Building Real-time Awareness of Out-of-distribution in Trajectory Prediction for Autonomous Vehicles

Tongfei Guo[1], **Taposh Banerjee**[2], **Lili Su**[1]

[1]Northeastern University, Boston, Massachusetts, USA
[2]University of Pittsburgh, Pittsburgh, USA
guo.t@northeastern.edu, taposh.banerjee@pitt.edu, l.su@northeastern.edu

Abstract

Ensuring trust and control in autonomous systems requires that they remain robust and reliable when facing the unpredictable complexity of the real world. In the context of agentic AI for autonomous driving, accurate trajectory prediction is foundational to safety-critical decision-making. However, due to discrepancies between training data and real-world conditions encountered during inference, even well-trained machine learning models may produce unreliable predictions. Such sim-to-real gaps (also known as imperfect training data) may be unavoidable due to the overwhelming complexity of data annotation and environment uncertainties. To support verifiable and trustworthy autonomy, we present a principled and computationally efficient framework for detecting when a model's predictions deviate from expected, in-distribution behavior. Leveraging the intuition that in-distribution (ID) scenes exhibit error patterns similar to training data, while out-of-distribution (OOD) scenes do not, we formulate OOD detection as a quickest change-point detection problem, enabling timely recognition of subtle or deceptive shifts in driving scenes that may compromise reliability. We address the challenging settings where the OOD scenes are deceptive, meaning that they are not easily detectable by human intuitions. Our solutions can handle the occurrence of OOD at any time during trajectory prediction inference. Experimental results on multiple real-world datasets demonstrate the effectiveness of our methods.

1 Introduction

AI technologies are the backbone of modern autonomous vehicles (AVs), and are reshaping transportation systems. Accurate trajectory prediction is essential for the safe operation of autonomous vehicles in real-world environments. However, even well-trained machine learning (ML) models may produce unreliable predictions due to discrepancies between training data and real-world conditions encountered during inference (Bahari et al. 2022). Specifically, many public datasets overrepresent certain driving scenes (e.g. straight lanes and high-ways) while significantly underrepresenting others (e.g. complex traffic circles or intersections), as illustrated in Figure 2. Moreover, real-world environments suffer a wide range of uncertainties such as sudden braking by nearby vehicles, large objects falling from other vehicles, or other deceptive yet life-threatening distribution shifts (Filos et al. 2020; Tang et al. 2020; Zhang et al. 2022). As a consequence, an autonomous vehicle may unexpectedly run into driving scenarios that are poorly represented by the training data, which we refer to as *out-of-distribution (OOD) data*. To ensure safety, it is critical to detect in real-time when a ML model's trajectory predictions become unreliable, allowing control to be seamlessly transferred back to human drivers.

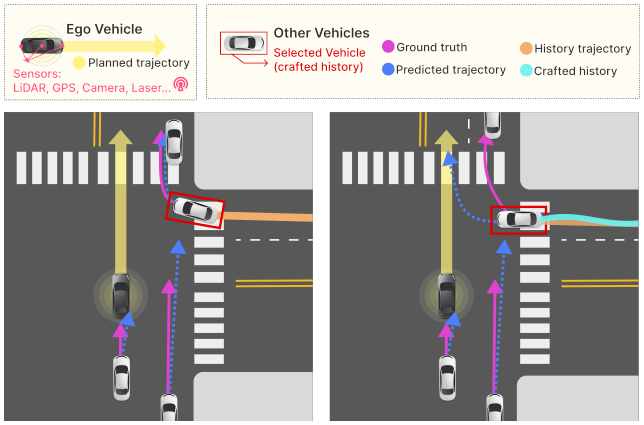


Figure 1: Illustration of the performance comparison of a given ML trajectory prediction model on ID and OOD scenes. In an ID scenario (left figure), the ego vehicle (gray car), on which the ML model is implemented, can accurately predict the trajectories of neighboring vehicles. In an OOD scenario (right figure), the target vehicle slightly deviates from its usual driving path due to unexpected debris from the vehicle ahead. As a result, the ego vehicle may mispredict the target vehicle's trajectory, mistakenly assuming it will move into the same lane. In response, the ego vehicle may suddenly brake, potentially causing a rear-end collision.

Prior works enhance trajectory prediction reliability through uncertainty quantification (UQ) (Neumeier et al. 2024; Pustynnikov and Eremeev 2021), including both parametric strategies (with specific probabilistic assumptions) and non-parametric ones (e.g., entropy-based measures). While these methods improve predictive confidence,

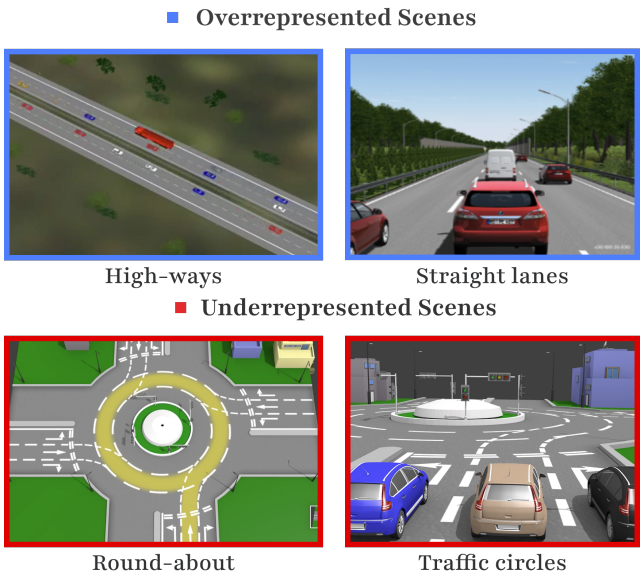


Figure 2: Illustration of overrepresent and underrepresent scenes for the deployed ML models. Images from PTV Vissim simulation.

they suffer from restrictive priors, computational inefficiency, and the assumption that training data fully represent real-world scenarios, limiting generalization to unseen cases. Beyond UQ-based reliability estimation, another related line of work focuses on identifying OOD inputs. Existing OOD detection for AVs largely targets vision tasks (e.g., object detection, segmentation) using scoring functions (MSP (Hendrycks and Gimpel 2016), energy scores (Liu et al. 2020)) or latent-space density estimation (Lee et al. 2018; Sun et al. 2022a). However, such approaches overlook temporal dependencies in sequential trajectory data, reducing their ability to capture gradual deviations, and often prioritize recognition accuracy over efficiency and scalability (Cui and Wang 2022).

