Multi-perspective Poisoning Detection using Graph Convolutional Network

Mario Luca Bernardi
Dept. of Engineering
University of Sannio
Benevento, Italy
bernardi@unisannio.it

Marta Cimitile
Dept. of Law and Digital Society
UnitelmaSapienza University
Rome, Italy
marta.cimitile@unitelmasapienza.it

Anna Vacca
Dept. of Law and Digital Society
UnitelmaSapienza University
Rome, Italy
anna.vacca@unitelmasapienza.it

***Abstract*—Federated Learning (FL) enables collaborative model training across distributed nodes while preserving data privacy. However, its decentralized architecture introduces significant security vulnerabilities, particularly susceptibility to poisoning attacks where malicious participants inject misleading information into the training process. In this paper, we propose a Multi-perspective Poisoning Detection approach that based on a Graph Convolutional Network architecture that integrates both temporal and spatial perspectives within a unified framework. This integration enables more robust and comprehensive drift analysis by applying simultaneous spatio-temporal evaluation to all detected drift instances. We evaluate the proposed approach on a new dataset generated starting from a public dataset.**

Experimental results demonstrate that the integrated hierarchical approach achieves superior detection accuracy with respect to the plain transformer-based autoencoder baseline and reduced false positive rates, enhancing the resilience of federated learning systems against sophisticated poisoning attacks while maintaining adaptability to legitimate concept drift.

***Index Terms*—poisoning attack, concept drift, transformer, gnn**

I. Introduction

Machine learning is increasingly used in everyday tasks. Distributed architectures such as Federated Learning (FL) can be used to train models on local data while preserving privacy.

However, decentralization introduces significant security vulnerabilities, including poisoning attacks that insert incorrect or deviant information into the training dataset [11], [18]. This can lead to serious consequences: models may fail to converge stably or, worse still, their final predictive outputs may be altered. Malicious participants can corrupt the training process by submitting manipulated model updates or training on compromised local data. These attacks can degrade model accuracy (untargeted attacks) or embed hidden backdoors that trigger misclassification of specific inputs (targeted attacks), ultimately compromising the integrity of automated decision-making and self-learning systems [7].

Although several defense strategies have been proposed in recent years, traditional defense mechanisms often assume static or Independently and Identically Distributed (IID) data, failing to consider the heterogeneity and non-stationary nature present in federated environments. Therefore, a robust defense framework must include contextual anomaly detection, leveraging metadata such as temporal and spatial coherence to distinguish true data drift from adversarial manipulation.

In a recent work [4], authors addressed this limitation by proposing the Concept Drift-based Data Poisoning Detection Approach (CoDaPD). CoDaPD was built upon the notion of concept drift [17], defined as variations in data distribution occurring over time, particularly prevalent in dynamic and non-stationary environments.

The underlying premise is that poisoning attacks on training data can generate fraudulent samples that mimic legitimate concept drift [10]. However, unlike genuine concept drift, adversarial-induced drift tends to be abrupt and spatially confined.

A notable limitation of this study lies in its two-steps structure. In the first step (temporal analisys), the main assumption is that if the local data stream deviates abruptly from the expected dynamics and a concept drift is present, this can be due to a possible malicious attack. In the second step (spatial analisys), the detected sudden drift undergo additional neighbor analysis and isolated drifting nodes are labeled as adversarial.

In this study, we propose a Multi-perspective Poisoning Detection approach consisting to overrun the described assumption by introducing a Hierarchical Transformers Autoencoder, including and integrating both the temporal and spatial perspective. This allows for a more effective disambiguation of genuine drifts from malicious ones by exploiting the complex dynamics that occur around the neighborhood of each node by applying both the spatial and temporal analysis to all the detected drift. This study also proposed an evaluation of the proposed approach on the Caltrans's PeMS (Performance Measurement System) dataset.

The next sections of the paper are organized as follows: related works (Section II), the fundamentals of poisoning attacks and conceptual drift (Section III), the detailed methodology (Section IV), the validation and the results obtained (Sections V and VI), and then conclude with a final discussion (Section VII).

II. Related Work

The security of distributed machine learning systems has become a critical research concern, particularly regarding poisoning attacks that corrupt the training process [20], [22].

Several defense mechanisms have been proposed to counter these threats in distributed environments. Baracaldo et al. [2] proposed leveraging contextual information about data provenance to detect malicious samples, while robust aggregation methods attempt to mitigate the influence of corrupted updates [6]. More recently, Laridi et al. [12] proposed federated autoencoder-based anomaly detection using summary statistics for threshold calculation, while authors in [23] introduced variance-minimization techniques that assign differential weights to local updates rather than eliminating outliers.

In the automotive domain, Wang et al. [21] addressed poisoning detection through bi-level optimization with variable perturbation parameters. Despite these advances, most existing solutions fail to account for the dynamic nature of data streams. As highlighted by recent surveys [1], [9], [15], concept drift represents a fundamental characteristic of real-world federated environments that defense mechanisms often overlook. Korycki and Krawczyk [10] demonstrated that adversarial instances injected into data streams can simulate concept drift, exploiting adaptation mechanisms to damage learning systems. Building upon these foundations, our previous work [4], [5] transferred these considerations to a distributed federated context where each client may be subject to adversarial drift. To the best of our knowledge, this represents the first application of drift analysis for poisoning detection in federated scenarios. Unlike existing approaches, CoDaPD enables attack identification based on the contextual dynamics of data generation and evolution. This paper introduces a further novelty with respect to [4] by proposing a Graph Convolutional Network to better integrate spatial and temporal drift analysis in a multi-perspective poisoning detection approach.

