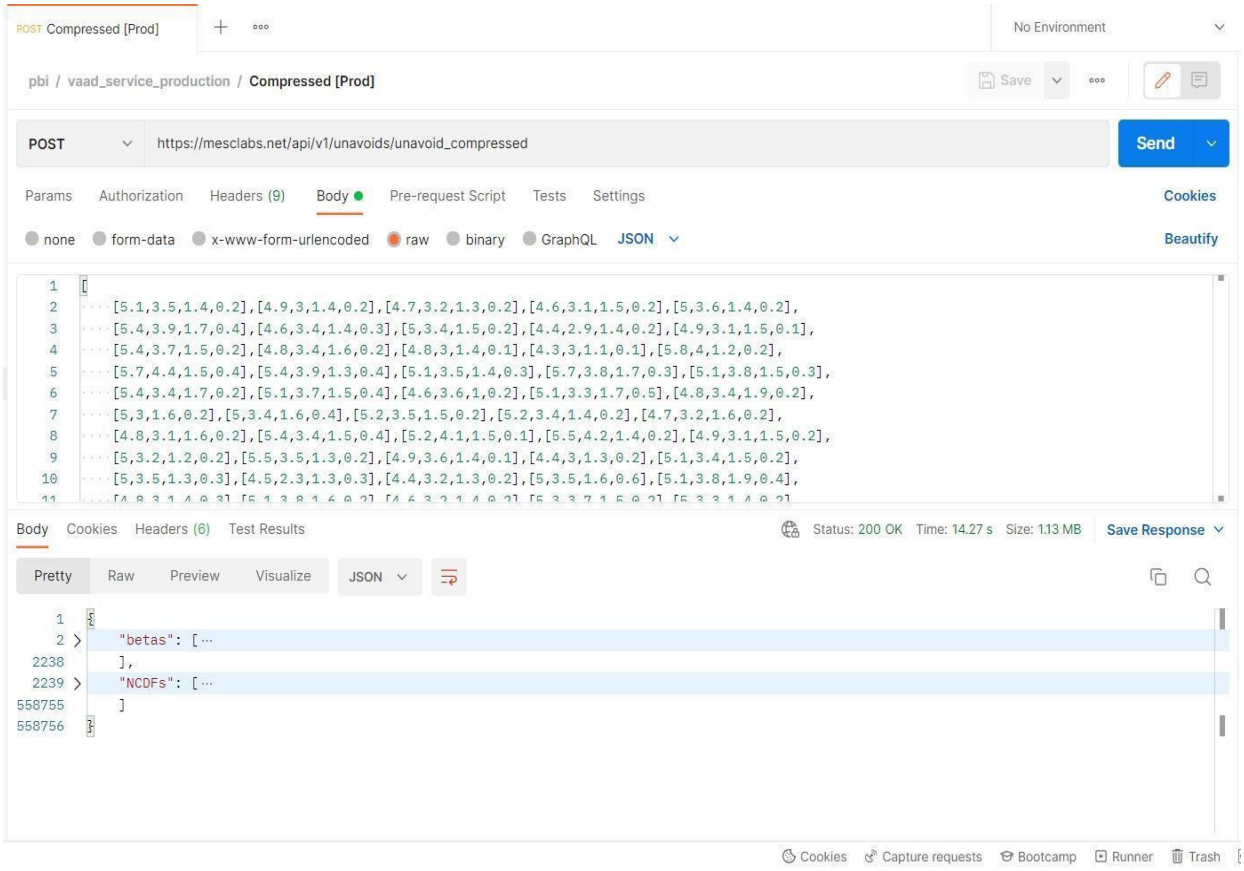

Figure 3.3: Utilizing UNAVOIDS API for Anomaly Detection in Iris Dataset.

3.2.3 Testing and Evaluating the Performance of the API

As part of the evaluation process for the UNAVOIDS RESTful API, it was essential to ensure its ability to handle large amounts of window data efficiently. Accordingly, the API was equipped with functionalities for data compression and decompression. This feature proved crucial given the large volumes of window data that could potentially exceed the allowed size for the request body.

In order to evaluate the efficiency of these operations, a range of experiments were carried out with different quantities of data points, specifically 750, 1500, and 3000. Table 3.1 delineates the overall duration encompassing both compression and decompression processes. Notably, these trials shed light on the efficacy and operational competence of the API's data compression and decompression abilities. It is important to mention, however, as evidenced by the data in the table, that the time spent on compression and decompression activities constitutes only a minor segment of the total time elapsed. This suggests that these operations were efficiently designed and do not significantly contribute to the overall processing time.