Observing these limitations, we adopt a sequential, error-driven perspective. We hypothesize that *in-distribution (ID)* driving scenes yield prediction error distributions consistent with training data, while OOD scenes deviate from them. Hence, we monitor the sequence of prediction errors and formulate trajectory-level OOD awareness as a quickest change-point detection (QCD) problem. Leveraging sequential analysis, we design lightweight and principled OOD detectors with formal guarantees on the trade-off between detection delay and false alarms.

Contributions We are the first to apply QCD methods for OOD detection in trajectory prediction across multiple real-world datasets. Our approach monitors only a scalar error variable, handles OOD occurrence at any inference step, and remains computationally efficient.

- We observe that pre- and post-change trajectory prediction errors are well-modeled as Gaussian Mixture Models, forming the basis of our OOD detection framework.
- We adapt the CUSUM algorithm for OOD detection, achieving faster and more reliable detection with minimal false alarms compared to classical sequential and leading OOD methods.
- We test robustness across varying levels of distribution knowledge (complete, partial, unknown), showing our method remains the most effective in challenging scenarios.

2 Related Work

Trajectory Prediction Safe autonomous driving requires forecasting the future motion of nearby agents. Trajectory prediction methods are commonly grouped into physics-based, classical learning, deep learning, and reinforcement learning (RL) approaches. Physics-based models (Anderson, Vasudevan, and Johnson-Roberson 2021) are efficient but oversimplify interactions. Classical approaches, including Gaussian Processes and Hidden Markov Models (Deo, Rangesh, and Trivedi 2018), extend prediction horizons yet rely on predefined maneuvers. Deep models (Kuefler et al. 2017) fuse map and interaction cues for high accuracy but depend on large datasets and face real-time trade-offs. RL-based imitation (Kuefler et al. 2017) enhances human-like decisions but remains computationally demanding. Since models often assume complete training coverage (Fang et al. 2022), their reliability degrades under distributional shifts, motivating robustness-aware prediction.

Uncertainty Quantification for Trajectory Prediction Prediction uncertainty arises from the stochastic nature of traffic dynamics. Uncertainty quantification (UQ) enables models to assess confidence via probabilistic or sampling-based estimation (Neumeier et al. 2024). Parametric methods assume specific distributions (e.g., Gaussian), whereas non-parametric ones rely on entropy or latent variance metrics (Pustynnikov and Eremeev 2021). Recent studies employ latent-space modeling or entropy-based metrics (Wiederer et al. 2023), improving calibration but still lacking generalization. Moreover, few efforts explore trajectory-level OOD detection, which we address through a principled, QCD-based formulation offering theoretical guarantees.

OOD Detection in Autonomous Vehicles OOD detection ensures model reliability when encountering unseen driving scenarios (Li et al. 2022). While prior work mainly targets visual perception via confidence or density scores (Hendrycks and Gimpel 2016; Lee et al. 2018), such frame-wise techniques neglect temporal dependencies crucial for trajectories. Combined UQ–OOD frameworks (Wiederer et al. 2023) improve robustness but are computationally costly. To overcome this, we adopt a temporal, model-agnostic framework that aggregates prediction errors over time for efficient online detection.

Change-point Detection Change-point detection identifies distributional shifts in time series (Tartakovsky, Polunchenko, and Sokolov 2012). The Quickest Change Detection (QCD) paradigm seeks minimal-delay detection under false-alarm constraints (Veeravalli and Banerjee 2014).

Among its formulations, the CUSUM algorithm (Page 1954; Moustakides 1986) offers asymptotic optimality by accumulating deviations from a baseline until a threshold is exceeded. Due to its efficiency and sequential nature, CUSUM aligns naturally with real-time OOD detection in trajectory prediction.

3 Problem Formulation

3.1 Trajectory Prediction

The training dataset $\mathcal{D} = \{S_j\}_{j=1}^N$ is a collection of driving scenes. Each S_j described by a triple $S_j = (\mathcal{X}_j, \mathcal{Y}_j, \mathcal{M})$, where $\mathcal{X}_j$ is the collections of observed trajectories, $\mathcal{Y}_j$ is the collection of future trajectories that we aim to predict based on $\mathcal{X}_j$, [1] and $\mathcal{M}$ is the map information. When a dataset does not have map information, we set $\mathcal{M} = \emptyset$. In a driving scene, an agent represents a moving object such as a vehicle, a motorcycle, or a pedestrian. For a given driving scene S_j, let m_j denote the number of agents in the scene. Hence, $\mathcal{X}_j$ and $\mathcal{Y}_j$ can be explicitly written out as $\mathcal{X}_j = \{x_j^1, \cdots, x_j^{m_j}\}$ and $\mathcal{Y}_j = \{y_j^1, \cdots, y_j^{m_j}\}$, where $x_j^i \in \mathbb{R}^{2\times L_O}$ and $y_j^i \in \mathbb{R}^{2\times L_P}$ for each agent i in the scene. Here, L_O and L_P are the numbers of time frames in the observed trajectory x_j^i and in the future trajectory y_j^i, respectively. It is worth noting that:
(1) Different from traditional supervised learning problem, in the trajectory prediction problem, the trajectory of an agent is a sequence of positions, resulting in possibly often overlapping $\mathcal{X}_j$ and $\mathcal{Y}_j$. (2) Different driving scenes may contain different numbers of agents as can be seen in Figure 2.

A machine-learning model f for the trajectory prediction task can be trained by minimizing the objective function that may take the form

$$\min_f \quad \frac{1}{N}\sum_{j=1}^{N} \ell\left(f(\mathcal{X}_j), \mathcal{Y}_j\right) + \lambda \times \text{regularization},$$

where ℓ is a loss function such as the negative log-likelihood (NLL) of the Laplace/Gaussian distributions (Peng et al. 2023; Tang et al. 2024), and $\lambda \geq 0$ is the regularization coefficient. One popular choice of regularization in the objective is cross entropy loss.

