# Anomaly based Incident Detection in Large Scale Smart Transportation Systems

Md. Jaminur Islam[**]
Western Michigan University
Kalamazoo, MI, USA
mohammadjaminur.islam@wmich.edu

Jose Paolo Talusan[*]
talusan.jose_paolo.tg3@is.naist.jp
Nara Institute of Sc. and Tech.
Nara, Japan

Shameek Bhattacharjee
Western Michigan University
Kalamazoo, MI, USA
shameek.bhattacharjee@wmich.edu

Francis Tiausas
Nara Institute of Sc. and Tech.
Nara, Japan
tiausas.francis_jerome.ta5@is.naist.jp

Sayyed Mohsen Vazirizade
Vanderbilt University
Nashville, TN, USA
s.m.vazirizade@vanderbilt.edu

Abhishek Dubey
Vanderbilt University
Nashville, TN, USA
abhishek.dubey@vanderbilt.edu

Keiichi Yasumoto
Nara Institute of Sc. and Tech.
Nara, Japan
yasumoto@is.naist.jp

Sajal K. Das
Missouri University of Science and Technology
Rolla, MO, USA
sdas@mst.edu

## ABSTRACT

Modern smart cities are focusing on smart transportation solutions to detect and mitigate the effects of various traffic incidents in the city. To materialize this, roadside units and ambient transportation sensors are being deployed to collect vehicular data that provides real-time traffic monitoring. In this paper, we first propose a real-time data-driven anomaly-based traffic incident detection framework for a city-scale smart transportation system. Specifically, we propose an incremental region growing approximation algorithm for optimal Spatio-temporal clustering of road segments and their data; such that road segments are strategically divided into highly correlated clusters. The highly correlated clusters enable identifying a Pythagorean Mean-based invariant as an anomaly detection metric that is highly stable under no incidents but shows a deviation in the presence of incidents. We learn the bounds of the invariants in a robust manner such that anomaly detection can generalize to unseen events, even when learning from real noisy data. We perform extensive experimental validation using mobility data collected from the City of Nashville, Tennessee, and prove that the method can detect incidents within each cluster in real-time.

## 1 INTRODUCTION

Rapid urbanization has proliferated the number of vehicles in cities leading to increasing congestion and a higher number of traffic accidents. For any traffic accident, delayed detection and response from first responders or emergency management agencies can worsen into heavy city-wide congestion and even in the loss of life. This delay is one of the most important challenges faced by communities across the globe [11].

To monitor the transportation infrastructure, three approaches have emerged to increase the visibility of real-time road conditions: (i) vehicular crowdsourcing, (ii) video-based anomaly detection; and (iii) sensor-based data collection.

Vehicular crowdsourcing involves cities leveraging commercial crowdsourcing platforms (such as Waze), to gather content reported by citizen users on these platforms to get real-time observations on traffic events. However, traffic incident detection is often unreliable and strong verification of the human reported data cannot be guaranteed in real-time.

Video anomaly detection[4] leverages cameras and sensors deployed by the city to detect traffic emergencies. This approach requires expensive edge devices, longer model training times, and continuous maintenance. Many environmental and connectivity constraints also negatively influence video quality and real-time availability. The computational resources needed to monitor and identify traffic incidents are high and not community scalable.

To avoid the above problems in these two paradigms, smart cities are deploying traffic sensors and Road Side Units (RSU) along roads and highways that collect traffic data from speed sensors or smart cars [8]. The RSU infrastructure is a typical IoT network that is decentralized, low-powered, and resource-constrained in nature.

However, given the ubiquity and number of devices, the RSU infrastructure can be utilized to work together in a distributed capacity, to design intelligent lightweight anomaly-based traffic incident detection in real-time that would otherwise be too computationally intensive, geographically impossible, or costly.

***Challenges:*** We view traffic incidents as anomalies that occur between otherwise normal traffic patterns. However, characterizing a normal traffic pattern that works at a large city scale is not straightforward due to (i) day-to-day variability of traffic, (ii) local neighborhood dependencies, (iii) a large number of speed sensors and road segments.

Hence, the nature of the problem falls under smart living CPS, which, unlike industrial CPS applications, are not just bound by tightly defined laws of physics. Therefore, the anomaly detection problem is much more challenging and requires novel advances compared to existing theories of anomaly detection in CPS.

[*]Both authors contributed equally to this research.

Furthermore, some previous works on smart metering [1] have attempted to solve the anomaly detection challenge in smart living CPS. However, such efforts used data collected from small experimental testbeds. Thus, the scale of the problem was smaller, and training data was free from noise. In contrast, our transportation CPS setting includes data collected from the wild, across a whole city. This needs to be accounted for in the design. Specifically, geospatial factors need to be blended with causal factors of the underlying structure of the data that characterizes benign situations.

While many prior works exist in this area, the effort in this paper takes the challenge for the whole city with a dataset analyzed over one year to account for all seasonal and human behavioral effects. The validation and the performance metrics reported are very robust compared to existing works in [7, 13, 14].

***Paper Contributions:*** We propose an unsupervised time series based anomaly detection framework for large-scale smart transportation networks that detects traffic incidents in real-time while maintaining a low false alarm rate. The framework automatically pinpoints the area of incident occurrence.

Specifically, we first show theoretical parallelism between the transportation problem and an existing anomaly detection metric (Harmonic to Arithmetic Mean ratios) previously developed for anomaly detection in smart energy systems. Second, we propose a region-growing approximation algorithm that allows to strategically partition the smart transportation CPS into clusters where the data is highly positively correlated. The strategic partitioning guarantees 1) high invariance of the anomaly detection metric and allows 2) decentralized cluster-wise implementation of our detection framework which enables the framework to pinpoint the area of the incident. Third, we propose a data cleaning and augmentation technique to enable learning the underlying structure of benign conditions from the data collected from the wild to reduce false alarms. Fourth, we give a technique to learn the bounds of the anomaly detection metric in each of the strategic partitions under normal traffic conditions to establish the anomaly detection criterion. Finally, we validate our approach through extensive large-scale experiments on real mobility datasets (6928 road segments over 1 year) acquired from the City of Nashville, Tennessee. Results show that our model is able to detect traffic incidents in real-time. The performance is measured by comparing our framework's decisions with a separate ground truth dataset containing actual incidents recorded by the Nashville Fire and Safety Department.

The rest of the paper is organized as follows: Section 2 discusses the related work. Section 3 introduces the transportation system model. Section 4 discusses the proposed framework. Experimental results are discussed in Section 5 followed by conclusions.

## 2 PREVIOUS WORK

Existing research on automatic incident detection for cyber-physical transportation systems broadly falls into two classes. They can be classified into *model-based* and *data-driven* approaches. Model-based approaches [6, 10] include probabilistic models [17], fuzzy C-means clustering [16], and state-based methods which used Kalman Filtering [9] to describe the state of monitored traffic so that usual traffic behaviors can be learned and unusual incidents can be detected. However, these methods require realistic assumptions for the target area and assume that their forecasting models are representative of true uncertainty in the data. Thus, they require extensive time-series validation.

Data-driven approaches, on the other hand, include classification methods which typically include nearest neighbors [12], neural networks, and support vector machines. These methods require labeled data to train and introduce new challenges regarding user data privacy. Techniques such as convolutional-LSTM models and neural networks [18], consume a lot of time comparing real-time data with historical data and have high computational costs limiting their effectiveness in decentralized deployments.

## 3 SYSTEM MODEL DESCRIPTION

The smart transportation CPS monitors the physical world of road conditions via TMC sensors that are deployed in each road segment. In our setting, there is one TMC sensor per road segment, so the number of TMC sensors equals the number of road segments. The data collected from TMC is used for various operational decisions that can control the appropriate volume of traffic to reduce the disturbance in mobility and travel times.

The TMC sensors are small computational units with minimal memory. Hence, each captured information is sent to a Road Side unit [15] (RSU). Each RSU receives data from multiple TMCs and has a larger computational power and memory. The RSUs usually have a wired backhaul link to an edge or cloud server, where data from all TMCs of an area of interest is accumulated. Depending on implementation variations, the RSU itself could also serve as a decentralized edge server for edge analytics. However, the system-level implementation is out of the scope of our paper. We provide a framework that can run on the edge or fog, based on the computational and networking capabilities available to the smart city.

For this paper, a traffic incident is an anomalous event such as *vehicular accidents, crime, or man-made disaster affecting traffic flow, fire, non-recurring high duration congestion* to which the police, emergency, and fire safety required a response. The ground truth information on incidents was collected from Nashville Fire and Safety Department. This ground truth information contains the location, time stamp, and date of each incident responded by the City of Nashville in the year 2019.