No Of Observations	UNAVOIDS time	Compression/ Decompression Time
750	13.5 sec	3.2 sec
1500	24.3 sec	7.9 sec
3000	52 sec	15.4 sec

Table 3.1: Total Elapsed Time (in seconds) for UNAVOIDS Functionalities and Compression/Decompression Across Different Numbers of Observations.

Table 3.2 offers an illustrative comparison of elapsed times between employing the API with and without data compression, while maintaining an identical number of observations. This

side-by-side examination accentuates the notable role of compression in enhancing the API's performance and reliability.

The process flow starts with the compression of the initial data, followed by its transmission to and receipt from the API. After reception, the data is decompressed and then processed using the UNAVOIDS' operation. The results of this operation are subsequently recompressed and dispatched back to the requester — a comprehensive sequence which demonstrates the full life cycle of the data handling within the system.

No Of Observations	API_Wihout_Compression	API_With_Compression
1200	38.38 sec	18.93 sec
2500	Failed	43.1 sec
5000	Failed	68.4 sec

Table 3.2 : Elapsed Time Comparison between API with Compression and API without Compression for the Same Number of Observations

3.3 Advantages of using UNAVOIDS through PyPI and API

The advantages of using UNAVOIDS through PyPI and API integration can be summarized below:

- Ease of Installation: The PyPI package ensures a straightforward installation process for UNAVOIDS removing the necessity for manual setup.
- Broad Accessibility: By being listed on PyPI, UNAVOIDS is readily available to a wide

range of users, enabling analysts, researchers, and developers to effortlessly incorporate its capabilities into their work.

- Smooth Integration: The RESTful API guarantees seamless integration of UNAVOIDS into existing cloud-based workflows, systems, and applications. This integration streamlines the processes of anomaly detection and visualization.
- Automated Workflows: The integration of the API allows users to automate anomaly detection and visualization tasks, thereby improving efficiency in security data analysis.

3.4 Limitations and Challenges Encountered

An ongoing challenge was handling large datasets when using the RESTful API **[14]**. The API has limitations on the amount of data that can be transmitted in a request or response body. To address this, a decision was made to choose between loading the data as a file stream or using compression and decompression techniques. Compression was chosen as it offered better performance and efficiency, enabling the effective transmission of large volumes of data within the API constraints.

Chapter 4 : Integration with DVP and Delivery as a PowerBI Custom Visual

4.1 Integration with DVP

4.1.1 Introduction to DVP (Data Visualization Platform)

The Data Visualization Platform (DVP) **[3], [7]** is a robust software that provides extensive visualization capabilities to researchers, scientists, and data analysts. A distinct advantage of DVP is its capability to seamlessly interact with Scientific Computing Environments (SCEs), functioning as an integrated system.

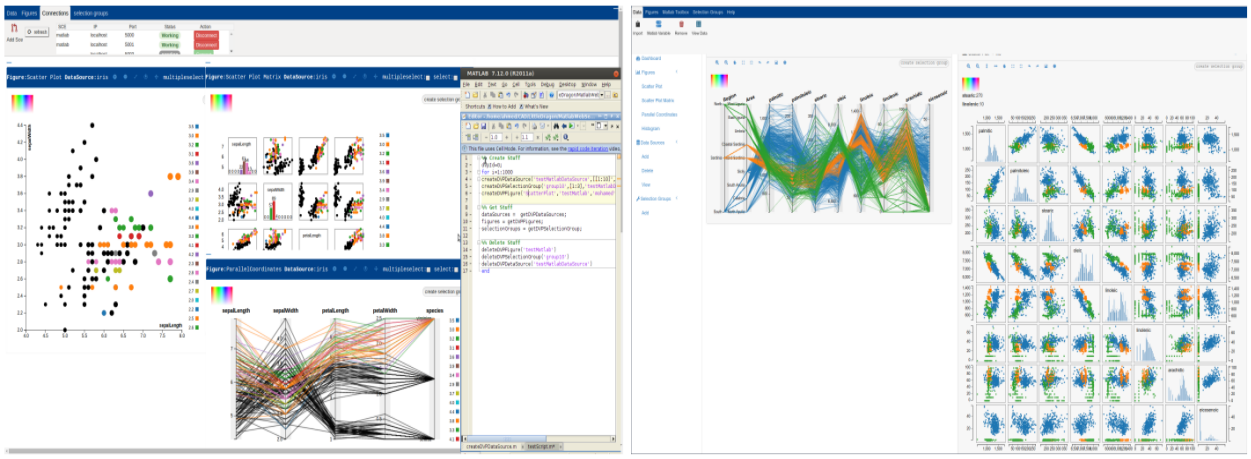

Figure 4.1: Two snapshots for DVP. Left: the DVP integrated locally to Matlab on the desktop and acts as a Matlab toolbox, where all variables are communicated from/to Matlab. Right: the online version of the DVP where all actions, interactions, and dynamics can be performed.