The specific training methods are out of the scope of this paper. Readers can find concrete instances of the objective functions in (Peng et al. 2023; Tang et al. 2024) and others. Notably, different from traditional ML tasks, where the model trained has fixed input and output dimensions, the trajectory prediction model f may have varying input and output dimensions.

Metrics of Prediction Errors Let $\hat{f}$ be a given trajectory prediction model. We follow existing literature on trajectory prediction model training to evaluate the generalization performance of $\hat{f}$ (Wang et al. 2019; Dendorfer, Elflein, and Leal-Taixé 2021). Let $\mathcal{D}_{\text{test}} = \{\mathcal{S}_j\}_{j=1}^{N_{\text{test}}}$ be the test dataset. Instead of evaluating the trajectory prediction errors of all neighboring agents, the prediction errors of an arbitrarily chosen agent are evaluated. Such chosen agent is referred to as *target agent* of scene $\mathcal{S}_j$, denoted as a_j. We use three common metrics in trajectory prediction: *Average Displacement Error* (ADE) and *Final Displacement Error* (FDE) and *Root Mean Squared Error* (RMSE). These metrics are widely adopted in prior works (Kim et al. 2020; Deo and Trivedi 2018; Ivanovic and Pavone 2022; Liu et al. 2024). The mathematical formulations are as follows:

- ADE $:= \frac{1}{N_{\text{test}}}\sum_{j=1}^{N_{\text{test}}} \frac{1}{L_P}\|\hat{f}(x_j^{t_j}) - y_j^{t_j}\|_2$,
- FDE $:= \frac{1}{N_{\text{test}}}\sum_{j=1}^{N_{\text{test}}} \|[\hat{f}(x_j^{t_j})]_{L_P} - [y_j^{t_j}]_{L_P}\|_2$, where $[\hat{f}(x_j^{t_j})]_{L_P}$ and $[y_j^{t_j}]_{L_P}$ represent the last positions in the predicted and ground-truth trajectories, respectively.
- RMSE $:= \frac{1}{N_{\text{test}}}\sum_{j=1}^{N_{\text{test}}} \sqrt{\frac{1}{L_P}\|\hat{f}(x_j^{t_j}) - y_j^{t_j}\|_2^2}$.

3.2 Casting OOD Awareness as a QCD Problem

In the context of trajectory prediction, we define OOD awareness as the ability to detect deviations in prediction performance over time. Specifically, we consider a sequence of prediction errors ϵ_t arising from a trajectory prediction model $\hat{f}$. The sequence of errors is defined as

$$\epsilon_t = d(\hat{f}(\mathcal{X}_t), \mathcal{Y}_t),$$

where $d(\cdot,\cdot)$ is a distance function measuring the discrepancy between the predicted future trajectory and the ground truth. As mentioned in 3.1, we consider three distance functions, i.e., ADE, FDE, and RMSE.

The AV observes the sequence $\{\epsilon_t\} = \{\epsilon_1, \epsilon_2, \ldots, \epsilon_t\}$ of the target agent (a) for every scene. Let the data distribution change at an unknown time γ. We denote the pre-change distribution of ϵ_t by $f_\phi(\epsilon_t)$ for $t < \gamma$ and the post-change distribution by $g_\theta(\epsilon_t)$ for $t \geq \gamma$, i.e.,

$$\epsilon_t \sim \begin{cases} f_\phi(\epsilon_t), & t < \gamma \text{ (ID Scene)}, \\ g_\theta(\epsilon_t), & t \geq \gamma \text{ (OOD Scene)}. \end{cases}$$

Let $\mathcal{A}$ be any algorithm. Let $\tau_{\mathcal{A}}$ be the time that a change is declared under algorithm $\mathcal{A}$. When the adopted $\mathcal{A}$ is clear from the context, we drop the subscript $\mathcal{A}$ in τ. We evaluate OOD detection algorithms using two metrics: (1) *Detection Delay*—time to detect an OOD event, reflecting responsiveness; and (2) *False Alarm Rate*—frequency of incorrect OOD flags, ensuring reliability. These metrics are widely used in (Lai, Fan, and Poor 2008; Lau, Tay, and Veeravalli 2018; Liang and Veeravalli 2024) due to their ability to capture the trade-off between sensitivity and robustness.

To ensure robust safety guarantees, we follow Lorden's minimax formulation (Lorden 1971), which quantifies the detection delay using the Worst-Case Average Detection Delay (WADD). The WADD is defined as

$$\text{WADD}(\tau) = \sup_{t\geq 1} \operatorname{ess\,sup} \mathbb{E}_t[(\tau - \gamma + 1)^+ \mid \epsilon_1, \ldots, \epsilon_{t-1}],$$

where $(\cdot)^+ = \max\{0, \cdot\}$, $\mathbb{E}_t[\cdot]$ denotes the expectation when the change occurs at time t, and $\operatorname{ess\,sup}(\cdot)$ refers to the essential supremum of a scalar random variable.

[1] Since trajectories are sequential data, $\mathcal{X}_j$ and $\mathcal{Y}_j$ may have overlaps.

Algorithm 1: CUSUM with Parameterized Models

Input: Threshold $b > 0$
Parameter: Learned parameters ϕ and θ
Output: Change point τ if detected

Initialize $W_0 \leftarrow 0$
Compute pre-change distribution $f_\phi(\epsilon_t)$
Compute post-change distribution $g_\theta(\epsilon_t)$
while true **do**
  Get a new observation of the prediction error ϵ_t
  Compute log-likelihood ratio: $\text{LLR}(\epsilon_t) = \log \frac{g_\theta(\epsilon_t)}{f_\phi(\epsilon_t)}$
  Update: $W_{t+1} \leftarrow \max(W_t + \text{LLR}(\epsilon_t), 0)$
  if $W_{t+1} \geq b$ **then**
    Declare a change point τ
    break
  end if
end while
return τ if detected

The false alarm rate follows the literature (Veeravalli and Banerjee 2014)

$$\text{FAR}(\tau) = \frac{1}{\mathbb{E}_\infty(\tau)},$$

where $\mathbb{E}_\infty$ denotes the expectation when the change never occurs.

