Scalable Pythagorean Mean based Incident Detection in Smart Transportation Systems

MD. JAMINUR ISLAM[*], Western Michigan University, USA
JOSE PAOLO TALUSAN[*], Vanderbilt University, USA
SHAMEEK BHATTACHARJEE[‡], Western Michigan University, USA
FRANCIS TIAUSAS, Nara Institute of Sc. and Tech., Japan
ABHISHEK DUBEY, Vanderbilt University, USA
KEIICHI YASUMOTO, Nara Institute of Sc. and Tech., Japan
SAJAL K. DAS, Missouri University of Science and Technology, USA

Modern smart cities need smart transportation solutions to quickly detect various traffic emergencies and incidents in the city to avoid cascading traffic disruptions. To materialize this, roadside units and ambient transportation sensors are being deployed to collect speed data that enables the monitoring of traffic conditions on each road segment. In this paper, we first propose a scalable 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. Second, using cluster-level detection, we propose a folded Gaussian classifier to pinpoint the particular segment in a cluster where the incident happened in an automated manner. We perform extensive experimental validation using mobility data collected from four cities in Tennessee, compare with the state-of-the-art ML methods, to prove that our method can detect incidents within each cluster in real-time and outperforms known ML methods.

CCS Concepts: • **Computing methodologies → Learning latent representations**; • **Applied computing → Transportation**.

Additional Key Words and Phrases: Weakly Unsupervised Learning, Anomaly Detection, Smart Transportation, Graph Algorithms, Cluster Analysis, Regression, Incident Detection, Approximation Algorithm.

[*]Both authors contributed equally to this research.
[‡] Corresponding Author

Authors' addresses: Md. Jaminur Islam, Western Michigan University, Kalamazoo, MI, USA, mohammadjaminur.islam@wmich.edu; Jose Paolo Talusan, Vanderbilt University, Nashville, TN, USA, jose.paolo.talusan@vanderbilt.edu; 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; 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.

2378-962X/2023/1-ART1 $15.00
https://doi.org/10.1145/3603381

ACM Reference Format:
Md. Jaminur Islam, Jose Paolo Talusan, Shameek Bhattacharjee‡, Francis Tiausas, Abhishek Dubey, Keiichi Yasumoto, and Sajal K. Das. 2023. Scalable Pythagorean Mean based Incident Detection in Smart Transportation Systems. *ACM Trans. Cyber-Phys. Syst.* 1, 1, Article 1 (January 2023), 25 pages. https://doi.org/10.1145/3603381

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 [17].

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[5] 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 [11]. 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, is 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.

Furthermore, 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 [10, 20, 21].

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 the incident.

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 the strategic partition of 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 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 from four cities in Tennessee. Results show that our model is able to detect traffic incidents in a cluster, in real-time. We extend our work in [12] by identifying the segment within the cluster, where the incident originated. Pinpointing the incident at this level lets responsible agencies make decisions faster. 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. Section 5 extends the framework for segment-level detection. Experimental results are discussed in Section 6 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 [8, 14] include probabilistic models [24], fuzzy C-means clustering [23], and state-based methods which used Kalman Filtering [13] 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, requiring extensive time-series validation.

Data-driven approaches, on the other hand, include classification methods which typically include nearest neighbors [18], 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 [25], 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 [22] (RSU). Each RSU receives data from multiple TMCs and has a larger computational power and memory. The RSUs usually have a wired back-haul 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, timestamp, and date of each incident responded by the City of Nashville in 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.

4 PROPOSED FRAMEWORK

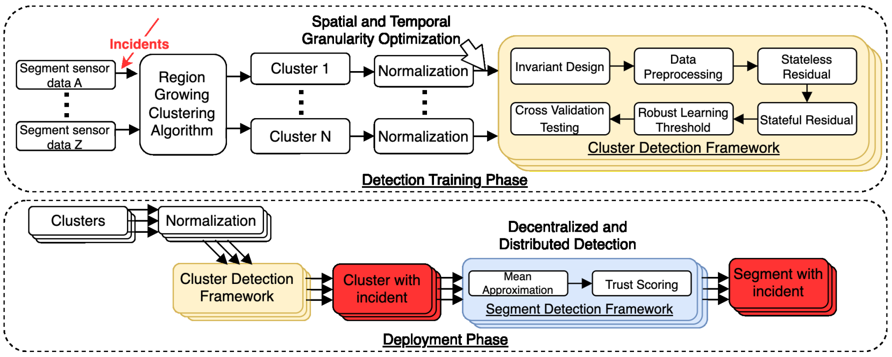


Fig. 1. Information architecture of the framework and the control flow of interaction between components. Aside from the region growing clustering, all components run in parallel. The cluster detection framework is generated in the detection training phase and is used on real-world data upon deployment.

First, we provide a high-level overview of the framework, its architecture can be seen in Fig. 1, followed by a summary of the notations used in this paper in Table 1. There are five logical modules in which the contribution is divided:

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.

Invariant Design: Involves metric derivation after the region growing approximation.

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 index t
$AM_{c_k}(t)$	Arithmetic Mean of cluster c_k at time index t
$Q_{c_k}(t)$	*Q-ratio* metric of cluster c_k at time index 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
$\tau^{max}_{c_k}(h), \tau^{min}_{c_k}(h)$	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
κ	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.

