A Dual-Model Anomaly Detection Algorithm for non-linear stream data in Smart City Environments

Anthony J. Bustamante*, Sarah Asad*, Daniela Nicklas†, Brent Lagesse*
*University of Washington Bothell
Email: absuarez@uw.edu, sasad23@uw.edu, lagesse@uw.edu
†Otto-Friedrich-University Bamberg
Email: daniela.nicklsa@uni-bamberg.de

***Abstract*—In this paper, we introduce a complementary and straightforward mechanism for anomaly detection tailored for smart city infrastructures, utilizing a combination of regression algorithms. Our methodology employs two distinct regression models to generate future predictions from a given dataset. The primary model is crafted to yield high-fidelity predictions, while the secondary model is purposefully designed to introduce a degree of noise. Both models work together as a defense against Flooding attacks through the detection of abnormal levels of data inflow (detection of outliers). We calculate the alignment cost, or Euclidean distance, between the predictions from these two models, establishing a threshold against which real future traffic can be evaluated. The alignment cost or euclidean distance of the actual traffic is computed in relation to the high-quality predictions and then compared with the established threshold to pinpoint anomalies. Through experimentation with various regression algorithms, including linear regression, support vector regression, decision trees, etc., we identified an optimal combination for peak performance. Our assessments, grounded in comprehensive smart city datasets, center on the process of transforming complex non-linear data into a more appropriate form to detect anomalous data points. Conclusively, the dual-model anomaly detection framework we propose stands out as an invaluable tool in defending smart city infrastructures from data irregularities and potential threats, highlighting the criticality of bespoke solutions in contemporary urban digital environments.**

***Index Terms*—Smart City, Security, Flooding attacks, Denial of Service Attacks(DoS), Anomaly Detection, Pattern Recognition, Outlier Detection.**

I. Introduction

The emergence of smart cities—urban environments enriched with interconnected devices, sensors, and advanced information systems promises transformative changes in urban life and governance [1]. Such ecosystems, however, bring forth a plethora of data that necessitates rigorous monitoring to ensure data integrity and security.

Flooding attacks(for instance DoS, DDoS, Replay attacks, MAC spoofing) in smart city environments exemplify critical challenges that can severely compromise the entire urban digital infrastructure [2]. Anomalies, if undetected, can lead to serious repercussions, affecting the decision-making processes, operational efficiency, and even public safety [3]. While a variety of techniques have been proposed for anomaly detection, smart city-specific datasets with their unique characteristics and high data volume pose significant challenges to these traditional methods.

To address this pressing matter, we propose a complementary technique to detect outliers. Past approaches for anomaly detection, particularly for time-series data, have often been limited in their adaptability and sensitivity, especially when applied to intricate and voluminous smart city datasets [4]. Our dual-model system, by contrast, has been designed with adaptability and simplicity at its core, ensuring robust detection capabilities with very high sensitivity and being able to handle complex non-linear traffic.

II. Related Work

The realm of anomaly detection in time series data, particularly in smart city environments, has seen various approaches. Prior works in this domain span statistical analysis, machine learning, deep learning, and time series analysis, each with its strengths and limitations in handling the complexity of smart city data [5]–[8].

Goldberg's work [9] emphasizes the use of volume-based anomaly detection in network traffic, which aligns with our approach in addressing large-scale data but lacks the simplicity and dual-model robustness of our method. The TadGAN paper [10] introduces generative adversarial networks for anomaly detection, a sophisticated method, but possibly overcomplex for certain applications, unlike our more straightforward regression-based approach. The paper on Dynamic Time Warping (DTW) [11] shares similarities with our use of DTW for alignment cost measurement. However, our method enhances this by integrating a dual regression model, offering increased sensitivity and adaptability for smart city datasets.

Our methodology distinguishes itself by its simplicity and robustness, leveraging basic regression algorithms capable of predicting future data trends effectively. This approach not only simplifies the anomaly detection process but also ensures high sensitivity in identifying even minor anomalies. Unlike existing methods that may require complex feature engineering or struggle with noise and evolving data patterns, our dual-model system is adept at handling such challenges, making it particularly suitable for the dynamic and voluminous datasets typical of smart city infrastructures.

By comparing the alignment cost or Euclidean distance of real traffic against high-quality predictions, our method offers a flexible and adaptable framework for anomaly detection that can effectively handle diverse datasets and evolving anomalies. Furthermore, our approach mitigates the sensitivity to noisy data and outliers typically associated with regression-based methods, making it a valuable contribution to the field of anomaly detection.

III. Problem Statement and Solution

Detecting anomalies in time series data and non-linear functions, especially those commonly observed in smart cities, poses significant challenges. The dynamic nature of this data, intertwined with noise and prevalent non-linear patterns, exacerbates these challenges [12]. Traditional statistical and machine learning techniques for anomaly detection often struggle to adapt to evolving patterns, handle noisy data efficiently, or require complex feature engineering [13]. While regression-based methods have potential in predictive modeling for such scenarios, they are notoriously vulnerable to noise and outliers [14] and measurement techniques like R2, MAE, MSE, etc. are not adequate to detect anomalies as stated in [15].