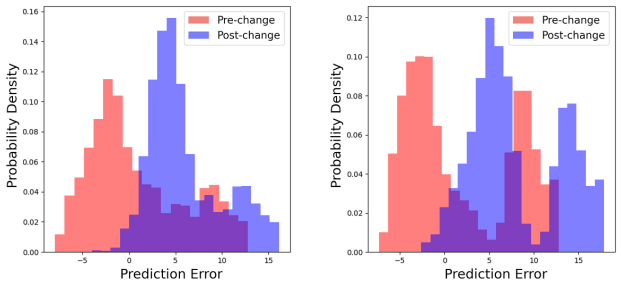
4 Methods

In this section, we present the key methods used for detecting OOD scenes, focusing on CUSUM as the primary algorithm, with Z-Score and Chi-Square tests serving as benchmark comparisons. CUSUM is a mature method known for its robust theoretical foundation and ability to quickly and reliably detect distributional shifts (Alippi and Roveri 2006; Manogaran and Lopez 2018; Sun et al. 2022b). In particular, CUSUM is designed to minimize detection delay while keeping the false alarm rate within acceptable bounds, providing formal performance guarantees in dynamic environments. As benchmarks, the Z-Score and Chi-Square methods are commonly used in literature for comparison (Cheadle et al. 2003; Mare, Moreira, and Rossi 2017; Wallis and Moore 1941; Chiang et al. 2024).

4.1 Background of CUSUM

Recall from Section 3.2 we use γ to denote this unknown change time and t_0 is the variable that indicates the current system time; its value increases by 1 as each time goes by. The problem of detecting OOD can be formulated as detecting a change in the distributions of the sequence of random prediction errors ϵ_t. As shown in Figure 3, we observe that on some popular datasets of trajectory prediction, the pre- and post-change distributions are Gaussian mixtures with two components. We consider three most popular scenarios when applying the CUSUM algorithm (Section 4.1, Section 4.1, and Section 4.1).

The CUSUM algorithm detects distributional shifts by monitoring cumulative deviations in the log-likelihood ratio



(a) GRIP++ (b) FQA

Figure 3: Illustration of the mixture Gaussian distributions for pre-change (red) and post-change (blue) error distributions in trajectory prediction, assessed using the ADE metric on the ApolloScape dataset. (a) Left: Results from the GRIP++ model. (b) Right: Results from the FQA model. Both examples highlight the observed shift in error distributions.

between pre-change (f_ϕ) and post-change (g_θ) distributions. The decision statistic W_t is computed iteratively

$$W_t = \begin{cases} 0, & \text{if } t = 0, \\ \max\{W_{t-1} + \log L(\epsilon_t), 0\}, & \text{if } t \geq 1, \end{cases} \quad (1)$$

where $L(\epsilon_t) = \frac{g_\theta(\epsilon_t)}{f_\phi(\epsilon_t)}$ is the likelihood ratio. A change is declared when W_t exceeds a threshold b

$$\tau = \inf\{t \geq 1 : W_t \geq b\}.$$

This method optimally balances the trade-off between false alarm rate $\text{FAR}(\tau < \gamma)$ and detection delay $\mathbb{E}[\tau - \gamma \mid \tau \geq \gamma]$. This method enables awareness of sequential decision reliability by continuously monitoring prediction errors. For implementation details, see Algorithm 1.

Theorem 4.1 ((Veeravalli and Banerjee 2014)). *For any $\alpha \in (0, 1)$, setting the threshold $b = |\log \alpha|$ ensures*

$$\textit{FAR}(\tau) \leq \alpha \quad \textit{and WADD}(\tau) = O\left(\frac{|\log \alpha|}{D_{KL}(g_\theta, f_\phi)}\right),$$

where $D_{KL}(g_\theta, f_\phi)$ is the Kullback-Leibler divergence between pre-change and post-change distributions.

Intuition After a change-point γ, the observations ϵ_t are distributed according to g_θ, and the expected log-likelihood ratio $\mathbb{E}_{t \geq \gamma}[\log(\frac{g_\theta(\epsilon_t)}{f_\phi(\epsilon_t)})]$ equals the Kullback-Leibler divergence $D_{KL}(g_\theta \parallel f_\phi)$, which is positive. Conversely, before γ, the observations are governed by f, and $\mathbb{E}_{t<\gamma}[\log(\frac{g_\theta(\epsilon_t)}{f_\phi(\epsilon_t)})] = -D_{KL}(f_\phi \parallel g_\theta)$, which is negative. Consequently, with an increasing number of observations, the cumulative sum $\sum_{t:t\geq\gamma} \log(\frac{g_\theta(\epsilon_t)}{f_\phi(\epsilon_t)})$ is expected to exceed the threshold b. The $\max\{\cdot, 0\}$ function in the statistic W_t reduces the impact of random fluctuations in the log-likelihood ratio and helps identify the change-point γ. The threshold b controls the sensitivity of the detection; increasing b makes the algorithm less prone to false alarms but also delays the detection of actual changes.

CUSUM with Complete Knowledge We assume that both the pre-change (f_ϕ) and post-change (g_θ) distributions follow Gaussian Mixture Models (GMMs). The probability density function for GMM can be defined as $p(\epsilon_t) = \sum_{i=1}^{K} w_i \cdot \mathcal{N}(\epsilon_t \mid \mu_i, \sigma_i^2)$, where K is the number of Gaussian components in the mixture, and w_i, μ_i, and σ_i^2, respectively, are the mixture weight, the mean, and the variance of component i. In our experimental results in Figure 3, $K = 2$.

