AN UNSUPERVISED BASED APPROACH TO DETECTING ANOMALIES IN HAZARD MONITORING NETWORKS.

by

JOHN HOOFT TOOMEY

A THESIS

Presented to the Department of Computer Sciences
in partial fulfillment of the requirements for the degree of
Bachelor of Science

June 2024

THESIS ABSTRACT

John Hooft Toomey

Bachelor of Science

Department of Computer Sciences

June 2024

Title: An Unsupervised Based Approach to Detecting Anomalies in Hazard Monitoring Networks.

Our society relies heavily on various critical infrastructures (e.g., hazard monitoring networks in the state of Oregon) dedicated to monitoring natural hazards (e.g., wildfires). The hazard monitoring networks comprise sensors and cameras interconnected by wide-area backbones, where failures can result in significant societal and economic loss. Consequently, monitoring of state of health of hazard monitoring networks is paramount.

While Machine Learning (ML) stands as one of the most groundbreaking advancements in Computer Science, its application to detecting and predicting anomalies in hazard monitoring networks is fraught with challenges. For one, each data transfer within hazard monitoring networks represents a distinct relationship, the creation of labeled training datasets for each connection is not feasible. Second, in the absence of labeled data, developing ML models to detect anomalies is impractical.

The main objective of this work is to enhance the robustness of hazard monitoring networks by addressing existing limitations in ML-driven anomaly detection. To this end, we propose the Anomaly Detection Tool (ADT)

framework which uses weakly supervised learning techniques (such as heuristics and lightweight ML algorithms) to capture distinct relationships, culminating in weak labels. These weak labels can then be mapped to one stronger label, thus creating a dataset containing labeled anomalies which can be used to notify network operators of potential network failures. Our research stands to benefit the reliability and security for Critical Infrastructures.

Note: The source code of the ADT framework is openly available to the community at: https://github.com/johnhooft/Anomaly-Detection-Tool

CURRICULUM VITAE

NAME OF AUTHOR: John Hooft Toomey

UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon, Eugene, OR, USA

DEGREE AWARDED:

Bachelor of Science, Computer Sciences, 2024, University of Oregon
Secondary Academic Discipline, Mathematics, 2024, University of Oregon

AREAS OF SPECIAL INTEREST:

Anomaly Detection
Machine Learning
Network Measurements

PROFESSIONAL EXPERIENCE:

Machine Learning Researcher, Oregon Networking Research Group,
Summer 2023 - Summer 2024
Computer Science Intern, Oregon Hazards Lab, Spring 2022 - Fall 2023

ACKNOWLEDGEMENTS

I would like to thank my research advisor and mentor Ram Durairajan for his continued guidance and support throughout the past year. His knowledge and encouragement were instrumental in the completion of this research. With this being my first research endeavour, I could not have overcome the many challenges I faced without his support.

I would also like to thank my parents who have always believed in me, encouraged me, and gave me confidence to peruse challenges outside my comfort zone. They are forever an inspiration to me.

I would lastly like to thank the University of Oregon Computer Science Department. Over the past four years, I have experienced significant growth both personally and professionally as a computer scientist, thanks in large part to the dedicated professors and the invaluable knowledge they imparted. I will always cherish my time here and the lasting impact it has had on my development.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Critical infrastructure (CI) networks (communication networks for emergency and public services) are increasingly relied upon to maintain the health, safety, security, and economic well being of citizens and governments. The increased trust in these networks to operate in critical scenarios means that we must maintain a certain level of confidence in these networks to operate as expected. To achieve this, the common solution is to devise a system to monitor, detect, and report anomalies in the network performance data. In the past this system has mainly been overseen by human operators, who monitor the 'State of Health" of a CI network, and flag anomalies they encounter. Employing human operators to continually monitor network performance data for anomalies is inefficient, it requires significant time and resources and after exposure to multiple anomaly events, operators are prone to fatigue. Thus, the development of an automated solution such as a decision support framework to help network operators identify anomalies in their CI is highly beneficial.

Real time hazard monitoring networks are an increasingly relevant subdomain of CI, and multiple research projects have been created to further the development of these CI hazard networks. For example, the University of Oregon and University of Washington collectively operate the Pacific Northwest Seismic Network (PNSN), to monitor earthquake and volcanic activity across the Pacific Northwest. The Northwestern University lead SAGE project is developing a CI network to help understand the impacts of global urbanization, natural disasters, such as flooding and wildfires, and climate change on natural ecosystems and city infrastructure. Finally, the Oregon Hazards Lab (OHAZ), a University of Oregon, a state affiliated

research laboratory, has developed a hazard monitoring network consisting of network devices such as radios, routers, and cell modems, to transmit data generated by sensors and cameras that detect potential wildfires and earthquakes. In all these examples, it is crucial to ensure the CI hazard monitoring network is operable during a natural hazard event. Currently there is no standard method to achieve this, and that is why it's prudent to develop a decision support framework to help assist network operators in monitoring these CI networks for data anomalies.

Previous attempts have been made to deploy systems that help increase the decision-making capabilities of the network operators, but due to their reliance on heuristic methods (discussed further in detail in Chapter 2), they lack the ability to detect non statistical anomalies. To address this limitation, previous researchers have used Machine Learning (ML) as a solution to detecting more subtle patterns within network performance data. While ML offers significant potential for anomaly detection in hazard monitoring networks, it faces major challenges. The unique relationships in each data transfer make it impractical to create labeled training datasets, and without labeled data, developing ML models using traditional supervised learning techniques is not possible.