Moreover, DVP is highly extensible, offering an extensive range of plotting methods suitable for diverse scientific disciplines. Importantly, it permits users to devise their custom plotting methods using JavaScript, thereby enabling the integration of advanced techniques and fostering a dynamic

user community.

Figure 4.1 (as appears in [3]) depicts two snapshots of DVP. The left snapshot demonstrates DVP integrated with MATLAB on a desktop, functioning as a comprehensive toolbox with variable communication. The right snapshot presents the online interface of DVP, supporting all operations, interactions, and dynamics.

DVP, in its current form (version 1.0), provides four types of interactive plots: Scatter Plot, Scatter Plot Matrix, Parallel Coordinates, and Histogram, all of which allow for functionalities such as zooming, panning, brushing, and linking. We leverage the cloud-based version of DVP, which is publicly available, to develop the visualization plot of the NCDF of UNAVOIDS, and to integrate it with the remaining plots of DVP. The integration process necessitated re-construction and code refinement due to compatibility issues between the current software versions and the one that DVP was initially developed with. As a result, we created a new system, which we have termed Visualization-Aided Anomaly Detection (VAAD), following the same suggested acronym of the authors of UNAVOIDS, when they suggested VAAD as a future work. The system is accessible from the official repository of the ISOT laboratory **[14].**

4.1.2 Integrating UNAVOIDS with DVP to Deliver VAAD Tool

VAAD (Visualization-Aided Anomaly Detection), aims to aid analysts in anomaly detection by effortlessly integrating the UN-AVOIDS detection algorithm, the NCDF visualization space, and other conventional visualization methods within a single interactive environment. The back-end engine performs the detection algorithm, while the plots in the NCDF visualization space and

original feature space offer visual depictions of the data. This unified approach allows analysts to explore and examine anomalies in a cohesive and interactive manner.

Moreover, the UNAVOIDS API (explained in the previous chapter) serves as a critical component as the back-end engine of UNAVOIDS either, responsible for generating the NCDFs and anomaly scores based on the given data. Subsequently, the VAAD **[14]** system handles visualizing and interacting with these results, permitting analysts to gain profound insights into the identified anomalies.

The incorporation of UNAVOIDS in VAAD was achieved utilizing D3.js **[9]**, a potent JavaScript library designed for creating dynamic and interactive data visualizations. D3.js played a pivotal role in plotting and enabling interactions within all the VAAD plots.

As depicted in **Figure 4.2**, the diagram outlines the workflow of VAAD. It initiates with a data source supplying data to VAAD. Within VAAD, a plot, such as a scatterplot, is created. The data is then dispatched to the UNAVOIDS API, which returns NCDFs. These NCDFs are plotted in the UNAVOIDS Visual, which can interact with other visuals.