CUSUM with Partial Knowledge In real-world, the exact parameters of the post-change distribution are often unknown. Instead, we leverage partial knowledge or side information to approximate the post-change distributions. In particular, we consider the case when the overall mean μ_g and variance σ_g^2 of the post-change distribution are known. We use $\mathcal{N}(\epsilon_t \,|\, \mu_g, \sigma_g^2)$ – a single Gaussian distribution with the given mean and variance – to replace the true post-change distribution g_θ in computing the likelihood ratio in Equation (1), i.e.,

$$L(\epsilon_t) = \frac{\hat{g}_\theta(\epsilon_t)}{f_\phi(\epsilon_t)}, \quad \text{where } \hat{g}_\theta(\epsilon_t) = \mathcal{N}(\epsilon_t \,|\, \mu_g, \sigma_g^2).$$

CUSUM with Unknown Knowledge Occasionally, the pre-change distribution may not be known, which may result from a lack of sufficient data to reconstruct the true distribution. We consider the case when the overall means (μ_f and μ_g) and variances (σ_f^2 and σ_g^2) of pre- and post- change distributions are known. We use single Gaussians to approximate the pre- and post-change distributions in computing the likelihood ratio in Equation (1), i.e.,

$$L(\epsilon_t) = \frac{\hat{g}_\theta(\epsilon_t)}{\hat{f}_\phi(\epsilon_t)},$$

where $\hat{f}_\theta(\epsilon_t) = \mathcal{N}(\epsilon_t \,|\, \mu_f, \sigma_f^2)$ and $\hat{g}_\theta(\epsilon_t) = \mathcal{N}(\epsilon_t \,|\, \mu_g, \sigma_g^2)$.

4.2 Benchmarks

Z-Score Time-series Detection The Z-Score method detects change points by measuring how much a data point deviates from a moving average. For each time point t, the Z-Score method calculates the moving average $\bar{\epsilon}_t$ over a defined window size w, as: $\bar{\epsilon}_t = \frac{1}{w}\sum_{r=t-w+1}^{t} \epsilon_r$, where ϵ_i represents the time points. Next, the moving standard deviation σ_t is computed as: $\sigma_t = \sqrt{\frac{1}{w}\sum_{r=t-w+1}^{t}(\epsilon_r - \bar{\epsilon}_t)^2}$. Using these, the Z score z_t for each data point ϵ_t at time t is then calculated as $z_t = \frac{\epsilon_t - \bar{\epsilon}_t}{\sigma_t}$.

A threshold is set for the Z-Score, often based on how sensitive the detection should be. When the absolute value of the Z-Score $|z_t|$ exceeds the threshold, it indicates that the data point ϵ_t is significantly deviating from the moving average. This declares a potential change point

$$\tau = \min\{t : |z_t| > b\}.$$

In summary, the Z-Score method detects change points by identifying data points whose deviations (e.g. variation, and standard deviation) exceed a pre-set threshold, indicating a shift in the underlying distribution.

Chi-Square Time-series Detection Chi-Square test commonly used to evaluate the independence of categorical variables and the fit between observed and expected frequencies. Specifically, we use Pearson's chi-square test to compare observed frequencies in the time-series with expected frequencies under normal behavior (Franke, Ho, and Christie 2012). Significant deviations from the expected values suggest potential anomalies. The test statistic is computed as follows: $\chi_t^2 = \sum_{r=t-w+1}^{t} \frac{(g_\theta(\epsilon_t) - f_\phi(\epsilon_t))^2}{f_\phi(\epsilon_t)}$, where w represents the number of time steps over which the test statistic is computed (i.e. window size). The null hypothesis assumes no significant difference between the frequencies. A large χ^2 value indicates an anomaly, and a change is declared when χ^2 exceeds a predefined threshold. For any given b, the change time τ is defined as

$$\tau = \min\{t : \chi_t^2 > b\}.$$

The Chi-Square test detects anomalies by comparing observed and expected frequencies in time-series data, declaring a change when the test statistic χ^2 exceeds a predefined threshold, signaling a shift in the underlying distribution.

5 Experiments and Results

Datasets Table 1 summarizes the characteristics of three datasets used. The ApolloScape (Huang et al. 2018), NGSIM (Federal Highway Administration 2020), and NuScenes (Caesar et al. 2020) datasets are widely used in trajectory prediction and autonomous vehicle research due to their diverse and complex real-world driving scenes. ApolloScape provides a rich multimodal dataset, including camera images, LiDAR point clouds, and approximately 50 minutes of manually annotated vehicle trajectories. NGSIM focuses on freeway traffic behavior, offering detailed vehicle trajectories recorded over 45 minutes on highways US-101 and I-80. NuScenes captures 1000 diverse driving scenes in Boston and Singapore, two cities known for their challenging traffic conditions. For trajectory prediction, we choose the history trajectory length (L_I) and future trajectory length (L_O) based on the recommendations provided by the dataset's authors. For each dataset, we randomly select 2500 scenes to serve as test cases.

Name	Scenario	Map	$\mathbf{L_I}$	$\mathbf{L_O}$
ApolloScape	Urban	×	6	6
NGSIM	Highway	×	15	25
NuScenes	Urban	✓	4	12

Table 1: Summary of datasets.

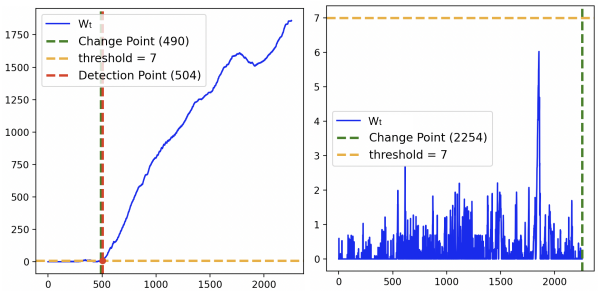
Name	Input features	Output format	Network
GRIP++	location + heading	single-prediction	Conv + GRU
FQA	location	single-prediction	LSTM

Table 2: Summary of models.

Trajectory Prediction Models Table 2 summarizes two benchmark models used for trajectory prediction. We select GRIP++ (Li, Ying, and Chuah 2019) and FQA(Kamra et al. 2020) as our trajectory prediction models due to their proven effectiveness on widely used datasets like ApolloScape and NGSIM from prior research. GRIP++ employs a graph-based structure with a two-layer GRU network, offering fast and accurate short- and long-term predictions, ranking #1 in the 2019 ApolloScape competition, making it an ideal fit for our experiments by minimizing model noise and enhancing algorithm performance evaluation. FQA introduces a fuzzy attention mechanism to model dynamic agent interactions, showing strong performance across diverse domains, including traffic and human motion which is well-suited for evaluating prediction performance in nuScenes dataset.