III. Background

Distributed cyber-physical and IoT infrastructures continuously collect telemetry from spatially deployed sensing devices (e.g., traffic stations, environmental monitoring units, industrial sensors).

In this context, concept drift is a phenomenon that occurs when the distribution of data in a stream (streaming data) changes over time. Thus, a machine learning model suddenly encounters data that no longer follows the same rules or statistical relationships as before [14].

The introduction of concept drift can introduce errors or misbehaviour which can lead to a drastic drop in model performance if not managed correctly. Concept drift can arise in different ways: it can be gradual when the changes are slow (evolving trends), or can be abrupt (new laws, pandemics, technological changes, accidents, malicious events). In this paper, we follow the definitions provided in [14] since it provides a general framework which we adhere to. To detect drift and discriminate if it is malicious or a genuine phenomenon, machine and deep learning approaches can be exploited. In this paper, we compare two models: a transformer-based autoencoder network which is used [4] to mainly detect the kind of drift from a temporal viewpoint (sudden or gradual) and our proposed model based on a spatio-temporal graph convolutional network adapted to the domain of distributed federated-learning context. In the remaining of the section more details on these two models are provided.

A. Transformer

A Transformer is a neural network architecture originally used for natural language processing. One of its distinctive features is its self-attention mechanism, which allows the model to "weigh" the relative importance of different parts of an input (examples of input could be the words in a sentence or parts of an image). Therefore, instead of analyzing data sequentially as with recurrent networks, the Transformer considers the entire context simultaneously and determines which elements are most relevant [8]. To perform these operations, the input is divided into units. Subsequently, through attention, each unit is compared with the others to determine how much it contributes to the overall meaning (relationship calculation). These relationships are then aggregated into vectors that capture both local content (details) and global content (overall structure) (representation construction). Finally, multiple layers of Transformers progressively refine understanding, moving from simple details to complex concepts (hierarchical processing).

B. Spatio-temporal GCN-based Classification

This setting naturally induces a graph representation, where nodes correspond to sensing units and edges encode spatial adjacency, functional coupling, or network connectivity. Graphs are typically irregular, with heterogeneous degrees and no canonical ordering of nodes, which complicates the definition of fundamental operations such as convolution. Moreover, the classical assumption of instance independence fails, as each node observation is conditionally dependent on its neighborhood. These aspects are further amplified in monitoring scenarios, where node attributes evolve over time and the detection target is not a static label but rather a dynamic deviation pattern.

Referring to Table I, we introduce the graph formalization adopted for spatio-temporal monitoring and the GCN-based classification framework. A spatio-temporal monitoring graph is defined as $G = (V, E, \mathbf{X})$, where V is the set of sensing nodes (devices, stations, or software agents), and E is the set of edges encoding

spatial adjacency or coupling relations among nodes. Each node $v \in V$ is associated with a time series of measurements $\{\mathbf{s}_v(t)\}_{t=1}^{T}$, where $\mathbf{s}_v(t) \in \mathbb{R}^m$ is the multivariate telemetry observed at time t. To integrate temporal evidence into a graph learning pipeline, we map each node time series to a windowed descriptor $\mathbf{x}_v(\tau) \in \mathbb{R}^d$ over a window W_τ, and $\mathbf{X}(\tau) \in \mathbb{R}^{n \times d}$ stacks all node descriptors at window index τ.

Graph Convolutional Networks (GCNs) extend the convolution operation to graph-structured data by aggregating information from a node's local neighborhood. Given a graph $G = (V, E)$ with adjacency matrix $\mathbf{A}$ and node features $\mathbf{X}$, a GCN layer updates node representations by propagating and transforming information across edges. At each layer l, the hidden representation of node v is updated as:

$$\mathbf{h}_v^{(l+1)} = f_a\left(\sum_{u \in N(v) \cup \{v\}} \frac{1}{\sqrt{d_v d_u}} \mathbf{W}^{(l)} \mathbf{h}_u^{(l)}\right), \qquad (1)$$

where $\mathbf{h}_v^{(0)} = \mathbf{x}_v$, d_v and d_u are node degrees, $\mathbf{W}^{(l)}$ are learnable weight matrices, and $f_a(\cdot)$ is an activation function. By stacking L layers, each node's final representation $\mathbf{h}_v^{(L)}$ aggregates information from its L-hop neighborhood.

For node classification into K mutually exclusive classes, a linear layer followed by softmax is applied to the final representations:

$$\hat{\mathbf{y}}_v = \mathrm{softmax}\Big(\mathbf{W}_{\text{out}}\, \mathbf{h}_v^{(L)} + \mathbf{b}\Big) \in \mathbb{R}^K, \qquad (2)$$

where $\hat{y}_{v,c}$ represents the predicted probability that node v belongs to class c. The network is trained by minimizing the cross-entropy loss over labeled nodes. The key property of GCNs for spatio-temporal monitoring is that classification decisions are informed not only by the node's own features but also by the features of its spatial neighborhood.