Our solution to this problem is to employ the use of multiple lightweight unsupervised ML algorithms in combination with heuristic techniques to develop a novel decision support framework to help network operators to detect anomalies in time series data without the need for labeled training datasets. The proposed Anomaly Detection Tool (ADT) framework utilizes a two-pass technique aimed at increasing the accuracy of the weak labels produced by the framework. The unsupervised learning based approach of this framework addresses the limitations

that traditional supervised learning techniques have in network applications. While the methods used in this work are found to more reliable identify varying types of anomalies, the accuracy of the framework is still highly dependent on the quality and noise of the data. Furthermore, the labels produced by the six lightweight ML models and processed under the naive assumption that they all performed equally. Ideally, the decision of each ML model would be weighted depending on its performance which would lead to more accurate results. The ADT framework provides a strong starting point upon which future work could be conducted, such as weighting the decisions made by each model, supporting anomaly detection in multi-variate data sets, and integrating an end-to-end version of this framework to run on top of real time hazard monitoring networks. Our research stands to greatly benefit both the protection of hazard monitoring networks and the evolution of ML applications, offering increased reliability and security for hazard CI's.

CHAPTER II

BACKGROUND AND MOTIVATION

2.1 Overview

This project was driven by the acknowledged importance of real-time hazard monitoring CIs, the challenges they encounter that could jeopardize their functionality, and the possible repercussions if a hazard monitoring CI becomes inoperative during a natural hazard event. An increasing number of federally funded projects are focused on developing real-time hazard monitoring networks. Notable examples include ShakeAlert, an earthquake early warning system Given et al. (2018); ALERTWildfire, a wildfire monitoring network K. Smith et al. (2016); the Ocean Observatories Initiative (OOI), which conducts oceanic observations L. M. Smith et al. (2018); and SAGE, an NSF Midscale Research Infrastructure project *A Software-Defined Sensor Network* (n.d.), among others. With the growing efforts to develop these networks, it is equally crucial to advance and enhance the tools that maintain network operations and safeguard them from potential issues. This research aims to develop a decision support framework to assist network operators in anomaly detection, thereby improving the resiliency of hazard monitoring networks.

2.2 Data Sets Used

In collaboration with OHAZ, we leveraged their state of health monitoring system for their hazard monitoring network to evaluate the deficiencies in existing standard methods of network monitoring and anomaly detection. OHAZ's hazard monitoring network comprises 34 cameras and 220 seismic stations across the state

of Oregon Hamers (2023), interconnected by network devices that transmit data back to OHAZ for processing. OHAZ monitors the state of health of the network by frequently querying performance data from their network devices, which include but are not limited to routers, radios, cell modems, switches, solar controllers, and the sensors and cameras themselves. This performance data consists of metrics such as round-trip times, receiving and transmission rates, and signal strength, which can be used to identify and diagnose potential problems in the network. To evaluate the model's performance across diverse data types, we employed a range of datasets that varied in timespans, sample sizes, and sampling rates. Time spans ranged from a couple of days to over a year, and sampling rates ranged from intervals of every five minutes to every thirty minutes. This approach ensured that our model was robust and effective across multiple types of datasets, rather than being tailored to a single dataset type.

The initial state of health monitoring system OHAZ utilized was the open-source software Nagios, which runs periodic checks on critical network device performance parameters and notifies network operators when a performance parameter has reached critical levels *Nagios Open Source* (2024). While Nagios works fine to query and log network device performance data, its ability to provide decision support through anomaly detection falls short of what is needed by network operators. To understand why anomaly detection is challenging it is important to understand the two main classifications of anomalies.

2.3 Types of Anomalies Considered

The two main types of anomalies are global (point) anomalies and contextual anomalies. A global anomaly refers to one or several individual cases that are

deviant with respect to the rest of the data. A contextual anomaly appears normal at first, but is deviant when an explicitly selected context is taken into account Foorthuis (2021).

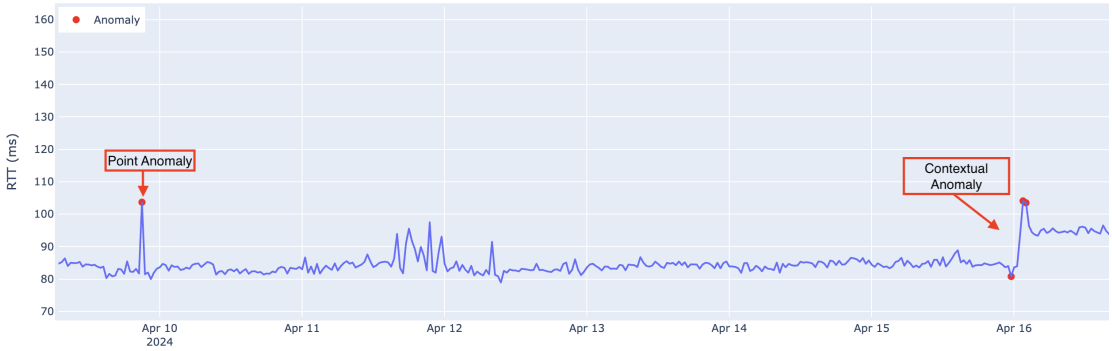

Figure 1. Plot of Round-Trip Time (RTT) data containing Point and Contextual anomalies.

Figure 1 above contains a labeled example of a point anomaly and contextual anomaly. The point anomaly on April 10th lies far above the average of the data, thus making it an obvious outlier. While on April 16th, the contextual anomaly is represented by an upward shift in the pattern of the data. The points marked as contextual anomalies are not necessarily outside of the statistical average of the data around it, but the change in the data's distribution and the increase in the average distance between data points caused by the shift in the expected pattern results in a contextual anomaly. More specifically, a global anomaly is determined by comparing the statistical value of a given data point to the statistical average of the rest of the data set. Whereas contextual anomalies are determined by considering both the statistical value of the data point and the spatial context that the point sits in in relation to the data points around it.

With global and contextual anomalies in mind, we can analyze the alerting services of Nagios to understand its shortcomings in detecting contextual anomalies and thus providing useful decision support for network operators. The Nagios monitoring system utilizes operator-determined threshold values to decide when to send alerts, notifying the operator if a data point is above or below the specified minimum or maximum threshold. This is problematic both because the threshold value is purely heuristic and not determined by any sort of analytic technique, and because threshold detection does not effectively detect contextual anomalies. The nature of contextual anomalies means that they can lie within the statistical average of the data and fall under the potential threshold value.