Our goal in this paper is to develop a framework and learn the parameters that automatically detect congestion in real-time in the test/deployment stage. The ground truth information during the testing set is used to measure the incident detection accuracy of our anomaly detection framework. The ground truth information during the training phase is used to cross-reference for data augmentation and cleaning that enables efficient learning of the underlying structure of data corresponding to benign conditions in the transportation CPS. Each TMC at the end of a time window $t$ sends the following information to the RSU: **timestamp**, **road segment ID**, **mean speed over the $t$-th time window**). The TMC sensor is located at the center of each road segment. Therefore, the distance between two road segments is the distance between the midpoint of any two road segments. The TMCs capture ambient speeds as vehicles pass by over a particular road segment.

## 4 PROPOSED FRAMEWORK

First, we provide a high-level overview of the framework. We summarize the notations used in this paper in Table 1.

***Theoretical Intuition***: We discuss the choice of harmonic mean to arithmetic mean ratio metric [1] as an anomaly detection metric, its relevance to the problem, and its advantages and modifications necessary to fit the transportation application.

***Region Growing Approximation***: For the metric to achieve invariance, we need spatial and temporal partitions of the high dimensional data at which the positive correlation within each partition is maximized, which is achieved through a region growing approximation algorithm.

***Pre-processing and Augmentation***: Due to the characteristics of real-world traffic data, the invariant contains the effects of accidents. This poses a practical problem for unsupervised learning problems such as anomaly detection. Therefore, our framework invokes a data cleaning and sanitization technique to augment synthetic benign samples of the invariant.

***Learning normal operating range of invariant***: Once the cleaning has been done, we obtain a low dimensional invariant that is a suitable candidate for pattern recognition of this invariant that remains stable when there are no incidents.

***Anomaly Detection Criterion***: We identify the best hyperparameter inputs to the training algorithm that gives the best output.

**Table 1: List of symbols.**

| Symbol | Description |
|---|---|
| $C$ | Total clusters in the target area |
| $c_k$ | $k^{th}$ cluster within the set of clusters $C$ |
| $S$ | Set of segments in the target area |
| $n$ | Number of segments |
| $S_{c_k}$ | Set of segments located in cluster $c_k$ |
| $s^l_{c_k}$ | $l^{th}$ segment in the $k^{th}$ cluster |
| $p$ | Speed correlation |
| $p^{(min)}$ | Correlation threshold |
| $p_{cut}$ | Cut off correlation value |
| $t$ | Time slot, based on temporal granularity |
| $m$ | Number of time slots |
| $d_{s^l_{c_k}}(t)$ | Mean speed $d$ at segment $l$ within $S_{c_k}$ at time $t$ |
| $HM_{c_k}(t)$ | Harmonic Mean of cluster $c_k$ at time window $t$ |
| $AM_{c_k}(t)$ | Arithmetic Mean of cluster $c_k$ at time window $t$ |
| $Q_{c_k}(t)$ | *Q-ratio* metric of cluster $c_k$ at time window $t$ |
| $\Gamma^{high}_{c_k}(t), \Gamma^{low}_{c_k}(t)$ | Upper and lower Safe margins for the ratio of cluster $c_k$ at time $t$, exceeding these results in non-zero residuals |
| $\tau^{max}_{c_k}(h), \tau^{min}_{c_k}(h)$ | Upper and lower standard limits of cluster $c_k$ over historical data $h$ |
| $\nabla_{c_k}(t)$ | Residuals for the ratios of cluster $c_k$, a non-zero residual indicates a possible anomaly |
| $\kappa$ | Scalar Factor Hyperparameter |
| $SF$ | Sliding Frame Size Hyperparameter |

### 4.1 Theoretical Intuition

For a large scale CPS application such as smart transportation, the anomaly detection metric should have the following properties:
(1) Invariance Under Benign Conditions: Under no incidents, the metric should show minimal change across time and across history. This is important to reduce false alarms given the low *base rate* of incident occurrence.
(2) Deviation Under Incidents: Under incidents, the metric should have properties that cause quick and discernible deviation in the metric. This is important to increase detection accuracy.

As a starting point, we leverage a recent result from [1] that showed that a collection of positively correlated random variables sampled repeatedly over time can be represented as a time series of *ratio between the harmonic to the arithmetic mean* of the aggregate data; and can be used as an anomaly detection metric. This is because this the metric is stationary in its time series as long as a positive co-variance structure can be preserved. Any unforeseen data falsification attack that disturbs the space-time covariance structure will cause deviations in the otherwise stationary time series of Harmonic Means to Arithmetic Means. In the following, we explain the theoretical explanation of why the HM to AM ratio is a good starting point for our problem and examine what novel theoretical and applied contributions are necessary to make it work for incident detection for a transportation CPS.

#### *4.1.1* Invariance Under Benign Conditions.

Here we explain why the harmonic to the arithmetic mean ratio is a candidate for an anomaly detection metric that is invariant under benign conditions.

The basic premise is that humans react with some shared driving behavior based on the time of the day, traffic level on the road, road type (highway or city lanes), and road width, etc. Such shared driving behavior in the absence of incidents causes driving speeds to increase or decrease together, or remain similar that in turn manifests itself as *having high positive correlation* among data points.

One of the achievements of [1] is that it proved that the upper bound on the absolute difference between the arithmetic mean and harmonic mean of the data collected from a positively correlated system depends on two things: (1) minimum possible value of the data (denoted by $d_{min}$) and (2) the average difference in data observed between any two arbitrary sensing end-points averaged over an appropriate time granularity (say, $T$), (denoted by $\xi(T)$).

As long as it can be guaranteed that $d_{min}$ and $\xi(T)$ do not change with time, the invariance in the harmonic to the arithmetic mean ratio is guaranteed. We identified, however, that $\xi(T)$ does not change *only under strategic spatial and temporal partitions which is nontrivial to achieve for a transportation CPS.*

Note that, the studies [1, 2] worked with a small experimental micro-grid. Furthermore, weather in a city affects all areas equally which implicitly preserves similar city-wide power consumption patterns. For the above reasons, a positive correlation was implicitly guaranteed in the advanced metering infrastructure (AMI) application. However, this is not the case with transportation applications.

In a transportation CPS, data is collected from the wild, and cities are a complex mix of narrower lanes and highways. The data is also affected by the uniqueness of the neighborhoods (e.g. downtown vs uptown) and thus a positive co-variance structure is not implicitly guaranteed. Therefore, a computationally tractable clustering method is required to achieve invariance.

#### *4.1.2* ***Deviation Under Incidents***.

Another key achievement of [1] and [2] is that it proved that any short-lived disturbance on the covariance structure will lead to deviation in any metric that combines Harmonic and Arithmetic Mean calculated from a highly positively correlated set of random variables. This is attributed to an asymmetry in Schur Concavity properties. The Harmonic Mean is strictly Schur Concave while Arithmetic Mean is Schur Convex. This imbalance causes deviations in the HM to AM ratio metric whenever any event triggers a decrease in the correlation.

*4.1.3* **Domain-Specific Challenges:** We need to realize the following domain-specific adaptations:

First, in [1], the strength of the positive correlation was implicit in smart metering CPS. However, in transportation CPS, several localized factors affect traffic data patterns in sub-areas of the city. This requires intelligent clustering that preserves a high space-time covariance structure strategically. Second, [1] was designed for power consumption data from smart meters for a small experimental micro-grid. In such applications, geospatial factors play little role, which is not the case with city-wide smart transportation CPS. This requires bounding the clustering region size. Third, AMI application had only one observation per hour and the framework proposed was suitable for attack detection and not incident detection. The time for detection of attacks was in the order of hours. In our CPS use case, incident detection needs to happen within minutes. This requires too many detection rounds, increases the false alarm reduction challenge. Fourth, in [1] the data was free from anomalies due to a controlled environment of an experimental micro-grid. Instead, this application contains data from the wild from a real city and therefore framework adaptions are necessary to learn the underlying structure of the benign pattern of the CPS.

## 4.2 Region Growing Clustering Algorithm

This is the main theoretical core of the contribution which mainly addresses the first two challenges. We need to strategically group the road segments into spatial clusters such that the speed data has maximum positive correlation which leads to the highest invariance. At the same time, the clustering needs to be geographically proximate for disturbances in the co-variance structure to have a causal link to the traffic incidents.