Model	Dataset	ADE (m)	FDE (m)	RMSE (m)
GRIP++	ApolloScape	1.68 / 4.15	2.11 / 8.73	6.16 / 6.26
	NGSIM	4.35 / 7.89	7.50 / 14.82	1.23 / 3.36
	NuScenes	5.82 / 8.16	5.03 / 8.78	5.74 / 5.33
FQA	ApolloScape	1.87 / 4.95	2.91 / 9.53	6.96 / 6.53
	NGSIM	4.63 / 8.69	7.88 / 15.62	1.63 / 4.16
	NuScenes	6.62 / 8.96	5.83 / 9.58	6.54 / 6.13

Table 3: Average prediction error before and after the perturbation.



(a) OOD Scene (b) ID Scene

Figure 4: Statistical evolution of CUSUM detection given the prediction errors from both ID and OOD scenes. A perturbation is introduced at time step $\gamma = 490$ (i.e., the change point occurs at step 490). At time step 507, the statistic W_t surpasses the threshold ($b = 7$), signaling an OOD detection. This results in a detection delay of 17 samples. The experiment is conducted using the GRIP++ model on the ApolloScape dataset, with prediction errors evaluated using the ADE metric.

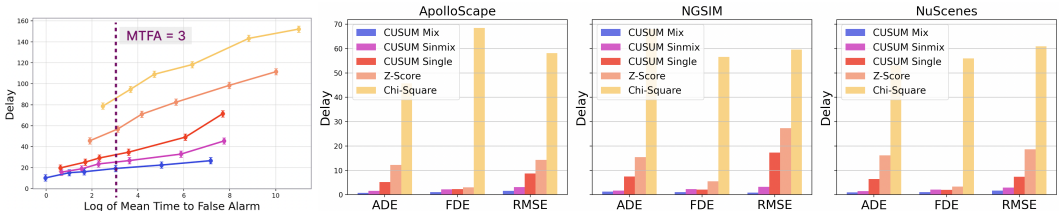
Deceptive OOD Scene Generation We generate deceptive OOD scenes by introducing subtle, physically constrained changes in driving behaviors. These changes, often imperceptible to human observers, are designed to significantly degrade the prediction performance of machine learning models. Inspired by adversarial perturbation techniques (Zhang et al. 2022), we apply these perturbations in a single-frame prediction setting for each dataset-model combination. The results, presented in Table 3, demonstrate that such perturbations consistently worsen trajectory prediction accuracy. On average, ADE increases by 148.8%, FDE by 176.2%, and RMSE by 50.1%. Notably, FDE experiences the highest increase (176.2%), highlighting greater vulnerability in long-term predictions compared to short-term or overall error measures. These findings suggest that even minor trajectory shifts could have real-world consequences if OOD scenes cannot be detected.

Gaussian Mixture Model Observation Figure 3 presents the mixture Gaussian distributions of prediction errors for pre-change (f_ϕ) and post-change (g_θ) error distributions. We evaluate the ApolloScape dataset using the ADE metric, comparing results from two distinct models: (a) GRIP++ (Figure 3a) and (b) FQA (Figure 3b). Both models consistently exhibit the same trend, underscoring the robustness of our findings and validating the presence of distributional shifts in prediction errors.

CUSUM Change-point Detection Figure 4 illustrates how the CUSUM algorithm detects abnormal shifts in AV trajectory prediction by tracking changes in prediction errors. It monitors the cumulative statistic W_t, summing log-likelihood ratios between pre- and post-change distributions, and declares a change when $W_t > b$. In the OOD scene (Figure 4a), a perturbation at step 490 causes W_t to exceed $b = 7$ at step 507, triggering detection with a 17-sample delay. In contrast, in the ID scene (Figure 4b), prediction errors remain stable and W_t stays below b, confirming no false alarms.

Delay vs. MTFA We evaluate the trade-off between detection delay and mean time to false alarm (MTFA), balancing responsiveness and reliability (Figure 5a). CUSUM Mix achieves the lowest delays—under 20 samples across thresholds—due to GMM-based modeling of both pre- and post-change distributions. CUSUM Sinmix, using a single Gaussian for post-change, delays detection to 25 samples at MTFA = 3, while CUSUM Single reaches 30. Chi-Square performs worst, nearly four times slower than CUSUM Mix, and Z-Score, though better, still lags behind all CUSUM variants. Figure 6 further confirms CUSUM's advantage in balancing speed and false-alarm control.

Average Detection Delay To enable a fair comparison of detection delays across different algorithms, we first ensured that all methods operated under the MTFA. This was achieved by carefully adjusting the detection threshold for each algorithm so that they can reach equivalent MTFA. Once aligned, we measured the detection delay—defined as the number of samples between the true change point and the algorithm's detection. As shown in Figure 5b, CUSUM Mix delivered the fastest response, with an average delay of just 3 samples, followed by CUSUM Sinmix at 5 samples and CUSUM Single at 8 samples. In comparison, the Z-Score method detected changes with a longer delay of 15 samples, and the Chi-Square method, included as a baseline, had the slowest response at 50 samples. To ensure statistical robustness, we fixed the change point at time step 1000 and repeated each experiment 5000 times. These results clearly demonstrate the efficiency of CUSUM-based methods in minimizing detection delay, making them highly



(a) Delay v.s. MTFA (b) Average Detection Delay

Figure 5: (a) Performance of Delay over Log of Mean Time to False Alarm (MTFA) of ApolloScape dataset, using GRIP++ model and ADE metric, as the threshold b increases from left to right. (b) Average detection delay of the tested OOD detection algorithms under same control of time to false alarm across different metrics. The delay is measured using **ADE**, **FDE**, and **RMSE**, providing a comparative analysis of algorithm performance in identifying anomalies. Lower delay values indicate faster detection.