2.4 Prior Efforts and their Limitations

Prior efforts have sought to develop techniques to address anomaly detection limitations in current systems like Nagios. Some of these efforts have focused on applying ML techniques to detect anomalies in a network. However, as detailed in Lavinia, Durairajan, Rejaie and Willinger (2020), there is a significant lack of data, particularly labeled data, or a scalable method for labeling data. This deficiency renders traditional supervised ML approaches impractical for network applications.

In conventional supervised ML, a model is trained and evaluated on a set of labeled data, enabling it to learn the patterns between the data features and the corresponding labels Goodfellow, Bengio and Courville (2016). However, in a hazard monitoring network like the one utilized by OHAZ, which comprises various types of connections (radio, cellular, or satellite), each connection has its own unique characteristics. For example, a transfer rate or round-trip time that

is normal for one device connection might be abnormal for another. To apply ML models to every connection in a network, a labeled dataset must be created for each connection. This requirement is highly impractical for a network such as OHAZ's, which contains numerous connections, making the generation of labeled datasets for each connection extremely time-consuming. Therefore, to effectively apply ML techniques to networks, an alternative approach must be considered. Instead of using supervised learning methods, unsupervised approaches can be employed. Unsupervised ML methods allow models to learn with no human-labeled data. This approach circumvents the issue of generating labeled datasets, allowing for models to be created for multiple unique connections.

CHAPTER III

DESIGN AND IMPLEMENTATION

3.1 Overview of Anomaly Detection Tool Framework

With contextual and global anomaly detection in mind and motivated by the above-mentioned importance of anomaly detection in CI hazard monitoring networks, we designed the Anomaly Detection Tool (ADT) Framework. While conventional supervised learning models require significant labeling efforts, the proposed ADT framework allows for the training of models without pre-labeled data. Additionally, to enhance accessibility and improve the effectiveness of the proposed framework as a decision support tool, ADT includes a user-friendly web browser dashboard, enabling network operators to apply data science and ML techniques to their network data without needing prior experience. As detailed in Figure 2, the ADT Dashboard interacts with the ADT framework through API calls.

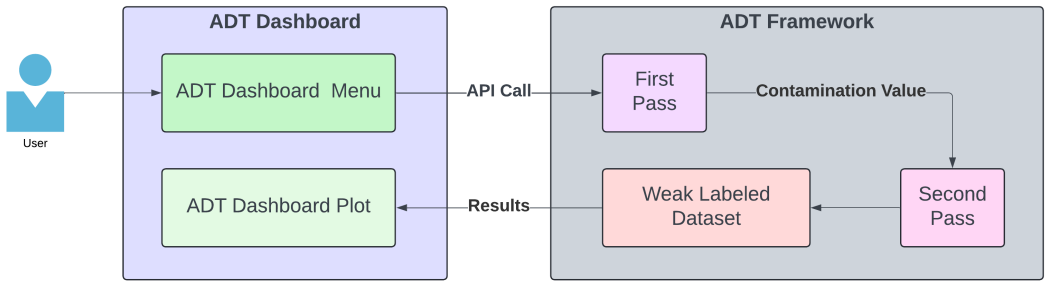

Figure 2. ADT Dashboard Framework Interaction Model

Unlike previous approaches that heavily rely on statistical analysis and basic machine learning models, this framework employs a two-pass technique

incorporating six lightweight unsupervised machine learning algorithms. The initial pass utilizes K-Nearest Neighbors (KNN) and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) for preliminary anomaly classification, which aids in estimating a contamination value *DBSCAN* (n.d.); *KNeighborsClassifier* (n.d.). This value is then used in the second pass for more precise anomaly classification, which includes the additional models: Isolation Forest, One-Class SVM, Local Outlier Factor (LOF), and Elliptical Envelope *EllipticEnvelope* (n.d.); *IsolationForest* (n.d.); *LocalOutlierFactor* (n.d.); *OneClassSVM* (n.d.). This multi-algorithm, multi-pass strategy is designed to enhance the robustness and accuracy of both contextual and global anomaly detection in varied and complex network connections. Overall, this work offers a potentially superior approach by reducing the dependency on labeled data, enhancing user accessibility, and employing a comprehensive, two-pass algorithmic strategy that promises significant improvements in anomaly detection compared to current systems like Nagios.

3.2 Initial Design Efforts

The preliminary approach to implementing the ADT framework consisted of applying basic statistical analysis techniques in conjunction with simple ML models to detect anomalies in network performance data. Inspired by previous research tackling a similar problem detailed in "Denoising Internet Delay Measurements using Weak Supervision" Muthukumar and Durairajan (2019), we used the K-means and K-Nearest-Neighbors (KNN) algorithms along with a mean plus standard deviation technique to flag potential anomalies. The K-Means clustering algorithm attempts to cluster the data into similar groups; then data

points can be classified as anomalous or not depending on how far away they are from their cluster centroid. The K-Means clustering algorithm was implemented with the Tslearn TimeSeriesKMeans library *tslearn.clustering.TimeSeriesKMeans — tslearn 0.6.3 documentation* (n.d.), where after fitting the model to the data, it can be transformed to a cluster-distance space. The cluster-distance space allows for the distance from a given data point to its parent cluster to be extracted. From this, we were able to collect all the distances from every data point to its parent cluster, and data points that had a distance greater than a standard deviation above the mean were marked as anomalies. While this approach to determining anomalies with a K-Means algorithm is potentially viable, its performance was less than ideal due to the nature of time series data. The tslearn Time Series K-Means algorithm uses Dynamic Time Warping (DTW) in an attempt to cluster time series subsequences that are similar in shape and temporal pattern *tslearn.clustering.TimeSeriesKMeans — tslearn 0.6.3 documentation* (n.d.). While this approach can be effective in certain cases, the paper "Clustering of Time Series Subsequences is Meaningless" raises concerns about the meaningfulness of clustering subsequences in time series data due to issues such as noise, high dimensionality, and the lack of a meaningful distance metric Keogh and Lin (2005). Because of this, we opted not to use the K-Means algorithm in the ADT Framework. The KNN approach we utilized was also centered around classifying the data points based on their spatial context relative to the data points around it, specifically what was the average distance from a given point to its K closest neighboring data points. Since the KNN algorithm was utilized in the final version of the framework, it is detailed in the framework architecture section below. Finally, the statistical approach was also deemed too

heuristic and unable to detect contextual anomalies, thus was also omitted from the framework.