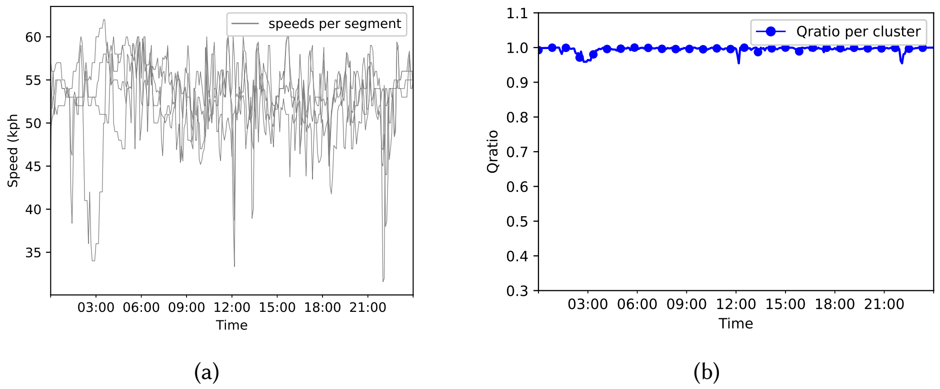


Fig. 2. Comparison of (a) Raw speed data and (b) Harmonic to the arithmetic mean ratio within a cluster.

4.1.1 **Invariance Under Benign Conditions.** 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*. We show in Fig. 2a that relying only on raw data from the sensors (harmonic mean) can lead to noisy data. However, the corresponding harmonic to arithmetic mean ratio is still stable under such conditions, Fig.2b. Note that, the studies [1, 2] worked with a small experimental micro-grid. Furthermore, weather 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 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

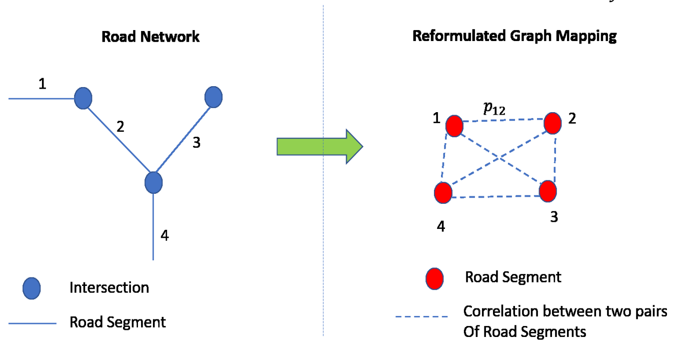

Road Network
1
2
3
4
Intersection
Road Segment
Reformulated Graph Mapping
p_{12}
1
2
3
4
Road Segment
Correlation between two pairs Of Road Segments

Fig. 3. Problem reformulation to graph problem.

road segments (vertices) it connects. Fig. 3 shows the remapped graph abstraction from the original road network to our reformulated graph mapping.

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 low false alarms and improved detection.

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 speed correlation between the two road segments which are further can be denoted in the transformed graph G as the two vertices, such as v_i and $v_{j \neq i}$, where $v_i, v_j \in V$.
Edge Preference Hyper-parameter (p_{cut}): 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:

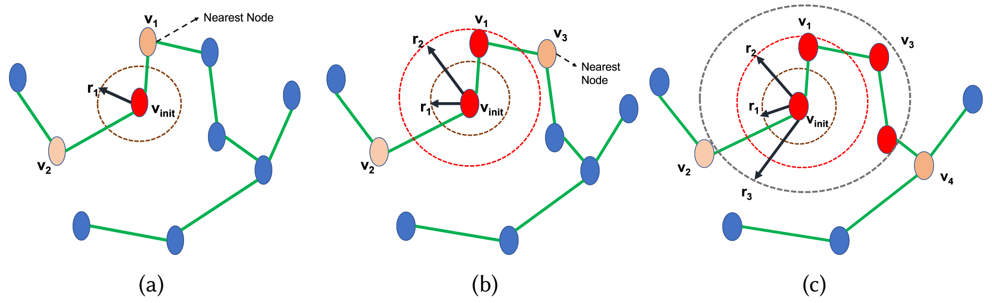


Fig. 4. (a) A region around the initial node. (b) Increased radius for the region based on the nearest segment whose one connecting end is inside the previous region. (c) A region where the volume is greater than cut.

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

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 [4], 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.