That is why in our proposed solution, we combine two regression algorithms with time series distance measures, which will more effectively detect anomalies in smart city time series data than traditional statistical and machine learning techniques. This effectiveness is hypothesized to be due to the method's increased resilience to noise and its capacity to adapt to evolving data patterns. We generate two distinct predictions on future data behaviors using two separate regression algorithms:

- P_{best}: It is the prediction that is most similar to the real data, we derived it from our top-tier regression algorithm, and this prediction is optimized for accuracy.
- $P_{\text{reference}}$: It is another prediction or dataset that is similar to the actual data but is not necessarily optimized for high accuracy, we sourced it from our secondary regression algorithm, this prediction intentionally has slightly noisier results. If necessary, we can introduce additional noise to expand our anomaly detection threshold.

The central tenet of our approach is that the alignment cost or Euclidean distance (either through Dynamic Time Warping (DTW) or standard Euclidean measures) between the two predictions (P_{best} and $P_{\text{reference}}$) provides a spectrum within which we anticipate the future real traffic to align.

The methodology's core premise is to use the alignment cost or Euclidean distance between the real traffic data (T_{real}) and the optimal prediction (P_{best}) as a metric for anomaly detection. This approach is grounded in the assumption that normal traffic behavior will fall within a predefined threshold, determined by the disparity between the two predictive models. When the real traffic deviates significantly from this threshold, it is identified as an anomaly. This technique theoretically improves anomaly detection by balancing the precision of regression models with the robustness of a threshold-based approach, thereby enhancing adaptability and reducing vulnerability to noise and outliers. In other words, the crux of our methodology is the assumption that this computed alignment cost or distance should fall within the threshold delineated by the two predictions. If the real traffic deviates beyond this boundary, it signifies an anomaly.

Our method offers the following advantages:

1) A merger of the predictive capabilities of regression models with the adaptability of a threshold-centric approach.
2) A significant reduction in the susceptibility to noise and outliers, a common issue with regression-based methods. This is achieved by combining them with time series distance measure techniques.
3) The inherent flexibility of the method makes it compatible with any regression algorithm(using batch learning or online learning), thus ensuring its applicability across diverse datasets and use cases.

IV. Methodology

A. Regression Algorithms

This study is narrowed to two regression algorithms that produced the best results for our use case: **Decision Tree Regressor (DTR)** and **Random Forest Regressor (RFR)**. We also tried multiple regression algorithms with different techniques, including batch learning and online learning. In our case, DTR and RFR were the models that performed the best (using batch learning). Our approach is not specifically limited to these regression algorithms and other regressors can be plugged in as needed for different applications or as new regressors are devised. In table III, we dive deeper into the reasons why we chose DTR and RFR. As stated in our conclusions we chose these two models because they offered the best R2, MAE, MSE, and RMSE results for our case(as a way to standardize our technique), however, the methodology would work also with any other algorithm that provides reasonable good predictions.

B. Threshold Calculation

1) Dynamic Time Warping (DTW): DTW is a technique used for measuring similarity between two temporal sequences that may vary in time, intensity or speed. It calculates an alignment cost that represents the best alignment between two sequences regardless of their non-linear variations in time [16].

2) Euclidean Distance: Euclidean distance is a metric used to measure the similarity between two vectors [17].

3) Threshold Calculation: In this study, we calculate thresholds using both the alignment cost (DTW) and the Euclidean distance. The goal is to establish a threshold that represents the expected distance between two future predictions: one that is the best possible prediction and another that serves as a reference point. The reference prediction is preferably a bit irregular, allowing for more flexibility in the threshold.

Given two future predictions A and B, the alignment cost threshold is calculated as:

$$T_{ac} = \text{DTW}(A, B) \tag{1}$$

and the Euclidean distance threshold is calculated as:

$$T_{ed} = d_{euc}(A, B) \tag{2}$$

where $\text{DTW}(A, B)$ is the alignment cost between A and B, and $d_{euc}(A, B)$ is the Euclidean distance between A and B.

C. Anomaly Detection

Taking into account the equations in 1 and 2, we then analyze the real traffic (T_{real}) by calculating the alignment cost or Euclidean distance between the real traffic and the best prediction (P_{best}). If the calculated distance is within the previously computed threshold, the real traffic is considered normal; otherwise, it is considered anomalous. Specifically, the real traffic C is considered normal if:

$$\text{DTW}(A, C) \leq T_{ac} \quad \text{or} \quad d_{euc}(A, C) \leq T_{ed} \tag{3}$$

where T_{ac} is the alignment cost threshold, and T_{ed} is the Euclidean distance threshold.

The threshold calculation approach was implemented as part of our experiment. Data can be sourced from various formats or directly collected in real-time. In our experiment, we analyze data both in real-time (running simulation by replaying real-world data) and from CSV files.

