T-Assess: An Efficient Data Quality Assessment System Tailored for Trajectory Data

Junhao Zhu
Zhejiang University
zhujunhao@zju.edu.cn

Tao Wang
Zhejiang University
wangtop@zju.edu.cn

Danlei Hu
Zhejiang University
dlhu@zju.edu.cn

Ziquan Fang
Zhejiang University
zqfang@zju.edu.cn

Lu Chen, Yunjun Gao
Zhejiang University
{luchen,gaoyj}@zju.edu.cn

Tianyi Li, Christian S. Jensen
Aalborg University
{tianyi,csj}@cs.aau.dk

ABSTRACT

With the widespread use of GPS-enabled devices and services, trajectory data fuels services in a variety of fields, such as transportation and smart cities. However, trajectory data often contains errors stemming from inaccurate GPS measurements, low sampling rates, and transmission interruptions, yielding low-quality trajectory data with negative effects on downstream services. Therefore, a crucial yet tedious endeavor is to assess the quality of trajectory data, serving as a guide for subsequent data cleaning and analyses. Despite some studies addressing general-purpose data quality assessment, no studies exist that are tailored specifically for trajectory data.

To more effectively diagnose the quality of trajectory data, we propose T-Assess, an automated trajectory data quality assessment system. T-Assess is built on three fundamental principles: i) extensive coverage, ii) versatility, and iii) efficiency. To achieve comprehensive coverage, we propose assessment criteria spanning validity, completeness, consistency, and fairness. To provide high versatility, T-Assess supports both offline and online evaluations for full-batch trajectory datasets as well as real-time trajectory streams. In addition, we incorporate an evaluation optimization strategy to achieve assessment efficiency. Extensive experiments on three real-life benchmark datasets offer insight into the effectiveness of T-Assess at quantifying trajectory data quality and offers evidence that it is superior to the state-of-the-art data quality systems.

PVLDB Reference Format:
Junhao Zhu, Tao Wang, Danlei Hu, Ziquan Fang, Lu Chen, Yunjun Gao, and Tianyi Li, Christian S. Jensen. T-Assess: An Efficient Data Quality Assessment System Tailored for Trajectory Data. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/ZJU-DAILY/T-Assess.

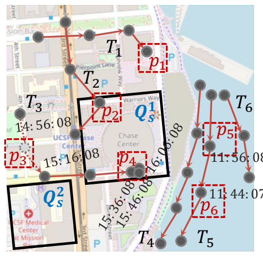
Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

1 INTRODUCTION

With the proliferation of GPS-enabled devices, massive trajectory data of moving objects such as vehicles is being accumulated that can fuel important real-word applications in fields such as smart cities, covering transportation and urban planning [26], as well as consumer services such as POI recommendation [25, 37], to name but a few. However, due to a variety of deficiencies, data quality issues typically occur during GPS data collection and trajectory generation. Figure 1 illustrates three types of common trajectory errors. For example, trajectory T_1 that is generated by a vehicle contains a point p_1 that is positioned incorrectly inside a building. Such quality issues degrade the effectiveness of downstream trajectory driven applications. Therefore, a crucial yet tedious endeavor is to enable trajectory data quality assessment, that may guide (targeted) trajectory cleaning, thereby improving the quality of applications.

Several existing studies [7, 19, 20, 29, 32] focus on the quantification of data quality. First, task-agnostic data quality assessment [7, 24, 28, 29, 31, 33] leverages data characteristics to quantify various quality dimensions. Second, task-aware data quality assessment [6, 12, 13, 27] considers data quality within the context of specific tasks, especially in the realm of machine learning (ML). These existing studies target general-purpose data quality assessment and therefore do not contend well with aspects specific to trajectories, such as the following:

- ***Fail to identify spatio-temporal patterns.*** Trajectory data captures spatio-temporal patterns, representing constraints on individual points in a trajectory (*e.g.*, location scope) or constraints between points in a trajectory (*e.g.*, speed constraints). However, existing data quality assessment methods only focus on patterns in general data and fall short at capturing violations of spatio-temporal patterns.
- ***Fail to capture inter-dependencies.*** Inter-dependencies refer to relations across trajectories in the dataset. For instance, trajectories usually exhibit specific group behaviors, *e.g.*, morning/evening peaks. However, the most related studies [24, 33] focus on detecting errors within a single univariate time-series, lacking the capability to consider inter-dependencies.
- ***Fail to realize topographical contexts.*** Topographical context is defined as constraints on trajectories caused by the specific geographic environment. For example, trajectories capturing the movements of cars in urban areas have to be consistent with the road network of the city. However, existing data quality assessment methods struggle with leveraging topographical context as they ignore auxiliary information like road networks and are



Dimension	Trajectory	Description
Validity	T_5	p_5, p_6 violate the timestamp monotonicity
Completeness	T_3	p_3 is a missing point
Consistency	T_1	p_1 is positioned incorrectly inside a building
	T_2	p_2 suffers from a location shift
	T_3	The vehicle should not stay on the road (p_4)
	T_6	T_6 deviates from the course of T_4, T_5

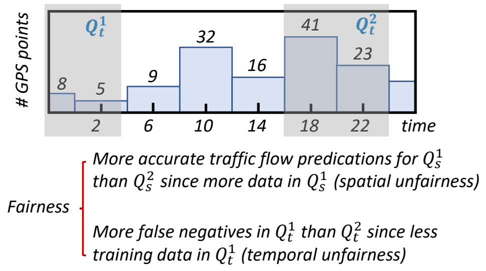


Figure 1: Examples of common types of erroneous trajectories.

unable to capture underlying relations between topographical contexts and trajectories.

In addition, spatio-temporal data is often utilized as it is being generated in ***streaming fashion***. However, most of the existing data quality studies target static data and cannot be extended to contend effectively with streaming data. Although a few systems [29] support incremental quality evaluation, they still face efficiency challenges when intricate constraint checks are involved. Motivated by the importance of trajectory data and this state-of-the-art, we introduce T-Assess, an automated quality assessment system specifically tailored for trajectory data. T-Assess is designed based on three principles:

(i) Extensive coverage. To achieve a broad understanding of trajectory data quality, assessments should cover trajectory irregularities and error cases broadly. Understanding the positive impact of data quality assessment on downstream applications and the shortages of existing trajectory data quality assessment methods is imperative to achieve a system that can cover the most typical errors in trajectory data. Considering the unique characteristics of trajectory data, our system offers constraints in four dimensions, facilitating a broad-based quality evaluation.

(ii) Versatility. Real-life applications call for both offline (*e.g.*, urban planing [26], POI recommendations [25, 37]) and online trajectory analyses (*e.g.*, real-time congestion monitoring [35], vessel monitoring [23]). Thus, T-Assess is designed to evaluate the quality of both historical and streaming trajectory data, supporting both offline and online trajectory analyses.

(iii) Efficiency. Arrival rates of streaming trajectories are often in the range from 1 per second to 1 per minute [3, 39]. Efficient data quality assessment that can keep up with such rates is crucial for subsequent data cleaning and analysis. A naive approach to evaluate trajectory data quality is a full scan of the data, identifying all points in a trajectory dataset that violate constraints. However, this is time-consuming, especially for massive trajectory data. To achieve efficiency, we first deploy T-Assess on a distributed dataflow engine (*e.g.*, Spark [2] and Kafka [1]) for historical and streaming trajectory data, thereby leveraging parallel processing capabilities. Inspired by the intuition that trajectories with the same spatial features share the same data quality, we approximate overall data quality by evaluating a subset of trajectories, which reduces lots of redundant calculations. Specifically, we propose a cluster-selection strategy consisting of efficient trajectory similarity search and representative selection, which achieves high efficiency and low estimation error.

The major contributions of the paper are as follows.

- We propose T-Assess, an efficient data quality assessment system tailored for trajectory data that leverages a distributed dataflow engine to enable both offline and online evaluations. To the best of our knowledge, this is the first study that targets the evaluation of trajectory data quality.
- We provide a comprehensive set of trajectory data quality metrics in terms of four dimensions (*i.e.*, validity, completeness, consistency, and fairness) that take into account the special characteristics of trajectory data.
- We design an evaluation optimization strategy that leverages spatio-temporal correlations between trajectories to improve data quality assessment efficiency within acceptable error margins.
- We report on extensive experiments that offer insight into the effectiveness and efficiency of T-Assess, including its optimization.

The rest of the paper is organized as follows. We present the problem addressed in Section 2. We provide an overview of the architecture of T-Assess in Section 3. Sections 4 and 5 present the data quality evaluation methods and evaluation optimization, respectively. Section 6 covers the experimental study. Section 7 reviews related work, and Section 8 concludes and offers directions for future work.

2 PRELIMINARIES

We first introduce two trajectory data models and then present the quality assessment problem addressed.

2.1 Trajectory Data Models

Trajectory datasets can be collected in *batch* or *streaming* modes. We thus provide data models for batch and streaming trajectory datasets.