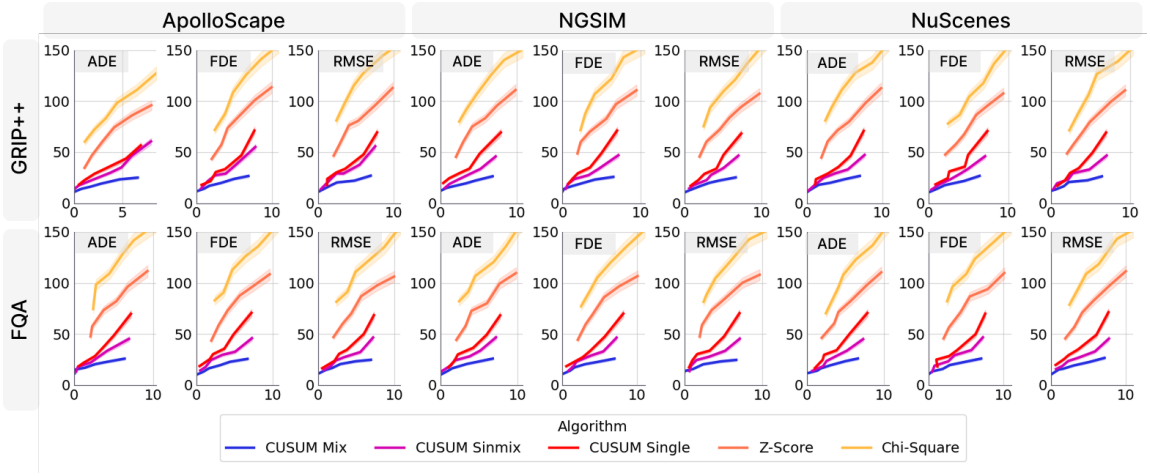


Figure 6: Performance of Delay over MTFA, comparing GRIP++ (top row) and FQA (bottom row). Each row is grouped in threes: the first three columns represent ApolloScape datasets, followed by NGSIM, and NuScenes. Within each group, metrics are arranged as **ADE**, **FDE**, and **RMSE**, validating trends and demonstrating consistent performance across diverse scenarios.

suitable for real-time applications where rapid and reliable change detection is critical.

6 Conclusions

We propose lightweight Quick Change Detection (QCD) methods for detecting out-of-distribution (OOD) scenes in real-world trajectory prediction datasets, introducing a novel approach in this field. Our method monitors a scalar variable of prediction errors, enabling OOD detection at any point during inference. Unlike prior studies that rely heavily on simulated environments, we rigorously address this challenge using three real-world datasets—ApolloScape, NGSIM, and NuScenes—alongside two state-of-the-art prediction models, GRIP++ and FQA. Our work demonstrates the feasibility and effectiveness of applying CUSUM-based algorithms for timely and accurate OOD detection in AV systems. Experimental results show that CUSUM Mix consistently achieves superior detection performance with minimal false alarms, particularly when modeled using GMMs across all dataset-model combinations. These findings provide a robust solution for enhancing the safety and reliability of AV systems through timely model adjustments in dynamic environments. For future work, we suggest exploring a tiered alarm system with context-aware alerts to prioritize critical warnings while reducing computational overhead, as well as leveraging imitation learning to adapt the system to OOD scenes.

References

Alippi, C.; and Roveri, M. 2006. An adaptive cusum-based test for signal change detection. In *2006 IEEE international symposium on circuits and systems*, 4–pp. IEEE.

Anderson, C.; Vasudevan, R.; and Johnson-Roberson, M. 2021. A kinematic model for trajectory prediction in general highway scenarios. *IEEE Robotics and Automation Letters*, 6(4): 6757–6764.

Bahari, M.; Saadatnejad, S.; Rahimi, A.; Shaverdikondori, M.; Shahidzadeh, A. H.; Moosavi-Dezfooli, S.-M.; and Alahi, A. 2022. Vehicle trajectory prediction works, but not everywhere. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17123–17133.

Caesar, H.; Bankiti, V.; Lang, A. H.; Vora, S.; Liong, V. E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; and Beijbom,

O. 2020. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11621–11631. nill: nill.

Cheadle, C.; Vawter, M. P.; Freed, W. J.; and Becker, K. G. 2003. Analysis of microarray data using Z score transformation. *The Journal of molecular diagnostics*, 5(2): 73–81.

Chiang, S.; Zito, J.; Rao, V. R.; and Vannucci, M. 2024. Time-Series Analysis. In *Statistical Methods in Epilepsy*, 166–200. nill: Chapman and Hall/CRC.

Cui, P.; and Wang, J. 2022. Out-of-distribution (ood) detection based on deep learning: A review. *Electronics*, 11(21): 3500.

Dendorfer, P.; Elflein, S.; and Leal-Taixé, L. 2021. Mg-gan: A multi-generator model preventing out-of-distribution samples in pedestrian trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 13158–13167. nill: nill.

Deo, N.; Rangesh, A.; and Trivedi, M. M. 2018. How would surround vehicles move? A unified framework for maneuver classification and motion prediction. *IEEE Transactions on Intelligent Vehicles*, 3(2): 129–140.

Deo, N.; and Trivedi, M. M. 2018. Convolutional social pooling for vehicle trajectory prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 1468–1476. nill: nill.

Fang, Z.; Li, Y.; Lu, J.; Dong, J.; Han, B.; and Liu, F. 2022. Is out-of-distribution detection learnable? *Advances in Neural Information Processing Systems*, 35: 37199–37213.

Federal Highway Administration. 2020. Traffic Analysis Tools: Next Generation Simulation. [Accessed: Sep. 13, 2024].

Filos, A.; Tigkas, P.; McAllister, R.; Rhinehart, N.; Levine, S.; and Gal, Y. 2020. Can autonomous vehicles identify, recover from, and adapt to distribution shifts? In *International Conference on Machine Learning*, 3145–3153. PMLR, nill: nill.

Franke, T. M.; Ho, T.; and Christie, C. A. 2012. The chi-square test: Often used and more often misinterpreted. *American journal of evaluation*, 33(3): 448–458.

Hendrycks, D.; and Gimpel, K. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, nill(nill): nill.

Huang, X.; Cheng, X.; Geng, Q.; Cao, B.; Zhou, D.; Wang, P.; Lin, Y.; and Yang, R. 2018. The apolloscape dataset for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 954–960. nill: nill.