All the road segments exhibiting correlations above a threshold may be grouped together to form a cluster. Thereafter, if $C = \{c_1, ...c_k, ...c_K\}$ is a candidate cluster set and $s_i$ and $s_j$ are any two road segments where $i \neq j$, $1 \leq i, j \leq n$ such that $s_i$ and $s_j$ are in the same cluster $c_k$, we can formalize the problem as the following:

$$\begin{aligned} \max \quad & \sum_{c \in C} \sum_{\{s_i, s_j\} \in c} Cor(s_i, s_j) \\ \text{s.t.} \quad & Cor(s_i, s_j) > p^{(min)} \end{aligned} \tag{1}$$

In the above optimization $Cor(s_i, s_j)$ represents the correlation between two road segments and $p^{(min)}$ is a threshold. The above optimization problem is *NP* hard since with $|S|$ number of road segments, there is an exponential number of possible solutions which is computationally intractable. We need an approximation to the exact solution. This is done by first converting the clustering problem into a graph problem.

**Reformulation into a Graph Problem** We convert our optimal clustering problem into a graph problem, where we visualize each *road segment as a vertex* on the graph $G'$ and the road segment connections as an edge. The weight of an edge is equal to the correlation between the road segments (vertices) it connects. Fig. 1 shows the remapped graph abstraction from the original road network to our reformulated graph mapping.

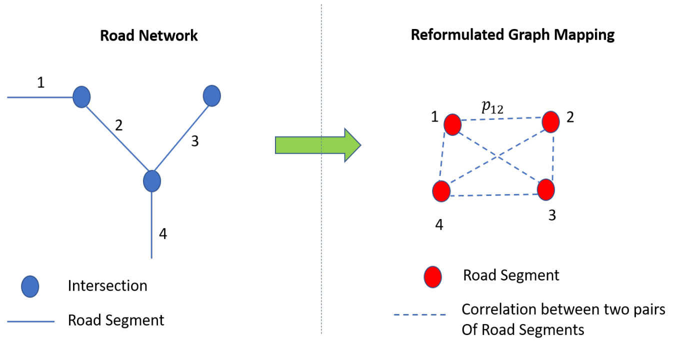

Road Network
1
2
3
4
Intersection
Road Segment
Reformulated Graph Mapping
$p_{12}$
1
2
3
4
Road Segment
Correlation between two pairs Of Road Segments

**Figure 1: Problem Reformulation to Graph Problem**

Theoretically, a correlation may exist between any pair of road segments. Therefore the initial graph $G'$ is a complete graph. However, since all road segments are not necessarily positively correlated (e.g. geographically distant, city roads to highways in the same geographical area), there will be edges with negative or zero weights and relatively low weights. Let there be a bound on the minimum correlation value $p_{cut} > 0$ necessary to be considered a feasible edge of the graph. All edges whose weights are less than $p_{cut}$ are pruned from the complete graph. A low $p_{cut}$ affects the level of invariance in the ratio invariant which is key to a low false alarm and improved detection performance.

Formally, this *reduced* graph is denoted as $G = (V, E)$, where $V$ is the set of vertices and $E$ is the set of edges. The set of vertices and edges are indexed by $v \in V$ and $e \in E$. Each edge $e$ is assigned a weight $p_e$ that is equal to the correlation between its two vertices, $v_i$ and $v_{j \neq i}$, where $v_i, v_j \in V$.

Edge Preference Hyper-parameter: From the feasible set, we introduce a notion of desirability to form the strongest grouping of clusters. Note, only using Euclidean distance is not appropriate for causal link because a narrow lane may have a highway road segment running over it. Geographically they are close, but even if one incident is affecting a ramp connecting the two, the correlation will not be as strong due to inherent differences in their physical characteristics. Also, some roads are long and can see an incident's effect quickly propagate, and segments not geographically very close still become affected by that same incident when not geographically close. Hence, we bring in a notion of edge preference hyper-parameter $p_{cut} < p^{(min)} < 1$. Using $p^{(min)}$ separates all edges $E$ into two subsets. We let the set $E^{s'}$ include all edges whose $p_e < p^{(min)}$ while the set $E^s$ include those edges whose $p_e \geq p^{(min)}$. This separation improves causal linkage.

**Distance Weight Variable $x_e$:** As explained earlier, geographically closer road segments will be affected by the same incident. Hence, the distance should be factored in the clustering too. Each edge $e \in E$ can be visualized as associated with a weight variable, $x_e \in (0, 1]$. The weight $x_e$ equals the normalized distance between two vertices (road segments) such that $x_e = \frac{dd_e}{dd_{max}}$, where $dd_e$ is the distances between two vertices of edge $e$ and $dd_{max}$ is the maximum distance among all distances between any pair of vertices.

Many optimization problems are formulated as error minimization problems where error is an unfavorable outcome that needs to be minimized. In our setting, two kinds of errors happen for any candidate solution (cluster). First, the two end vertices $\{v_i, v_j\}$ of an edge $e$ has correlation value $p_e > p^{(min)}$ but they are in two different candidate clusters (*positive error*). Second, the two end vertices $\{v_i, v_j\}$ of an edge $e$ has correlation value $p_e < p^{min}$ but they are in the same cluster *negative error*. By minimizing these two errors, the optimal clustering can be achieved, maximizing the correlations in a cluster.

**Transformed Optimization Problem:** In the graph-theoretic mapping of the original network, the original optimization can then be re-written as the following:

$$\begin{aligned}&\arg\min_{C} \quad \Big[\sum_{e\in E^s} p_e x_e + \sum_{e\in E^{s'}} p_e(1-x_e)\Big] \\ &\text{s.t.} \quad x_e \in (0,1]\end{aligned} \tag{2}$$

In the above optimization, the first term includes all positive errors for a given candidate solution $C$, when $e \in E^s$ (they have $p_e \geq p^{(min)}$) and $\{v_i, v_j\}$ are not in the same candidate cluster $C$. Similarly, the second term includes all negative errors for the same candidate solution $C$, when $e \in E^{s'}$ (they have $p_e < p^{(min)}$) and $\{v_i, v_j\}$ are in the same candidate solution $C$. We need to find the solution $C$ which jointly minimizes both errors.

In [3], a clustering problem for a weighted graph which has the same form as Eqn. 2, was solved. Their study revealed that the relaxed form of the integer problem has an $\Omega(\log n)$ integrality gap. Hence, the best-known factor of the approximation can be $O(\log n)$. Hence, it ensures theoretical guarantees to our approach.

**Approximation Algorithm:** To understand the core idea of the approximation algorithm, which is based on growing a region with radius $r$ from some random starting point, we first need to define some key elements.

**Region:** A $Region(v_{init}, r)$ is the set of road segments $v \in V$ that are within the area with radius $r$ from an initial vertex $v_{init}$.

**Cut**: A $cut(c)$, where $c \in C$, is the sum of the weights of the edges $e \in E^s$ with $p_e > p^{(min)}$ where each edge $e$ has one vertex $v_i$, within $c$ and the other $v_j$, is outside.

$$cut(c) = \sum_{\substack{\{v_i,v_j\}\cap c=1 \\ e\in E^s}} p_e \tag{3}$$

**Volume**: $vol(c)$ is also the total sum of the weights of the edges $e \in E^s$, with $p_e > p^{(min)}$, where at least one vertex $v_i$ or $v_j$ is inside the cluster.

$$vol(c) = \sum_{\substack{\{v_i,v_j\}\in c \\ e\in E^s}} p_e x_e \tag{4}$$

The algorithm 1 returns the set of different clusters. The formation of one cluster happens via the region growing process. The core of the region growing approximation corresponds to the lines 3-7 in Algorithm 1 which decides what is included in one cluster, while the rest of the algorithm repeats the process of finding clusters for the whole graph.

It starts at a random vertex $v_{init}$ with an initial radius $r_1 = r_{init} > 0$, that forms an initial region $Region(v_{init}, r_1)$. To illustrate, Fig. 2a represents an initial region centered on a random vertex (shaded in red) with radius $r_1$.

**Algorithm 1:** Approximation Algorithm

```
Input: $G = (V, E)$
Output: $C = c_1, ..., c_K$
Initialize: $k = 1, C = \{\}$
1  begin
2      while $G$ is not $\emptyset$ do
3          $select\ random\ v_{init} \in V$
4          $r \rightarrow r_{init}$
5          while $cut(Region(v_{init}, r)) \geq vol(Region(v_{init}, r))$ do
6              $r \rightarrow r + \min_{v\in V,\ v\notin Region(v_{init},r)} (d_{v_{init},v} - r)$
7          $c_k = Region(v_{init}, r)$
8          $insert(c_k, C)$
9          $remove(c_k, G)$
10         $k \rightarrow k + 1$
11     return $C = c_1, c_2, ...c_K$
```

Next it finds nearest vertex $v$, in the neighborhood of the initial region. To find the nearest vertex, it lists all other vertices $v_{near}$ such that each vertex in set $v_{near}$ includes only vertices that are directly connected to the region $Region(v_{init}, r_1)$. The term "directly connected to the region" implies that an edge exists between a vertex in $v_{near}$ and any vertex inside $Region(v_{init}, r_1)$ (any vertex shaded red). For illustration, in Fig. 2a, $v_1$ and $v_2$ are the only vertices directly connected to the $Region(v_{init}, r_1)$.