Definition 1 (Batch trajectory dataset). *A batch trajectory dataset is a set of trajectories, denoted by $\mathcal{T}_B = \{T_1, T_2, \ldots, T_N\}$, where N is the number of trajectories. Each trajectory T is a* ***bounded****, time-ordered sequence of GPS points, denoted by $T = \langle p_1, p_2, \ldots, p_n \rangle$, where n represents the number of points and each GPS point $p = (x, y, t)$ consists of a longitude x, a latitude y, and a timestamp t.*

A batch trajectory dataset remains unchanged once collected. We thus assume that a batch trajectory dataset remains unchanged during data quality assessment.

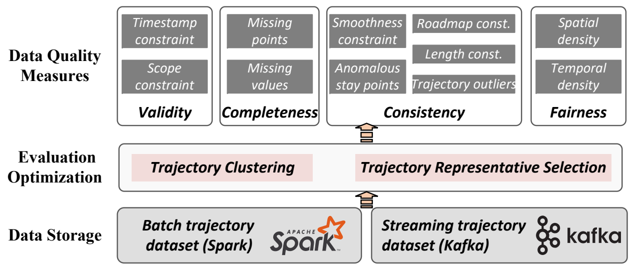


Figure 2: The system architecture of T-Assess.

Definition 2 (Streaming trajectory dataset). *A streaming trajectory dataset $\mathcal{T}_S$ is a set of streaming trajectories. Each streaming trajectory T is an **unbounded**, time-ordered sequence of GPS points, denoted by $T = \langle p_1, p_2, \dots \rangle$. That is, in the streaming setting, GPS points keep arriving over time.*

We use p_i to denote the i-th GPS point in a trajectory T and $p_i.x$ (resp. $p_i.y$ and $p_i.t$) to denote the longitude x (resp. latitude y and timestamp t) of the GPS point p_i. We drop subscripts for the general case.

Note that, the road network is a crucial topographical context for detecting moving patterns in trajectory data [10, 19]. Therefore, if available, the road network is also used as one of the inputs for the data quality assessment system and assists in detecting data quality issues.

2.2 Trajectory Data Quality Assessment

To support both batch and streaming trajectory datasets, offline and online quality assessments are considered.

Definition 3 (Offline quality assessment). *Given a batch trajectory dataset $\mathcal{T}_B$ and a set of trajectory data quality measures (i.e., constraints, to be detailed in Section 4.1), offline quality assessment aims to count the numbers of data points and trajectories in $\mathcal{T}_B$ that fail to satisfy the quality measures.*

Unlike in offline quality assessment, the number of points and trajectories that violate the pre-defined constraints in the online setting is dynamic. To support the online trajectory data quality assessment, following previous studies [11, 38], we adopt a sliding window model.

Definition 4 (Sliding window). *Given a length W and the current timestamp t_c, a sliding window contains points in the streaming trajectory dataset $\mathcal{T}_S$ located in the time period $[t_c - W, t_c] (t_c \geq W)$.*

While Definition 4 adopts a time-based sliding window, our problems can be extended to count-based sliding windows that cover the W most recent GPS points of each trajectory stream. Using the sliding window, online assessment is defined as follows.

Definition 5 (Online quality assessment). *Given a streaming trajectory dataset $\mathcal{T}_S$, a sliding window, and a set of trajectory data quality measures, online data quality assessment aims to count the numbers of data points and trajectories in the sliding window that fail to satisfy pre-defined quality measures.*

3 T-ASSESS ARCHITECTURE

To enable extensive, versatile, and efficient trajectory data quality assessment, the T-Assess automatic data quality assessment system adopts architecture illustrated in Figure 2.

T-Assess employs the distributed platforms Spark [2] and Kafka [1] for batch and streaming trajectory datasets, respectively. T-Assess operates on Array, a data type provided by Spark and Kafka for partitioned sequence-like data. T-Assess casts trajectories in a dataset to the Array data type, and applies constraint evaluations implemented on top of the Array to analyze the trajectories. These platforms leverage parallel computing to facilitate high-performance batch and streaming processing.

Even with the advantages of distributed computing, efficiency challenges arise when intricate constraint checks are involved. To address this, T-Assess incorporates an evaluation optimization strategy in a second layer that evaluates data quality approximately with error bounds. The optimization strategy consists of two steps: trajectory clustering and trajectory representative selection. The details are covered in Section 5. This way, T-Assess performs constraint checks on selected trajectory representatives rather than the entire dataset, which makes a trade-off between correctness and efficiency.

In the subsequent layer, T-Assess exposes a user-facing API that enables the formulation of a variety of data quality measures for trajectory data. Considering the unique characteristics of trajectories and potential downstream analyses, the proposed data quality measures span four dimensions, to be detailed in Section 4. With these data quality measures, T-Assess provides users with a comprehensive understanding of the quality of trajectory datasets.

4 DATA QUALITY EVALUATION

We first motivate and describe the detailed data quality dimensions, and then we present how to conduct offline and online evaluations in terms of the above four dimensions.

4.1 Data Quality Dimensions

4.1.1 ***Motivation***. Reviewing the data quality literature [7, 19, 20, 29, 32], existing quality metrics either do not target trajectory data or combine to offer only partial coverage. For example, Su et al. [33] consider two constraints for validity, and these do not enable to capture of relevant potential errors in trajectory data. As mentioned in the introduction, existing task-agnostic quality assessments do not take into account trajectory data characteristics, while task-aware ones target specific application contexts. T-Assess combines the advantages of both and provides a broad range of quality metrics spanning four dimensions. These metrics consider both the trajectory characteristics and potential impacts on applications, effectively addressing the "**fails**" mentioned in the introduction.

4.1.2 ***Design***. We proceed to cover the quality metrics using the examples of errors illustrated in Figure 1.

Validity refers to the degree to which trajectory data conforms to basic spatio-temporal patterns. It addresses the **first "fail"** stated in the introduction by including constraints for individual points in a trajectory, namely *timestamp* and *scope constraint*. The timestamp constraint ensures that timestamps of trajectory points increase monotonically over time. For example, p_5 and p_6 in T_5 violate the timestamp constraint since the timestamp of p_6 is smaller than that of p_5. The scope constraint restricts the location scope of GPS points. For example, the longitude of a GPS point cannot exceed the range of $[-180, 180]$.

Completeness concerns the integrity and informativeness of trajectory data. It is included in most data quality studies [8, 24, 29, 32] and can guide for data cleaning (*e.g.*, position imputation [21]). Here, we consider *missing points* and *missing values*. A missing point exists if the time interval between two consecutive points is abnormally long. For instance, p_3 is regarded as a missing point since the time interval between points before and after p_3 (20s) is larger than the average sampling interval (10s) of T_3. We regard a GPS point as a missing value if it has no longitude or latitude value.

Consistency is gauged by the extent to which a set of semantic rules is upheld. It addresses three "fails" by enabling constraints between the points in a trajectory, constraints concerning interdependencies, and constraints concerning topographical contexts. Specifically, we consider these: *smoothness constraint* (addressing the **first "fail"**), *length constraint* and *trajectory outliers* (addressing the **second "fail"**), *anomalous stay points* and *road network constraint* (addressing the **third "fail"**). The smoothness constraint restricts location shifts within a trajectory. For example, p_2 in T_2 violates the smoothness constraint as it deviates abnormally from other points in T_2. The length constraint restricts the length of a trajectory in a dataset, with an abnormally short trajectory regarded as a violation. A trajectory deviating from the proper course is considered a trajectory outlier. For instance, T_6 in Figure 1 is a trajectory outlier as it deviates from its course, as captured by T_4 and T_5. Considering topographical contexts, a point is regarded as anomalous when its location is inappropriate. For example, p_4 in T_3 is a stay point that stays in one place for a long time, and it is anomalous because it stays on the road (typically forbidden). The road network constraint specifies that trajectories generated by vehicles have to be constrained to the road network. For example, p_1 in T_1 violates the road network constraint, as it is positioned inside a building.

Fairness evaluates the degree of biased information in a trajectory dataset, which affects machine learning results in subsequent applications. We focus on *spatial* and *temporal density*. Spatial density measures the density of GPS points in a specific region compared to the overall density of the dataset, indicating data skewness in the spatial dimension. Taking Q_s^1 and Q_s^2 in Figure 1 as an example, assuming traffic flow prediction as a subsequent task, it is expected that models will produce more accurate predictions for region Q_s^1 due to its better GPS point coverage compared to Q_s^2. Temporal density measures the density of GPS points in a specific timespan compared to the overall density, indicating data skewness in the temporal dimension. For instance, considering Q_t^1 and Q_t^2 in Figure 1 and anomaly detection across timespans as a subsequent task, it is expected that more false negatives may occur in region Q_t^1 than Q_t^2 due to less training data in Q_t^1. *Fairness* alerts data practitioners to perform data augmentation to address data skewness.