V. Experiment Setup

In this section, we outline the experimental configuration employed to assess the efficiency of our proposed approach in identifying anomalies within the data stream from the smart city. (Figure 1). In the next sub-sections, we illustrate the deployed topology, data collection process, and the specifics of our anomaly detection implementation.

A. Deployment Topology

Our experimental setup, we were provided access to a smart city environment, which consisted of six Wi-Fi probing sensors that are deployed across the busiest areas of Bamberg, Germany (see Figure 1) [14]. The purpose of these sensors is to capture anonymized probe requests from Wi-Fi-enabled devices, generating real-time data that represents the overall network traffic in the city.

As shown in Figure 1, the Wi-Fi sensors communicate with a processing backend server using TCP. The backend server receives JSON-formatted requests from the sensors and implements our proposed anomaly detection algorithm. The processed data, either tagged as normal or anomalous, is then stored in a database. APIs can subsequently consume the stored data for additional processing.

B. Data Collection

The Wi-Fi sensors collect probe requests, which are small data packets sent by Wi-Fi-enabled devices to discover nearby Wi-Fi networks as seen in Figure 1. These probe requests contain information such as the device's MAC address, timestamps, and additional details about the street and city, among other things. In our experimental setup, the sensors capture these probe requests and anonymize the data to protect individuals' privacy. The anonymized data is then encapsulated in JSON format before being sent to the backend server via TCP/HTTPS. The JSON data included the following fields:

- **eventype**: Event defined in our sensors based on their position and status.
- **epocutc**: Timestamp about when the event took place.
- **zone**: Sensor's location in the city.
- **mac_address**: Hash of the MAC addresses of devices, hashed using SHA-224.
- **RSSI**: RSSI value of each device in the vicinity.
- **techtype**: Technology type used for data collection. For this study, only WiFi receptors were employed.

At the backend server, the incoming JSON requests are parsed, and the relevant information is extracted for further processing. The data is then pre-processed and prepared for anomaly detection.

C. Anomaly Detection and Defense Mechanisms

Our anomaly detection algorithm is implemented on the backend server. It analyzes the incoming traffic, represented by probe request data from the Wi-Fi sensors, to identify any deviations from expected patterns. The primary focus of our experiment is to detect flooding attacks(anomaly in the flow of traffic) that could disrupt the city's network infrastructure.

If the algorithm detects an anomaly, the server can trigger predefined protection mechanisms, such as another defense mechanism, alerting network administrators, blocking suspicious traffic, or adjusting network parameters to mitigate the impact of the attack. Conversely, if no anomalies are detected, the traffic is considered legitimate and allowed to proceed to the database, where APIs can access and consume the data for various applications.

It is worth nothing that the experimental setup described in this chapter was tested for flooding attacks in forms of DoS, Replay attacks and MAC spoofing. Since, the nature of the paper is identifying anomalies in the traffic sent to our backend server, we refer to these different attacks as flooding attacks in the rest of the paper.

VI. Evaluation and Results

A. Pre-processing and Feature Engineering

Time series and non-linear datasets inherently contain complexities that demand careful pre-processing. In our study, dates (time-variable) serve as independent variables, with a counter value or count as the dependent variable (as shown in