It then calculates the euclidean distance from $v_{init}$ to each vertex in $v_{near}$ and selects the vertex $v$ that is nearest $v_{init}$. Let this smallest distance (from $v_{init}$) be denoted as $d_{(v_{init},v)}$. This distance is used as the radius of the new region such that $r_2 = d_{(v_{init},v)}$. Since $r_2 > r_1$, the region grows from the initial region, hence the term region growing approximation. This is illustrated in Fig. 2b, where $r_2$ is formed by the distance between the nearest vertex $v_1$ and $v_{init}$.

Note, that with the new region, the set $v_{near}$ now includes $v_2$ and $v_3$. For simplicity, let's drop the suffix of the radius parameter such that the radius at any iteration of region growth is denoted simply by $r$. The region then continuously grows until the $vol(Region(v_{init}, r))$ is greater than $cut(Region(v_{init}, r))$ (See line 5 in Algorithm 1). When this stopping condition is met, the region stops growing and we achieve our first cluster $c_k = c_1$ where $k = 1$. We insert this first cluster into our final cluster set denoted by $C$ (See line 8 in Algorithm 1).

All the vertices in cluster $c_1$ are then removed from the graph $G$ to avoid duplication while generating other clusters (line 9 in Algorithm 1). Finally, $k$ is increased by 1 for the next iteration. The process starts again with a new initial region centered around a *new* random vertex $v_{init}$. It generates the cluster $c_k = c_2$ by executing the lines $3 - 7$ which is added to the cluster set $C$ before removing the vertices in cluster $c_2$ from graph $G$. Algorithm 1 continues until there are no vertices left to cluster (see line 2 in Algorithm 1). Once the graph $G$ is empty, the set of clusters $C$ is returned.

**Complexity Analysis:** The approximation algorithm takes polynomial time to cluster all the segments. To prove that, we can assume that no two vertices are at the same distance level from the initial random vertex $v_{init}$. Therefore, it will be safe to assume that at each iteration of the inner loop a single node will be added in $region(v_{init}, r)$. If there are $m_1$ number of vertices for the first cluster, then there will $n - m_1$ number of vertices available to cluster

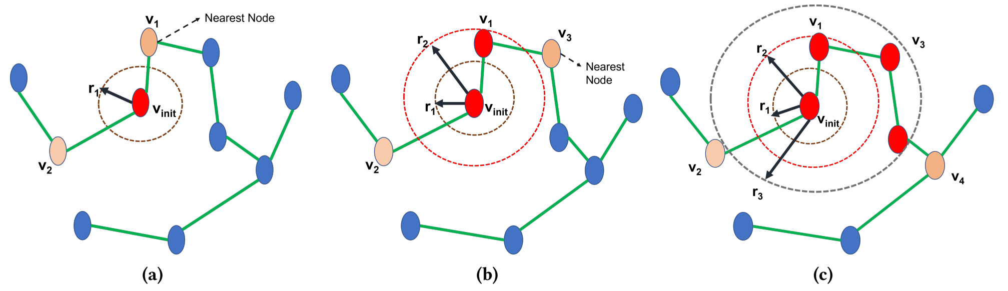


**Figure 2: (a) A region around the initial node. (b) Increased radius for the region based on the nearest segment whose one connecting end is inside the previous region. (c) A region where the volume is greater than cut.**

in the next iteration. If this same process runs $k^{th}$ number of times until there will be no vertex left to cluster, it can be shown that $n - m_1 - m_2 - ... - m_k = 0$ where $m_2, m_3, .., m_k$ are the number of vertices in successive clusters. This implies that $n = m_1 + m_2 + ... + m_k$ which is the total number of segments. Therefore, for $n$ number of segments, the algorithm performs clustering at most $n$ number of times. Hence, the complexity of the algorithm is bounded by $O(n)$.

## 4.3 Ratio Invariant per Cluster

The clustering process ensures clusters that maximize the correlation strategically. Let any cluster $c_k$ have $|S_{c_k}|$ number of road segments. Then, we calculate a ratio metric $Q_{c_k}(t)$ for every cluster $c_k$ at each time window $t$, which is the invariant. The ratio metric is defined as the ratio of the harmonic mean $HM_{c_k}(t)$ and arithmetic mean $AM_{c_k}(t)$ of data collected from all TMC road segments within a cluster such that:

$$HM_{c_k}(t) = \frac{S_{c_k}}{\sum_{l=0}^{|S_{c_k}|} \frac{1}{d_{S^l_{c_k}}}} \qquad AM_{c_k}(t) = \frac{\sum_{l=0}^{|S_{c_k}|} d_{S^l_{c_k}}}{S_{c_k}} \tag{5}$$

where $d_{S^l_{c_k}}$ is the aggregate speed reported by the $l$-th TMC for a time window $t$. Consequently, the ratio sample of the cluster $c_k$ at any time window $t$ is calculate by the following:

$$Q_{c_k}(t) = \frac{HM_{c_k}(t)}{AM_{c_k}(t)} \tag{6}$$

To illustrate the importance of the approximation algorithm for clustering to maximize positive correlation strategically, we compare the plots of time series of the ratio samples for the same time frame of the same day in Fig. 3 for the same area. Fig. 3a is a cluster with high data correlation (0.87) and Fig. 3b is a cluster with low data correlation (0.37). Observe that the time series of ratio samples in Fig. 3a is highly stable under benign traffic conditions (stationarity and low variance) and shows a sharp deviation on the incident that happened at 13:00 hrs. In contrast, Fig. 3b that did not maximize correlation has poor stability and does not show clear deviation in its time series when the incident happens.

An important thing to note is that every incident is unique in its manifestation and the method has to generalize for various clusters. Hence, we need to learn the underlying general structure of the ratio time series. However, the training data collected from the wild have incidents, and the data collected from connected transportation is very noisy due to human behavioral randomness. This is unlike traditional industrial CPS where the data patterns

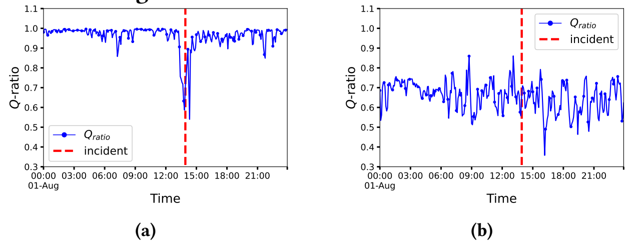


**Figure 3: Effect of cluster level correlation on invariance. (a) high correlation (0.87). (b) low data correlation (0.35)**

are only governed by tightly modeled laws of physics. Hence, we cannot simply learn the ratio samples themselves.

## 4.4 Data Pre-processing and Augmentation

Real-world mobility data collected from the wild (not from the experimental testbed), pose a practical problem for unsupervised learning problems such as anomaly detection, due to the presence of various incidents in the training phase. This prevents the learning of the underlying structure of benign data patterns. We need a mechanism to bypass this problem which we discuss here.

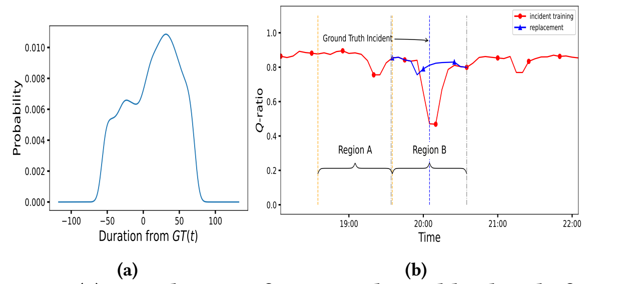


**Figure 4: (a) Distribution of Temporal Neighborhood of Disturbances across all incidents (b) Data Augmentation**

The intuition is to use the time and location stamp of the ground truth incidents and superimpose them on the ratio time series of the cluster which falls under the location of a particular incident. Then we identify the neighborhood of the time series of $Q_{c_k}(t)$ around all incidents to learn the portions of the time series that were disturbed. Unless these disturbances are cleaned out, it will prevent learning the structure of the benign behavior.

Note, ground truth incident recording itself is noisy due to human-in-the-loop issues. We observed in many cases, they are recorded much after the physical world has been affected by the incident. In other cases, the incident is reported and recorded instantly but it takes some time for the physical world to get really affected (e.g. in sparse traffic scenarios).