We use the above metrics to evaluate the trajectory data quality in general scenarios. However, different quality assessment requirements may exist in specific application scenarios. Our system can be easily extended to support more data quality metrics.

4.2 Evaluation Implementation

Table 1 lists the metrics available in our paper for evaluating trajectory data quality. In our implementation, we employ classical methods to evaluate the metrics as described in Table 1. The system ultimately returns counts of violations of the quality metrics as the outputs. Instead of aiming for SOTA performance in a specific quality metric evaluation, our system prioritizes offering a broad-based quality assessment. Our system can incorporate any of the SOTA techniques for these quality metric evaluations.

4.2.1 *Timestamp constraint.*

The timestamp constraint is defined as follows.

$$\forall 0 < i \le j \le n, p_i.t \le p_j.t. \tag{1}$$

In both offline and online settings, T-Assess checks the timestamp constraint by traversing GPS points in a dataset. Specifically, for every two consecutive points p_{i+1} and p_i, our system verifies whether they violate Ineq. 1. In online evaluations, the last GPS point of each trajectory within the current time window is evaluated in the upcoming time window.

4.2.2 *Scope constraint.*

The scope constraint is defined as follows.

$$x_{min} \le p_i.x \le x_{max} \quad \text{and} \quad y_{min} \le p_i.y \le y_{max} \tag{2}$$

where x_{min} and x_{max} (resp. y_{min} and y_{max}) represent minimum and maximum longitudes (resp. latitudes) of the range under consideration. Unless stated, the defaults are $x_{min} = -180$, $x_{max} = 180$, $y_{min} = -90$, and $y_{max} = 90$. In both offline and online settings, our system checks the scope constraint for every GPS points by traversing through the dataset.

4.2.3 *Missing points.*

Taking into account the inevitable precision errors of the device, given the sampling time interval Δt of the device, we propose that there is no missing point between p_i and p_{i+1} if the following holds,

$$p_{i+1}.t - p_i.t < 2\Delta t. \tag{3}$$

The primary challenge in identifying missing points lies in determining the sampling period Δt. Our system regards the most frequent time interval among all consecutive GPS points as the sampling period. Given the sampling period Δt, for any consecutive points p_{i+1} and p_i, the number of missing points between p_{i+1} and p_i is estimated as $\lfloor \frac{p_{i+1}.t - p_i.t}{2\Delta t} - 1 \rfloor$.

4.2.4 *Missing values.*

For a GPS point p_i, we consider it an incomplete point with missing values if

$$p_i.x = \texttt{NULL} \quad \text{or} \quad p_i.y = \texttt{NULL}. \tag{4}$$

In both offline and online modes, our system checks whether GPS points satisfy Eq. 4 by traversing every GPS points.

4.2.5 *Smoothness constraint.*

We identify abrupt location shifts in trajectories, treating these as violations of the smoothness constraint. In both offline and online settings, the Kalman filter [40] is deployed in our system to detect large deviations.

Table 1: Metrics for evaluating trajectory data quality.

Metric	Definition
	Dimension *Validity*
Timestamp constraint	$p_i.t \leq p_j.t,\ \forall 0 < i \leq j \leq n$.
Scope constraint	$(x_{min} \leq p_i.x \leq x_{max}) \wedge (y_{min} \leq p_i.y \leq y_{max})$, where x_{min} and x_{max} (resp. y_{min} and y_{max}) represent minimum and maximum longitudes (resp. latitudes) of the range under consideration.
	Dimension *Completeness*
Missing point	$p_{i+1}.t - p_i.t < 2\Delta t$, where Δt is the sampling time interval.
Missing value	$(p_i.x \neq \texttt{NULL}) \wedge (p_i.y \neq \texttt{NULL})$.
	Dimension *Consistency*
Smoothness constraint	Following the previous definition [40], treat abrupt location shifts in trajectories as violations of the smoothness constraint.
Length constraint	$\sum_{1\leq i\leq n-1} dist(p_{i+1}, p_i) \geq L_{th}$, where $dist(*,*)$ is a distance function and L_{th} denotes the trajectory length threshold.
Trajectory outlier	Following the previous definition [18, 22], a GPS point p_i from trajectory T is deemed a point outlier if the number of its neighbors — defined as points from other trajectories with a distance of less than σ to p_i — is less than a threshold η. If a trajectory T contains more than a pre-set number ρ of these point outliers, T is considered an outlier trajectory.
Anomalous stay point	Regard point outliers that are not in stay point clusters as anomalous stay points.
Road network constraint	Following the previous definition [15], in a directed graph-form road network $G = (V, E)$, where a node $v \in V$ represents an intersection or a road end and an edge $e \in E$ represents a road segment, a path P is a sequence of connected segments, *i.e.*, $P : e_1 \to e_2 \to \cdots \to e_n$. The roadmap constraint requires each trajectory T to correspond with a path P in G.
	Dimension *Fairness*
Spatial density	$\frac{\lvert\mathcal{T}(Q_s)\rvert}{\sum_{T\in\mathcal{T}}\lvert T\rvert} \cdot \frac{Area}{Area(Q_s)}$, where Q_s is a rectangular region, $\mathcal{T}(Q_s)$ is a set of GPS points from $\mathcal{T}$ within Q_s, $\sum_{T\in\mathcal{T}}\lvert T\rvert$ is the total dataset's GPS points, and $Area$ and $Area(Q_s)$ denote the dataset's total area and the area of Q_s, respectively.
Temporal density	$\frac{\lvert\mathcal{T}(Q_t)\rvert}{\sum_{T\in\mathcal{T}}\lvert T\rvert} \cdot \frac{\max_{p_i\in\mathcal{T}}(p_i.t) - \min_{p_i\in\mathcal{T}}(p_i.t)}{t_{max}-t_{min}}$, where $Q_t = [t_{min}, t_{max}]$ is a user-specified timespan, $\mathcal{T}(Q_t)$ is a set of GPS points in Q_t, and $\max_{p_i\in\mathcal{T}}(p_i.t) - \min_{p_i\in\mathcal{T}}(p_i.t)$ is the dataset's total timespan.

4.2.6 *Length constraint.*
The length constraint is defined as:

$$\sum_{1\leq i\leq n-1} dist(p_{i+1}, p_i) \geq L_{th} \tag{5}$$

where the distance function $dist(*,*)$ is implemented as haversine distance and L_{th} denotes the trajectory length threshold. Any trajectory shorter than L_{th} is considered erroneous.

Based on the 3-sigma rule, the trajectory length threshold L_{th} is set to three times the standard deviation of the average trajectory length. In the offline setting, T-Assess compares the length of each trajectory with L_{th} via Eq. 5. In the online setting, the key problem is determining when a trajectory stream ends since only buffered GPS points in the trajectory are accessible. To solve it, T-Assess determines that a trajectory stream has ended if no inflow of GPS points is registered in k consecutive time windows.

4.2.7 *Trajectory outliers.*
Following the previous definition [18, 22], given a distance threshold σ, for a GPS point p_i from the trajectory T, the neighbors of the point p_i is denoted by $\mathcal{N}(p_i) = \{p_j \in \mathcal{T} - T \mid dist(p_i, p_j) \leq \sigma\}$. If the size of the neighbors is smaller than a threshold η, *i.e.*, $|\mathcal{N}(p_i)| \leq \eta$, the point p_i is considered as a point outlier. If the number of point outliers contained in a trajectory T surpasses a threshold ρ, the trajectory T is regarded as an outlier.

In the offline setting, T-Assess is equipped with TRAOD [18], one of the most classical trajectory outlier detection methods, which first partitions the trajectory and then detects sub-trajectory outliers based on distance measures. In the online setting, T-Assess leverages an online variant of TRAOD, PTMDS [16], which extends TRAOD for streaming trajectory outlier detection.

4.2.8 *Anomalous stay points.*
A stay point stands for a region where the moving object stays for a while. Meaningful stay points in different trajectories, indicating particular geographic meanings (*e.g.*, parking areas), can be well clustered. On the other hand, anomalous stay points resulting from accidents (*e.g.*, device failures) tend to appear randomly. Hence, we take outliers (*i.e.*, data points not in the stay point clusters) as the anomalous stay points.

In the offline setting, we employ hierarchical TrajDBSCAN [34] to capture stay points and outliers. Specifically, a minor-neighborhood TrajDBSCAN is first applied to all points in the dataset, leading to stay points clustering together. A wide-reaching TrajDBSCAN is then applied to the identified stay points, disclosing outliers that fail to belong to any clusters–these are the anomalous stay points. In the online setting, TrajDBSCAN is replaced with dynamic DBSCAN [14] to cater to streaming trajectories.

4.2.9 *Road network constraint.*
Given a road network $G = (V, E)$ in the form of directed graph (each node $v \in V$ represents an intersection or a road end, each edge $e \in E$ represents a road segment), a path P is a sequence of connected road segments, *i.e.*, $P : e_1 \to e_2 \to \cdots \to e_n$, and the roadmap constraint specifies that every trajectory T has to match with a path P in G.