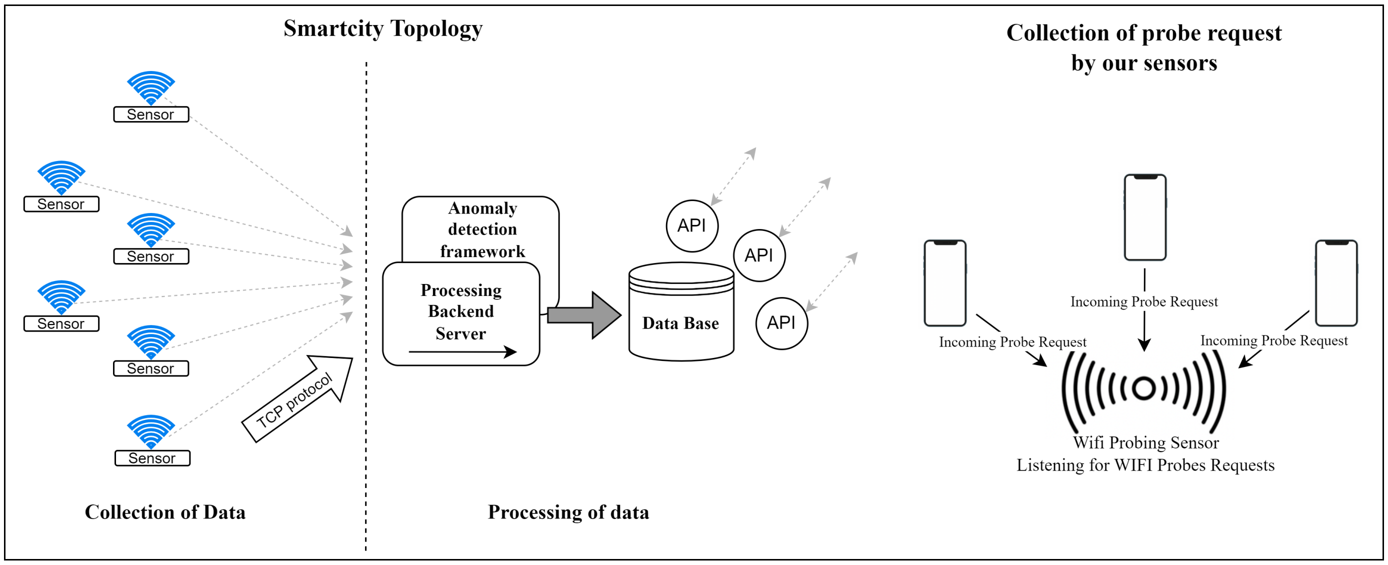

Fig. 1. Smart City Architecture Used For Our Experiments and Demonstration of WiFi Probe Reception by the Sensor

Table I). The fundamental steps of our pre-processing method are: Date Parsing, Resampling, Handling Missing Values, Stationarity, and Feature Engineering. Our approach bins the timestamps on data into 5-minute windows. Additionally, it is worth noting that we initially used a 5-minute window as it served as a foundational aspect of our project, primarily due to the need for consistency across various project components. Nonetheless, this can be modified according to the user's needs. We also tested our approach with different window gaps, such as 1 minute and 2 minutes, and obtained similar results.

Besides the fields in Table I, interaction terms were also introduced in the data to capture inter-feature relationships and improve model performance (Feature engineering part indicated in section IV). These interactions were obtained by multiplying two features together to create a new feature, although some other types of interactions can also be established to tailor the model to the problem's needs. A sample of the data after feature engineering is shown in Table II.

TABLE I
SAMPLE DATA FROM THE DATASETS

Year	Month	Day	Hour	Minute	Second	Counter
2023	1	30	12	33	2	2
2023	1	30	12	33	4	3
…	…	…	…	…	…	…
2023	7	10	0	0	8	1
2023	7	10	0	0	18	1

B. Model Training

Our model was trained on a 3 million entry dataset. We used an 80/20% split of training to testing data and selected the best performing models. For our research, we utilized `scikit-learn` library but also tried online-learning libraries like River. Furthermore, the hyperparameters used for training our models were set to their default values in `scikit-learn` (except `number of estimators = 1` and `random_state = 0` for RFR). We experimented with various configurations, but in our specific scenario, they did not yield a substantial impact.

C. Threshold Generation

Algorithm 1 describes the threshold generation process.

Algorithm 1 Anomaly Detection Using DTW and Euclidean Distance

```
Require: Future predictions: $A$, $B$, Real traffic data: $C$
Ensure: Anomaly detection in real traffic data
 1: function CALCULATEDTW($X$, $Y$)
 2:     return Dynamic Time Warping distance between $X$ and $Y$
 3: end function
 4: function CALCULATEEUCLIDEAN($X$, $Y$)
 5:     return Euclidean distance between $X$ and $Y$
 6: end function
 7: procedure GENERATETHRESHOLDS($A$, $B$)
 8:     $T_{ac} \leftarrow$ CALCULATEDTW($A$, $B$)
 9:     $T_{ed} \leftarrow$ CALCULATEEUCLIDEAN($A$, $B$)
10:     return $T_{ac}$, $T_{ed}$
11: end procedure
12: procedure CHECKANOMALY($C$, $T_{ac}$, $T_{ed}$)
13:     $D_{ac} \leftarrow$ CALCULATEDTW($A$, $C$)
14:     $D_{ed} \leftarrow$ CALCULATEEUCLIDEAN($A$, $C$)
15:     if $D_{ac} \leq T_{ac}$ or $D_{ed} \leq T_{ed}$ then
16:         return False                          ▷ Traffic is normal
17:     else
18:         return True                           ▷ Traffic is anomalous
19:     end if
20: end procedure
21: $(T_{ac}, T_{ed}) \leftarrow$ GENERATETHRESHOLDS($A$, $B$)
22: $isAnomalous \leftarrow$ CHECKANOMALY($C$, $T_{ac}$, $T_{ed}$)
23: if $isAnomalous$ then
24:     Handle anomaly in traffic data
25: else
26:     Continue normal operations
27: end if
```

The thresholds generated from the predictive models were used to evaluate real traffic data. They provided an effective

TABLE II
SAMPLE DATA AFTER FEATURE ENGINEERING

Table I values(except counter and year)	**Interactions**						**Counter**
	month_day	**day_hour**	**month_hour**	**hour_min**	**hour_sec**	**min_sec**	
For our final dataset	month*day	day*hour	month*hour	hour*min	hour*sec	min*sec	2
we copied the same values from	1	84	12	550	120	1	3
Table I, except	...	...	...	...	...	...	...
the values in counter and year	1	30	72	451	1288	100	5

means of identifying abnormal traffic patterns and alerting for potential flooding attacks.

D. Problem to solve

Figure 2 illustrates the category of Non-linear Time Series challenges relevant to Smart Cities. The dataset spans 7 months. Although, for the graph presented in this paper (Figure 2), we are depicting data from just a single week.