TABLE I: Notation.

Symbol	Description
$\lvert\cdot\rvert$	Cardinality of a set.
G	A generic graph.
V	Set of nodes, with $n = \lvert V\rvert$.
E	Set of edges, with $l = \lvert E\rvert$.
e_{ij}	Edge $e_{ij} \in E$ from node i to j.
$N(v)$	Neighborhood set of node v (1-hop, unless otherwise stated).
$\mathbf{A}, \mathbf{A}^T$	Adjacency matrix of a graph and its transpose.
$\mathbf{X}$	Node-attribute matrix.
$\mathbf{x}_v$	Attribute vector of node v.
$\mathbf{h}_v^{(l)}$	Hidden representation of node v at GCN layer l.
$f_a(\cdot)$	Activation function (e.g., ReLU).
$\mathbf{W}^{(l)}$	Learnable weight matrix at layer l.
W_τ	A temporal window indexed by τ.
y_v	Class label of node v.
$\hat{\mathbf{y}}_v$	Predicted class probability distribution for node v.

IV. The approach

The Multi-perspective Poisoning Detection (MPPD) architecture is designed to identify and neutralize adversarial concept drift within distributed environments. As illustrated in Figure 1, the framework comprises n distinct client nodes (each node corresponding to a mapped station), each producing local models derived from their individual data streams. However each client node includes a *Local Model Manager* performing the training of local models. The *Local Model Manager* uses, as underlying model, the DCRNN (Diffusion Convolutional Recurrent Neural Network) model [13] for the traffic detection.

The trained models are subsequently transferred to the *Poisoning Detector* component of the *Federation Server* for aggregation. This component also receives from the *Windows Aggregator* a graph-based representation of all the client data windows (they are sent by each node from the *Local Data Manager*).

The *Poisoning Detector* is designed to identify the poisoned models, preventing them from corrupting the global learning process. The poisoned model detection is performed by using Graph Convolutional Network [16], which considers both the temporal and spatial drift to evaluate drift graduality and drift's propagation within the network. The Aggregation Manager synthesizes validated local models into a unified federated global model. To maintain system integrity, it incorporates a filtering mechanism that discards any models identified with adversarial drift. This process utilizes a specialized federated learning strategy known as DQFed [3] to perform the final aggregation.

The *Global Model Manager* then receives the resulting global model and redistributes it to the participating clients, thereby closing the federated learning loop. This systematic framework enables MPPD to preserve model robustness against poisoning attempts while remaining responsive to legitimate concept drifts. A granular analysis of the main components and of their contribution to adversarial drift detection and mitigation follows in the subsequent subsections.

A. Poisoning Detector

We address data poisoning detection by framing it as a multi-class node classification problem. The key insight is that poisoning attacks induce *locally incoherent* temporal patterns: a compromised node exhibits anomalous drift while its neighbors remain stable or show uncorrelated dynamics. In contrast, genuine anomalies caused by real-world phenomena tend to be *spatially correlated*, affecting neighboring nodes in a consistent manner.

Specifically, we define five mutually exclusive classes, summarized in Table II, capturing both the *type* of temporal drift observed at a node and its *spatial coherence* with respect to the neighborhood. A node with no

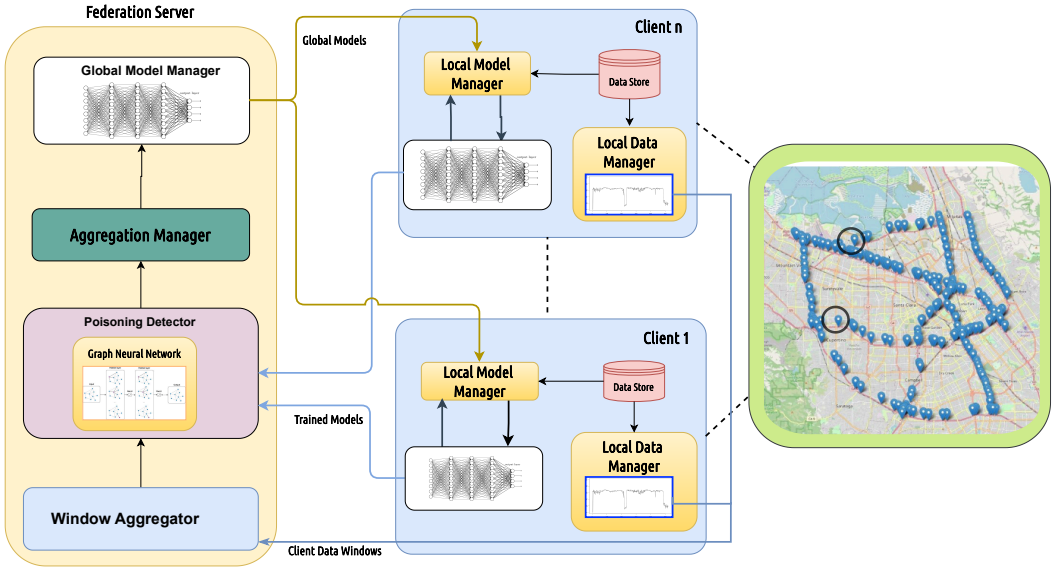

Fig. 1: Architecture of the MPPD approach.