In both offline and online settings, our system incorporates OHMM [15] to find the correspondence between each point to a road segment in the roadmap. Once the correspondence does not exist, the point and the corresponding trajectory violate the roadmap constraint.

4.2.10 *Spatial density.*
Given a rectangle range region $Q_s = [p_{min}, p_{max}]$ specified by users, where p_{min} (resp. p_{max}) represents the bottom left (resp. top right) point of the range region, the set of GPS points from the trajectory dataset $\mathcal{T}$ within the given range region is denoted as $\mathcal{T}(Q_s)$. Then, the spatial density $\rho_s(Q_s, \mathcal{T})$ is

defined as:

$$\rho_s(Q_s, \mathcal{T}) = \frac{|\mathcal{T}(Q_s)|}{\sum_{T \in \mathcal{T}} |T|} \cdot \frac{Area}{Area(Q_s)} \tag{6}$$

where $\sum_{T \in \mathcal{T}} |T|$ represents the total number of GPS points in the dataset, $Area(Q_s)$ is the area of the given range region, and $Area$ represents the total area covered by the dataset. The density $\rho_s(Q_s, \mathcal{T})$ is restricted to the range $[0, 1]$, and indicates the degree of similarity in GPS point distribution between the specified range and the dataset.

In both offline and online settings, we construct a quad-tree for all GPS points in the dataset, in order to accelerate the range query to find $\mathcal{T}(Q_s)$ via breadth-first search.

4.2.11 Temporal density. Given a timespan $Q_t = [t_{min}, t_{max}]$ specified by users, the set of GPS points within the timespan is denoted by $\mathcal{T}(Q_t)$, the temporal density $\rho_t(Q_t, \mathcal{T})$ is defined as

$$\rho_t(Q_t, \mathcal{T}) = \frac{|\mathcal{T}(Q_t)|}{\sum_{T \in \mathcal{T}} |T|} \cdot \frac{\max_{p_i \in \mathcal{T}}(p_i.t) - \min_{p_i \in \mathcal{T}}(p_i.t)}{t_{max} - t_{min}} \tag{7}$$

where $\max_{p_i \in \mathcal{T}}(p_i.t) - \min_{p_i \in \mathcal{T}}(p_i.t)$ represents the timespan of the entire dataset. As with spatial density, temporal density $\rho_t(Q_t, \mathcal{T})$ also falls within $[0, 1]$.

To facilitate efficient timespan query, in both offline and online settings, we build segment trees for each leaf of the quad-tree. With the segment trees, to find GPS points in the given timespan, we traverse segment trees by breadth-first search and prune the nodes in the trees that are not in the given timespan.

5 EVALUATION OPTIMIZATION

We proceed to cover the propose system's optimization strategy by first motivating the strategy and then providing algorithm details.

5.1 Overview

While enabling perfect accuracy, data quality assessment via full data scan is prohibitively time-consuming, especially with massive trajectory data, even when leveraging a distributed dataflow engine. Thus, assuming the cost of quality assessment is proportional to the number and the lengths of trajectories in a dataset, we propose conducting data quality assessments on representative trajectories rather than on all trajectories. The key intuition is that *spatially similar trajectories share similar data quality.* The core challenges are those of *identifying spatially similar trajectories* and *selecting representative trajectories.*

5.2 Trajectory Clustering

5.2.1 ***Motivation.*** To find similar trajectories, the straightforward solution is pair-wise comparisons using some trajectory distance notion, such as Frechet distance. However, this has high time complexity [17]. To address this issue, we opt for trajectory clustering. Although different trajectory clustering methods exist, many of them require intricate trajectory feature engineering for subsequent clustering steps, making them computationally expensive [30]. Consequently, we propose an algorithm for generating trajectory clusters that avoids complex feature engineering, resulting in a more efficient process.

Algorithm 1: Trajectory clustering

```
Input: A grid-based trajectory dataset $\mathcal{T}^G = \{T_1^G, T_2^G, \ldots, T_M^G\}$
Output: A set $C$ of trajectory clusters
1  sort $\mathcal{T}^G$ in descending order of the trajectory length
2  initialize $C \leftarrow \emptyset$
3  initialize the set of cluster centers as empty
4  for each $T_i^G$ in $\mathcal{T}^G$ do
5      $has\_cluster \leftarrow$ False
6      for each $C_j$ in clusters $C$ do
7          $c_j \leftarrow$ the cluster center of $C_j$
8          $o_{ij} \leftarrow$ the number of overlaps between $T_i^G$ and $c_j$
9          if $o_{ij} \geq O$ then
10             add $T_i$ as a cluster member to the cluster $C_j$
11             $has\_cluster \leftarrow$ True
12             break
13     if $has\_cluster$ is False then
14         a new cluster $C_{new}$ is added to $C$
15         $T_i^G$ becomes the center of $C_{new}$
16 return The set $C$ of trajectory clusters
```

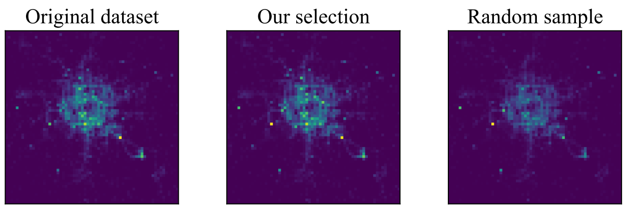


Figure 3: Visualization of various trajectory representative selection strategy on T-drive.

5.2.2 ***Design.*** To improve the efficiency of trajectory clustering, we convert free space-represented trajectories into grid space-represented trajectories. Specifically, given a two-dimensional free space, we construct a grid index G by partitioning the space into $2^\theta \times 2^\theta$ equal-sized grid cells, where 2^θ is referred to as the grid resolution. Each raw trajectory T can be converted into a grid-based trajectory T^G by replacing each point p_i in T with the ID of its grid cell. The trajectory clustering algorithm is presented in Algorithm 1. Given a grid-based trajectory dataset, we first sort them in descending order of the length of trajectories. Then we initialize an empty list that stores the centers of each cluster (we regard the longest trajectory in the cluster as the center of the cluster). During the clustering, we traverse the sorted grid-based trajectories, and for each visited trajectory T^G, we check whether it has at least O grid cells that intersect with any of the current centers. If T^G has at least O grid cells that overlap with cluster center c, we add it to the corresponding cluster C centered by c; otherwise, a new cluster C_{new} is added to the set of clusters and T^G becomes the center of C_{new}. This process continues until all trajectories are visited.

Complexity analysis. For a dataset of N trajectories and K clusters generated by the algorithm, its complexity is $O(N \log N + K(N-K))$.

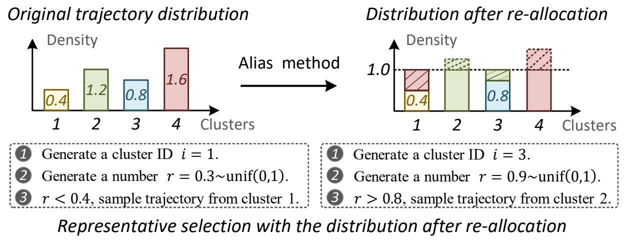

Original trajectory distribution
Density
0.4
1.2
0.8
1.6
1 2 3 4 Clusters
Alias method
Distribution after re-allocation
Density
1.0
0.4
0.8
1 2 3 4 Clusters
❶ Generate a cluster ID $i = 1$.
❷ Generate a number $r = 0.3 \sim \text{unif}(0,1)$.
❸ $r < 0.4$, sample trajectory from cluster 1.
❶ Generate a cluster ID $i = 3$.
❷ Generate a number $r = 0.9 \sim \text{unif}(0,1)$.
❸ $r > 0.8$, sample trajectory from cluster 2.
Representative selection with the distribution after re-allocation

Figure 4: A toy example to illustrate how the trajectory representative selection strategy works.