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

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          $r \longrightarrow r_{init}$
5          while $cut(Region(v_{init}, r)) \geq vol(Region(v_{init}, r))$ do
6              $r \longrightarrow r + \min_{\substack{v \in V \\ v \notin Region(v_{init}, r)}} (d_{v_{init}, v} - r)$
7          $c_k = Region(v_{init}, r)$
8          $insert(c_k, C)$
9          $remove(c_k, G)$
10         $k \longrightarrow k + 1$
11     return $C = c_1, c_2, ...c_K$
```

The algorithm returns the set of different clusters. The formation of one cluster happens via the region growing process. Algorithm 1 starts by checking if the graph G has more nodes to cluster in line 1. Nodes that are chosen as part of a cluster are then removed from G. The core of the region growing approximation are 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} at line 3 with an initial radius $r_1 = r_{init} > 0$ in line 4, that forms an initial region $Region(v_{init}, r_1)$.

In line 5 the algorithm checks if a cluster has met the terminating conditions based on both cut and volume. In Line 6 the algorithm grows the radius to include the nearest node outside of the cluster and repeats the process until reaches the terminating criteria. In Line 7 it forms the cluster c_k, and the formed cluster is added to the final cluster set C. The nodes in the cluster are then removed from the graph G. Then, the algorithm repeats the entire process until all nodes are clustered. To illustrate the algorithm 1, Fig. 4a represents an initial region centered on a random vertex (shaded in red) with radius r_1. Next, it finds the 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. 4a, 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. 4b, 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, we drop the suffix of the radius parameter such that the radius at any iteration of region growth is simply r. Then, the above process of region growing happens continuously The region then continuously grows until the stopping condition is met and we have 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. 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. Once the graph G is empty, the set of clusters C is returned. The approximation algorithm takes polynomial time to cluster all the segments. We proved the complexity in our previous work in [12] which is $O(n)$ where n is the number of road segments.

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 index 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 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 index t. Consequently, the ratio sample of the cluster c_k at any time index 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 the time series of the ratio samples for the same time and area in Fig. 5. Fig. 5a is a cluster with a high data correlation (0.87) and Fig. 5b is a cluster with a low data correlation (0.37). Observe that the time series of ratio samples in Fig. 5a 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. 5b that did not maximize correlation has poor stability and does not show clear deviation in its time series when the incident happens.

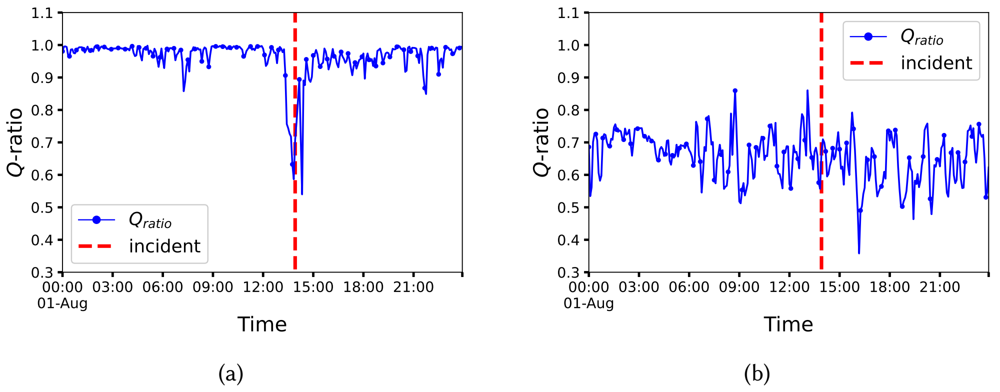


Fig. 5. Effect of cluster level correlation on invariance: (a) High correlation (0.87). (b) Low correlation (0.35).

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 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 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.

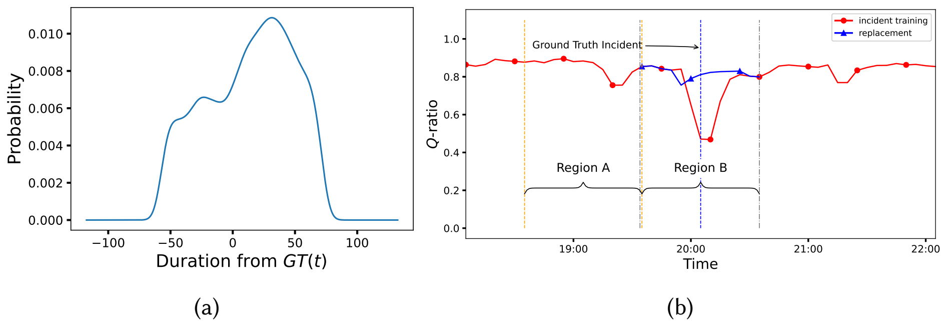


Fig. 6. (a) Distribution of temporal neighborhood of disturbances across all incidents (b) Data augmentation.

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. 6b 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. 6a. In Fig. 6b, this region is marked as region B. In Fig. 6b, 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. 6b. 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 6b 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 [7]) 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 σ^{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 index 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 index 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. 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:

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

Residual Under Curve A non-zero residual indicates the possible presence of an anomaly. However,

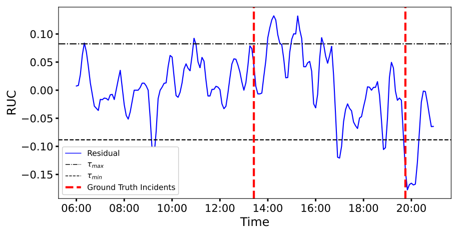


Fig. 7. Detection illustration: RUC of i^{th} cluster.

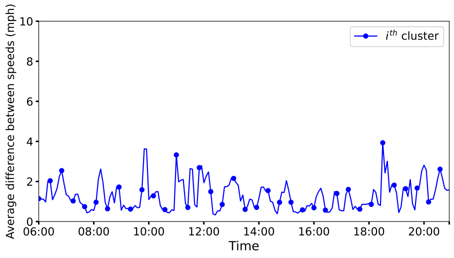


Fig. 8. Average difference in speeds.

to confirm, a sum of residuals is calculated over a fixed optimal time window size which can be called as sliding frame size. The summation is termed as *residuals under curve*. It is calculated by:

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

Algorithm 2: Calculate $\tau_{c_k}^{max}(h)$