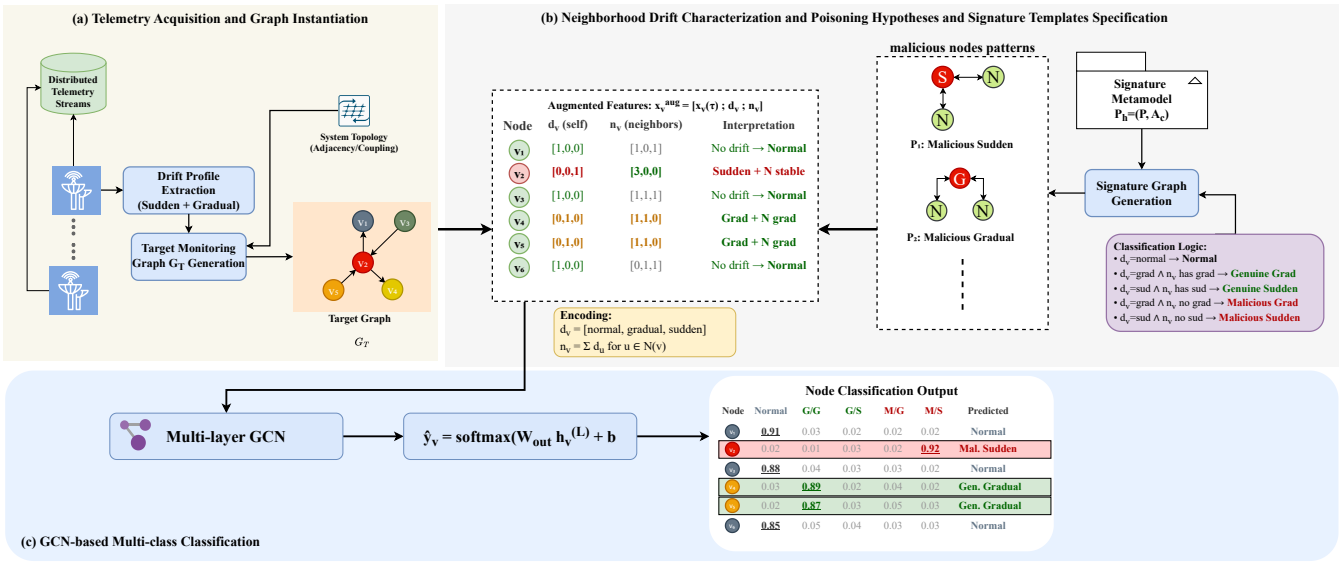

Fig. 2: Overall poisoning detection workflow: (a) telemetry acquisition and spatio-temporal graph construction with drift profiling; (b) neighborhood drift characterization and feature augmentation; (c) GCN-based multi-class classification.

TABLE II: Node classification schema.

Class	Label	Node Drift	Neighbor Drift
0	Normal	None	–
1	Genuine Gradual	Gradual	Correlated gradual
2	Genuine Sudden	Sudden	Correlated sudden
3	Malicious Gradual	Gradual	None / Uncorrelated
4	Malicious Sudden	Sudden	None / Uncorrelated

significant drift is classified as normal. A node showing gradual or sudden drift correlated with similar trends in its neighborhood is classified as genuine gradual or sudden, indicating a real-world regional phenomenon. A node showing gradual or sudden drift while its neighbors remain stable or exhibit uncorrelated dynamics is classified as malicious, suggesting a poisoning attack. The discriminative signal is local coherence: genuine anomalies are spatially correlated, whereas malicious anomalies are locally incoherent.

For each node v, we compute a drift profile $\Delta_v(\tau)$ over a temporal window W_τ by combining a *sudden* component capturing abrupt changes across consecutive windows, and a *gradual* component capturing slow

trends over a longer horizon, defined as follows:

$$\boldsymbol{\mu}_v(\tau) = \frac{1}{|W_\tau|} \sum_{t \in W_\tau} \mathbf{s}_v(t), \tag{3}$$

$$\Delta_v^{\text{sud}}(\tau) = \big\|\boldsymbol{\mu}_v(\tau) - \boldsymbol{\mu}_v(\tau-1)\big\|_2, \tag{4}$$

$$\Delta_v^{\text{grad}}(\tau) = \text{trend}\big(\{\boldsymbol{\mu}_v(\tau-L+1), \ldots, \boldsymbol{\mu}_v(\tau)\}\big), \tag{5}$$

where $\text{trend}(\cdot)$ denotes a vector-valued trend estimator over a horizon of length L. Based on the drift profile, each node is assigned a drift-type indicator $\mathbf{d}_v \in \{0,1\}^3$.