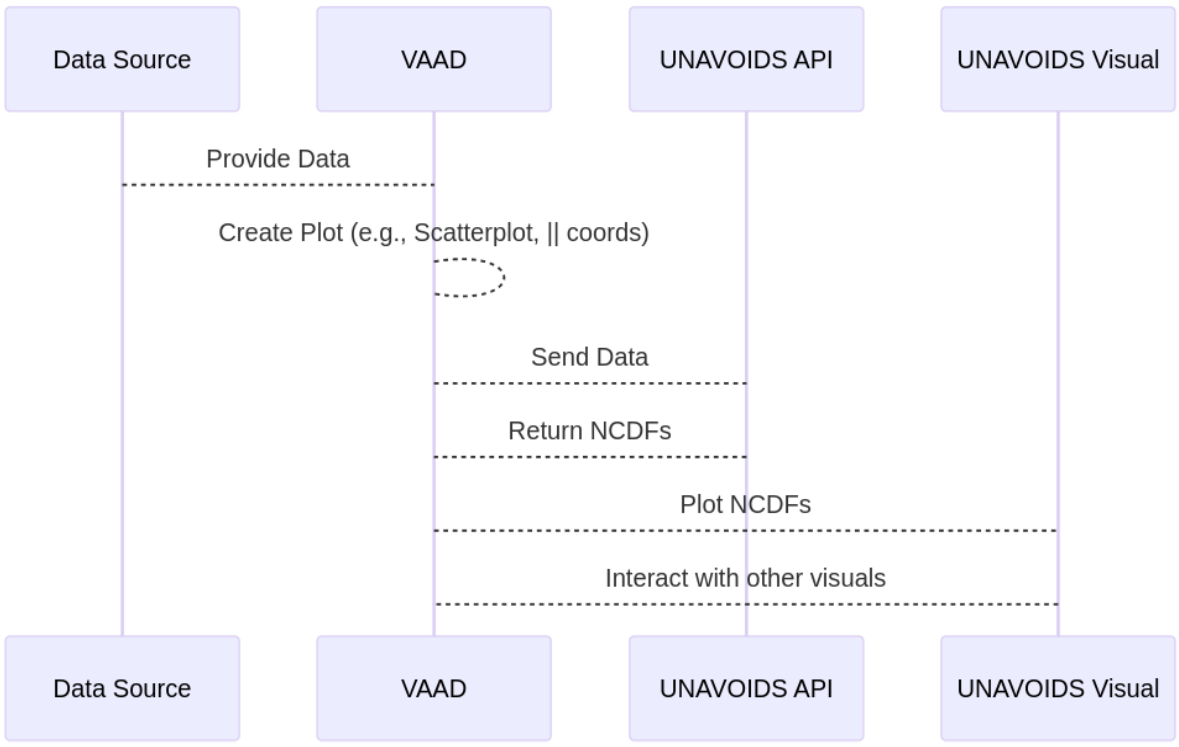

Figure 4.2: Flow Diagram Depicting the Process of VAAD Operation - From Data Upload and Preliminary Plotting to Unavoids API Interaction and Final Visualization

4.1.3 Practical Examples Demonstrating the Real-World Value of VAAD

Let's consider two practical examples that vividly illustrate the real-world value of VAAD. These examples utilize the MaliciousDoH-CSVs and BenignDoH-NonDoH-CSVs datasets **[**11**]**. By these two examples we emphasize the power of UNAVOIDS, the value of our implementations and deliverables, and the importance to data analysts.

In the first use case and As illustrated in **Figure 4.3**, We extracted a subset of data from the MaliciousDoH-CSVs Dataset and identified distinct anomalies in the parallel coordinates plot.

However, understanding these anomalies requires familiarity with parallel coordinates. As demonstrated in the figure on the left, UNAVOIDS accurately detects and labels anomalies as outliers.

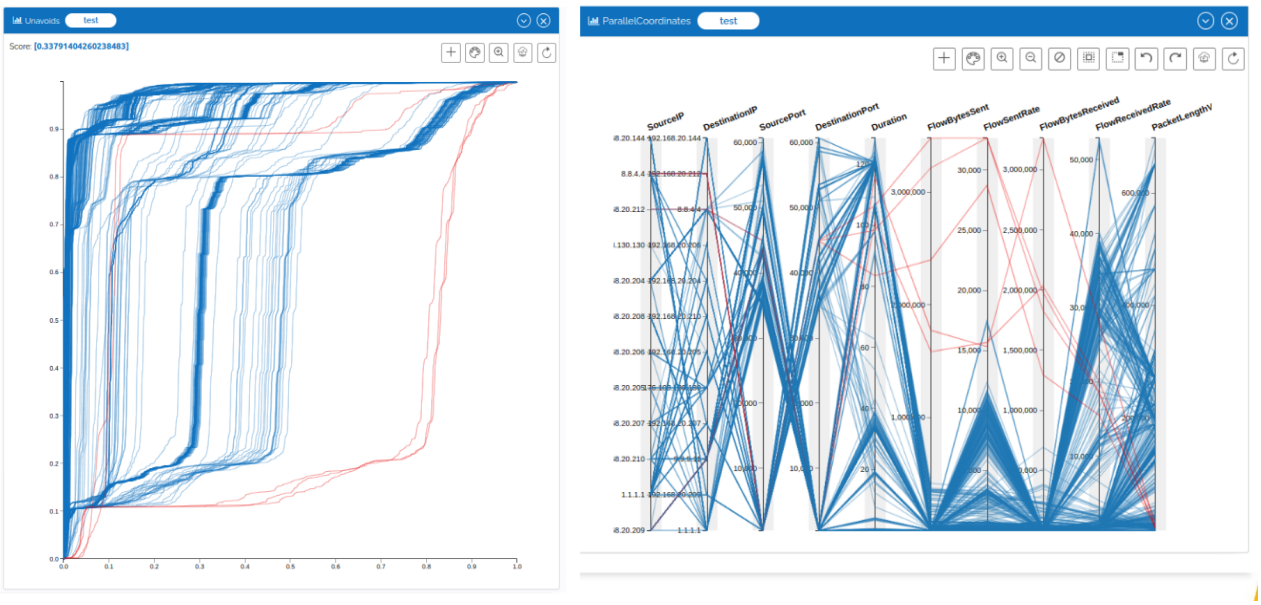

Figure 4.3 : Anomaly Detection Using VAAD: Clear Anomaly Visualization Across Multiple Figures.