```
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 $\tau_{c_k}^{max}(h) = \frac{1}{\eta^+} \text{argmin}_\tau \left| \sum_{\mathbb{CC}} cc_{max} - \sum_{\mathbb{PP}} pp_{max} \right|$
```

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 differently 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 index 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 index $t^{current}$. Then, the incident detection criterion is given

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

Fig. 7 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.

Why do ratios and RUC deviate? As and when an accident occurs within a subarea of a city, the immediate neighborhood of the location where the accident happened, experiences a decrease

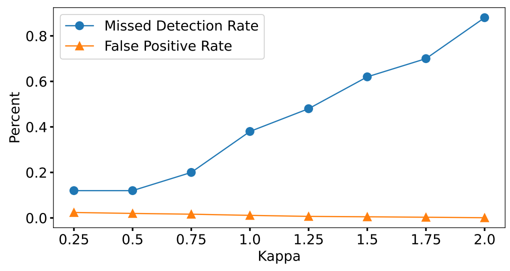


Fig. 9. Effect of varying κ on the detection of a given cluster.

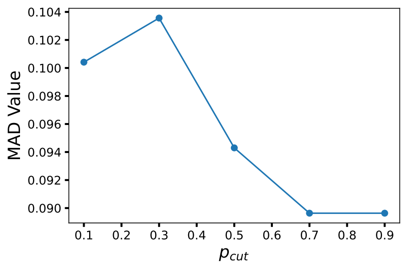


Fig. 10. MAD over p_{cut} for k^{th} cluster.

in vehicle speeds instantly. However, this reduction in speed takes time to propagate beyond this immediate neighborhood until it affects the whole cluster. This delay in propagation causes the deviation in the signature and can be detected by the metric as an anomaly. Fig. 8, is an illustration that shows that the average difference between any two pairs of TMC values *within the identified clusters* over time $\xi(T)$ do not vary much, which is required for ratio stability.

4.6 Hyperparameter Tuning

There are four different hyperparameters. The first set of hyperparameters is p_{cut} and $p^{(min)}$ which affect the clustering process and the distribution of ratios. The second parameter set includes κ and SF values which affect the standard limits. The distribution will produce different standard limits for every combination of κ and SF the same ratio. To learn the p_{cut}, we measure the deviation in invariance (ratios) Q^{MAD} for different margins 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 of p_{cut}, shown in Fig. 10, is recommended since too much positive correlation reduces the sensitivity to smaller incidents.

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 κ and SF. The parameter κ 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 κ 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 κ and SF from the above equation are selected as the optimal hyperparameters for the considered cluster. Each cluster c_k has its own optimal selection of κ and SF. Figure 9 shows the effect of the hyperparameter κ on the detection performance of the k^{th} cluster. A detection model with large κ reduces the total number of false alarms however, it increases the number of missed incidents detected. The equation above ensures an acceptable performance.

Table 2. Cluster Information.

	$p^{(min)}$		
Cluster Info	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

Table 3. $\mu^{MR}_{c_k}$ based rating levels.

Scenario	Discrete Rating Level (rl)
$\theta^{S^l_{c_k}}_{diff} \leq \Delta_{abs}$	4
$\Delta_{abs} < \theta^{S^l_{c_k}}_{diff} \leq 2\Delta_{abs}$	3
$2\Delta_{abs} < \theta^{S^l_{c_k}}_{diff} \leq 3\Delta_{abs}$	2
otherwise	1

5 SEGMENT LEVEL DETECTION

The primary motivation of segment-level incident detection is to provide optimal resource dispatch for agents that manage traffic incidents. Though cluster-level detection [12] helps to identify the incidents in a city block, it can not provide the exact location. Without this information, the emergency response will be more time-consuming since the department would need to manually locate the road affected or wait for citizens to call in. For example, one cluster can have 15 road segments within 600 meter radius. The agents would have to allot resources to locate the incident area before providing the service. In this context, our segment-level detection actually brings locality which allows them to respond and dispatch optimally. Another motivation was to address the scalability issue. A city or a region can have a vast number of road segments. For example, a region with 500 roads would require 500 detection models introducing plenty of overhead. In contrast, the cluster level first detects whether there is a problem with only one detection model. Then, only if there is an anomaly at the segment level would detection be initiated. Hence, the segment-level detection runs only when needed, it saves computation and networking resources that support such computations.

The incident detection framework detects incidents at a cluster level where each cluster c_k is made up of a set of S_{c_k} segments. Any segment $S^l_{c_k}$(l^{th} in cluster c_k) can be the origin of an actual incident that cannot be identified in the current framework. This motivated us to propose an extended architecture that would locate the origin of the incident after detecting the incident at a cluster level. The extended framework is a significant amendment to the prior framework that can help commuters save time by identifying segments and generating alternative routes.

For segment-level detection, first a cluster c_k under a detected incident at time t is considered to narrow down the origin (segment $S^l_{c_k}$) of the incident. Intuitively, in the cluster c_k, only the affected segment will show a change(decrease) in speed values. Therefore,the average speed $\mu_{c_k}(t)$ at time t also changes(decreases) for the cluster c_k. However, an approximation of the unaffected mean $\mu_{c_k}(t)$ of the cluster c_k can identify the affected road segment(where (segment)the incident occurred) by comparing the road segment speed $d_{S^l_{c_k}}(t)$ with the mean. Therefore, we approximate the value of mean $\mu_{c_k}(t)$ and denote it as corrected mean by $\mu^{MR}_{c_k}(t)$. Further, depending on the distance of the speed of each segment $d_{S^l_{c_k}}(t)$ with $\mu^{MR}_{c_k}(t)$ at time t, each road segment is assigned a rating level between 1 to 4. We calculate the rating for a time window of length T for a segment. A key point is that throughout the window T the approximation of mean $\mu^{MR}_{c_k}(t)$ is calculated only once at t. Using the rating values, a trust score per segment is calculated which lets us apply binary classification to identify impacted and non-impacted road segments in cluster c_k under the incident.