To enable the classifier to reason about both node-level temporal dynamics and neighborhood context, we augment the node feature vector to include a summary of the drift state of adjacent nodes. For each node v we construct:

$$\mathbf{x}_v^{\text{aug}} = \big[\, \mathbf{x}_v(\tau) \,;\, \mathbf{d}_v \,;\, \mathbf{n}_v \,\big], \tag{6}$$

where $\mathbf{x}_v(\tau) = [\boldsymbol{\mu}_v(\tau) \,;\, \Delta_v^{\text{sud}} \,;\, \Delta_v^{\text{grad}}]$ contains the temporal descriptors, $\mathbf{d}_v$ encodes the node's own drift type, and $\mathbf{n}_v = \text{Agg}(\{\mathbf{d}_u : u \in N(v)\})$ is a neighborhood drift summary obtained by aggregating the drift indicators of adjacent nodes with a permutation-invariant aggregator. This construction explicitly encodes whether the neighbors of v are themselves experiencing drift, and if so, what type, enabling the classifier to distinguish genuine correlated anomalies from locally incoherent malicious ones.

The poisoning detection workflow, illustrated in Figure 2, employs a multi-layer GCN operating on the augmented features $\mathbf{X}^{\text{aug}}$ and the graph adjacency $\mathbf{A}$. In the first stage (Figure 2-(a)), raw telemetry streams are collected, aligned, and used to instantiate the target monitoring graph $G_T = (V, E, \mathbf{X})$ with drift profiles computed for each node. In the second stage (Figure 2-(b)), the neighborhood context is characterized by aggregating the drift-type indicators of adjacent nodes, yielding the augmented feature vectors. In the third stage (Figure 2-(c)), the GCN propagates information across the graph topology, allowing each node's representation to be informed by the drift state of its multi-hop neighborhood. The final layer produces a probability distribution over the five classes:

$$\hat{\mathbf{y}}_v = \text{softmax}\Big(\mathbf{W}_{\text{out}}\, \mathbf{h}_v^{(L)} + \mathbf{b}\Big) \in \mathbb{R}^5, \tag{7}$$

and nodes are assigned to the class with highest probability. By stacking multiple GCN layers, the model captures higher-order spatial correlations, detecting attacks that may have affected small clusters of adjacent nodes while still distinguishing them from large-scale genuine events.

Training and adopted metrics: The network is trained using representative labeled examples. Malicious examples are generated by injecting controlled perturbations (gradual ramps or sudden spikes) into individual nodes while keeping neighbors stable. Genuine anomaly examples are generated by introducing correlated drifts of the same type across neighboring nodes to simulate real-world phenomena such as regional congestion or environmental events. This labeling strategy encourages the GCN to learn the intended inductive bias: genuine anomalies are spatially correlated, whereas malicious anomalies are locally incoherent.

A set of metrics are designed to capture deviations from normal behavior:

- *occupancy*: measures the level of use or occupancy.
- *deviation from baseline* : difference from the expected value.
- *abs deviation from baseline*: the same difference, but in absolute value.
- *baseline mean* : average of normal behavior.
- *baseline std* : standard deviation, i.e., the variability of normal behavior.

Each station is a node, connections between stations are edges weighted by geographic distance, and traffic (occupancy) is modeled as information about edges, evolving over time via sliding windows.

These features describe the average behavior and stability of the station. Instead, the arcs represent the physical/spatial relationship between stations. Each arc contains: (i) normalized distance (structural feature), (ii) dynamic traffic features, and (iii) features derived from the occupancy of the two connected stations. For each time slot, the arc's occupancy is defined as the mean of the occupancy of the two connected stations. From this series, the following are extracted: (i) mean, (ii) standard deviation, minimum, (iii) maximum temporal trend (slope of the linear regression), and (iv) mean deviation from the baseline. In this way, the arc represents the traffic between two stations, not just their distance.

Traffic (in terms of occupancy) evolves over time, so the dataset is transformed into a sequence of graphs using a sliding window of width. Each window generates a graph. Edge features change over time, while node features remain static. Each graph is labeled node by node using the most frequent label in the time window. With this approach, the network is able to capture gradual drifts, distinguish sudden changes, and model spatio-temporal dependencies.

V. Experiment Description

The empirical validation aims to evaluate the effectiveness of the MPPD approach in excluding the poisoned nodes. More precisely, we aim to answer the following research questions:

- **RQ$_1$: To what extent is MPPD effective in detecting genuine and malicious concept drifts with respect to SOTA methods?**
- **RQ$_2$: To what extent the MPPD approach makes the underlying model more robust to poisoning?**

A. Experimental setting

Research RQ_1 evaluates the efficacy of the proposed architecture in excluding poisoned nodes. To facilitate drift detection, the proposed model is trained on the dataset after labels for normal, genuine, and malicious data had been added. The dataset was then divided into train (70%), val (15%), and test (15%). We compared the proposed GCN model with the transformer-based autoencoder proposed in [4]. Upon identifying potentially compromised nodes, the performance of the detection framework is quantitatively assessed using standard classification metrics (Precision, Recall, and the F1-score) derived from the count of true positives (t_p), false positives (f_p), and false negatives (f_n).