**Temporal Disturbance Period Selection:** We know that prominent incidents in the city cause large congestion that gets captured in the congestion factor metric available with the dataset. Additionally, the moving average of the invariant decreases near incidents. We utilize the decrease to differentiate between benign and noisy ratios (invariant) which are then used to select a neighborhood around the incident ground truth timestamp $GT(t)$. This can be visualized through Fig. 4b where region $B$ is such a neighborhood.

From the time stamp where the incident was recorded (minutes=0), we check how many incidents showed a low moving average of ratio samples for a window before and after (minutes=0). One can find $ar\%$ of the incidents create a decrease in the ratio time series for less than Minutes $=\pm y$ minutes. Hence, the temporal neighborhood of ratio time series sample around the $G(t)$ timestamp that needs to be cleaned and discarded is on average y minutes before and after the $GT(t)$ shown in Fig. 4a. In Fig. 4b, this region is marked as region B. In Fig. 4b, we showed region markings assuming, $ar = 60\%$, the $y$ is around 30 minutes before and after any corresponding ground truth time stamps. Similarly, any confidence interval can be used for cleaning.

**Ratio Sample Cleaning and Augmentation:** To clean the incident neighborhood, we discard the ratios of region B from the training examples of ratio samples and replace them with the cumulative moving average (CMA) of an equal length of time just before the start of region B (temporal disturbance window of a cluster).

As an illustration, the cumulative sliding moving average of the ratio samples from region A are copied into the discarded ratio samples from region B, as demonstrated by Fig. 4b. The CMA for any cluster $c_k$ at time $t$, is calculated by the following:

$$Q^{MA}_{c_k}(t) = \frac{(t-1)Q^{MA}_{c_k}(t-1) + Q_{c_k}(t)}{t} \tag{7}$$

This process is executed for all ratio sample neighborhoods of ground truth incidents found in all clusters during the training phase. The CMA of *region A* is then used to replace the signature in *region B*. Figure 4b shows the incident signature being replaced by the cleaned data. This allows the model to learn the underlying structure of the data without incidents.

## 4.5 Detection Framework Design

After cleaning effects of ground truth recorded incidents, there are other behavioral randomness and noise that make lowering false alarms challenging without sacrificing the detection accuracy. Therefore, a two-tier approach (NIST recommended [5]) to learning the thresholds and an appropriate anomaly detection criterion is essential. The two-tier principal mandates short-term and long-term errors of any underlying detection metric. We adapt this idea in our context in the following manner:

*4.5.1 First Tier Stateless Residuals.* The first tier uses the time series distribution of the ratios $Q_{c_k}$ to set up a varying threshold that follows the ratio distribution for each cluster $c_k$ where $k \in \{1, \cdots, K\}$. A particular ratio $Q_{c_k}(t)$ can be greater than or less than the mean ratio $Q^{Mean}_{c_k}(t)$. The acceptable margin creates the upper and lower side boundary using the mean ratio of a cluster $Q^{Mean}_{c_k}(t)$ and the standard deviation $\sigma^{c_k}$. The upper boundary is denoted as $\Gamma^{high}_{c_k}(t)$ and the lower boundary is denoted as $\Gamma^{low}_{c_k}(t)$. The boundaries are termed as safe margins which can be calculated using the following equations:

$$\Gamma^{high}_{c_k}(t) = Q^{Mean}_{c_k}(t) + \kappa\sigma^{c_k} \tag{8}$$

$$\Gamma^{low}_{c_k}(t) = Q^{Mean}_{c_k}(t) - \kappa\sigma^{c_k} \tag{9}$$

*4.5.2 Second Tier Stateful Residuals.* The second tier consists of two thresholds. These thresholds are termed as standard limits in [1]. The upper side standard limit is $\tau^{min}_{c_k}(h)$ and the lower side standard limit is $\tau^{max}_{c_k}(h)$. Setting up these two thresholds is not as straightforward as tier 1. To calculate the thresholds, the first step is to get the residuals $\nabla_{c_k}(t)$ of each time window using the safe margin. This residual will be used to again calculate the residual under curve $RUC_{c_k}(t)$ over a sliding frame of size $SF$ for each time window and the sub-region. Finally the $RUC_{c_k}(t)$'s are used to learn the standard limits by the given algorithm 2

***Residual*** is defined as the difference between the safe margin and the ratio which is outside the boundary. If a ratio is higher than $\Gamma^{high}_{c_k}(t)$, the residual will be positive and if a ratio is less than $\Gamma^{low}_{c_k}(t)$, the residual will be negative. The following equation calculates the residual:

$$\nabla_{c_k}(t) : \begin{Bmatrix} = Q_{c_k}(t) - \Gamma^{high}_{c_k}(t), & \text{if} \quad Q_{c_k}(t) > \Gamma^{high}_{c_k}(T); \\ = Q_{c_k}(t) - \Gamma^{low}_{c_k}(t), & \text{if} \quad Q_{c_k}(t) < \Gamma^{low}_{c_k}(t); \\ = 0, & \text{otherwise}; \end{Bmatrix} \tag{10}$$

***Residual Under Curve*** A non-zero residual indicates the possible presence of an anomaly. However, to confirm, a sum of residuals is calculated over a fix optimal time window size which can be called as sliding frame size. The summation is termed as the *residuals under curve.* It is calculated using the equation below.

$$RUC_{c_k}(t) = \sum_{j=t-FS}^{t} \nabla_{c_k}(k) \tag{11}$$

**Algorithm 2:** Calculate $\tau^{max}_{c_k}(h)$

```
1 for $c_k, t, \tau$ do
2     if $(RUC_{c_k}(t) < \tau$ then
3         $cc_{max} : \frac{|\tau - RUC_{c_k}(t)|}{2}$
4         $\mathbb{CC} \leftarrow cc_{max}$
5     else
6         $pp_{max} = |RUC_{c_k}(t) - \tau|2$
7         $\mathbb{PP} \leftarrow pp_{max}$
8 $\tau^{max}_{c_k}(h) = \frac{1}{\eta^+} \text{argmin}_\tau \left| \sum_{\mathbb{CC}} cc_{max} - \sum_{\mathbb{PP}} pp_{max} \right|$
```

***Learning Standard Limit*** The computed $RUC_{c_k}$ is later used to learn the standard limit using Algo. 2. The algorithm treats the interior and exterior RUC in a different manner by multiplying two different weights. An interior-point contributes less to the overall loss and an exterior point contributes more. The algorithm minimizes the difference between the loss of interior and exterior points to learn the optimal standard limit both for the higher and lower sides. Eventually, both of the learned thresholds use the same algorithm, here we have shown only for the $\tau_{c_k}^{max}(h)$. For $\tau_{c_k}^{min}(h)$ only the negative $RUC_{c_k}(-)$ are used whereas for $\tau_{c_k}^{max}$ only the positive $RUC_{c_k}(+)$ are used.

*4.5.3* **Anomaly Detection Criterion.** Similarly, the RUC can be calculated at every time window in the test set. Let $RUC_{c_k}(T^c)$ is the RUC value for the cluster $c_k$ in the test set at the current time window $T^{current}$. Then, the incident detection criterion is given

$$RUC_{c_k}(T^{current}): \begin{cases} \in [\tau_{c_k}^{min}(h), \tau_{c_k}^{max}(h)] & \text{No Incident;} \\ \notin [\tau_{c_k}^{min}(h), \tau_{c_k}^{max}(h)], & \text{Incident Inferred;} \end{cases} \tag{12}$$

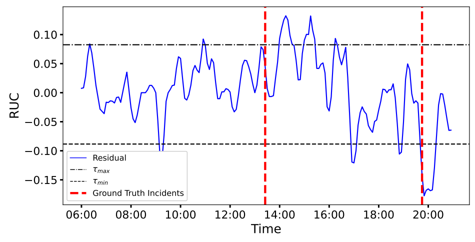
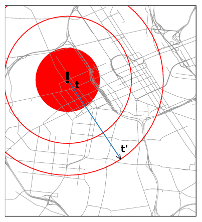
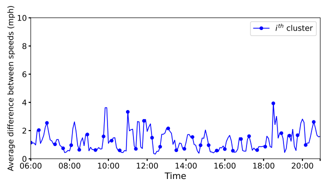


**Figure 5: Detection Illustration: RUC of $i^{th}$ cluster.**

Fig. 5 illustrates the incident detection where the vertical lines are the ground truth incidents and the horizontal lines represent the standard limits. we can see that RUC(T) metric goes beyond the learned standard limit near the growth truth time stamps.