Overall, segment-level detection is divided into two major parts, the first is approximating the cluster mean and the second is the trust scoring model. The entire architecture is redefined to detect the road segments under incident for smart transportation problems from [3], where a trust score-based framework was first used to detect individual smart meters under attack.

5.1 Mean Approximation at Incident Detection Time

It is proven in [3] that if the individual values are decreased due to a data poisoning attack or other means, the means (arithmetic and harmonic) are also decreased. In such cases, the general mean μ can be corrected by using $\mu = AM + (AM - HM)$. Similarly, due to incidents, we assume that the speed values are decreased in a smart transportation system. Thereafter, the same mean correction can be utilized to get the approximation of the mean. Therefore, the correction is done by $\mu^{MR}_{c_k}(t) = AM_{c_k}(t) + (AM_{c_k}(t) - HM_{c_k}(t))$, and is calculated each time t when an incident is detected. Further, to measure the distance of the speed values $d_{S^l_{c_k}}(t)$ of each segments $S^l_{c_k}(t)$ the unbiased standard deviation is approximated. For corrected standard deviation, we simply take the standard deviation in the window just before the occurrence of the incident. That means for standard deviation correction we will take $\sigma^{MR}_{c_k}(t) = \sigma_{c_k}(t-1)$. Again, both the approximated mean $\mu^{MR}_{c_k}(t)$ and standard deviation $\sigma^{MR}_{c_k}(t)$ are calculated only once for the window length T.

5.2 Trust Scoring Model

Trust scores for each segment are computed in three steps. **First**, the corrected or approximated mean $\mu^{MR}_{c_k}(t)$ and standard deviation $\sigma^{MR}_{c_k}(t)$ are used to calculate ratings per segment for a fixed window of length $T = 30$ minutes. **Second**, each segment is assigned a weight based on the ratings received in the first step. **Lastly**, the weights are converted to a trust score using an inverse power law (IPL) based kernel function. In the following, each of these three steps is presented in detail.

5.2.1 Segment Rating. A rating of a segment $S^l_{c_k}$ at time t is an integer value that points out the closeness of the speed value $d_{S^l_{c_k}}(t)$ of the segment with the corrected mean value $\mu^{MR}_{c_k}(t)$ of a cluster c_k. A segment can have a different rating value based on time t within a window T. As mean value $\mu^{MR}_{c_k}(t)$ stays same for such window T, we remove the t from $\mu^{MR}_{c_k}(t)$ thus it becomes $\mu^{MR}_{c_k}$. Now, we calculate the absolute difference between the $d_{S^l_{c_k}}(t)$ and $\mu^{MR}_{c_k}$ which is denoted as $\Delta_{abs} = \sigma_{MR}$ for distance under one standard deviation. Thereafter, if the distance for a segment $S^l_{c_k}$ in cluster c_k has absolute difference denoted by $\theta^{S^l_{c_k}}_{diff} = d_{S^l_{c_k}}(t) - \mu^{c_k}_{MR}$, the segment get the rating according to the policy presented by Table 3. The Table 3 shows 4 different rating values where the highest rating is 4 which is closest to $\mu^{MR}_{c_k}$, and 1 is the lowest rating that denotes the furthest distance from the $\mu^{MR}_{c_k}$. The ratings for all segments in a cluster c_k are represented by a rating vector $r^{S^l_{c_k}}$. This rating vector is sorted as $r^{S^l_{c_k}}_{sort}$ in ascending order to assign weights to each segment depending on the rating value.

5.2.2 Segment Weight. Here each rating $r^{S^l_{c_k}}_{sort}$, for each segment $S^l_{c_k}$ in cluster c_k at time t gets a weight based on the Gaussian rating distribution . The weight for each rating are denoted as $W^{S^l_{c_k}}$ The lower rating gets lower weights and vice versa. After getting the rating, they are sorted which makes it easier to give lower weights to smaller ratings through Eqn 15 by dividing the rating space over the considered time window. In the following, a weight parameter $x(t)$ distributed between 1 to 4 is calculated as:

$$x(t) = 1 + \frac{(TK-1)t}{T-1} \quad \forall \quad t = 0, 1, ..., T-1 \tag{15}$$

In Eqn 15, $TK = 4$ denotes the total number of discrete rating levels in the system and T is the length of the window. Now, $x(t)$ from Eqn 15 is used to calculate the distance from highest rating level $\mu_{BR} = 4$. The larger the distance, the smaller the density value on the shape of the Gaussian distribution. The following equation is used to calculate the density as weights

$$cw_{S^l_{c_k}}(t) = \frac{1}{\sigma^{dr}_{S^l_{c_k}} \sqrt{2\pi}} \exp - \frac{(x(t) - \mu_{BR})^2}{2\left(\sigma^{dr}_{S^l_{c_k}}\right)^2} \tag{16}$$

In Eqn 16, $\sigma^{dr}_{S^l_{c_k}}$ represents the standard deviation of discrete ratings of each segment from $\mu_{BR} = 4$ in a window length T. The $\sigma^{dr}_{S^l_{c_k}}$ for each segment will be different based on different observations compared to common mixture data, which captures certain individual differences in speed. The raw weights from Eqn. 16, are normalized using $w_{S^l_{c_k}}(t) = \frac{cw_{S^l_{c_k}}(t)}{\sum_{t=0}^{T-1} cw_{S^l_{c_k}}(t)}$ Thereafter through an indicator function which is denoted as $I(rl, t)$, all the weights $w_{S^l_{c_k}}(t)$ corresponding to each unique rating level rl (between 1 to 4) withing T are added up to get the cumulative weight associated with each rating level. The cumulative weight is by $WD(rl)$ which is calculated by the following equation

$$WD(rl) = \sum_{t=0}^{T-1} w_t \quad I(rl, t) \tag{17}$$

In Eqn 17, the indicator function $I(rl, t)$ indicates whether any particular rating level rl occurs in the time slot t and It is written as below

$$I(rl, t) = \begin{cases} 1 & \text{if rl occurred in time slot t} \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

The cumulative weights for each rating level are considered together to get the final weight for each segment $S^l_{c_k}$. The weight is presented by $R_{S^l_{c_k}}$ of the l^{th} segment of k^{th} cluster which is a continuous value between 1 and 4 and is given by:

5.2.3 Trust Score. The weight ($R_{S^l_{c_k}}$) is used to get the trust score by injecting into an inverse power law (IPL) based kernel function. IPL transforms the weight value based on magnitude such as transforming larger weight to increase it, and vice versa. If a weight $R_{S^l_{c_k}}$ is 4, it means the highest trustworthiness which is followed by an exponential 'discounting' of trust, as $R_{S^l_{c_k}}$ decreases. The final transformed value which is the trust score is between 0 and 1. The following Eqn 19 is showing IPL based trust score calculation.

$$TR_{S^l_{c_k}} = \frac{(R_{S^l_{c_k}})^\eta}{(TK)^\eta}, \quad TR_{S^l_{c_k}} \in \{0, 1\} \tag{19}$$

In Eqn 19, η is a scaling factor controlling the rate of discounting. The Eqn 19, gives exponentially less trust to $R_{S^l_{c_k}}$ as it decreases from the maximum value of 4, in adherence to the Folded Gaussian shape of the rating distribution of road segments. The scaling factor η depends on the skewness of folded Gaussian in the benign data set. The Eqn. 19 produces trust values such that impacted and non-impacted road segments by any incident are linearly separable, which enables the calculation of an unsupervised threshold for classification.

5.2.4 Hyper Parameter Learning. The segment-level detection performance depends on the parameter η which is a scaling factor learned by maximizing the following objective

$$\eta^* = \underset{\forall \eta > 0}{\arg\max}(TPR) \tag{20}$$

In Eq.20 we select η that maximizes TPR. Further, the optimal value of η is data specific, hence we tune the parameter for each separate data. Parameter tuning was done by searching through a range from 0.25 to 10.0 in steps of 0.25 with 5 cross-fold validation. We selected the η with the highest overall accuracy across the cross-validation set for four cities.

5.3 Classification and Evaluation

Here, the trust scores are considered as input to an unsupervised binary classifier. The binary classifier learns a threshold based on which it divides the input set into two classes. The threshold is learned through a binary classification algorithm, K-medoids [19], which is quick and sufficient for the use case. The binary classification has two outputs either 0 or 1 where 0 refers to non-impacted and 1 denotes an impacted road segment. After getting the classification results of the road segments, the performance is evaluated utilizing ground truth information. In the following neighborhood selection, validation, and performance evaluation are presented as part of the evaluation.
Neighborhood Selection: The neighborhood area is identified by locating the real road segment where the incident occurred utilizing ground truth data. The neighborhood consists of several road segments depending on max hop distance hd which is an integer denoting the distance from the origin. Here, the max hd is set to 4 to get a neighborhood around the incident. Each segment in the neighborhood has a level of hd between 1 to 4 based on the hop distance from the incident segment. The lowest hop distance is 0 which denotes the incident segment itself. The highest hop distance is $hd = 4$ denoting the furthest segments. In the following, the neighborhood segments are, compared against the segments classified as impacted by the binary classifier.
Validation: To validate, the segments with hop distances 0 and 1 are considered to keep it more realistic. Intuitively, it requires significant time to propagate the incident impact up to $hd = 4$. Therefore, considering level>1 will unnecessarily create a bias in validation. The classification of a road segment is considered as **success** if it is classified as impacted and the road segment in the neighborhood has a hop distance of either 0 or 1. On the contrary, the classification of a road segment is considered as **failure** if the road segment is classified as impacted but the road segment is either not found or has a hop distance greater than 1.
Performance Evaluation: For performance evaluation, we simply count the successful classification while validating.For example if a cluster c_k has $|S_{c_k}|$ number of segment, the binary classification will have S_{c_k} predictions. Now, out of $|S_{c_k}|$, if the number of successful classification is $count_{Succ}$, the success rate is defined as $\mathbf{SR} = \frac{count_{Succ}}{|S_{c_k}|}$

6 EXPERIMENTAL EVALUATION

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

6.1 Details of Dataset

To evaluate our framework, we use one year-long traffic data collected from the city of Nashville, Tennessee by roadside sensors at five-minute intervals. This dataset is bigger in duration and in terms of coverage area compared to validation used in existing works [10, 21]. 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 weekdays of 2019, focusing on the period between 6:00 AM to 9:00 PM since during the weekend and late night, the traffic has discrete patterns. Details are shown in Table 4.

Table 4. Detail of datasets.

Data Sources	Properties	Nashville	Memphis	Knoxville	Chattanooga
Road network	# intersections	6,928	12,075	5,513	5,121
	# streets	19,493	27,144	11,786	11,532
Traffic incidents	# instances	8,116	12,113	4,745	3,470
Sensors	# count	6,928	12,081	6,073	5,560
Collection period		2019	2021	2020	

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.

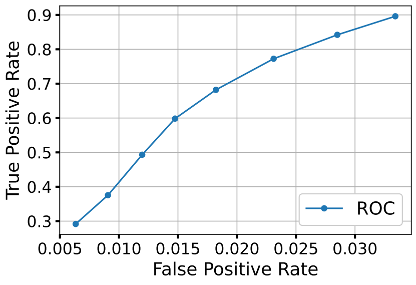
6.1.1 Training Dataset Details. To train, we focus on Southwest(-87.0506, 35.9895) and Northeast $(-86.5275, 36.4168)$. Then, the segments inside the area are clustered following the clustering process discussed in Sect. 4.2. To cluster, the road segments from the transformed graph problem where correlation p_e is assigned as the weights for each edge 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 were limited to fewer clusters that correspond to the busiest parts of the city. Hence, we selected 25 clusters with the most reported incidents from the ground truth and used it to evaluate the performance of incident detection. Limiting the dataset gives us the opportunity to validate our framework quickly since testing on clusters with little or no incidents would not prove whether our method can actually detect various kinds of incidents in accident-prone areas. 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, there are 12 ratio samples augmented per incident.