3.3 Anomaly Detection Tool Framework Architecture

The ADT framework is designed by utilizing six lightweight ML algorithms, in combination with a 2-pass technique to produce a weakly labeled dataset.

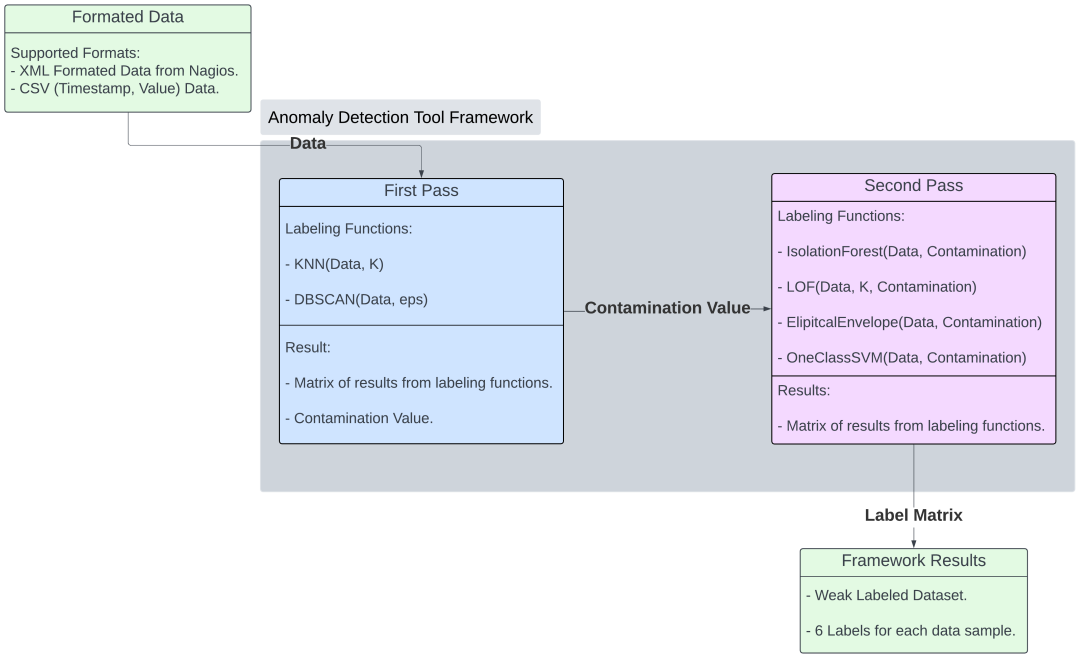

Figure 3. Anomaly Detection Tool Framework Architecture

Figure 3 above details a higher-level architecture model for the final version of the ADT framework. The user has the option of formatting their data into CSV (Timestamp, Value) format, or if their data has been queried from the Nagios RRD database, it can be left in its native XML format. This framework centers around the concept of a two-pass technique, where the data is run through the first pass containing the K-Nearest Neighbors (KNN) and Density-Based Spatial

Clustering of Applications with Noise (DBSCAN) algorithms. The first pass defines a contamination value, which is an estimate of the percentage of data points in the dataset that are deemed anomalous by the two algorithms. This contamination value is utilized by the algorithms in the second pass to help produce more accurate anomaly classification.

3.3.1 First Pass

The KNN algorithm in the first pass utilizes the Scikit-learn KNeighborsClassifier implementation and has a hyperparameter K, which represents the number of nearest neighbors it will consider when fitting the model to the data *KNeighborsClassifier* (n.d.). A K value of 3 would mean that for each data point, it would identify its 3 nearest neighboring data points. Since this framework is designed to be able to quickly generate models for network connections, we must pick a technique to set the hyperparameters for each model that doesn't require human intervention. Thus, the hyperparameter K is defined by using a common heuristic technique of taking the square root of the total number of data points and rounding down to the nearest integer. The approach to classify anomalies using the KNN algorithm is as follows. Once the K hyperparameter is defined, the KNN algorithm is fitted to the dataset. From that, for every data point in the dataset we calculate the average distance from that point to its K closest neighbors. With this, we calculate the median and the standard deviation of the average distance data and data points are classified as anomalous if their average distance was a standard deviation above the median. We chose to use the median instead of the mean as it's less affected by statistical outliers. Each data point classified as an anomaly by the KNN algorithm is kept track of in an array.

The DBSCAN algorithm in the first pass uses Scikit-learn DBSCAN implementation and has a hyperparameter "eps". According to the Scikit-learn documentation, the eps value is "the maximum distance between two samples for one to be considered in the neighborhood of the other" and is the most important DBSCAN parameter to choose depending on the dataset *DBSCAN* (n.d.). To pick this value we utilized a technique outlined in the paper "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise" Ester, Kriegel, Sander and Xu (1996). This approach uses the previously described KNN algorithm to calculate the average distance of a point to its nearest neighbors, and then these K-distances are plotted in ascending order. The aim is to determine the "knee", which corresponds to the optimal eps parameter. The knee (or elbow) is defined as the visible point in a graph where a curve visibly bends, particularly from a low slope to a high slope or vice versa. Figure 4 below shows an example of KNN distances sorted in ascending order and the vertical line indicates where the "knee" is.