As illustrated in **Figure 4.4**, Two anomalies that are not immediately observable in other visuals, are successfully detected by UNAVOIDS. This example underscores the superior anomaly detection capabilities of UNAVOIDS, even in scenarios where traditional visualizations may fail to identify irregularities.

When analyzing **Figure 4.3** in the context of the ground truth, we find that the anomalies identified by our UNAVOIDS system are indeed accurate, reflecting genuine outliers that align with the ground truth. Conversely, when examining **Figure 4.4**, we note that the anomalies identified by UNAVOIDS are false positives. While UNAVOIDS demonstrates a robust ability to uncover irregularities that might elude traditional visualization methods, this instance underscores the essential role of ground truth in refining our anomaly detection procedures. This observation does not diminish UNAVOIDS' capabilities; rather, it emphasizes the necessity for ongoing fine-tuning to align with ground truth, ensuring more precise and accurate anomaly detection. This scenario reiterates the importance of an iterative process in the development of robust visual analytic systems, guided by the unerring standard of ground truth.

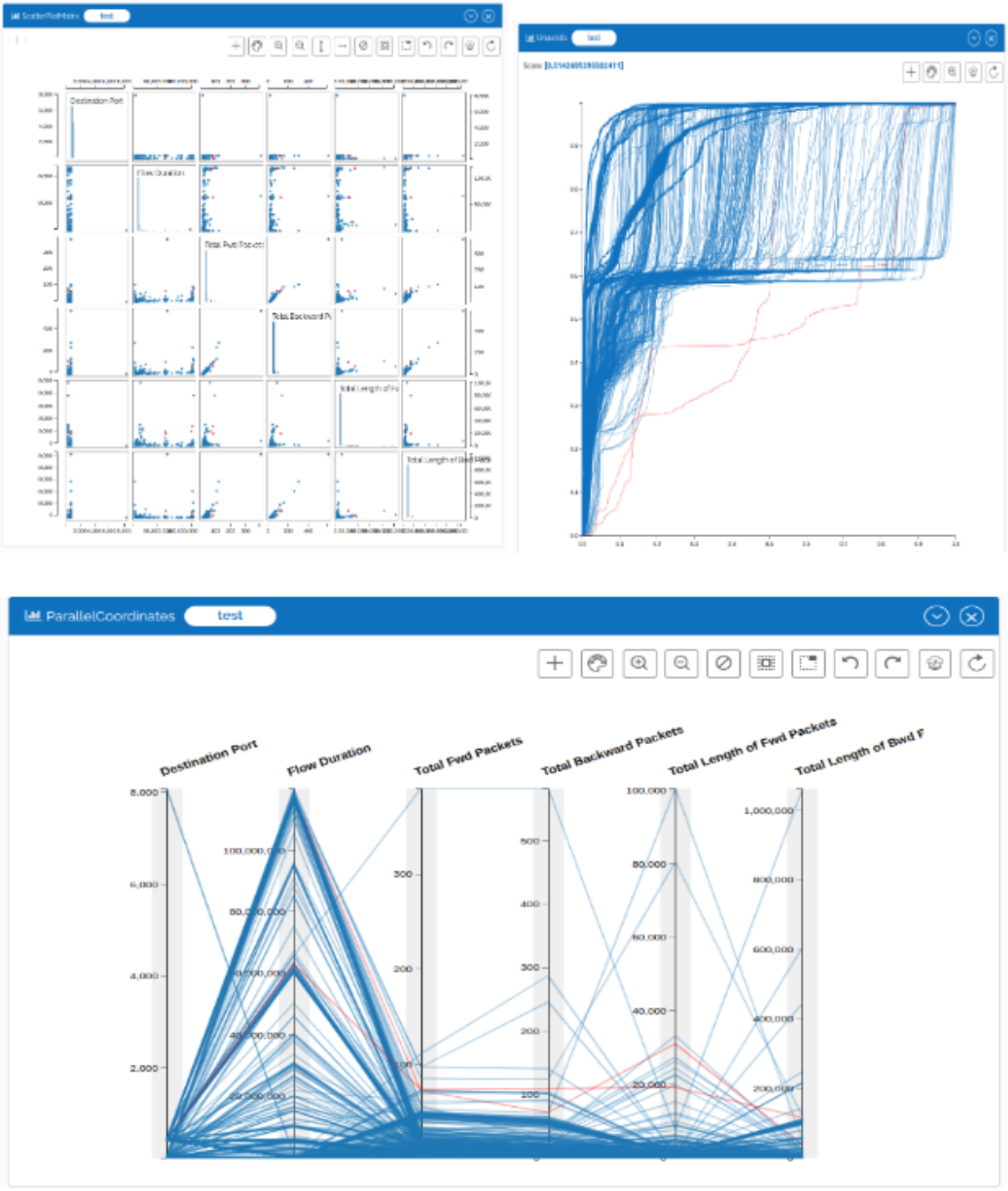

Figure 4.4 : Superior Anomaly Detection with VAAD: Identifying Anomalies Unobservable in Traditional Visuals