6.1.2 Testing Dataset Details. This section evaluates our decentralized implementation of a light-weight anomaly detection framework. We tested two months (Nov. and Dec.) from the dataset. In these two months, there were a total of 851 incidents recorded in 580 active segments. We present how our technique enables us to detect these incidents which can lead to actionable information.

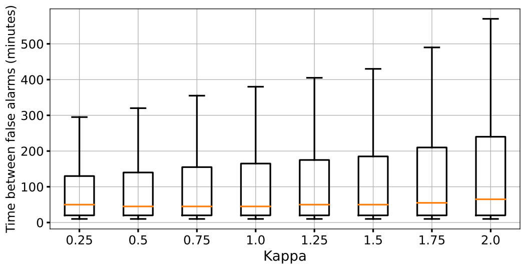
6.2 Performance Results

This section shows a sensitivity analysis of our performance to changing hyperparameters instead of learned hyperparameters. Then, we report the performance of the optimal learned model with hyperparameters using the fitness function described in Section 4.6.

6.2.1 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.

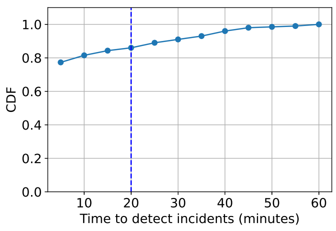


(a)

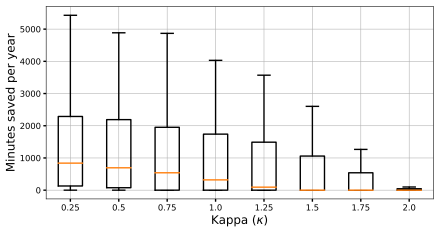


(b)

Fig. 11. (a) Average ROC curve for 25 clusters (b) Mean time between false positives with different κ for c_k.



(a)



(b)

Fig. 12. (a) Time to detect incidents (b) Impact on route travel times under different κ.

Detection Rate and False Alarms Fig. 11a 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 detection attempts and the false positives are few. Fig. 11b shows the rarity of these false positives even when using $\kappa = 0.25$ which has the best overall detection rate. Specifically, Fig. 11b gives an idea on the expected time between two false alarms for various κ.

Mean Time to Detection A key performance indicator of usability in a CPS application is time to detection. Fig. 12a 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.

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:

$$\triangle t = RD\left(\frac{1}{IS} - \frac{1}{FS}\right) \tag{21}$$

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 12b show the amount of travel time saved on a macro level depending on the hyperparameter and granularity used respectively.

6.2.2 Overall Performance with Learnt Hyperparameter. We applied the learnt values of κ, SF, p_{cut} and p_{min} for each of the 25 clusters. The final result is an average from all 25 clusters for all traffic incidents over the 2 months. The average true positive detection rate $TPR = 0.90$ and $FPR = 0.03$.

6.3 Comparative Analysis

We evaluate our framework further by comparing it with different baseline models and by using the additional dataset from three of the largest cities in the state of Tennessee.

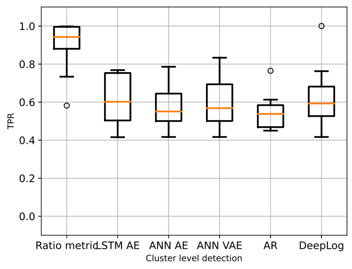
We compare our cluster-level incident detection against prediction-based (e.g., AR, DeepLog-LSTM) and reconstruction-based (e.g., AE, VAE, EncDec-AD) anomaly detectors. The models are trained on harmonic speed data instead of the ratio metric we previously derived. We identify incidents on the time series data per segment per cluster and augment them following the method described in Section 4.4. Figures 13a and 13b compare the true positive rates and false positive rates of cluster-level detection of the framework and baseline models. All results are obtained using the optimal hyperparameters SF and κ for each cluster. We used the same clustering for all methods. Hence, clustering-wise there is no performance gap. Later, we show that our detection part of the framework outperforms the other methods. And it is worth noting that the performance of our framework depends entirely on the quality of the in-variance which is the end product of the clustering strategy. Our detection model blends with the clustering approach resulting in improved performance compared to other methods. A summary for each model is as follows:

(1) AR [15] models use linear regression to calculate a sample's deviance from its predicted value which is then used to identify outliers. This model can handle multivariate time series data by training independent linear regression models for each dimension and computes the anomaly score for each sample. The score can be based on the mean, maximization, or median weighted deviance of each dimension.
(2) DeepLog-LSTM [6] is a deep neural network that uses LSTM to model a system log as a natural language sequence. It learns the log patterns from normal executions to detect anomalies.
(3) AE [9] is an autoencoder model base on a multi-layer perceptron (MLP) that encodes data into a lower dimensional latent space which is then reconstructed back to the original structure by a decoder. Anomalies are identified based on the reconstruction errors.
(4) VAE is similar to an AE. However, it outputs parameters of a pre-defined distribution in the latent space for every input.
(5) EncDec-AD [16] replaces the MLP layers in the AE with LSTM layers.

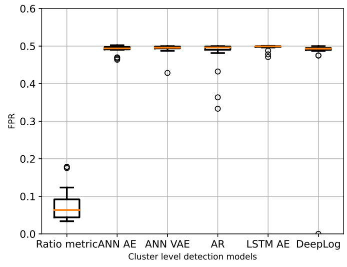
We extend our framework to Memphis, Knoxville, and Chattanooga. These three cities, along with Nashville are the four largest cities in Tennessee by population. We use one year of long traffic data for each city in our training and experimental evaluation. The data provider is the same as the Nashville dataset. Each contains "real-time" harmonic mean flow speeds. We perform the same division of data for training, cross-validation, and testing. Each city generated varying numbers of clusters, however, we only select 25 clusters with the most number of incidents for each city.

In contrast to the Nashville dataset, incidents for these three cities were gathered from crowd-sourced incident reports through Waze. Incidents were limited to only include accident reports. Data collection periods were also varied across three years, to show the ability of our framework to handle spatially and temporally different data. Fig. 14a and 14b show the ability of the framework to adapt and scale to other areas with minimal to no customization.

Overall, the results show the advantage of our framework against baseline models and when used in other cities and during different periods. Our framework offers higher TPRs and lower

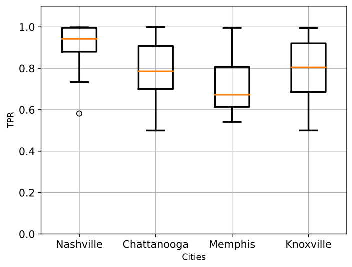


(a)

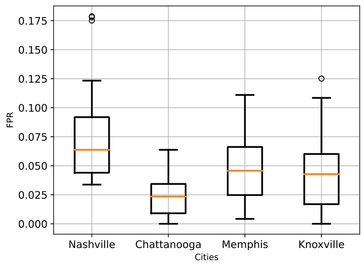


(b)