**(a)**



**(b)**

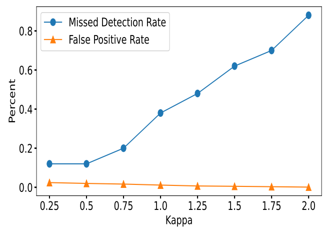
**Figure 6: (a) Propagation of Initial Incident. (b) Average Difference in Speeds.**

**Why ratios and RUC deviate?** As and when an accident occurs within a subarea (a cluster) of a city, the immediate neighborhood of the location where the accident happened, experiences a decrease in vehicle speeds instantly. However, this reduction in speed takes time to propagate beyond this immediate neighborhood until it affects the whole subarea. In Fig. 6a an accident occurred at time $t$ inside a cluster $c$, the reduction in speed in the immediate neighborhood (red circle) takes time $(t + n)$ to propagate outwards. The shaded region has decreasing speeds compared to the unshaded region that does not decrease creating a drop in correlation. This delay in propagation causes the deviation in the signature and can be detected by the metric as an anomaly. Fig. 6b, is an illustration that shows that the average difference between any two pair of TMC values *within the identified clusters* over time $\xi(T)$ are not varying too much, which is required for ratio stability.

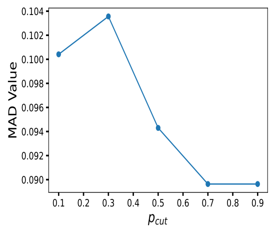
## 4.6 Hyperparameter Learning

There are four different types of hyperparameters. First set of hyperparameters is $p_{cut}$ and $p^{(min)}$ which affect the clustering process and the distribution of ratios. Second parameter set includes $\kappa$ and $SF$ values which affect the standard limits. For every combination of $\kappa$ and $SF$ the same ratio distribution will produce different standard limits. Here, we will discuss how the hyperparameters are learnt.

To learn the $p_{cut}$, we measure the deviation in invariance (ratios) $Q^{MAD}$ for different margin of $p_{cut}$. The $Q^{MAD}$ is used to select the $p_{cut}$ value since it directly affects the level of invariance in the ratiometric. Since the lowest median absolute deviation in the series imply the most stability, it implies that the smallest $p_{cut}$ for which the minimum value of $Q^{MAD}$ stops decreasing across consecutive values of candidate $p_{cut}$ is desirable. The smallest value is useful of $p_{cut}$ is recommended since too much positive correlation reduces the sensitivity to smaller incidents. It is shown in Fig. 7b.



**(a)**



**(b)**

**Figure 7: (a) Effect of varying $\kappa$ on the detection performance of a given cluster. (b) MAD over $p_{cut}$ for $k^{th}$ cluster.**

As we increase $p_{cut}$ from 0.0 to 0.99, the mean absolute deviation of the ratios in a cluster decreases. This trend continues until a certain point where the deviation stabilizes. Accounting this, we settle on $p_{cut} = 0.7$ as a lower bound. The hyper-parameter $p^{(min)}$ controls the area coverage and the performance of the cluster-wise incident detection. We learn it by the following:

$$\underset{p^{(min)}}{\arg\max}(CovR + TPR - FPR) \tag{13}$$

where $CovR$ is the coverage rate of road segments while clustering, $TPR$ true positive rate of detection, $FPR$ are the false positive rate, for incident detection. The above equation ensures the maximization of performance by reducing the false positives at the same time and increasing the coverage percentage. The approximate solution cover a majority of the area which had a total of 6,928 road segments. As we increase the correlation threshold, $p^{(min)}$, the radius becomes smaller with fewer segments being included in each cluster. However, as a result, there are now more clusters generated, resulting a larger coverage of the target area. The performance for different values for $p^{(min)}$ are given in Table 2.

For the incident detection model we learn the optimal value for the hyper parameters $\kappa$ and $SF$. The parameter $\kappa$ is a value in $(0, 3)$ and $SF$ is a sliding frame size which varies among the integer values in the set $\{3, 5, 7, 9\}$ . The optimal values of $\kappa$ and $SF$ are learnt by

$$\underset{\kappa,SF}{\arg\min} \quad (MDR + FPR) \tag{14}$$

where $MDR$ is the missed detection rate and $FPR$ is the false positive rate. The $\kappa$ and $SF$ from the above equation are selected as the optimal hyperparameters for the considered cluster.Each cluster $c_k$

**Table 2: Cluster Information**

| Cluster Info | $p^{(min)}$ 0.75 | 0.85 | 0.95 |
|---|---|---|---|
| count | 317 | 354 | 472 |
| mean | 16.06 | 14.54 | 11.81 |
| min | 4 | 4 | 4 |
| max | 161 | 132 | 112 |
| ave. data correlation | 0.863 | 0.862 | 0.867 |
| ave. radius (m) | 611.11 | 610.07 | 490.03 |
| area coverage | 73% | 74% | 80% |
| True Positive Rate | 0.916092 | 0.924653 | 0.926340 |
| False Positive Rate | 0.010727 | 0.032051 | 0.030957 |

has its own optimal selection of $\kappa$ and $SF$. Figure 7a shows the effect of the hyperparameter $\kappa$ on the detection performance of the $k^{th}$ cluster. A detection model with large $\kappa$ reduces the total number of false alarms however, it increases the number of missed incidents detected. The equation above ensures an acceptable performance.

## 5 EXPERIMENTAL EVALUATION

In this section, we introduce the dataset used for evaluation and the performance of our framework in terms of metrics such as incident detection rate, false alarm rate, time to detection of incidents, the impact of undetected incidents on the CPS application.

### 5.1 Details of Dataset

To evaluate our incident detection methods, we use one year (2019) long traffic mobility data collected from the city of Nashville, Tennessee. This dataset contains mobility data collected by the sensors at five-minute intervals. To the best of our knowledge, this dataset is bigger in duration and also in terms of area of coverage compared to validation used to existing works [7, 14]. For ground truth incidents, we use another dataset collected from Nashville's Fire and Emergency Response Department during the same year. In all phases of the experiment, we only consider the weekdays of 2019, focusing on the period between 6:00 AM to 9:00 PM since during night hours and weekend hours the traffic has discrete patterns. Details are shown in Table 3.

**Table 3: Detail of datasets**

| Data Sources | Properties | Values |
|---|---|---|
| Road network | # intersections<br># streets | 6928<br>19493 |
| Traffic incidents<br>Sensors | # instances<br># count | 8116<br>6928 |
| Data collection period | 01/01/2019 - 12/31/2019 | |

**Experimental Setup**: The twelve months of data is divided across training, cross-validation, and testing sets. The training phase learns the model for a combination of hyperparameters The cross-validation set is used to find the best hyperparameters that give the best outcome. The best hyperparameters are fitted to the model to find the final learned model that is used for testing.

The first eight months (Jan. to Aug.) are used for training. The next two months, (Sept. and Oct.), used for cross-validation. The final two months (Nov. and Dec.) are used to test the model itself.

*5.1.1 Training Dataset Details.* For training, we focus on the geographic coverage area Southwest($-87.050630, 35.989510$) and Northeast ($-86.527560, 36.416830$). Then, the segments inside the area are clustered following the clustering process discussed in Sect. 4.2. To cluster, the road segments from the transformed graph problem where correlation $p_e$ is assigned as the weights for each edges $e$ in the graph. We consider a cutoff correlation value $p_{cut}$ and a minimum level of correlation $p^{min}$ which leads to more invariance.

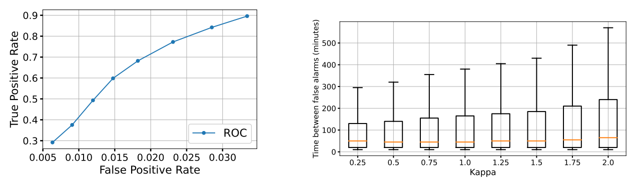
The clustering generated 354 clusters. However, we found that most incidents in the ground truth dataset were restricted to fewer clusters that correspond to the busiest parts of the city. Hence, we selected 25 clusters with the most reported incidents in the ground truth dataset. These 25 clusters were used to evaluate the performance of incident detection. For the temporal disturbance due to the ground truth incidents, the $ar = 60\%$ was used from the distribution of duration from $GT(t)$ variable. This corresponds to $y = \pm 30$ minutes around the neighborhood from all $GT(t)$. Since $t$ is slotted every 5 minutes, it boils down to 12 ratio samples cleaned and augmented per incident.

*5.1.2 Details of Testing dataset.* In this section, we evaluate our decentralized implementation of a lightweight anomaly detection framework. We used two months (Nov. and Dec.) from the dataset for testing. In these two months, there were a total of 851 incidents recorded in 580 active segments. We present how our technique gives us the ability to detect these incidents which can lead towards valuable actionable information.