4.2 Developing UNAVOIDS as a Power BI Custom Visual

4.2.1 Introduction to Microsoft Power BI

Microsoft Power BI **[10]** is a comprehensive and interactive data visualization software product developed by Microsoft, primarily catering to business intelligence needs. As a key component of the Microsoft Power Platform, Power BI seamlessly integrates a collection of software services, applications, and connectors to transform disparate data sources into coherent and visually captivating insights. Users can input data from various sources, including databases, webpages, and structured files such as spreadsheets, CSV, XML, and JSON. The Power BI suite comprises both cloud-based BI services, known as "Power BI Services," and a desktop-based interface called "Power BI Desktop". This software solution offers a range of features, including data preparation, data discovery, interactive dashboards, and data warehousing capabilities. A distinguishing feature of Power BI is its capability to integrate custom visualizations created by users, enabling them to extend its functionalities with tailored visuals to suit their specific needs. With our development of the UNAVOIDS' NCDF space as a Microsoft PowerBI visual (as explained in the next section), UNAVOIDS will be published on the Power BI Visuals Marketplace, allowing users to effortlessly incorporate it into their Power BI reports without the need for local environment setup. This means that users can harness the power of UNAVOIDS alongside other Power BI visuals in a seamless and fully interactive manner.

4.2.2 Developing of UNAVOIDS as a Power BI Custom Visual

We created a Microsoft Power BI custom visual to display the NCDF space of UNAVOIDS from scratch. During the implementation, special attention was given to ensuring the interactivity of the UNAVOIDS visual with other Power BI visuals in the same environment (call report in PowerBI

jargon), all sharing the same data source. As depicted in **Figure 4.5**, the UNAVOIDS visual in the Power BI report exhibits bidirectional interactivity with other visuals, allowing users to seamlessly explore and analyze data across multiple visualizations within the report.