Fig. 13. Cluster level detection with various methods: (a) True positive rates and (b) False positive rates.

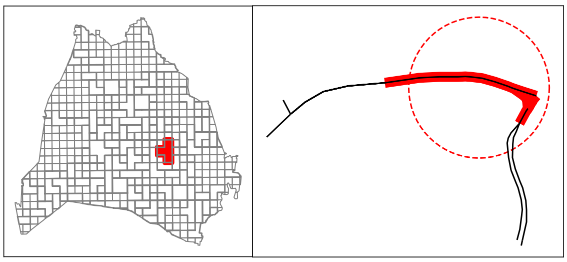


(a)

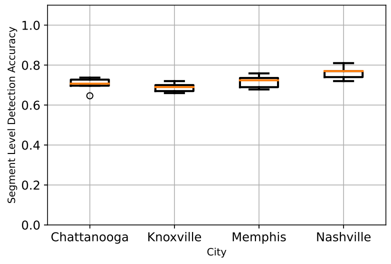


(b)

Fig. 14. Cluster level detection in various cities, TN: (a) True positive rates and (b) False positive rates.



(a)



(b)

Fig. 15. (a) Cluster and segment level mapping. (b) Segment level detection accuracy cities in TN.

FPRs across the board. Proving that it can be effective in the first step to not only detect incidents but to identify the extent of its effects both temporally and spatially.

6.4 Segment Level Incident Detection

After an incident is detected in any cluster, the framework must then be able to detect incidents at the segment level. However, incident reports are inherently noisy, not always aligning perfectly with segment information (harmonic speed data) which in contrast, is gathered in real-time. Reported incidents may be incomplete or incorrectly logged, they may also be reported late. Thus, to be able to correctly evaluate the framework, we have to specify a region around the reported incident origin to search. Detections are considered successful if the framework can detect segments located at or one hop away from the ground truth as discussed in Sec. 5.3. The left side of Fig. 15a shows an incident detected in one cluster within Nashville and on the right a group of segments in that cluster, we highlight the reported point of origin and its adjacent segments in red. We use the results from the cluster-level detection and select 100 successful incident detections from 25 clusters uniformly at random. We set the time window as the duration of time from the start of detection to the end, defined as the duration when the ratio decreased and then returned to normal. We do the same for each of the four cities and then calculate the rating levels of each segment for each incident window in each offending cluster. Figure 15b shows the accuracy of the proposed framework at detecting the affected segments given a successful detection at the cluster level. Detection at the segment level is only considered successful when it can detect both the origin and the adjacent segments. Given these requirements and the scale of the problem, where each dataset has 100 incidents across 1000s segments, the framework can identify the affected segments >75% of the time. The framework is lightweight and quick, and detection time is less than a second per incident, which enables it to be placed even on resource-constrained devices.

7 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 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. Further, we extend our incident detection framework to enable individual road segment detection under incident by trust score assignment. These enable our approach to be deployed in a decentralized manner while maintaining high-performance incident detection in a real-time manner.

ACKNOWLEDGMENTS

This research was supported by National Science Foundation, USA grants, SATC-2030611, CNS-1818942, CNS-1818901, SATC-2030624, OAC-2017289. We thank the Tennessee department of transportation and Nashville fire department for help in obtaining the relevant data for this research.

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] Shameek Bhattacharjee, Aditya Thakur, and Sajal K. Das. 2018. Towards Fast and Semi-Supervised Identification of Smart Meters Launching Data Falsification Attacks. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security* (Incheon, Republic of Korea) *(ASIACCS '18)*. Association for Computing Machinery, New York, NY, USA, 173–185. https://doi.org/10.1145/3196494.3196551

[4] 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.

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

[6] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (Dallas, Texas, USA) *(CCS '17)*. Association for Computing Machinery, New York, NY, USA, 1285–1298. https://doi.org/10.1145/3133956.3134015

[7] 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).

[8] 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

[9] Simon Hawkins, Hongxing He, Graham Williams, and Rohan Baxter. 2002. Outlier Detection Using Replicator Neural Networks. In *Data Warehousing and Knowledge Discovery*, Yahiko Kambayashi, Werner Winiwarter, and Masatoshi Arikawa (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 170–180.

[10] 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

[11] 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

[12] Jaminur Islam, Jose Paolo Talusan, Shameek Bhattacharjee, Francis Tiausas, Sayyed Mohsen Vazirizade, Abhishek Dubey, Keiichi Yasumoto, and Sajal K. Das. 2022. Anomaly based Incident Detection in Large Scale Smart Transportation Systems. In *2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS)*. 215–224. https://doi.org/10.1109/ICCPS54341.2022.00026

[13] 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

[14] 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

[15] Kwei-Herng Lai, Daochen Zha, Guanchu Wang, Junjie Xu, Yue Zhao, Devesh Kumar, Yile Chen, Purav Zumkhawaka, Minyang Wan, Diego Martinez, and Xia Hu. 2020. TODS: An Automated Time Series Outlier Detection System. *CoRR* abs/2009.09822 (2020). arXiv:2009.09822 https://arxiv.org/abs/2009.09822

[16] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2016. LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection. *CoRR* abs/1607.00148 (2016). arXiv:1607.00148 http://arxiv.org/abs/1607.00148

[17] 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

[18] 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

[19] Hae-Sang Park and Chi-Hyuck Jun. 2009. A simple and fast algorithm for K-medoids clustering. *Expert Systems with Applications* 36, 2, Part 2 (2009), 3336–3341. https://doi.org/10.1016/j.eswa.2008.01.039

[20] 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)*.

[21] 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

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

[23] 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

[24] 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

[25] 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