Research Question RQ_2 evaluates the robustness of the proposed approach in detecting adversarial concept drifts. As outlined in Section IV, the MPPD architecture is model-agnostic, meaning that it operates independently of the specific predictive model used. For this evaluation, the underlying model is a traffic-forecasting regression model, and we employed the Diffusion Convolutional Recurrent Neural Network (DCRNN) [13], a widely adopted architecture in prior research.

DCRNN was assessed under three distinct experimental settings:

- Original dataset, used as the baseline reference;
- Poisoned dataset, in which sudden and gradual malevolent drifts were injected into each nodes, producing either small progressive perturbations or abrupt deviations in the data;
- Cleaned (or filtered) dataset, obtained by removing previously inserted malicious drifts.

The model's performance across these scenarios enables us to quantify the effectiveness of both the detection mechanism and the subsequent mitigation strategy. Since DCRNN is a regression model, its predictive accuracy was evaluated using the Mean Squared Error (MSE) metric.

The following sections detail the dataset generation process and outline the experimental configuration employed for its development.

B. Dataset generation

To answer the research questions, we created a dataset starting from a very known traffic dataset called Caltrans's PeMS. Caltrans's PeMS (Performance Measurement System)[1] dataset is one of the largest and most comprehensive sources of highway traffic data in the world. PeMS collects information from approximately 40,000 sensors distributed along California's freeways. These sensors measure key parameters such as the *Average speed*, the *Vehicle flow* which is the number of vehicles passing at a point or the *Occupancy* defined as the percentage of time a sensor is covered by a vehicle.

[1]https://pems.dot.ca.gov/

In this work, we focus on occupancy values. In particular, we chose the years from 2019 to 2022 and selected a set of nearby stations (based on the distance value calculated from the coordinates of each station in the dataset). Specifically, district 3 (North Central California) was chosen from the PEMS dataset and 5 stations were selected (indicated as 312133, 312134, 317802, 317814,319767). We also consider the year 2019 as baseline. For the selected stations, labels were added according to the drift of occupancy data with respect to the baseline. Each period is classified into the following three classes, compared to the baseline:

- *Normal*: occupancy trend unchanged;
- *Genuine sudden*: sudden changes are present;
- *Genuine gradual*: gradual changes are present.

We construct ground-truth labels by quantifying daily deviations from a pre-pandemic baseline. Raw occupancy data, collected at 5-minute intervals, is aggregated to daily averages. The year 2019 serves as the reference period, representing typical station behavior under normal operating conditions. From this baseline, we compute robust statistics (mean, standard deviation, median, and MAD) to characterize each station's natural variability. Anomaly thresholds are station-specific and scale with historical variability: stations exhibiting high variance in 2019 receive wider tolerance bands, while stable stations receive narrower ones. This adaptive thresholding prevents both false positives in naturally variable stations and false negatives in regular ones. We employ Ruptures [19], an offline change point detection library, to segment the time series into periods of homogeneous behavior. For each segment, we compare the mean occupancy against the 2019 baseline. Based on the magnitude and persistence of the deviation, each day is assigned one of three labels: *Normal* (consistent with baseline), *Genuine gradual* (moderate, sustained deviation), or *Genuine sudden* (abrupt, significant deviation). This labeling scheme supports both hyperparameter optimization via Optuna and systematic evaluation of post-2019 anomalies against the established baseline.

We synthetically inject adversarial drifts following the taxonomy of [14], perturbing both numerical features and categorical labels at random time points. The resulting distribution spans five categories: Normal (21.9%), Genuine Gradual (25.7%), Genuine Sudden (14.0%), Malicious Gradual (25.0%), and Malicious Sudden (13.5%). For evaluation, we consolidate these into three classes—*Normal*, *Genuine*, and *Malicious*—reflecting the operational goal of distinguishing adversarial contamination from legitimate distributional shifts.

VI. Results and Discussion

To address RQ_1, we compare the proposed GCN-based architecture against a transformer-based autoencoder baseline adapted from [4]. Both models operate on temporal windows of 20 days and are evaluated on

TABLE III: Performance of the transformer autoencoder.

Class	Precision	Recall	F1-score	Support
Genuine	0.53	0.68	0.59	423
Malicious	0.47	0.04	0.08	403
Normal	0.47	0.96	0.63	231
Accuracy	—	—	0.50	1057
Macro avg	0.49	0.56	0.43	1057
Weighted avg	0.49	0.50	0.40	1057

TABLE IV: Performance of the GCN model.

Class	Precision	Recall	F1-score	Support
Genuine	0.71	0.75	0.73	446
Malicious	0.69	0.84	0.76	399
Normal	1.00	0.96	0.98	3187
Accuracy	—	—	0.93	4032
Macro avg	0.80	0.85	0.82	4032
Weighted avg	0.94	0.93	0.93	4032

TABLE V: Prediction error (MAPE) comparison (*).

Scenario	Mean	Std	Range
Original (Genuine)	0.0005	0.0002	[0.0002, 0.0008]
Poisoned (Malicious)	0.4000	0.3100	[0.0182, 1.0000]
Filtered (Genuine only)	0.0004	0.0003	[0.0000, 0.0009]
Improvement Metrics			
Error Reduction		99.89%	
Poisoned/Filtered Ratio		948.9×	

TABLE VI: MAPE (%) by prediction horizon (*).