3.3.2 Second Pass

The second pass of the ADT framework consists of 4 separate lightweight ML algorithms. All four of these algorithms, Isolation Forest, Elliptical Envelope, Local Outlier Factor, and One Class Support Vector Machine, are unsupervised ML algorithms designed to detect anomalies and are using the implementation from the Scikit-learn python library *EllipticEnvelope* (n.d.); *IsolationForest* (n.d.); *LocalOutlierFactor* (n.d.); *OneClassSVM* (n.d.). Elliptical Envelope uses covariance estimation on Gaussian distribution data and tries to make an elliptical cluster that fits most of the data; data points lying outside of the elliptical cluster

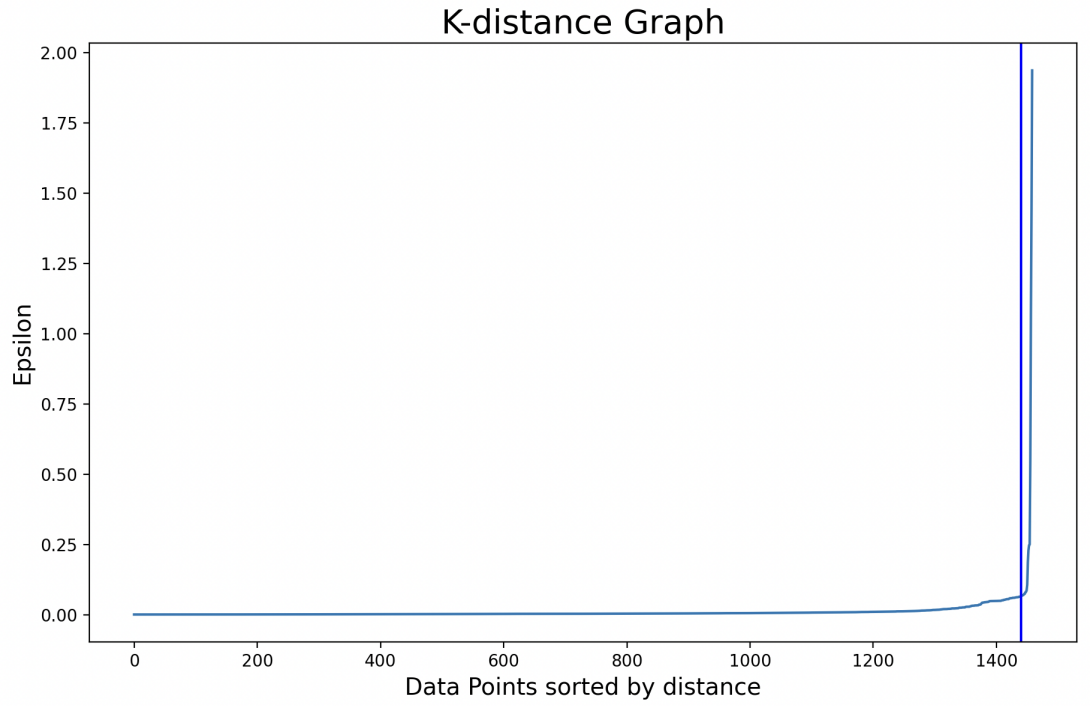

Figure 4. K-distance graph representation of how eps value is determined based on knee location.

are considered anomalous *EllipticEnvelope* (n.d.). Isolation Forest is similar to Random Forest and uses a form of decision tree to classify the anomalies in a dataset *IsolationForest* (n.d.). The Local Outlier Factor algorithm computes the local density deviation of a given data point with respect to its neighbors and samples with substantially lower density than their neighbors are classified as anomalies *LocalOutlierFactor* (n.d.). Finally, the One Class Support Vector Machine is another outlier and anomaly detection algorithm that unlike traditional Support Vector Machines, exclusively trains on data points from a single class *OneClassSVM* (n.d.). Each of these algorithms require the contamination hyperparameter to accurately fit to the data, and thus are organized into the second pass. Once the contamination hyperparameter has been passed to the second pass, the algorithms are fitted to the data and make predictions on what

data points they consider anomalous by labeling each point with a –1 for an anomaly, or 1 for a non-anomaly. The anomaly predictions of each model are stored in their own separate array similarly to the method described in the first pass. These anomaly index arrays are used to build a label matrix which is part of the output of the ADT framework

3.4 ADT Framework Results

After the data is run through both the first and second pass, we then sum each prediction from all six of the models to generate a weak labeled data set. To create this weak labeled data set we iterate through each of the six predications for every data point and count how many of them predicted the point to be an anomaly. This information is stored in a pandas DataFrame, where each row represents a data sample, and the columns names are as follows; timestamp, value, anomaly_value. The anomaly_value column represents the number of models that consider that point to be anomalous. We recognize that our current method consists of a naive majority vote technique to process the weak labels, presents a potential flaw. Not every model in the framework is created equally and depending on the features of a given data set, the models will have varying levels of accuracy. Ideally, the labels produced by a given model would be weighted based on the model's performance. For example, if KNN was found to perform extremely well on a particular data set, its labels would be weighted higher in comparison to the other less accurate models in the framework. But due to the nature of unsupervised ML, there are no "gold" labels which represent with 100 % accuracy what the label of a data point should be. This means the traditional way of calculating a model's accuracy by comparing its results to a test data set is not possible. One potential solution, proposed in the

future work section, involves determining a model's accuracy by assessing the level of agreement between its decisions and the results of other models. Subsequently, the model's labels would be weighted accordingly to enhance the overall accuracy of the framework. This potential area for future work could help address some of the limitations in the ADT's results, which are discussed in the following section.

3.5 ADT Results Advantages and Limitations

The Anomaly Detection Tool (ADT) Framework offers significant advantages in the realm of network anomaly detection by leveraging both statistical and spatial analysis of data points. By examining the spatial context of each data point relative to its neighbors, the framework can effectively identify anomalies that might be missed by traditional methods. Additionally, the ADT framework's use of unsupervised learning allows for the application of machine learning techniques without the need for labeled datasets, making it particularly useful in network applications, where obtaining labeled data is challenging and impractical. This capability enables network operators to deploy sophisticated anomaly detection models rapidly and efficiently, thereby improving the security and reliability of critical network infrastructure.

However, the ADT framework is not without its limitations. One notable drawback is its susceptibility to noisy data, which can significantly decrease the accuracy of the results. In environments where data quality is variable the frameworks results tend to include more false positives or missed anomalies.

For example, Figure 5 is a plot of RTT values over the course of 3 days. The dataset is relatively small in size and noisy in nature (contains no regular patterns).