Algorithm 2: Trajectory representative selection

```
Input: A trajectory dataset $\mathcal{T} = \{T_1, T_2, \ldots, T_M\}$, trajectory
       clusters $C = \{C_1, C_2, \cdots, C_K\}$, the number of samples $N_s$
Output: A set $\mathcal{T}_S$ of sampled trajectories
1  initialize $\mathcal{T}_S \leftarrow \emptyset$
2  initialize an alias array $A$ as empty
3  rescale the size of each clusters, denoted as $d_i$ for each cluster $C_i$
4  initialize queues $Q_{>1}$ and $Q_{<1}$ as empty
5  for each cluster id $i$ in $C$ do
6      if $d_i > 1$ then $Q_{>1}$.push($i$)
7      else if $d_i < 1$ then $Q_{<1}$.push($i$)
8  while $Q_{>1}$ is not empty and $Q_{<1}$ is not empty do
9      $l \leftarrow Q_{>1}$.pop(), $s \leftarrow Q_{<1}$.pop()
10     $A[s] = l$
11     $d_l = d_l - (1 - d_s)$
12     if $d_l > 1$ then $Q_{>1}$.push($l$)
13 while $N_s > 0$ do
14     generate a cluster id $i$ at random
15     generate a random number $r \sim Uniform(0, 1)$
16     if $r < d_i$ then
17         sample the longest trajectory $T$ from the cluster $C_i$
18     else
19         sample the longest trajectory $T$ from the cluster $C_{A[i]}$
20     $\mathcal{T}_S = \mathcal{T}_S \cup \{T\}$
21     $N_s = N_s - 1$
22 return The set $\mathcal{T}_S$ of sampled trajectories
```

5.3 Trajectory Representative Selection

5.3.1 ***Motivation.*** Trajectory clustering groups spatially similar trajectories into the same cluster. Rather than considering all trajectories in a dataset, selecting a subset of trajectories from each cluster when performing a quality assessment can significantly improve efficiency. However, randomly selecting trajectories from clusters would severely distort the trajectory distribution of the dataset, rendering the quality assessment ineffective. Instead, we propose a trajectory representative selection strategy, which can preserve the trajectory distribution as well as ensure high performance in the selection process. As illustrated by Figure 3, we observe that the trajectory distribution of the dataset sampled by our selection strategy closely resembles that of the original dataset compared to the randomly sampled dataset.

5.3.2 ***Design.*** Sampling according to a specific distribution is more challenging and tricky than sampling from a uniform distribution. Inspired by this, the selection strategy revolves around sampling representatives from a uniform distribution but still following the trajectory distribution of the dataset. To achieve this, we re-allocate trajectories in different clusters to make the size of each cluster uniform, without changing the actual sizes of clusters and the affiliations of the trajectories, as illustrated by Figure 4.

We utilize the alias method [36] for trajectory reallocation. The process is as follows. Given clusters $C = \{C_1, C_2, \ldots, C_K\}$, we rescale the values of the sizes of clusters such that the mean value of all cluster sizes is 1. We denote d_i as the rescaled value of the size of cluster C_i, calculated as $d_i = \frac{|C_i| \cdot K}{\sum_{j=1}^{K} |C_j|}$. Following this preprocessing, the reallocation proceeds as follows: (1) An alias array A is initialized as empty. (2) Two queues, $Q_{>1}$ and $Q_{<1}$, are created, where $Q_{>1}$ stores IDs of clusters with $d_i > 1$ and $Q_{<1}$ holds IDs of clusters with $d_i < 1$. (3) Cluster IDs i and j are popped from $Q_{>1}$ and $Q_{<1}$. We place ID i in the j-th position of the alias array (*i.e.*, $A[j] = i$) and set $d_i = d_i - (1 - d_j)$. If $d_i > 1$, we put i back into $Q_{>1}$; otherwise, we omit it. (4) *Step 3* is repeated until either $Q_{>1}$ or $Q_{<1}$ is empty.

After trajectory reallocation, we sample representatives from the uniform distribution while preserving the original distribution. We randomly generate a cluster ID $i \in [1, K]$ and a random number r from a uniform distribution $Uniform(0, 1)$. If $r \leq d_i$, the longest trajectory from the i-th cluster is chosen; otherwise, the longest trajectory from the $A[i]$-th cluster is chosen. This process continues until N_s trajectories are sampled.

Complexity analysis. Given K clusters and N_s samples, the time complexity of trajectory reallocation is $O(K)$ and the time complexity of sampling a trajectory is $O(1)$.

5.4 Error Analysis

By employing the evaluation optimization, the error incurred by approximate evaluation is small and bounded. Given a selected representative T^s and a trajectory T, the violation count of T derived by the system is approximated by that of T^s. The approximate error for a pair of T^s and T is defined as the difference between two violation counts, denoted by x. For a dataset of N trajectories to be assessed, N_s trajectory representatives are sampled from the dataset and form N pairs. The accumulative error is defined as the sum of approximate errors for all pairs, denoted by $S_{N_s} = \sum_{i=1}^{N} x_i$. We have the following error guarantee analysis:

THEOREM 1 (ERROR BOUND). *The probability that S_{N_s} exceeds a threshold ϵ is upper bounded by $Pr(S_{N_s} \geq \epsilon) \leq \frac{(N-N_s)(1-O/|T|)}{\epsilon}$.*

PROOF SKETCH. According to trajectory clustering, T and T^s have at least O grid cell overlaps. The approximate error x satisfies $0 \leq x \leq 1 - \frac{O}{|T|}$ (0 for the zero-error estimation, $1 - \frac{O}{|T|}$ for the case where the estimation is totally wrong in non-overlapping parts). For N_s representatives, they are accurately evaluated; for each of the remaining $(N - N_s)$ trajectories, the error is bounded by $1 - \frac{O}{|T|}$. Thus, by *Markov's Inequality*, we have,

$$Pr(S_{N_s} \geq \epsilon) \leq \frac{\mathbb{E}[S_{N_s}]}{\epsilon} \leq \frac{(N - N_s)(1 - O/|T|)}{\epsilon}$$

Table 2: Statistics of the datasets used in experiments.

Attributes	#trajectories	#GPS points	time range	sampling rate
T-drive	10,357	15,000,000	7 days	1~177 s
Rome	54,482	2,509,107	30 days	7 s
AIS	1,833	13,941,382	59 days	1~90 s

Table 3: The types and amounts of injected noise.

Noise type	**T-drive**				
	origin	addnoise-1	addnoise-2	addnoise-3	addnoise-4
Timestamp swap	0%	3%	9%	9%	3%
Point deletion	0%	3%	15%	9%	15%
Location shifts	0%	9%	0%	9%	15%
Total	0%	15%	24%	27%	33%

Noise type	**Rome**				
	origin	addnoise-1	addnoise-2	addnoise-3	addnoise-4
Timestamp swap	0%	1%	3%	3%	3%
Point deletion	0%	1%	5%	3%	5%
Location shifts	0%	3%	0%	3%	5%
Total	0%	5%	8%	9%	13%

□

6 EXPERIMENTS

In this section, we conduct experiments with the aim of answering the following research questions:

RQ1 Can T-Assess act as an indication of data quality for downstream data analysis tasks?

RQ2 Do the optimizations for T-Assess effectively realize the trade-off between effectiveness and efficiency?

RQ3 Do the parameters have significant impacts on T-Assess?

In the following, we present the experimental settings followed by answering the above research questions.

6.1 Experimental Setup

Datasets. The experiments are conducted on three real-world datasets, *i.e.*, T-drive [39], Rome [4], and AIS [3]. Statistics of the datasets are presented in Table 2.

- **T-drive** [39]. This dataset contains 10,357 trajectories of taxis during a period of 7 days.
- **Rome** [4]. This dataset contains 54,482 mobility traces of taxi cabs in Rome, Italy, collected over 30 days.
- **AIS** [3]. This dataset contains 1,833 trajectories of vessels during a two-month period.

Baselines. Deequ [29] and TsFile [33] are data quality assessment systems for general-purpose and time-series data, respectively, each of which can support a small set of quality metrics. For fair comparison, we combine them and propose a new data quality assessment system, termed STDeequ, to serve as a baseline. STDeequ supports timestamp constraint checking, missing value detection, and out-of-range value detection.

Parameter settings. By default, the number of samples N_s in the sample-based optimization is set to 50% of total number of trajectories in a dataset. The grid resolution parameter 2^{θ} is set to 32. The overlap threshold O is set to the average trajectory length of the grid-based trajectories. The length of the time window W for the online evaluation are set to 1 minute. All experiments concerning distributed computing are conducted on a cluster of 10 nodes. Each node is equipped with two 12-core processors (E5-2620v3 2.40GHz), 128GB RAM, CentOS 7, Spark 3.3.2, and Kafka 3.3.2.

Metrics. We quantify each quality dimension (except fairness) as the proportion of trajectories that do not contain violations of constraints that belong to the corresponding dimension. For fairness, we use spatial and temporal density to quantify the fairness of the dataset since they are already normalized within the unit range. To improve legibility, abbreviations are used for various metrics — Validity (VA), Completeness (CM), Consistency (CS), Spatial density (FA-S), and Temporal density (FA-T). For offline evaluation efficiency, we use the average evaluation time for a single trajectory as the metric; for the efficiency of online evaluations, we use the time it takes the system to process a data point from the moment it is received as the metric.

6.2 Indication Performance (RQ1)

To demonstrate the potential applications and efficacy of T-Assess, we conduct a comparative analysis with STDeequ as a trajectory data quality indicator for downstream tasks. Specifically, we employ trajectory similarity search and trajectory clustering as case studies, both of which play a fundamental role in various trajectory data analysis applications. We generate 5 variants of datasets by injecting varying amounts of noise and perform two trajectory similarity search and trajectory clustering algorithms (we choose ST2Vec [10] and E^{2}DTC [9] as they are SOTA in each task) on these perturbed datasets. The both of two downstream applications need the road network as one of inputs. We observe whether algorithm performance is affected in proportion to the data quality obtained from data quality assessment system. This aims to determine the extent to which the quality assessment system can function as a data quality indicator for downstream tasks.