Ivanovic, B.; and Pavone, M. 2022. Injecting planning-awareness into prediction and detection evaluation. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, 821–828. IEEE, nill: nill.

Kamra, N.; Zhu, H.; Trivedi, D. K.; Zhang, M.; and Liu, Y. 2020. Multi-agent trajectory prediction with fuzzy query attention. *Advances in Neural Information Processing Systems*, 33: 22530–22541.

Kim, H.; Kim, D.; Kim, G.; Cho, J.; and Huh, K. 2020. Multi-head attention based probabilistic vehicle trajectory prediction. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, 1720–1725. IEEE, nill: nill.

Kuefler, A.; Morton, J.; Wheeler, T.; and Kochenderfer, M. 2017. Imitating driver behavior with generative adversarial networks. In *2017 IEEE intelligent vehicles symposium (IV)*, 204–211. IEEE.

Lai, L.; Fan, Y.; and Poor, H. V. 2008. Quickest detection in cognitive radio: A sequential change detection framework. In *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*, 1–5. IEEE, nill: nill.

Lau, T. S.; Tay, W. P.; and Veeravalli, V. V. 2018. A binning approach to quickest change detection with unknown post-change distribution. *IEEE Transactions on Signal Processing*, 67(3): 609–621.

Lee, K.; Lee, K.; Lee, H.; and Shin, J. 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31(nill): nill.

Li, K.; Chen, K.; Wang, H.; Hong, L.; Ye, C.; Han, J.; Chen, Y.; Zhang, W.; Xu, C.; Yeung, D.-Y.; et al. 2022. Coda: A real-world road corner case dataset for object detection in autonomous driving. In *European Conference on Computer Vision*, 406–423. Springer, nill: nill.

Li, X.; Ying, X.; and Chuah, M. C. 2019. Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving. *arXiv preprint arXiv:1907.07792*, nill(nill): nill.

Liang, Y.; and Veeravalli, V. V. 2024. Quickest Change Detection with Post-Change Density Estimation. *IEEE Transactions on Information Theory*, nill(nill): nill.

Liu, M.; Cheng, H.; Chen, L.; Broszio, H.; Li, J.; Zhao, R.; Sester, M.; and Yang, M. Y. 2024. Laformer: Trajectory prediction for autonomous driving with lane-aware scene constraints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2039–2049. nill: nill.

Liu, W.; Wang, X.; Owens, J.; and Li, Y. 2020. Energy-based out-of-distribution detection. *Advances in neural information processing systems*, 33: 21464–21475.

Lorden, G. 1971. Procedures for reacting to a change in distribution. *The annals of mathematical statistics*, nill(nill): 1897–1908.

Manogaran, G.; and Lopez, D. 2018. Spatial cumulative sum algorithm with big data analytics for climate change detection. *Computers & Electrical Engineering*, 65: 207–221.

Mare, D. S.; Moreira, F.; and Rossi, R. 2017. Nonstationary Z-score measures. *European Journal of Operational Research*, 260(1): 348–358.

Moustakides, G. V. 1986. Optimal stopping times for detecting changes in distributions. *the Annals of Statistics*, 14(4): 1379–1387.

Neumeier, M.; Dorn, S.; Botsch, M.; and Utschick, W. 2024. Reliable trajectory prediction and uncertainty quantification with conditioned diffusion models. In *Proceedings of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 3461–3470.

Page, E. S. 1954. Continuous inspection schemes. *Biometrika*, 41(1/2): 100–115.

Peng, M.; Wang, J.; Song, D.; Miao, F.; and Su, L. 2023. Privacy-Preserving and Uncertainty-Aware Federated Trajectory Prediction for Connected Autonomous Vehicles. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 11141–11147. IEEE, nill: nill.

Pustynnikov, A.; and Eremeev, D. 2021. Estimating uncertainty for vehicle motion prediction on yandex shifts dataset. *arXiv preprint arXiv:2112.08355*.

Sun, Y.; Ming, Y.; Zhu, X.; and Li, Y. 2022a. Out-of-distribution detection with deep nearest neighbors. In *International Conference on Machine Learning*, 20827–20840. PMLR, nill: nill.

Sun, Z.; Zou, S.; Zhang, R.; and Li, Q. 2022b. Quickest change detection in anonymous heterogeneous sensor networks. *IEEE Transactions on Signal Processing*, 70: 1041–1055.

Tang, X.; Jia, T.; Hu, X.; Huang, Y.; Deng, Z.; and Pu, H. 2020. Naturalistic data-driven predictive energy management for plug-in hybrid electric vehicles. *IEEE Transactions on Transportation Electrification*, 7(2): 497–508.

Tang, X.; Kan, M.; Shan, S.; Ji, Z.; Bai, J.; and Chen, X. 2024. Hpnet: Dynamic trajectory forecasting with historical prediction attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 15261–15270. nill: nill.

Tartakovsky, A. G.; Polunchenko, A. S.; and Sokolov, G. 2012. Efficient computer network anomaly detection by changepoint detection methods. *IEEE Journal of Selected Topics in Signal Processing*, 7(1): 4–11.

Veeravalli, V. V.; and Banerjee, T. 2014. Quickest change detection. In *Academic press library in signal processing*, volume 3, 209–255. nill: Elsevier.

Wallis, W. A.; and Moore, G. H. 1941. A significance test for time series analysis. *Journal of the American Statistical Association*, 36(215): 401–409.

Wang, C.; Ma, L.; Li, R.; Durrani, T. S.; and Zhang, H. 2019. Exploring trajectory prediction through machine learning methods. *IEEE Access*, 7: 101441–101452.

Wiederer, J.; Schmidt, J.; Kressel, U.; Dietmayer, K.; and Belagiannis, V. 2023. Joint out-of-distribution detection and uncertainty estimation for trajectory prediction. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 5487–5494. IEEE.

Zhang, Q.; Hu, S.; Sun, J.; Chen, Q. A.; and Mao, Z. M. 2022. On adversarial robustness of trajectory prediction for autonomous vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 15159–15168. nill: nill.