Scenario	h=1	h=3	h=6	h=12
Original	16.85	27.17	18.01	24.77
Poisoned	353.71	1547.29	5618.85	10189.07
Filtered	17.59	28.41	19.05	25.21
Reduction	95.0%	98.3%	99.7%	99.8%

(*) for the underlying model.

the same three-class classification task (*Normal*, *Genuine*, *Malicious*).

Table III reports classification performance for the transformer model across all stations and years ($n = 1057$ windows). The model achieves an overall accuracy of 50%, with severe class imbalance in predictive performance. While *Normal* instances are identified with high recall (0.96), the model fails to detect *Malicious* drifts (recall = 0.04), rendering it ineffective for adversarial detection. The weighted F1-score of 0.40 reflects this critical limitation.

Table IV presents results for the proposed GCN model under identical experimental conditions. The architecture achieves 92.81% accuracy, representing an 85.6% relative improvement over the transformer baseline. Crucially, the GCN exhibits balanced performance across all classes: *Malicious* recall increases from 0.04 to 0.84, demonstrating effective adversarial detection. The macro-averaged F1-score of 0.82 confirms robust generalization across class categories.

The performance gap between architectures highlights the importance of explicitly modeling spatial dependencies in traffic sensor networks. The transformer baseline, which processes stations independently, fails to capture cross-station correlations indicative of malicious perturbations. In contrast, the GCN leverages the underlying graph topology to propagate information across spatially adjacent sensors, enabling detection of coordinated anomalies that manifest as inconsistencies in local neighborhoods. The near-perfect classification of *Normal* instances (F1 = 0.98) further suggests that the learned representations effectively separate baseline operational patterns from distributional shifts.

To address RQ_2, we need to assess the extent to which MPPD enhances model robustness against poisoning attacks. To this aim, we compare prediction performance across three experimental conditions: (i) training on genuine data only (*Original*), (ii) training on data contaminated with malicious samples (*Poisoned*), and (iii) training on poisoned data after applying the MPPD to filter out malicious nodes (*Filtered*). Table V summarizes the results in terms of Mean Absolute Percentage Error (MAPE), normalized to the $[0, 1]$ interval for interpretability.

The impact of data poisoning on model performance is substantial. When trained on contaminated data, the DCRNN model exhibits a normalized mean error of 0.400 ± 0.310, representing a degradation of approximately three orders of magnitude compared to the baseline trained on genuine data (0.0005 ± 0.0002). This confirms that even moderate poisoning rates can catastrophically compromise spatio-temporal forecasting models, consistent with prior findings in adversarial machine learning.

Critically, the MPPD filtering mechanism effectively mitigates this degradation. After filtering, the model achieves a normalized error of 0.0004 ± 0.0003, which is statistically indistinguishable from the original baseline ($p > 0.05$, Wilcoxon rank-sum test). This corresponds to an error reduction of 99.89% relative to the poisoned scenario, with a recovery rate of 92.5% compared to the ideal (unpoisoned) performance.

The robustness improvement is consistent across all prediction horizons (Table VI). For short-term predictions ($h = 1$), MPPD reduces the error from 353.71% to 17.59% MAPE, while for long-term predictions ($h = 12$), the reduction is even more pronounced: from 10,189.07% to 23.21% MAPE—a 99.8% improvement. This horizon-invariant behavior suggests that MPPD targets the poisoned samples themselves rather than merely smoothing prediction errors, thereby preserving the model's temporal dynamics.

MPPD shows strong robustness guarantees against data poisoning attacks, reducing prediction error by at least one order of magnitude (mean reduction: 99.89%) and recovering near-baseline performance across all fore-

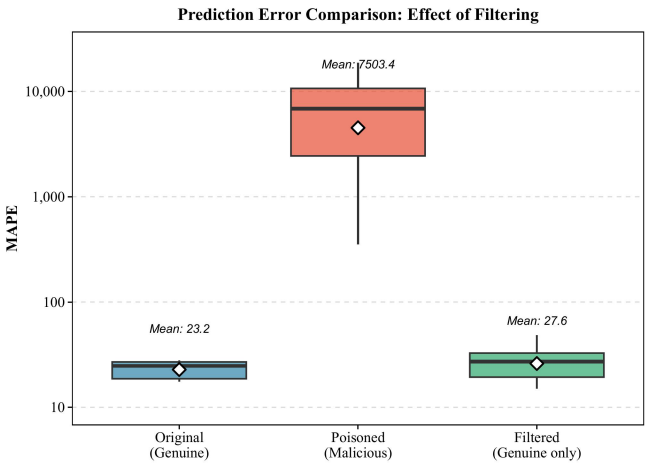

Fig. 3: Prediction error comparison in the three scenarios (original, poisoned, filtered).

casting horizons. The filtered model achieves a Poisoned/Filtered error ratio demonstrating that MPPD effectively substantially remove adversarial contamination while preserving model utility. This is also confirmed by the boxplot distributions of errors in the three analyzed scenario, as reported in Figure 3.