Settings. To synthesize noisy datasets, we consider two kinds of noise: (1) data quality issues that can be detected by both bases-line STDeequ and T-Assess; (2) data quality issues that only can be detected by T-Assess. Regarding that STDeequ only supports three types of noise detection, we inject "easy-detected" noise by swapping timestamps and deleting GPS points, which can be detect by both methods. In addition, we also inject "hard-detected" noise by shifting locations, which can be only captured by T-Assess. To make the data quality evaluation more difficult, the amount of injected noise varies among datasets, instead of linearly increasing injected noise in the datasets. Note that, if we linearly increase inject noise of all types in the datasets, even a method can only detect one type of injected noise, it is very easy for this method to detect the decrement of data quality. Table 3 lists the types and the amounts of noise injected into the datasets.

Results. Table 4 presents the results of the effectiveness comparison between T-Assess and STDeequ. We rank the 5 variants of datasets based on the performance of ST2Vec, E^{2}DTC, and the data quality assessed by STDeequ and T-Assess, respectively. Subsequently, we calculate the Pearson correlation between ranks by algorithm performance and ranks by quality assessment systems. We find that

Table 4: Effectiveness as a trajectory data quality indicator for downstream tasks.

Datasets	Variants	Trajectory Similarity Search				Trajectory Clustering				Data Quality Assessment			
		ST2Vec [10]				E^{2}DTC [9]				STDeequ		T-Assess (ours)	
		HR@10	HR@50	R10@50	Rank	UACC	NMI	RI	Rank	Avg. score	Corr.	Avg. score	Corr.
T-Drive	origin	0.4916	0.6010	0.8397	(1)	0.3407	0.3424	0.1620	(1)	0.62 (1)	0.75	0.63 (1)	0.85
	addnoise-1	0.4624	0.5868	0.8361	(2)	0.3066	0.2858	0.1806	(2)	0.59 (2)		0.58 (2)	
	addnoise-2	0.4700	0.5807	0.8227	(4)	0.2953	0.2428	0.1267	(3)	0.49 (5)		0.54 (3)	
	addnoise-3	0.4442	0.5620	0.8016	(5)	0.2883	0.2476	0.1214	(4)	0.52 (4)		0.53 (4)	
	addnoise-4	0.4524	0.5772	0.8455	(3)	0.2747	0.2270	0.1093	(5)	0.53 (3)		0.52 (5)	
Rome	origin	0.4890	0.5961	0.8395	(1)	0.3696	0.4019	0.2309	(1)	0.95 (1)	0.65	0.90 (1)	0.95
	addnoise-1	0.4529	0.5762	0.8031	(2)	0.2853	0.3068	0.1453	(2)	0.94 (2)		0.89 (2)	
	addnoise-2	0.4445	0.5623	0.7998	(3)	0.3254	0.3333	0.1787	(4)	0.91 (5)		0.88 (3)	
	addnoise-3	0.4349	0.5529	0.7815	(4)	0.3274	0.3468	0.1879	(3)	0.91 (4)		0.87 (4)	
	addnoise-4	0.3834	0.5051	0.7221	(5)	0.3441	0.3637	0.2032	(5)	0.91 (3)		0.86 (5)	

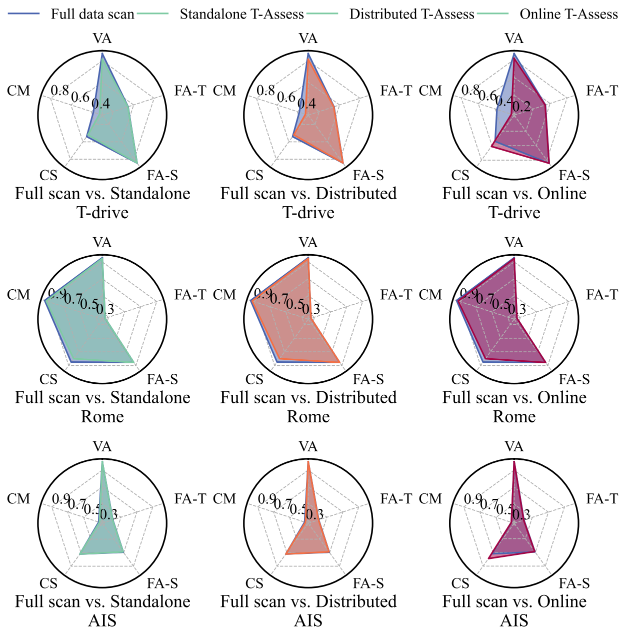


Figure 5: Effectiveness of T-Assess.

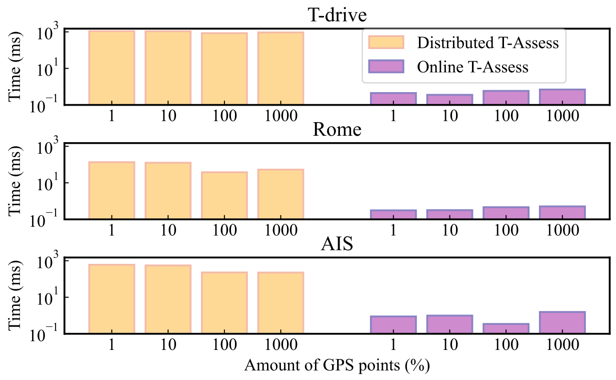


Figure 6: Scalability of T-Assess.

the correlation coefficients between ranks by algorithms performance and ranks by T-Assess get 0.85 and 0.95 on T-Drive and Rome, respectively, surpassing the correlation coefficients for STDeequ (0.75 and 0.65 on T-Drive and Rome respectively). These results demonstrate that T-Assess serves as a more accurate data quality indicator for downstream data analysis applications. This superiority is attributed to T-Assess providing much more comprehensive data quality measures compared to STDeequ.

6.3 Ablation Study (RQ2)

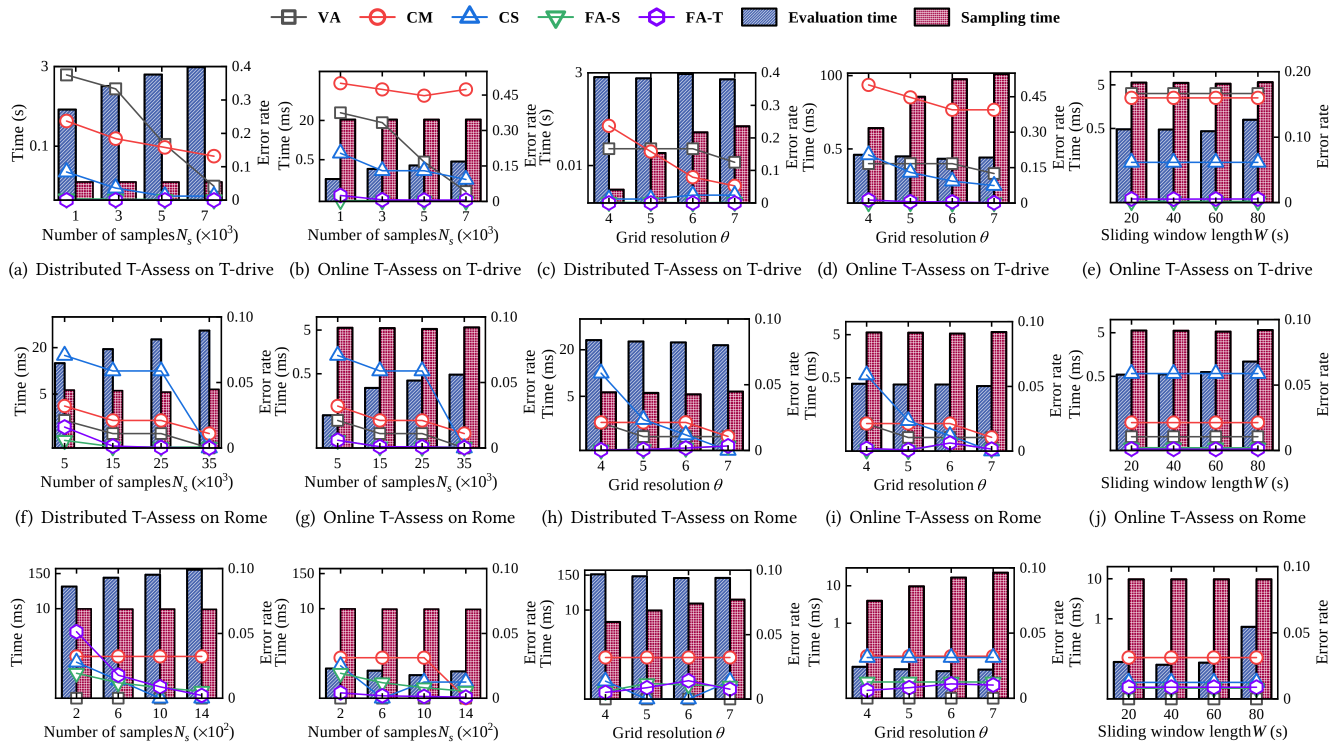
To demonstrate the effectiveness and efficiency of optimizations implemented in T-Assess, we compare the full-data-scan quality assessment (*i.e.*, the non-optimized solution) with three optimized variants of T-Assess: standalone T-Assess with sampling, distributed T-Assess, and online T-Assess.