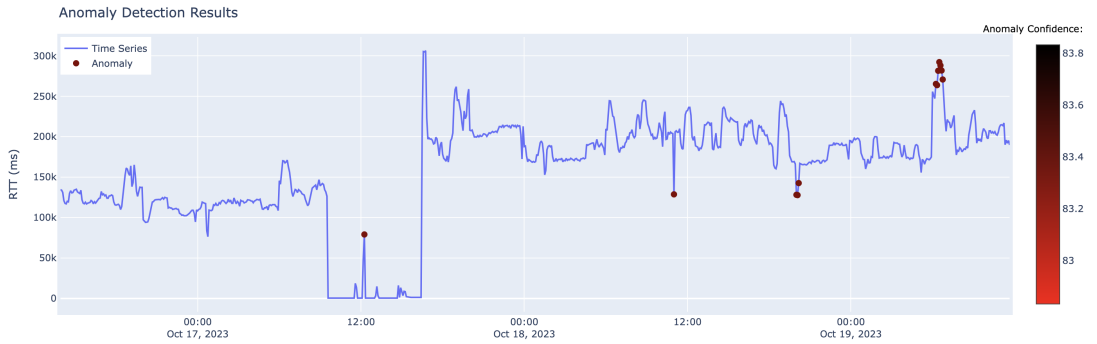

Figure 5. Anomaly Detection Results for a Small Noisy Round-Trip Time (RTT) Dataset.

The frameworks results contain zero points unanimously agreed to be an anomaly. Furthermore, areas in the data with large pattern shifts such as October 17th, 2023, at 12:00, would be inferred as a contextual anomaly to an observer but are not marked as such by the framework. Due to the low quality of the dataset, the framework is unable to provide confident anomaly detection results.

Moreover, the framework performs optimally with larger datasets that contain clearer anomalies. The increased context provided by larger datasets allows the models to make more accurate classifications, whereas smaller datasets with less pronounced anomalies may not provide sufficient information for the models to function effectively.

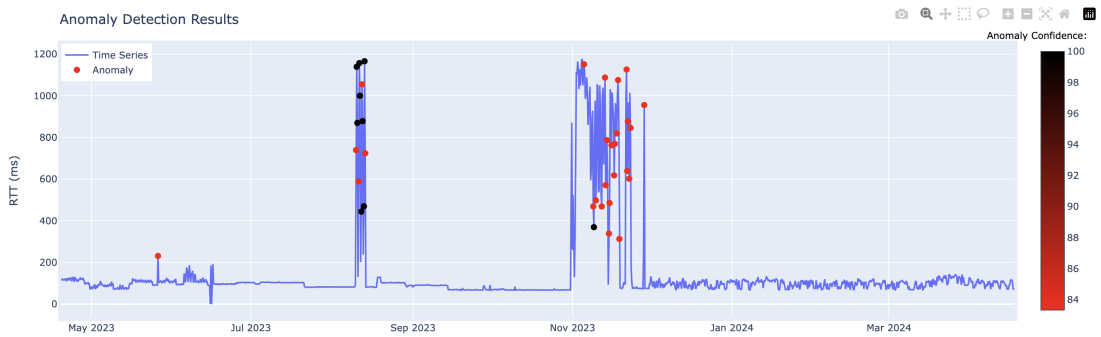

Figure 6. Anomaly Detection Results on a Large Time Scale RTT Dataset.

For example, Figure 6 above is the ADT results on a much larger dataset with less noise and more prominent anomalies. The frameworks results are more confident (higher agreement between models), and the results contain a low number of false positives or missed anomalies.

As demonstrated, the ADT framework's efficacy is highly dependent on the quality and size of the input data, which could limit its applicability in certain contexts. These limitations highlight the need for ongoing refinement and adaptation of the framework to ensure its reliability and accuracy across diverse operational environments. We've detailed the results produces by the ADT framework, but future work on developing unsupervised ML frameworks for network performance anomaly detection could benefit from quantifying the frameworks results in a manner where the accuracy and effectiveness can be better reported.

3.6 Anomaly Detection Tool Dashboard

As mentioned earlier, this research was conducted both with the intent of developing an improved decision support framework for detecting contextual and global anomalies in time series data, but also to lower the barrier of entry for network operators to apply data science and ML techniques to their network monitoring. To address this, we developed a web browser-based software that allows network operators to utilize the capabilities of the framework without needing prior knowledge of how to train and tune ML models. By designing an intuitive user interface, network operators can more effectively use the ADT framework as an anomaly detection decision support tool for their CI hazard monitoring network.

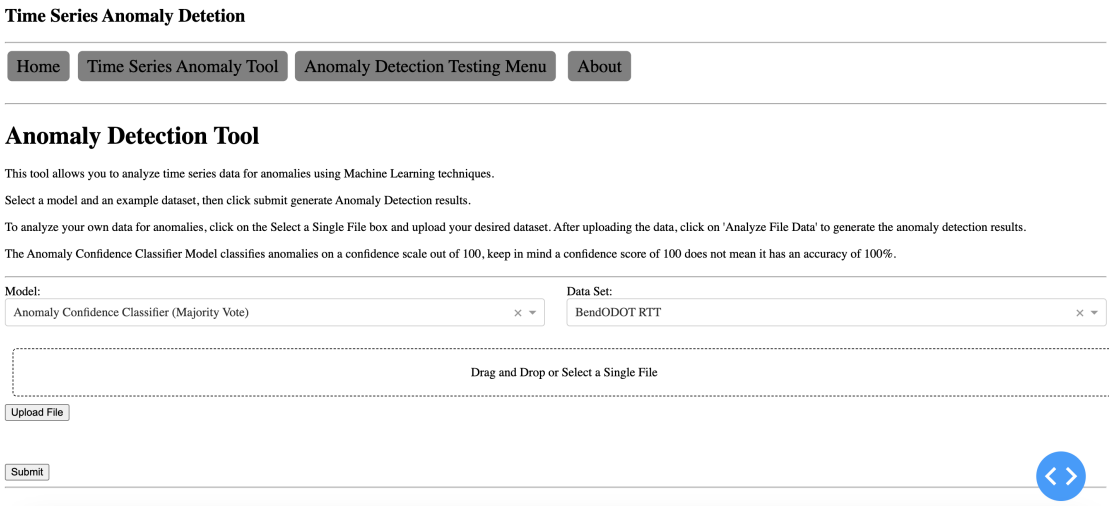

Figure 7. Anomaly Detection Tool Dashboard Menu.