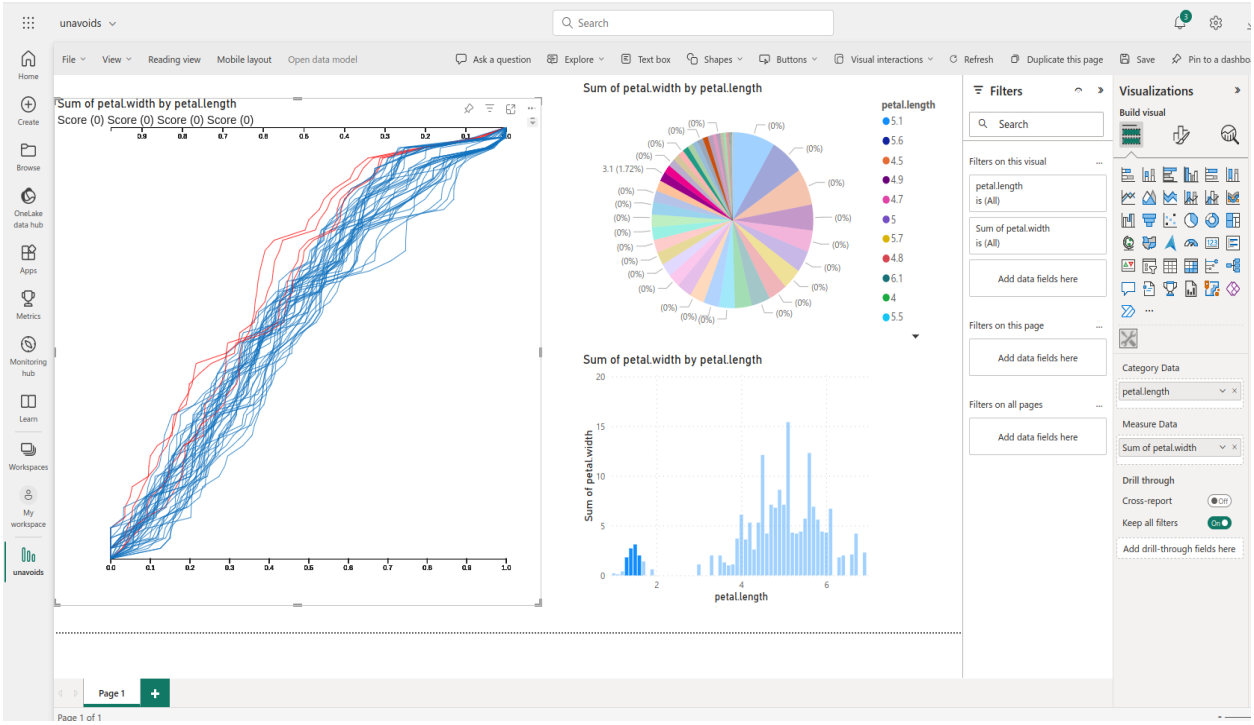

Figure 4.5 : Interactive Integration of UNAVOIDS Visual with Power BI Report

As depicted in **Figure 4.6**, we have a snippet of code that is used to create the custom visual. This code snippet is a function named `unavoids` that is used to visualize the Neighborhood Cumulative Distribution Function (NCDF) space of a given dataset. The function takes four parameters: `ncdfs` (the NCDFs of the data), `width` and `height` (the dimensions of the visualization), and `selected_indexes` (the indexes of selected data points).

Next, the function defines the scales for the x and y axes based on the data's range. It then draws

lines for each data point in the NCDF space, color-coding them based on their group and setting their width. All other implementation details are available on the official repository of this project [14].

```
# Importing required modules
import powerbi from "powerbi-visuals-api"
import "./../style/visual.less"

# Importing PowerBI Visuals API interfaces
import VisualConstructorOptions = powerbi.extensibility.visual.VisualConstructorOptions
import VisualUpdateOptions = powerbi.extensibility.visual.VisualUpdateOptions
import IVisual = powerbi.extensibility.visual.IVisual

import * as d3 from "d3"

# Defining the Visual class
export class Visual implements IVisual {
    private previousDataViews: DataView[] = []
    private svg: d3.Selection<SVGGElement, any, any, any>
    private divElem: d3.Selection<HTMLElement, any, any, any>
    private real_data: number[][]
    private selected_data: number[][]
    private selected_indexes: number[]
    private NCDFs: number[][]
    private betas: number[][]
    private host: powerbi.extensibility.visual.IVisualHost
    private dataViews
    private selectionManager: powerbi.extensibility.ISelectionManager
    private element: HTMLElement
    private brushSelection: d3.Selection<any, any, any, any>
    private brush: d3.BrushBehavior<any>

    # Constructor method for the Visual class
    constructor(options: VisualConstructorOptions) {
        /* ... */
    }

    # Method to handle updates to the visual
    public update(options: VisualUpdateOptions) {
        /* ... */
    }

    # Method to draw data
    public drow_data(){
        /* ... */
    }


    # Method to compare two arrays of numbers
    public isEqual(a: number[], b: number[]): boolean {
        /* ... */
    }


    # Method to handle unavoidable actions
    public unavoids(ncdfs, width, height, selected_indexes) {
        /* ... */
    }

    # Method to handle the end of the visual brush
    private VisualbrushEnded(event, lines): void{
        /* ... */
    }

    # Method to update other visuals
    private UpdateOtherVisualsFull(selected_indexes): void {
        /* ... */
    }

    # Method to fetch Unavoids data
    private fetch_unavoids_data(data){
        /* ... */
    }
}
```

Figure 4.6 : Custom Visual Creation for Microsoft PowerBI: A Code Snapshot

4.3 The Impact of UNAVOIDS Integration with DVP and Power BI

The integration of UNAVOIDS with both DVP **[3], [7]** and Power BI **[10]** can provide a significant impact on data analysis and visualization workflows. By combining the anomaly detection capabilities of UNAVOIDS with the interactive and customizable features of a visualization platform (either DVP, Microsoft PowerBI, or any other future development), users are empowered to gain deeper insights and uncover hidden patterns in their data. This enables users to leverage the strengths of UNAVOIDS alongside other plots or visuals in the visualization system creating a comprehensive and interactive data analysis experience.

4.4 Limitations and Challenges Encountered

The development of UNAVOIDS encountered several challenges that required innovative solutions. One significant challenge was creating an interactive visual representation due to the unique nature of the data. Each curve in the UNAVOIDS visual represents a single observation, and all the values fall between 0 and 1. Overcoming this challenge involved mapping the indices of the curves and implementing a reflection mechanism to establish interactivity between the selected curve and other visuals. This required careful consideration and a creative approach to ensure an effective and intuitive user experience.