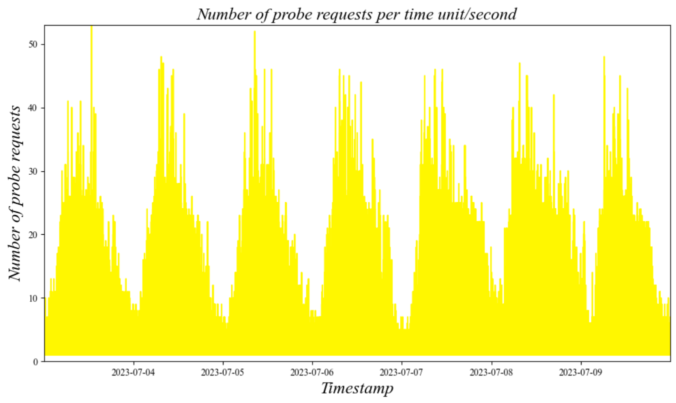

Fig. 2. One week example of the dataset.

E. Predictions vs real traffic

In Figure 3 we can see the plots for the real traffic that we are evaluating with our algorithm, the predictions with our best model (in red), which is almost identical to the real traffic (in blue), and lastly the noisy prediction with our second model that we used as a reference point to create our threshold(in purple).

In our dual-model approach, the secondary prediction was derived from what we discerned as the second-best regression model. To create a meaningful differential for establishing a threshold, we deliberately introduced a controlled amount of random noise. It is essential to note that during our experimental phase, we evaluated both the Euclidean distance and the alignment cost across a myriad of models, ultimately selecting those that demonstrated superior performance in the smart-city domain. Given:

- P_1 as the prediction of the primary (best) model.
- P_2 as the prediction of the secondary (second-best) model before noise introduction.
- N as the random noise vector.
- ED as the Euclidean distance.
- AC as the alignment cost.

Note: The noise `N` introduced in this experiment was generated by multiplying the results of our second-best model with a random number ranging from 1 to 3.5, however, people can use any other type of technique to introduce noise to expand the threshold.

The prediction with noise for the secondary model can be represented as:

$$P_2' = P_2 + N \tag{4}$$

The Euclidean distance and alignment cost can then be computed between P_1 and P_2' to determine anomalies:

$$ED(P_1, P_2') \tag{5}$$

$$AC(P_1, P_2') \tag{6}$$

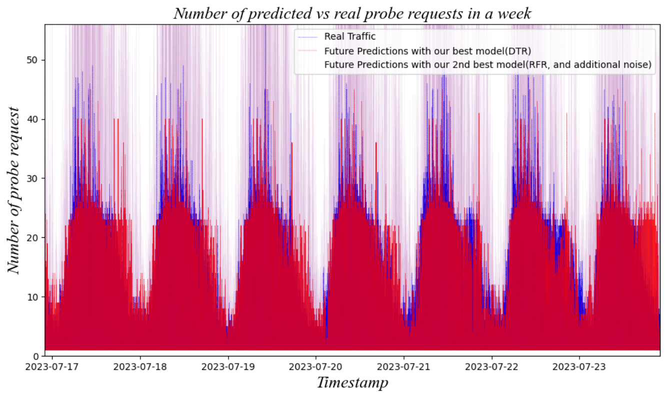

Fig. 3. Comparison for our best model vs reference point model vs real traffic

Moreover, we also measured the performance of our selected models using multiple evaluation metrics including the R^2 score [18], MAE [19], MSE [20], and RMSE [21]. These metrics offer insights into the accuracy and reliability of our regression models. The results from our evaluation are summarized in the table III:

TABLE III
SUMMARY OF REGRESSOR ERROR RESULTS

Evaluation methods	**DTR**	**RFR (without noise)**
R2	-0.809517785	-0.816383466
MAE	3.149929931	3.190064165
MSE	34.34288167	34.4731856
RMSE	5.860279999	5.871387025

Upon examining the values presented in the table, we observe that the $R2$ values for DTR marginally surpass those for RFR, and similar trends appear across MAE, MSE, and RMSE. The data stream emanating from smart city infrastructures is characteristically inundated with noise and exhibits non-linearity. Thus, achieving perfect evaluation scores can be elusive, especially in complex scenarios, this observation resonates with findings in the literature, where the intricacies of dealing with noisy and non-linear data in networking-monitoring systems lead to less-than-perfect $R2$ values [15] Our research emphasizes the significance of adaptive modeling in smart city environments, focusing not merely on achieving high scores but on building models that adeptly discern anomalies amidst the vastness of urban digital data streams.

F. Analysis of Euclidean Distance and Alignment Cost Outcomes

Figure 4 delineates the derived threshold juxtaposed against the genuine Euclidean distance and Alignment cost associated with real-world traffic. This congruence substantiates that our algorithmic approach is robust and aptly suitable for handling challenges inherent to time series data as well as nonlinear complexities (Characteristics of smart city traffic).