### 5.2 Performance Results

We give the overall performance of the optimal learned model with a model selection of hyperparameters using the fitness function described in Section 4.6 at the end of this subsection. Before, that we show sensitivity analysis of our performance to changing hyperparameters instead of learned hyperparameters. This exercise gives deeper insight for generality against other unseen datasets where there may be a concept drift between the cross-validation and the test sets, which creates unpredictable performance when hyperparameters are learned from cross-validation instead.

*5.2.1 Sensitivity Analysis of Performance.* Here we give the sensitivity analysis of performance where the $kappa$ is not learned but varied as a free parameter to check its effect on the changing performance. The performance metrics include time to detection, true positive rate, false-positive rate, expected time between false alarms, and impact of undetected incidents.



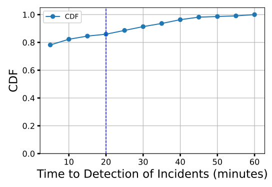
(a) (b)

**Figure 8: (a) Average ROC curve for 25 clusters (b) Mean time between false positives based on different $\kappa$ for $c_k$.**

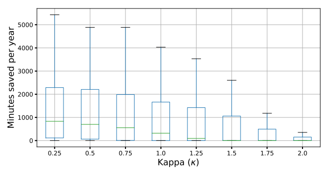
<u>Detection Rate and False Alarms</u> Fig. 8a shows the average ROC curve across the 25 clusters, underscoring the performance of our framework. One can see that at 90% true positive detection rate, the false alarm/false positive rate (FPR) is only 0.030. The low FPR is a significant achievement because: (1) anomaly detection methods are

prone to false alarms (2) due to lower rates of emergency/incidents, the cost of FPR is usually high for any CPS. Each cluster has 16,560 total detection attempts and the false positives are few and far in between. Fig. 8b shows the rarity of these false positives even when using $\kappa = 0.25$ which has the best overall detection rate. Specifically, Fig. 8b gives an idea on the expected time between two false alarms for various $\kappa$ values.

Mean Time to Detection A key performance indicator of usability in a CPS application is time to detection. Fig. 9a shows that 78% incidents were detected in the first 5 minutes and 90% incidents were detected within 30 minutes. Quick time to detection is essential to warn commuters earlier and control the flow of traffic to prevent congestion spread.



(a)



(b)

**Figure 9: (a) Time to Detection of Incidents (b) Impact on route travel times under different $\kappa$**

Impacts of Undetected Incidents By successfully detecting an incident, a traffic congestion will be prevented or mitigated and the commuters will be diverted to different routes, avoiding travel delays. We generated 200 routes that pass through segments with incidents and computed the travel time of each route with incidents. Using the true positive rates, we calculated what percentage of incidents will be detected. Detected incidents allow the system to notify commuters early and thus will experience less delay in their travel times. The time saved per vehicle is given by the following:

$$\triangle t = RD\left(\frac{1}{IS} - \frac{1}{FS}\right) \tag{15}$$

where $RD$ is the total distance in a given route, $IS$ is the speed due to incidents and $FS$ is the free-flow speed which is experienced when affected areas are avoided. Assuming on average that 10,000 vehicles pass by any segment per year, we can identify the impact of our anomaly detector on the travel time saved over a year. Figure 9b show the amount of travel time saved on a macro level depending on the hyperparameter and granularity used respectively.

*5.2.2 Overall Performance with Learnt Hyperparameter.* We applied the learnt values of $\kappa$, $SF$, $p_{cut}$ and $p_{min}$ for each of the 25 clusters. The final performance is an average from all 25 clusters for all traffic incidents over the full 2 months. Doing this, we found the average true positive detection rate $TPR = 0.90$ and $FPR = 0.03$.

## 6 CONCLUSION

We proposed an unsupervised time series-based anomaly detection framework for city-scale smart transportation CPS. We discuss how an existing anomaly detection metric (Harmonic to Arithmetic Mean ratios) can be applied to a transportation problem, by using a strategic partitioning of city area into positively correlated clusters that guarantee high invariance in detection metric. We utilize a data augmentation technique to enable unsupervised learning of the anomaly detection technique and learn the bounds of the technique under the sanitized, normal traffic conditions to establish anomaly detection criteria. Results show that our proposed unsupervised anomaly detection framework allows strategic partitions to independently generate, sanitize, learn and detect anomalies with high accuracy and low false-positive rates. This enables our approach to be deployed in a decentralized manner while maintaining high-performance anomaly detection in a real-time manner.

## REFERENCES

[1] Shameek Bhattacharjee and Sajal K. Das. 2021. Detection and Forensics against Stealthy Data Falsification in Smart Metering Infrastructure. *IEEE Transactions on Dependable and Secure Computing* 18, 1 (2021), 356–371. https://doi.org/10.1109/TDSC.2018.2889729

[2] Shameek Bhattacharjee, Venkata Praveen Kumar Madhavarapu, Simone Silvestri, and Sajal K. Das. 2021. Attack Context Embedded Data Driven Trust Diagnostics in Smart Metering Infrastructure. *ACM Trans. Priv. Secur.* 24, 2, Article 9 (Jan. 2021), 36 pages. https://doi.org/10.1145/3426739

[3] Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. 2006. Correlation clustering in general weighted graphs. *Theoretical Computer Science* 361, 2 (2006), 172–187. https://doi.org/10.1016/j.tcs.2006.05.008 Approximation and Online Algorithms.

[4] Keval Doshi and Yasin Yilmaz. 2021. An Efficient Approach for Anomaly Detection in Traffic Videos. *arXiv preprint arXiv:2104.09758* (2021).

[5] David I. Urbina Jairo Giraldo Alvaro A. Cardenas Junia Valente Mustafa Faisal. 2016. Survey and New Directions for Physics-Based Attack Detection in Control Systems. *NIST* NIST GCR 16-010 (2016).

[6] Yaser E. Hawas. 2007. A fuzzy-based system for incident detection in urban street networks. *Transportation Research Part C: Emerging Technologies* 15, 2 (2007), 69–95. https://doi.org/10.1016/j.trc.2007.02.001

[7] Yue Hu and Daniel B. Work. 2021. Robust Tensor Recovery with Fiber Outliers for Traffic Events. *ACM Trans. Knowl. Discov. Data* 15, 1, Article 6 (Dec. 2021), 27 pages. https://doi.org/10.1145/3417337

[8] Zafar Iqbal and Majid Iqbal Khan. 2018. Automatic incident detection in smart city using multiple traffic flow parameters via V2X communication. *International Journal of Distributed Sensor Networks* 14, 11 (2018), 1550147718815845. https://doi.org/10.1177/1550147718815845

[9] S.J. Julier and J.K. Uhlmann. 2004. Unscented filtering and nonlinear estimation. *Proc. IEEE* 92, 3 (2004), 401–422. https://doi.org/10.1109/JPROC.2003.823141

[10] Akira Kinoshita, Atsuhiro Takasu, and Jun Adachi. 2015. Real-time traffic incident detection using a probabilistic topic model. *Information Systems* 54 (2015), 169–188. https://doi.org/10.1016/j.is.2015.07.002

[11] Ayan Mukhopadhyay, Geoffrey Pettet, Sayyed Mohsen Vazirizade, Di Lu, Alejandro Jaimes, Said El Said, Hiba Baroud, Yevgeniy Vorobeychik, Mykel Kochenderfer, and Abhishek Dubey. 2022. A Review of Incident Prediction, Resource Allocation, and Dispatch Models for Emergency Management. *Accident Analysis & Prevention* 165 (2022), 106501. https://doi.org/10.1016/j.aap.2021.106501

[12] Murat Ozbayoglu, Gokhan Kucukayan, and Erdogan Dogdu. 2016. A real-time autonomous highway accident detection model based on big data processing and computational intelligence. In *2016 IEEE International Conference on Big Data (Big Data)*. 1807–1813. https://doi.org/10.1109/BigData.2016.7840798

[13] Yasas Senarath, Ayan Mukhopadhyay, Sayyed Vazirizade, hemant Purohit, Saideep Nannapaneni, and Abhishek Dubey. 2021. Practitioner-Centric Approach for Early Incident Detection Using Crowdsourced Data for Emergency Services. In *21st IEEE International Conference on Data Mining (ICDM 2021).*

[14] Yasas Senarath, Saideep Nannapaneni, Hemant Purohit, and Abhishek Dubey. 2020. Emergency Incident Detection from Crowdsourced Waze Data using Bayesian Information Fusion. In *2020 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. 187–194. https://doi.org/10.1109/WIIAT50758.2020.00029