The software utilizes Plotly's Dash library, which is specially designed to facilitate the building of web dashboards utilizing the Plotly graphing abilities to display data. The web dashboard is comprised of a menu system to select what type of model to run and on what data set. The user also has the option of uploading their own data set, so long as it complies with the XML or CSV format specified. Once the user has selected a model and data set, the ADT Dashboard will make an API call to the framework and receive the anomaly detection results. The user can then view and analyze the results through the interactive plot generated with the help of the Plotly Library.

The plot displays where potential anomalies are located by marking the specific points with an orange, red, or black circle. The color of the anomalous point is dependent on the "confidence" score which is determined from the results the framework produces. This confidence score is on a scale out of 100 and represents what fraction of the labels for that point agreed that it was an anomaly. For example, if six out of the six labels agreed it was an anomaly, the confidence score

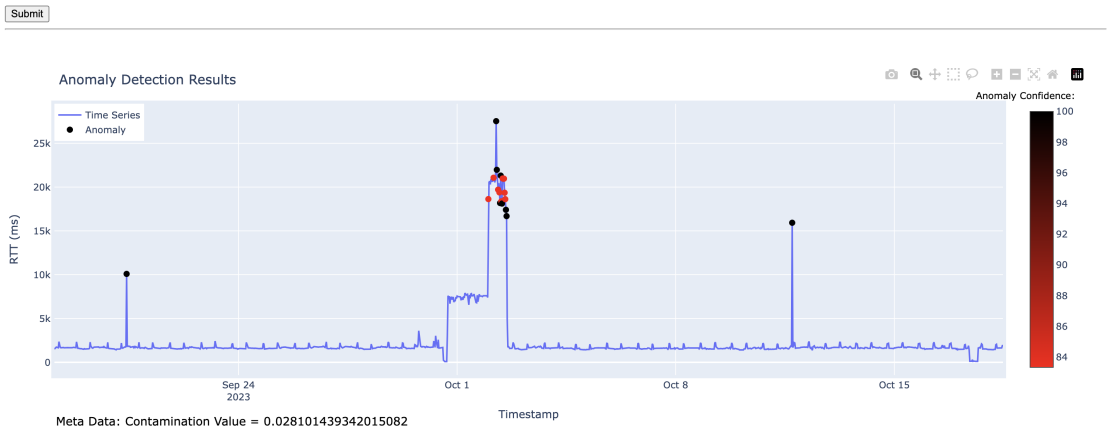

Figure 8. Anomaly Detection Tool Dashboard Results Plot.

would be 100. The confidence score is not a representation of the accuracy of the model, only its agreement levels. Since the framework is designed to help detect potential anomalies, and is not meant to classify if or what type of failure is occurring in the network, we chose to represent the results of the framework with a "confidence score" as that will best provide decision support for the network operator to identify anomalies in their CI network.

These design choices were validated by conducting a presentation for the main network operators at Oregon Hazards Lab. The presentation audience consisted of two of the main network operators at OHAZ, along with the OHAZ director and of course the author and research advisor of this Thesis paper. We briefed them on the varying types of anomalies, the challenges of traditional ML applications to networks, and our goals in developing this framework and accompanying software. We conducted a live demonstration detailing the functionality of the software, which provided us with useful feedback on what we did right and areas we can improve on. Most notably, the network operators expressed that while the software was useful for visualizing results, integration into existing systems was more ideal. Fortunately, the framework was designed with that in mind, as the web software

simply visualizes the results produced by an API call to the ADT framework. This means that those API calls could be used to integrate the framework into any system provided that the framework has access to the data, and the user specifies the correct data path. The network operators also mentioned that they are interested in extending this anomaly detection to include support for multivariate datasets, which would allow them to better validate anomalies and classify the types of failures produced by a set of anomalies, this is discussed more in future work.

CHAPTER IV

SUMMARY AND FUTURE WORK

4.1 Summary

The Anomaly Detection Tool (ADT) Framework, combined with its web-based software, offers a robust solution for improving the reliability of CI networks such as the Oregon Hazards Lab's hazard monitoring network. By leveraging unsupervised ML techniques, the ADT Framework aims to be a decision support tool for detecting both global and contextual anomalies in time series data without requiring extensive labeled datasets. This approach addresses key limitations of traditional supervised learning methods in network applications, particularly the impracticality of generating labeled data for each unique connection in a large network.

The framework's two-pass technique, involving an initial pass with K-Nearest Neighbors (KNN) and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) to estimate a contamination value, followed by a second pass using four additional lightweight algorithms (Isolation Forest, Elliptical Envelope, Local Outlier Factor, and One Class Support Vector Machine), ensures a more accurate classification of anomalies. The use of a simple majority vote technique to combine the outputs of these algorithms results in a weakly labeled dataset that can be easily interpreted by network operators through the web dashboard.

The accompanying web-based software enhances the usability of the ADT Framework by providing an intuitive interface for network operators to apply data science techniques and visualize anomaly detection results without needing prior expertise in ML. This integration not only lowers the barrier to entry for

utilizing advanced data analysis tools but also serves as a decision support tool for operators by providing actionable insights to maintain confidence in the health and performance of their critical infrastructure networks.

4.2 Future Work

While this research details a promising approach to the problem of applying unsupervised ML techniques to networks for anomaly detection, there is still much to do which presents opportunities for future work. The main areas we identified that could potentially be explored in future work are, weighting weak labels based on agreement, scaling this approach to work with multivariate data, and improving the integration of the framework into real time systems for better evaluation and testing.