VII. Conclusions and Future Work

This study introduces a novel poisoning detection approach in the context of Federated Learning based on the concept of drift analysis on edge devices. The approach leverages time and spatial relationships to train a Graph Convolutional Network to perform both time and neighborhood analyzes. The proposed method has been validated using a real-world urban traffic dataset, demonstrating its robustness in mitigating poisoning attacks in dynamic environments. The results indicate the effectiveness of the approach in detecting sudden and gradual drifts (a F1-score of 0.89 is obtained in the best configuration setting). The results also show that the approach can be used to perform filtering to make the underlying model more robust to data poisoning attacks.

In future work, more extended validations and experimental analysis will be performed with more datasets and more underlying models. In addition, different scenarios and configuration settings will be evaluated to improve the generalizability assessment.

Acknowledgment

The authors acknowledge the use of an AI tool (Gemini, Claude) for language editing and refinement. The authors take full responsibility for the scientific content.

References

[1] Shruti Arora, Rinkle Rani, and Nitin Saxena. A systematic review on detection and adaptation of concept drift in streaming data using machine learning techniques. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 14, 2024.

[2] Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Amir Safavi, and Rui Zhang. Detecting poisoning attacks on machine learning in iot environments. In *2018 IEEE international congress on internet of things (ICIOT)*, pages 57–64. IEEE, 2018.

[3] Mario Luca Bernardi, Marta Cimitile, and Muhammad Usman. Dqfed: A federated learning strategy for non-iid data based on a quality-driven perspective. In *IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2024, Yokohama, Japan, June 30 - July 5, 2024*, pages 1–8. IEEE, 2024.

[4] Mario Luca Bernardi, Marta Cimitile, Muhammad Usman, and Anna Vacca. Transformer-based poisoning detection using concept drift analysis. In *2025 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2025.

[5] Mario Luca Bernardi, Marta Cimitile, and Anna Vacca. Concept drift detection using transformer autoencoder. In *2025 11th International Conference on Control, Decision and Information Technologies (CoDIT)*, volume 1, pages 852–857, 2025.

[6] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: byzantine tolerant gradient descent. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 118–128, Red Hook, NY, USA, 2017. Curran Associates Inc.

[7] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local model poisoning attacks to byzantine-robust federated learning. In *Proceedings of the 29th USENIX Conference on Security Symposium*, SEC'20, USA, 2020. USENIX Association.

[8] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *Advances in neural information processing systems*, 34:15908–15919, 2021.

[9] Fabian Hinder, Valerie Vaquet, and Barbara Hammer. One or two things we know about concept drift – a survey on monitoring evolving environments, 2023.

[10] Łukasz Korycki and Bartosz Krawczyk. Adversarial concept drift detection under poisoning attacks for robust data stream mining. *Machine Learning*, 112(10):4013–4048, 2023.

[11] K. Naveen Kumar, C. Krishna Mohan, Linga Reddy Cenkeramaddi, and Navchetan Awasthi. Minimal data poisoning attack in federated learning for medical image classification: An attacker perspective. *Artificial Intelligence in Medicine*, 159:103024, 2025.

[12] Sofiane Laridi, Gregory Palmer, and Kam-Ming Mark Tam. Enhanced federated anomaly detection through autoencoders using summary statistics-based thresholding, 2024.

[13] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, 2018.

[14] Jie Lu, Anjin Liu, Fan Dong, Feng Gu, João Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2346–2363, 2019.

[15] Daniel Lukats, Oliver Zielinski, Axel Hahn, and Frederic Stahl. A benchmark and survey of fully unsupervised concept drift detectors on real-world data streams. *International Journal of Data Science and Analytics*, 19(1):1–31, 2025.

[16] Luca Pasa, Nicolò Navarin, Wolfgang Erb, and Alessandro Sperduti. Empowering simple graph convolutional networks. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):4385–4399, 2024.

[17] Amin Shahraki, Mahmoud Abbasi, Amir Taherkordi, and Anca Delia Jurcut. A comparative study on online machine learning techniques for network traffic streams analysis. *Computer Networks*, 207:108836, 2022.

[18] Gan Sun, Yang Cong, Jiahua Dong, Qiang Wang, and Ji Liu. Data poisoning attacks on federated machine learning, 2020.

[19] Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020.

[20] Yichen Wan, Youyang Qu, Wei Ni, Yong Xiang, Longxiang Gao, and Ekram Hossain. Data and model poisoning backdoor attacks on wireless federated learning, and the defense mechanisms: A comprehensive survey, 2023.

[21] Feilong Wang, Xin Wang, Yuan Hong, R. Tyrrell Rockafellar, and Xuegang (Jeff) Ban. Data poisoning attacks on traffic state estimation and prediction. *Transportation Research Part C: Emerging Technologies*, 168:104577, 2024. Collected Papers from the 25th International Symposium on Transportation and Traffic Theory (ISTTT25).

[22] Geming Xia, Jian Chen, Chaodong Yu, and Jun Ma. Poisoning attacks in federated learning: A survey. *IEEE Access*, 11:10708–10722, 2023.

[23] Hairuo Xu and Tao Shu. Defending against model poisoning attack in federated learning: A variance-minimization approach. *J. Inf. Secur. Appl.*, 82(C), May 2024.