Effectiveness. Figure 5 presents the evaluation results in four dimensions using different methods, represented by area coverage in radar plots. The area discrepancies of each method compared to the full data scan are also noted around subfigures. The areas covered by standalone T-Assess (green area), distributed T-Assess (orange area), and online T-Assess (red area) are all close to the area covered by full-data-scan (blue area), especially on Rome dataset, which verifies the effectiveness of our T-Assess. Note that, the online T-Assess performs not well in the completeness dimension on T-drive. This is because, only a small number of GPS points contained in each time window, so that it is hard to estimate the sampling period Δt accurately. In contrast, in Rome dataset, the sampling rate is relatively stable, which is very easy to estimate. In addition, there exists a slight discrepancy between the areas covered by standalone and distributed T-Assess. This is because, the distributed platform needs to partition the data set to enable parallelism, resulting in slightly different cluster results.

Efficiency. Table 5 lists the trajectory data quality evaluation time for each dimension that variant methods take. We observe that the full data scan takes 3.83s and 0.16s to perform data quality assessment for each trajectory on T-drive and Rome, which is the most expensive. Standalone T-Assess completes the evaluation in half the time taken by the full data scan on both datasets, which is attributed to the sampling-based optimization strategy that cuts down unnecessary evaluations. Owing to distributing data and

Table 5: Runtime comparison of variants of T-Assess in milliseconds (ms).

Mode	Methods	T-drive					Rome					AIS				
		VA	CM	CS	FA	Total	VA	CM	CS	FA	Total	VA	CM	CS	FA	Total
Offline	Full data scan	3.19	3.86	3803.22	15.54	3825.82	0.16	0.16	161.45	1.17	162.95	1.63	1.99	1260.98	0.38	1264.98
	Standalone T-Assess	2.22	2.12	2076.57	11.49	2092.40 (+187.8%)	0.15	0.09	64.41	0.64	65.29 (+249.6%)	0.79	1.36	665.41	0.23	667.79 (+189.4%)
	Distributed T-Assess	1.45	0.29	413.63	2.03	417.40 (+916.6%)	0.07	0.02	25.13	0.35	25.57 (+637.3%)	0.53	0.70	135.08	0.04	136.35 (+927.7%)
Online	Online T-Assess	0.001	0.001	0.24	0.19	0.43	0.001	0.001	0.18	0.16	0.35	0.001	0.002	0.06	0.001	0.06



(a) Distributed T-Assess on T-drive (b) Online T-Assess on T-drive (c) Distributed T-Assess on T-drive (d) Online T-Assess on T-drive (e) Online T-Assess on T-drive

(f) Distributed T-Assess on Rome (g) Online T-Assess on Rome (h) Distributed T-Assess on Rome (i) Online T-Assess on Rome (j) Online T-Assess on Rome

(k) Distributed T-Assess on AIS (l) Online T-Assess on AIS (m) Distributed T-Assess on AIS (n) Online T-Assess on AIS (o) Online T-Assess on AIS

Figure 7: Effectiveness and Efficiency vs. number of samples N_s, grid resolution 2^θ, and sliding window length W.

resources among different system components, distributed T-Assess achieves almost 9 times improvements in evaluation time on both T-drive and Rome datasets compared to the full data scan. Online T-Assess only takes less than a millisecond to evaluate data quality for each arriving GPS data point. The high efficiency is achieved by the proposed optimization strategy as well as the power of the distributed streaming computing platform. In terms of scalability, Figure 6 illustrates the scalability of the system by varying the size of trajectory data. System's runtime almost remains constant as the dataset size varies.

6.4 Sensitivity Study (RQ3)

The effectiveness and efficiency of T-Assess might be influenced by three key parameters: the number of samples N_s, grid resolution 2^θ, and the sliding window length W. In this experiment, we use the runtime to evaluate the efficiency, while employing the error rate to evaluate the effectiveness. Let $E^*_{fds}(\mathcal{T})$ denote the data quality in dimension $*$ obtained by the full data scan and $E^*_{N_s,\theta}(\mathcal{T})$ denote the data quality in dimension $*$ obtained by T-Assess with parameter N_s and θ. The error rate in the quality dimension $*$ is calculated as:

$$Error\ rate^* = \frac{|E^*_{fds}(\mathcal{T}) - E^*_{N_s,\theta}(\mathcal{T})|}{E^*_{fds}(\mathcal{T})} \tag{8}$$

The number of samples. We investigate the impact of the number of samples N_s on the performance of T-Assess. Figures 7a, 7b, 7f, and 7g show the results by varying N_s from 1000 to 7000. As observed, for all quality dimensions, the error rates decrease with an increase of N_s. This decrease is attributed to T-Assess with the sampling-based optimization becoming closer to the full data scan as N_s increases. In addition, the runtime of the quality evaluation increases as N_s grows, while the runtime of the sampling remains almost constant, which further validates the $O(1)$ complexity of the trajectory representative selection strategy.

Grid resolution. To explore the impact of the grid resolution 2^θ on the performance of T-Assess, we vary the grid resolution θ from 4 to 7, resulting in the number of grid cells ranging from 32×32 to 128×128. Figures 7c, 7d, 7h, and 7i illustrate the corresponding results. As observed, the error rates in almost all quality dimensions drop and the running time of the sampling period increases with the growth of θ. This is because, grid-based trajectories become more precise with a larger resolution, resulting in a higher trajectory clustering accuracy but a longer clustering time. Meanwhile, the evaluation time remains constant regardless of the resolution. This is because the number of trajectories used for quality evaluations remains the same no matter how the resolution varies.

Sliding window length. We explore the impact of the length of the sliding window W on the performance of online T-Assess. Figures 7e and 7j depict the results of T-Assess with the length of the sliding window W varying from 20s to 80s. As observed, in terms of the effectiveness, the assessment results of online T-Assess remains stable; in terms of the efficiency, the running time increases with the growth of W. This is because the number of data points within a sliding window increases, resulting in longer processing time to perform constraint checks in the sliding window. Note that, the time complexity of some constraint checks (*e.g.*, the smoothness constraint) is superlinear with the number of data points.

7 RELATED WORK

The data-driven applications are ubiquitous, and their effectiveness depends on high-quality data. Thus, data quality assessment has attracted much attention in both industry and academia. Existing data quality assessment studies can be classified into two categories: task-agnostic data quality assessment [7, 24, 28, 29, 31, 33] and task-aware data quality assessment [6, 12, 13, 27].

Task-agnostic data quality assessment. Task-agnostic data quality assessment uses data characteristics as a means of quantifying different quality dimensions. For example, Deequ [29] is an automated data verification system that utilizes Spark to perform data quality checks (*i.e.*, find violations to constraints) on large datasets. Its extension [28] provides an optimization to reduce the system's overall runtime by incrementally performing data verification on ever-changing data. TsQuality [24] is a system for evaluating time-series data quality, where four basic time-series data quality measures are implemented as functions or operators. The above studies offer basic metrics for providing general-purpose data quality assessments and fall short in providing a sufficiently informative indication of the quality of specific types of data.

Task-aware data quality assessment. Task-aware data quality assessment allows data quality to be considered in the context of the target tasks, especially machine learning tasks. For example, Snoopy [27] is an automatic feasible study for machine learning via estimating irreducible errors that stem from data quality issues in datasets. The F4U system [12] estimates the correlation between dataset quality and analysis quality, which is regarded as a quality metric. Following the F4U system, Budach et al. [5] empirically explore the relationship between more data quality dimensions and the performance of more machine learning algorithms covering different analysis tasks. Although task-aware data quality assessment is more indicative of how data quality affects analysis tasks, it is designed for specific tasks and not generally applicable.

8 CONCLUSION

We present T-Assess, an efficient trajectory data quality assessment system with three salient features. First, the system supports constraints spanning four dimensions. Second, the system supports both offline and online evaluations for batch trajectory datasets and trajectory streams. Third, the system features an evaluation optimization strategy to improve evaluation efficiency. Extensive experiments demonstrate the effectiveness and efficiency of T-Assess. In the future, we plan to incorporate deep learning techniques into data quality assessments to learn more complicated features in trajectory data.