Chapter 5 : Conclusion and Future Directions

5.1 Summary of Findings and Achievements

This project delivered important software tools and packages to the field of anomaly detection, in particular, and the field of data visualization, in general. The integration of UNAVOIDS **[1]**, **[2]** with various platforms, including the development of a PyPI package, the implementation of a RESTful API , the delivery of a visualization platform (VAAD), and the creation of a Power BI custom visual, all have provided users with versatile and user-friendly tools for detecting and visualizing anomalies in their data. All implementations and documentations are available on the official repository of the ISOT laboratory **[14]**.

In particular, the delivery of the VAAD software and the PowerBI visual, have enhanced the visualization capabilities of UNAVOIDS. By seamlessly integrating UNAVOIDS in these platforms, users can explore and interact with anomaly visualizations in a dynamic and intuitive manner. The interactive features, such as zooming, panning, and brushing, provide users with enhanced flexibility and control over their data analysis.

5.2 Future Work and Potential Enhancements

Looking to the future, there are several directions for further improvement and expansion. One avenue is to continue enhancing the performance and scalability of UNAVOIDS, particularly for analyzing large-scale and real-time data. Additionally, ongoing research and development efforts can focus on expanding the range of visualization techniques and methods within DVP **[3]**, **[7]** to cater to the diverse needs of different scientific fields.

Furthermore, collaboration with domain experts and practitioners can provide valuable insights and use cases to further refine and optimize UNAVOIDS for specific applications and industries. Continued user feedback and engagement will be essential in driving the evolution of UNAVOIDS, ensuring its relevance and effectiveness in addressing emerging challenges in anomaly detection and data analysis.

5.3 Closing Remarks.

In summary, the development of UNAVOIDS deliveries has been characterized by innovation, collaboration, and the pursuit of excellence in anomaly detection and visualization. By merging advanced algorithms, user-friendly platforms, and seamless integration capabilities, UNAVOIDS has become a valuable tool for data analysts, researchers, and scientists across various domains. We hope that UNAVOIDS will continue to empower users, enable new discoveries, and drive advancements in anomaly detection and data analysis.

References

[1] Yousef, W. A., Traoré, I., & Briguglio, W. (2021). UN-AVOIDS: Unsupervised and Nonparametric Approach for Visualizing Outliers and Invariant Detection Scoring.

[2] Yousef, W. A., Traore, I., & Briguglio, W. (Pending). Unsupervised and nonparametric approach for visualizing outliers and invariant detection scoring. US Patent 63168686.

[3] Yousef, W. A., Abouelkahire, A. A., Marzouk, O. S., Mohamed, S. K., & Alaggan, M. N. (2019). DVP: Data visualization platform. arXiv preprint arXiv:1906.11738.

[4] ISOT Laboratory. (n.d.). UNAVOIDS Code. Retrieved from https://github.com/isotlaboratory/UNAVOIDS-Code

[5] Code Ocean. (n.d.). UNAVOIDS demo. Retrieved from https://codeocean.com/capsule/2122579/tree/v1

[6] DiRienzo, N. (n.d.). Intro to anomaly detection. Retrieved from https://bookdown.org/ndirienzo/ista_321_data_mining/intro-to-anomaly-detection.html

[7] MESCLABS. (n.d.). DVP. Retrieved from https://mesclabs.com/dvp/

[8] Kaggle. (n.d.). iris.csv dataset. Retrieved from https://www.kaggle.com/datasets/saurabh00007/iriscsv

[9] D3.js. (n.d.). Retrieved from https://d3js.org/

[10] Microsoft. (n.d.). Power BI. Retrieved from https://powerbi.microsoft.com/

[11] MontazeriShatoori, M., Davidson, L., Kaur, G., & Lashkari, A. H. (2020). Detection of DoH Tunnels using Time-series Classification of Encrypted Traffic. In The 5th IEEE Cyber Science and Technology Congress, Calgary, Canada, August 2020.

[12] M. Montazeri Shatoori, L. Davidson, G. Kaur, and A. H. Lashkari, “Detection of DoH tunnels using time-series classification of encrypted traffic,” in Proc. IEEE Int. Conf. Dependable, Autonomic Secure Comput., Int. Conf. Pervasive Intell. Comput., Int. Conf. Cloud Big Data Comput., Int. Conf. Cyber Sci. Technol. Congr. (DASC/PiCom/CBDCom/CyberSciTech), Aug. 2020, pp. 63–70.

[13] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy, 2018, pp. 108–116.

[14] Information Security and Object Technology (ISOT) Research Lab. (n.d.). Retrieved July 6, 2023, from https://github.com/isotlaboratory/A-UNAVOIDS-PYPI