As mentioned previously, the framework currently uses a naive majority vote technique to process six weak labels for each data sample, which is flawed because not all models in the framework have equal accuracy. Thus, developing a technique to weigh the labels based on each model's performance could improve accuracy. Since unsupervised machine learning lacks "gold" labels for perfect accuracy, traditional accuracy measurement isn't possible. However, a possible workaround is to weigh the labels based on the agreement among models. For example, if the labels produced by KNN were generally found to be agreed upon by one or more of the other models, then we could weigh its labels higher than a model where its labels are hardly agreed upon. Determining the agreement levels between labels to indirectly quantify a given model's performance could potentially provide more accurate anomaly detection and improved confidence in the framework's results.

When we presented this work to the network operators at OHAZ's, one of the pieces of feedback they gave us was the desire for the framework to handle multivariate dataset. Currently the framework can only process single-variate dataset, meaning it can only do one performance metric and a time in isolation. A multivariate dataset would consist of a combination of performance metrics (Round Trip Time, Transfer / Receive Rates, etc.) all on one time scale. This is a notable area for future work, as the benefit of multivariate datasets are that you could potentially validate the anomaly detection results of one variable against the corresponding variables in the dataset. Given a multivariate dataset, common failures could be classified depending on the values of performance metrics at the time of failure. This would greatly increase the efficiency of network monitoring in critical infrastructure as a model could be generated off these classifications of common failures and help assist network operators in identifying failures and preventing them from causing issues in the future.

Finally, integrating a decision support framework such as ADT into a real-time hazard monitoring network could yield several significant benefits. Firstly, it would allow for the generation of quantitative results on the number of anomalies the framework successfully detects, providing concrete evidence of its effectiveness. Secondly, deploying the framework in a real-time environment would offer valuable insights into how it performs at scale, identifying potential challenges and areas for optimization. Additionally, real-time integration would facilitate continuous improvement of the framework through ongoing feedback and data collection, enabling adaptive learning and refinement of the anomaly detection algorithms. Furthermore, by operating within a live hazard monitoring network, the framework could contribute to immediate operational improvements, enhancing the resilience

and reliability of critical infrastructure systems. Overall, such integration would be a crucial step towards validating and enhancing the practical applicability of the ADT framework, paving the way for broader adoption and implementation in various network monitoring contexts.

4.3 Lessons Learned

Developing the Anomaly Detection Framework detailed in this paper was my first endeavor in a research project of this scale. Throughout the course of my work on this problem, I encountered many unforeseen challenges. To find solutions, I had to learn many new skills, think creatively, and push myself to apply my knowledge to the problem at hand. My most important takeaways from this research are as follows:

1. Initially, my understanding of the intricacies of machine learning (ML) was limited. I spent significant time exploring various ML techniques, learning their nuances, and understanding their applicability to different situations. Ultimately, I gained insight into their strengths and limitations, enabling me to choose the most suitable technique for the problem.

2. I gained valuable insights into the crucial role of data in ML model development. I honed my skills in data analysis and preprocessing, ensuring the data was optimally prepared for training ML models.

3. My research provided a deeper appreciation for the critical infrastructure networks and their essential role in performance monitoring and anomaly detection. Understanding the resilience of networks and their ability to reliably transmit data, especially for natural hazard detection, became

evident. Developing solutions to enhance network monitoring is crucial for ensuring operational reliability.

References

DBSCAN. (n.d.). `https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html`. Scikit-learn.

EllipticEnvelope. (n.d.). `https://scikit-learn.org/stable/modules/generated/sklearn.covariance.EllipticEnvelope.html`. Scikit-learn.

Ester, M., Kriegel, H.-P., Sander, J. & Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96).*

Foorthuis, R. (2021). On the nature and types of anomalies: a review of deviations in data. *International Journal of Data Science and Analytics*, *12*(4), 297-331.

Given, D. D., Allen, R. M., Baltay, A. S., Bodin, P., Cochran, E. S., Creager, K., ... Heaton, T. H. (2018). *Revised technical implementation plan for the ShakeAlert system—an earthquake early warning system for the west coast of the United States* (Tech. Rep.). US Geological Survey.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning.* MIT Press. (`http://www.deeplearningbook.org`)

Hamers, L. (2023, April 17). *OHAZ builds disaster-resilient infrastructure network.* `https://environment.uoregon.edu/ohaz`. OHAZ | UO Environment.

IsolationForest. (n.d.). `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html`. Scikit-learn.

Keogh, E. & Lin, J. (2005). Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and Information*

Systems, *8*, 154-177.

KNeighborsClassifier. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html. Scikit-learn.

Lavinia, Y., Durairajan, R., Rejaie, R. & Willinger, W. (2020, August). Challenges in using ML for networking research: How to label if you must. In *Proceedings of the Workshop on Network Meets AI ML* (p. 21-27).

LocalOutlierFactor. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.LocalOutlierFactor.html. Scikit-learn.

Muthukumar, A. & Durairajan, R. (2019, December). Denoising internet delay measurements using weak supervision. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)* (p. 479-484). IEEE.

Nagios Open Source. (2024, March 13). https://www.nagios.org/.

OneClassSVM. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html. Scikit-learn.

Smith, K., Kent, G., Slater, D., Williams, M., McCarthy, M., Vernon, F., ... Braun, H.-W. (2016). Integrated multi-hazard regional networks: Earthquake warning/response, wildfire detection/response, and extreme weather tracking. In *Applied Geology in California.*

Smith, L. M., Barth, J. A., Kelley, D. S., Plueddemann, A., Rodero, I., Ulses, G. A., ... Weller, R. (2018). The ocean observatories initiative. *Oceanography*, *31*(1), 16-35.

A Software-Defined Sensor Network. (n.d.). https://www.anl.gov/mcs/sage-a-softwaredefined-sensor-network. SAGE.

tslearn.clustering.TimeSeriesKMeans — tslearn 0.6.3 documentation. (n.d.). https://tslearn.readthedocs.io/en/stable/gen_modules/clustering/tslearn.clustering.TimeSeriesKMeans.html. tslearn.