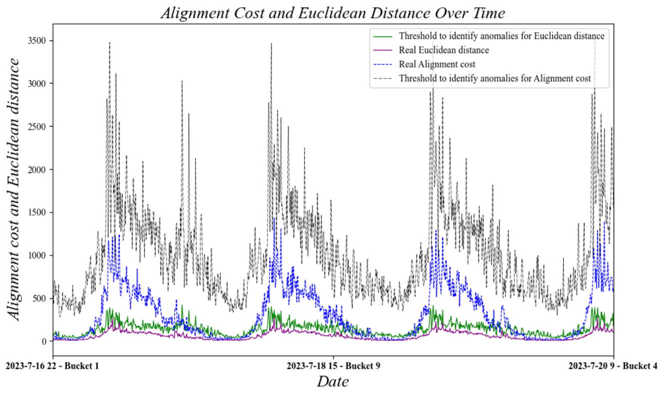

Fig. 4. Comparison between Projected Alignment Cost Threshold(black) and Actual Traffic Alignment Cost(blue) / Projected Euclidean Distance Threshold(green) and Actual Traffic Euclidean Distance(purple)

G. Benchmarking Proposed Methodology Against Statistical Techniques

Figure 5 displays the performance of various thresholds in relation to the true alignment cost. The distinct curves represent:

- The real alignment cost (dashed blue curve).
- The anomaly detection threshold derived from our proposed technique (solid black curve).
- An alternative threshold determined using the real statistics from the preceding day (green dash-dot curve)- ”We used the Dataset of the previous day and calculated the AC against our best prediction”.
- A threshold established by averaging statistical values from the past week (red curve)- ”We obtained the mean value of the previous week and then calculated the AC against our best prediction”.

Note that we did not compare the related work with our proposed method for not being precisely applicable to our project.

Our evaluation also extended to setting thresholds based on statistical analysis. Specifically, we assessed the alignment cost between our premier model and the values from the day prior, as well as the cumulative averages from the last seven days. As Figure 5 elucidates, these statistically derived thresholds did not resonate optimally with the real-world alignment costs. This stark contrast accentuates the superior efficacy of our proposed approach in accurately mirroring the genuine alignment costs.

$$\tau_{proposed} = f_{regression}(data) \tag{7}$$

$$\tau_{day} = f_{statistical}(data_{day-1}) \tag{8}$$

$$\tau_{week} = \frac{1}{7}\sum_{i=1}^{7} f_{statistical}(data_{day-i}) \tag{9}$$

Where:

- $\tau_{proposed}$ represents the threshold from our proposed methodology.
- τ_{day} denotes the threshold derived from the prior day’s data.
- τ_{week} symbolizes the threshold computed by calculating the mean of the last week’s data.

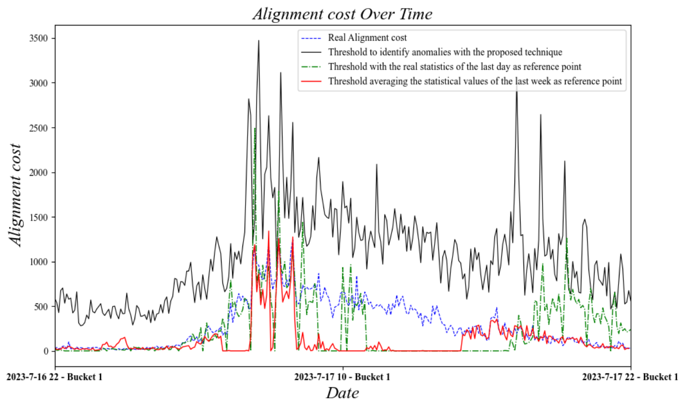

Fig. 5. Comparative Analysis: Proposed Model, Reference Point Model, and Real Traffic Alignment Cost

As we can see from Figure 5, our proposed method is more suitable than the other approaches, which is another way to demonstrate the value of our proposal.

H. Performance Analysis under Flooding Attack

Figure 6 showcases the system’s capability to detect anomalies even under subtle adversarial conditions, such as a simulated flooding attack (this encompasses DoS, DDoS, Replay attacks, and MAC spoofing). The alignment cost resulting from the fabricated ”flooding attack data” significantly exceeds the threshold defined by our methodology. This contrast reaffirms the sensitivity and robustness of our anomaly detection mechanism.

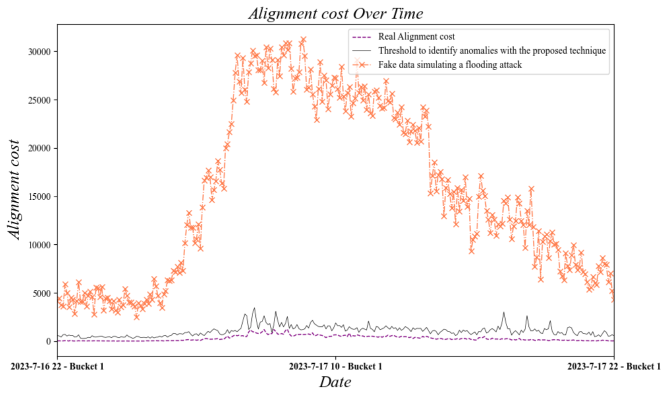

Fig. 6. Anomaly Detection Performance during a Simulated Flooding Attack

To simulate the attack, fake data was generated by dispatching random JSON queries at intervals ranging between 40 to 200 within a 5-minute span. This volume is markedly higher than the typical peak observed in legitimate traffic (approximately 50 queries). However, it is worth noting that in real-world attack scenarios, the volume of such queries could escalate into thousands. This underlines the superior sensitivity of our model: **if it can accurately identify anomalies with such low-level surges, detecting larger-scale attacks becomes exponentially more straightforward**.