REFERENCES

[1] 2012. Apache Kafka. https://kafka.apache.org.
[2] 2014. Apache Flink. http://flink.apache.org.
[3] 2020. AIS Project. https://marinecadastre.gov/ais.
[4] 2022. CRAWDAD roma/taxi. https://dx.doi.org/10.15783/C7QC7M.
[5] Lukas Budach, Moritz Feuerpfeil, Nina Ihde, Andrea Nathansen, Nele Sina Noack, Hendrik Patzlaff, Hazar Harmouch, and Felix Naumann. 2022. The Effects of Data Quality on ML-Model Performance. *CoRR* abs/2207.14529 (2022).
[6] Emily Caveness, Paul Suganthan G. C., Zhuo Peng, Neoklis Polyzotis, Sudip Roy, and Martin Zinkevich. 2020. TensorFlow Data Validation: Data Analysis and Validation in Continuous ML Pipelines. In *SIGMOD*. 2793–2796.
[7] Gao Cong, Wenfei Fan, Floris Geerts, Xibei Jia, and Shuai Ma. 2007. Improving Data Quality: Consistency and Accuracy. In *VLDB*. 315–326.
[8] Chenglong Fang, Feng Wang, Bin Yao, and Jianqiu Xu. 2022. GPSClean: A Framework for Cleaning and Repairing GPS Data. *ACM Trans. Intell. Syst. Technol.* 13, 3 (2022), 40:1–40:22.
[9] Ziquan Fang, Yuntao Du, Lu Chen, Yujia Hu, Yunjun Gao, and Gang Chen. 2021. E^2DTC: An End to End Deep Trajectory Clustering Framework via Self-Training. In *ICDE*. 696–707.
[10] Ziquan Fang, Yuntao Du, Xinjun Zhu, Danlei Hu, Lu Chen, Yunjun Gao, and Christian S. Jensen. 2022. Spatio-Temporal Trajectory Similarity Learning in Road Networks. In *KDD*. 347–356.
[11] Ziquan Fang, Shenghao Gong, Lu Chen, Jiachen Xu, Yunjun Gao, and Christian S. Jensen. 2023. Ghost: A General Framework for High-Performance Online Similarity Queries over Distributed Trajectory Streams. *Proc. ACM Manag. Data* 1, 2 (2023), 173:1–173:25.
[12] Daniele Foroni, Matteo Lissandrini, and Yannis Velegrakis. 2021. Estimating the extent of the effects of Data Quality through Observations. In *ICDE*. 1913–1918.
[13] Daniele Foroni, Matteo Lissandrini, and Yannis Velegrakis. 2021. The F4U System for Understanding the Effects of Data Quality. In *ICDE*. 2717–2720.
[14] Junhao Gan and Yufei Tao. 2017. Dynamic Density Based Clustering. In *SIGMOD*. 1493–1507.
[15] Chong Yang Goh, Justin Dauwels, Nikola Mitrovic, Muhammad Tayyab Asif, Ali Oran, and Patrick Jaillet. 2012. Online map-matching based on Hidden Markov model for real-time traffic sensing applications. In *ITSC*. 776–781.
[16] Usman Gohar. 2019. Scalable Techniques for Trajectory Outlier Detection.
[17] Danlei Hu, Lu Chen, Hanxi Fang, Ziquan Fang, Tianyi Li, and Yunjun Gao. 2024. Spatio-Temporal Trajectory Similarity Measures: A Comprehensive Survey and Quantitative Study. *IEEE Trans. Knowl. Data Eng.* 36, 5 (2024), 2191–2212.
[18] Jae-Gil Lee, Jiawei Han, and Xiaolei Li. 2008. Trajectory Outlier Detection: A Partition-and-Detect Framework. In *ICDE*. 140–149.
[19] Huan Li, Hua Lu, Christian S. Jensen, Bo Tang, and Muhammad Aamir Cheema. 2023. Spatial Data Quality in the Internet of Things: Management, Exploitation, and Prospects. *ACM Comput. Surv.* 55, 3 (2023), 57:1–57:41.
[20] Huan Li, Bo Tang, Hua Lu, Muhammad Aamir Cheema, and Christian S. Jensen. 2022. Spatial Data Quality in the IoT Era: Management and Exploitation. In *SIGMOD*. 2474–2482.
[21] Xiao Li, Huan Li, Harry Kai-Ho Chan, Hua Lu, and Christian S. Jensen. 2023. Data Imputation for Sparse Radio Maps in Indoor Positioning. In *ICDE*. 2235–2248.
[22] Jiali Mao, Tao Wang, Cheqing Jin, and Aoying Zhou. 2017. Feature Grouping-Based Outlier Detection Upon Streaming Trajectories. *IEEE Trans. Knowl. Data Eng.* 29, 12 (2017), 2696–2709.

[23] Duong Nguyen, Rodolphe Vadaine, Guillaume Hajduch, René Garello, and Ronan Fablet. 2022. GeoTrackNet - A Maritime Anomaly Detector Using Probabilistic Neural Network Representation of AIS Tracks and A Contrario Detection. *IEEE Trans. Intell. Transp. Syst.* 23, 6 (2022), 5655–5667.

[24] Yuanhui Qiu, Chenguang Fang, Shaoxu Song, Xiangdong Huang, Chen Wang, and Jianmin Wang. 2023. TsQuality: Measuring Time Series Data Quality in Apache IoTDB. *Proc. VLDB Endow.* 16, 12 (2023), 3982–3985.

[25] Xuan Rao, Lisi Chen, Yong Liu, Shuo Shang, Bin Yao, and Peng Han. 2022. Graph-Flashback Network for Next Location Recommendation. In *KDD*. 1463–1471.

[26] M. Mazhar Rathore, Awais Ahmad, Anand Paul, and Seungmin Rho. 2016. Urban planning and building smart cities based on the Internet of Things using Big Data analytics. *Comput. Networks* 101 (2016), 63–80.

[27] Cédric Renggli, Luka Rimanic, Luka Kolar, Wentao Wu, and Ce Zhang. 2023. Automatic Feasibility Study via Data Quality Analysis for ML: A Case-Study on Label Noise. In *ICDE*. 218–231.

[28] Sebastian Schelter, Stefan Grafberger, Philipp Schmidt, Tammo Rukat, Mario Kießling, Andrey Taptunov, Felix Bießmann, and Dustin Lange. 2019. Differential Data Quality Verification on Partitioned Data. In *ICDE*. 1940–1945.

[29] Sebastian Schelter, Dustin Lange, Philipp Schmidt, Meltem Celikel, Felix Bießmann, and Andreas Grafberger. 2018. Automating Large-Scale Data Quality Verification. *Proc. VLDB Endow.* 11, 12 (2018), 1781–1794.

[30] Raunak Shah, Koyel Mukherjee, Atharv Tyagi, Sai Keerthana Karnam, Dhruv Joshi, Shivam Pravin Bhosale, and Subrata Mitra. 2023. R2D2: Reducing Redundancy and Duplication in Data Lakes. *Proc. ACM Manag. Data* 1, 4 (2023), 268:1–268:25.

[31] Phanwadee Sinthong, Dhaval Patel, Nianjun Zhou, Shrey Shrivastava, Arun Iyengar, and Anuradha Bhamidipaty. 2021. DQDF: Data-Quality-Aware Dataframes. *Proc. VLDB Endow.* 15, 4 (2021), 949–957.

[32] Shaoxu Song and Aoqian Zhang. 2020. IoT Data Quality. In *CIKM*. 3517–3518.

[33] Yunxiang Su, Yikun Gong, and Shaoxu Song. 2023. Time Series Data Validity. *Proc. ACM Manag. Data* 1, 1 (2023), 85:1–85:26.

[34] Le Hung Tran, Quoc Viet Hung Nguyen, Ngoc Hoan Do, and Zhixian Yan. 2011. *Robust and hierarchical stop discovery in sparse and diverse trajectories.* Technical Report.

[35] Antonio Virdis, Giovanni Stea, and Gianluca Dini. 2021. SAPIENT: Enabling Real-Time Monitoring and Control in the Future Communication Infrastructure of Air Traffic Management. *IEEE Trans. Intell. Transp. Syst.* 22, 8 (2021), 4864–4875.

[36] Alastair J. Walker. 1977. An Efficient Method for Generating Discrete Random Variables with General Distributions. *ACM Trans. Math. Softw.* 3, 3 (1977), 253–256.

[37] Xinfeng Wang, Fumiyo Fukumoto, Jin Cui, Yoshimi Suzuki, Jiyi Li, and Dongjin Yu. 2023. EEDN: Enhanced Encoder-Decoder Network with Local and Global Context Learning for POI Recommendation. In *SIGIR*. 383–392.

[38] Yanwei Yu, Lei Cao, Elke A. Rundensteiner, and Qin Wang. 2014. Detecting moving object outliers in massive-scale trajectory streams. In *KDD*. 422–431.

[39] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. 2011. Driving with knowledge from the physical world. In *KDD*. 316–324.

[40] Yu Zheng. 2015. Trajectory Data Mining: An Overview. *ACM Trans. Intell. Syst. Technol.* 6, 3 (2015), 29:1–29:41.