[15] Siemens. 2018. Connected Vehicle Roadside Unit (RSU). Brochure. https://www.mobotrex.com/download/download-datasheet-for-siemens-connected-vehicle-roadside-unit-rsu/

[16] Hong Yang, Zhenyu Wang, Kun Xie, and Dong Dai. 2017. Use of ubiquitous probe vehicle data for identifying secondary crashes. *Transportation Research Part C: Emerging Technologies* 82 (2017), 138–160. https://doi.org/10.1016/j.trc.2017.06.016

[17] Kun Zhang and Michael A.P. Taylor. 2006. Effective arterial road incident detection: A Bayesian network based algorithm. *Transportation Research Part C: Emerging Technologies* 14, 6 (2006), 403–417. https://doi.org/10.1016/j.trc.2006.11.001

[18] Lin Zhu, Rajesh Krishnan, Aruna Sivakumar, Fangce Guo, and John W. Polak. 2019. Traffic Monitoring and Anomaly Detection based on Simulation of Luxembourg Road Network. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. 382–387. https://doi.org/10.1109/ITSC.2019.8917015

## A APPENDIX: ARTIFACT EVALUATION

In this section, the process of installing and running the software to detect incidents in a smart transportation system is discussed. Moreover, some of the results are presented by describing how they may be generated using our framework. Since the real data used for results in the main body of the paper is proprietary, the data cannot be disclosed. Therefore, we have prepared a dummy synthetic dataset specifically for artifact reproducibility. Therefore, while the exact framework is used, the results reported with the code will not be directly comparable to those in the main paper.

The synthetic dataset included in the artifact submission is generated from a fraction of the original dataset. The dataset is reduced both spatially and temporally as shown in Fig. 10 and Table 4. In Fig. 10 the area under the dotted line represents the randomly selected fraction of the entire Nashville city area for which the results will be generated. Under this area, there are only **1095** number of streets where as for the original data set we considered **19493** number of streets. It is possible to inspect for a larger area and the process will be same as described in the following sections. Table 4 summarizes the experimental details such the amount of data that is used for training, testing and validation.

For the purpose of the evaluation, we included the dataset and the Dockerfile in a public repository[1].

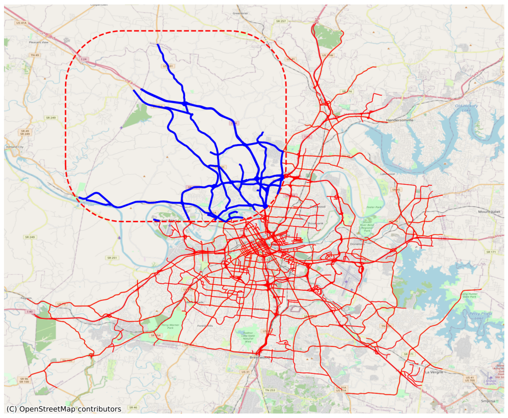


**Figure 10: Reduction of the Target Area for Artifact Evaluation, original in red lines, reduced in blue.**

**Table 4: Reduction of Dataset for Artifact Evaluation**

| Data Sources | Properties | Paper | Artifact |
|---|---|---|---|
| Road network | # intersections | 6928 | 598 |
| | # streets | 19493 | 1095 |
| Traffic incidents | # instances | 8116 | 89 |
| Sensors | # count | 6928 | 598 |
| Training dataset | | Jan. to Aug. | Oct. (1-14) |
| Testing dataset | | Sept. and Oct. | Nov. (1-14) |
| Validation dataset | | Nov. to Dec. | Dec. (1-14) |

### A.1 Requirements

The evaluation will be done inside a Docker container. Therefore, for the evaluation both `Docker` and `docker-compose` should be installed on the host platform. Though any operating system can be used to validate the evaluation our recommendation is to use a UNIX system. Any latest version of the Docker and docker-compose will work. However, the versions should be at least 20.10.5 build 55c4c88 for Docker and 1.27.4, build 40524192 for the docker-compose respectively. After installation of both Docker and docker-compose, verify that they are running before executing the evaluation.

[1]https://github.com/linusmotu/AnomalyDetection_ICCPS_Artifact

### A.2 Host Platform

For this evaluation, we prepared the Docker image on an Ubuntu device. The host platform should have at least 500MB free and a minimum of 8GB RAM. After installation, we ran the framework to generate the result. The instruction to run and generate the results will be discussed in the next section. The average run time varies depending on device configurations and it varies ranging from 300 to 1000 seconds. The entire evaluation uses the CPU only and a fast GPU is not required.

### A.3 Instructions

To start the evaluation, all the necessary files and required data are included in the repository. On the terminal on the host platform, execute the following commands.

(1) `git clone` https://github.com/linusmotu/AnomalyDetection_ICCPS_Artifact.git
(2) `cd AnomalyDetection_ICCPS_Artifact`
(3) Execute sudo `docker-compose up --build`, and wait. Note that sudo is required to bypass any permission issues.
(4) Results are saved as `.png` files in the `synthetic_figures` directory.
(5) Verify the resulting figures.

In table 5, the system configurations are listed.

**Table 5: Device Specification and Runtimes**

| Component | Specification |
|---|---|
| OS | Ubuntu 20.04.1 |
| CPU | AMD Ryzen Threadripper 3970X 32-Core |
| Architecture | x86_64 |
| GPU | NVIDIA GeForce RTX 3080 |
| Runtime | 300 sec |
| OS | Macbook Pro (2017) OSX 10.15.7 |
| Architecture | x86_64 |
| CPU | 2.9 GHz Quad-Core Intel Core i7 |
| GPU | Intel HD Graphics 630 |
| Runtime | 1068 sec |

### A.4 Generated Data

The results are varied based on the clusters generated by the clustering algorithm presented in Algorithm 1. The clustering algorithm creates 40 clusters on the considered area for the artifact evaluation. The summary of the table is given in Table 6. after generating the clusters.

The incident detection performed separately on each cluster, shows reasonably higher true positive performances which is approximately 80%. That indicates the frameworks compatibility with the main results regardless the scale of the problem. We can get much better accuracy if we can select an area where speed correlations show higher positive correlation. For this evaluation we allowed lower positive speed correlation to get decent cluster sizes.

**Table 6: Cluster Information**

| Cluster Info | $p^{(min)}$ 0.0 |
|---|---|
| no. of clusters | 40 |
| mean no. of segments | 7.525 |
| min | 4 |
| max | 21 |
| ave. data correlation | 0.007 |
| ave. radius (m) | 563.02 |
| area coverage | 50% |
| True Positive Rate | 0.794 |
| False Positive Rate | 0.268 |

It also confirms that the speed correlation based clustering has inversely proportional relation with the area coverage. The more the positive correlation threshold, the less will be the area coverage.

The following Fig. 11, shows the detection performance for a single cluster with varying kappa similar to the Fig. 7a. Fig 12 shows the ROC curve for the considered area and Fig. 13 shows the average time between two false alarms.

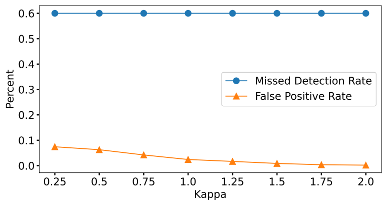


**Figure 11: Effect of varying $\kappa$ on the detection performance of a given cluster. Equivalent to Figure 7a in the paper.**

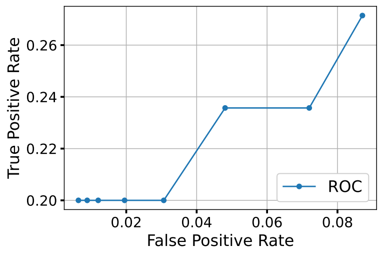


**Figure 12: Average ROC curve for 10 clusters. Equivalent to Figure 8a in the paper.**

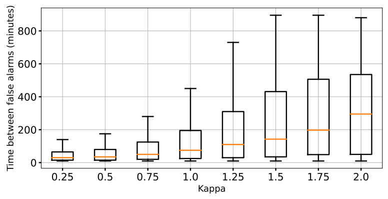


**Figure 13: Meantime between false positives. Equivalent to Figure 8b in the paper.**

In the main paper we used $p^{min} = 0.85$ and $p_{cut} = 0.7$ to get the best possible clusters. However, due to the anonymization done to generated the synthetic dataset, certain road segment relationships were lost. Thus, we could not generate the same quality of clusters as in the original paper. We had to select lower than standard pmin and pcut to generate clusters for the framework to run on. Thus, the results shown in Figures 11-13 deviate from the original results.

However, these results were generated in the exact framework as the original paper. Replacing the synthetic data with the original dataset will result in approximately similar outcome as the original paper.