Mathematically, the surge in traffic can be represented as:

$$\Delta Q_{attack} = Q_{attack} - Q_{normal} \tag{10}$$

Where: Q_{attack} denotes the number of queries during the flooding attack and Q_{normal} signifies the standard query count in genuine traffic. Given our observed values, we have: $\Delta Q_{attack} = 150$.

The modest value of ΔQ_{attack} and our model's ability to spot it elucidates its refined sensitivity. With the ever-evolving threat landscape, having an anomaly detection system with such sensitivity becomes paramount for maintaining robust security postures.

VII. Anomaly Detection Using the Anomaly Indicator

An essential aspect of anomaly detection, especially in complex environments like smart cities, is to have a quantifiable measure that can effectively differentiate between typical and anomalous data. To this end, we introduce an Anomaly Indicator (AnI) given by the formula:

$$\text{AnI} = \frac{\text{Threshold value}}{\text{Real value}} \tag{11}$$

The AnI serves as a decisive metric in our analysis. When AnI is greater than or equal to 1, it suggests that the observed traffic conforms to the expected patterns and does not exhibit any anomalies. Conversely, if AnI is less than 1, it indicates the presence of anomalous behavior in the traffic data.

This straightforward yet effective metric provides a clear boundary that distinguishes between benign and malign data. By using AnI, we can swiftly ascertain the nature of the traffic without delving into the intricacies of underlying patterns or dependencies.

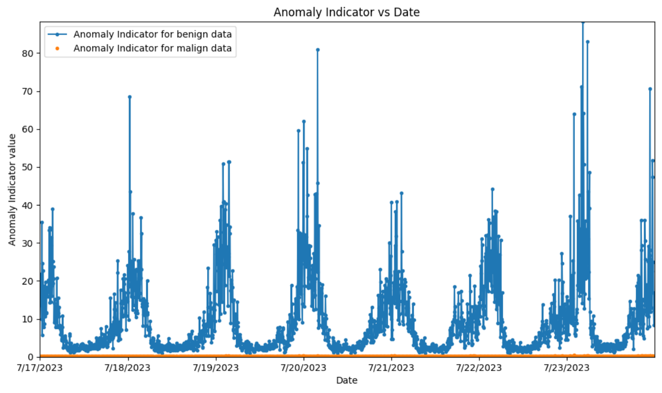

Fig. 7. Anomaly Indicator vs Date for Benign and Malign Data

From Figure 7, the calculated Anomaly Indicator (AnI) values further emphasize the accuracy of our model in differentiating between benign and malign data. Specifically, when using the formula 11, the benign data (depicted in blue) consistently results in AnI values closer to or greater than 1, indicating the absence of anomalies. In contrast, the malign data (depicted in coral) produces AnI values substantially below 1, signifying the presence of anomalies. This clear distinction not only highlights the efficacy of our anomaly detection mechanism but also underscores its vital role in safeguarding smart city infrastructures. The model's consistent performance across different data points further solidifies the robustness of our proposed approach.

Furthermore, during our experiments, we subjected our defense mechanism to a rigorous testing environment (simulation of our real environment). Over a span of 168 hours(one week), we dispatched approximately 80,640 probe requests to evaluate the system's robustness (We chose to use our Euclidean distance results). The results, as Table IV shows, were commendable.

TABLE IV
Performance Metrics of the Defense Mechanism

Metric	**Score**
Accuracy	100%
Precision	100%
Recall	100%
F1 Score	100%
True Positive Rate (TP)	100%
False Positive Rate (FP)	0%

While we are optimistic about the reproducibility and consistency of these results in real-world scenarios, we acknowledge the importance of comprehensive testing. As a part of our ongoing research, we are in the process of evaluating our defense mechanism against an even broader array of attack scenarios in our real infrastructure.

VIII. Conclusions

This study has unveiled a specialized solution to address the significant challenge of anomaly detection in smart cities, characterized by copious amounts of dynamic data, intricate fluctuations, noise, and evolving non-linearities. Our approach, leveraging the predictive capabilities of regression models coupled with the flexibility of a threshold-based method, has shown prime adaptability for complex non-linear volumes of data. It has been designed to inherently offer resilience against outliers, a prevalent shortcoming of conventional regression techniques, by factoring in noise when making predictions with the secondary model.

Moreover, the methodology adopted in this paper for reproducing results and comparing different models highlights a structured yet flexible approach. The choice of reference algorithms and the comparison metrics used demonstrate that while a standardized procedure was followed, there is room for variation that does not compromise the objectives. This research, though oriented towards smart cities, lays a foundation that could be beneficial in broader contexts of time series analysis and non-linear dynamics, suggesting a potential universality of the approach.

IX. Future Work

The forthcoming steps for this research include conducting additional experiments and refining our findings with the new data obtained. The present study serves as a proof of concept, demonstrating initial results. A key focus for practical implementation is adopting online or pseudo-online learning mechanisms. These mechanisms aim to update the model in real or near-real time as new information is received, moving towards using libraries like River for continuous learning. Our initial experiments were designed to re-train the model with each new batch of data from Wifi sensors. Transitioning from batch to dynamic learning is intended to improve the model's relevance in real-world settings, especially within the fast-changing landscapes of smart cities.

Beyond the current focus on smart cities, we plan to assess the methodology's effectiveness across a variety of datasets and environments. There is a pressing need to refine our approach, evaluate its performance on a wider scale, and broaden its utility beyond urban digital ecosystems. Future research will investigate the challenges within time series analysis and non-linear dynamics more thoroughly. Our aim is to harness the inherent versatility of our methodology, making it a valuable tool for addressing a broad spectrum of scientific and practical issues.

Acknowledgment

This work is partially funded by the National Science Foundation, Grant No. 1853953.

References

[1] H. Chourabi, T. Nam, S. Walker, J. R. Gil-Garcia, S. Mellouli, K. Nahon, T. A. Pardo, and H. J. Scholl, "Understanding smart cities: An integrative framework," in *2012 45th Hawaii International Conference on System Sciences*. IEEE, 2012, pp. 2289–2297.

[2] Y. Zhang, N. Meratnia, and P. J. Havinga, "Flooding attack detection in smart cities," in *2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS)*. IEEE, 2014, pp. 440–445.

[3] M. S. Hossain and G. Muhammad, "Anomaly detection in smart city using deep and machine learning models," *IEEE Access*, vol. 7, pp. 38 583–38 593, 2019.

[4] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[5] A. A. Cook, G. Mısırlı, and Z. Fan, "Anomaly detection for iot time-series data: A survey," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6481–6494, 2020.

[6] M. Hoh, A. Schöttl, H. Schaub, and F. Wenninger, "A generative model for anomaly detection in time series data," *Procedia Computer Science*, vol. 200, pp. 629–637, 2022.

[7] K. Choi, J. Yi, C. Park, and S. Yoon, "Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines," *IEEE Access*, vol. 9, pp. 120 043–120 065, 2021.

[8] M. Hu, Z. Ji, K. Yan, Y. Guo, X. Feng, J. Gong, X. Zhao, and L. Dong, "Detecting anomalies in time series data via a meta-feature based approach," *IEEE Access*, vol. 6, pp. 27 760–27 776, 2018.

[9] D. Goldberg and Y. Shan, "Volume-based anomaly detection in network traffic," Presented at HotCloud'15, 2015.

[10] A. Geiger, D. Liu, A. Cuesta-Infante, S. Alnegheimish, and K. Veeramachaneni, "Tadgan: Time series anomaly detection using generative adversarial networks," *Preprint*, 2020.

[11] D. M. Diab, B. AsSadhan, H. Binsalleeh, S. Lambotharan, K. G. Kyriakopoulos, and I. Ghafir, "Anomaly detection using dynamic time warping," in *2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*. IEEE, 2019, pp. 193–198.

[12] J. Li, H. Izakian, W. Pedrycz, and I. Jamal, "Clustering-based anomaly detection in multivariate time series data," *Applied Soft Computing*, vol. 100, p. 106919, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494620308577

[13] F. Stoffel, F. Fischer, and D. A. Keim, "Finding anomalies in time-series using visual correlation for interactive root cause analysis," in *Proceedings of the Tenth Workshop on Visualization for Cyber Security*, ser. VizSec '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 65–72. [Online]. Available: https://doi.org/10.1145/2517957.2517966

[14] T. Rütermann, A. Benabbas, and D. Nicklas, "Know thy quality: Assessment of device detection by wifi signals," in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2019, pp. 639–644.

[15] A. Bustamante, N. Ghimire, P. Sanghavi, A. Pokharel, and V. Irekponor, "Advantages of machine learning in networking-monitoring systems to size network appliances and identify incongruences in data networks," in *Trends in Artificial Intelligence and Computer Engineering. ICAETT 2021. Lecture Notes in Networks and Systems*, vol. 407. Cham: Springer, 2022.

[16] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[17] G. Strang, *Linear Algebra and Its Applications*, 4th ed. Cengage Learning, 2005.

[18] N. R. Draper and H. Smith, *Applied Regression Analysis*, 3rd ed. Wiley, 1998.

[19] R. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.

[20] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*. Springer, 2013.

[21] C. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate research*, vol. 30, no. 1, pp. 79–82, 2005.