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 & Zhejiang Key Laboratory of Big Data Intelligent Computing
zqfang@zju.edu.cn

Lu Chen, Yunjun Gao
Zhejiang University & Zhejiang Key Laboratory of Big Data Intelligent Computing
{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 four real-life benchmark datasets offer insight into the effectiveness of T-Assess at quantifying trajectory data quality beyond the capabilities of 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, 18(3): 666 - 674, 2024.
doi:10.14778/3712221.3712233

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

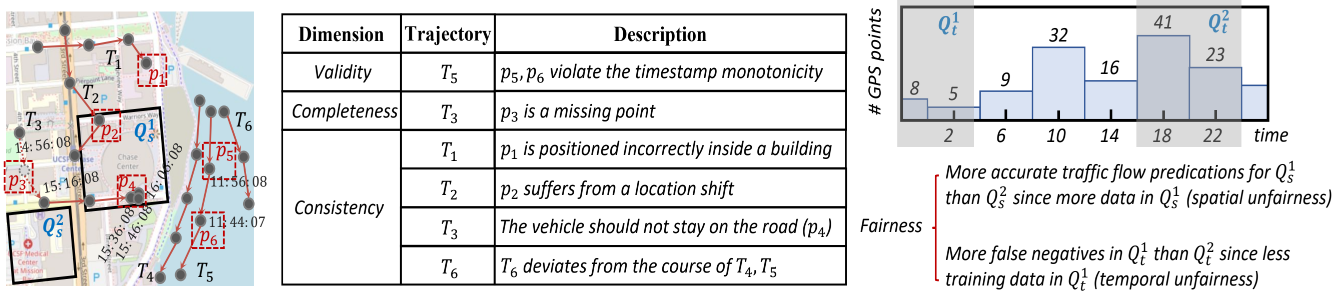
1 INTRODUCTION

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 [33], as well as consumer services such as POI recommendation [32, 46], 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 [10, 26, 27, 36, 39] focus on the quantification of data quality. First, task-agnostic data quality assessment [10, 31, 35, 36, 38, 41] leverages data characteristics to quantify various quality dimensions. Second, task-aware data quality assessment [8, 21, 22, 34] 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:

- ***Failure 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.
- ***Failure 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 [31, 41] focus on detecting errors within a single univariate time-series, lacking the capability to consider inter-dependencies.
- ***Failure 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

Proceedings of the VLDB Endowment, Vol. 18, No. 3 ISSN 2150-8097.
doi:10.14778/3712221.3712233



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

Fairness

More accurate traffic flow predications for Q_s^1 *than* Q_s^2 *since more data in* Q_s^1 *(spatial unfairness)*

More false negatives in Q_t^1 *than* Q_t^2 *since less training data in* Q_t^1 *(temporal unfairness)*

Figure 1: Examples of common types of erroneous trajectories.

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 unable to capture underlying relations between topographical contexts and trajectories.

In addition, spatio-temporal data is 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 [36] 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 [33], POI recommendations [32, 46]) and online trajectory analyses (*e.g.*, real-time congestion monitoring [42], vessel monitoring [30]). 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, 50]. 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 this 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 our knowledge, this is the first study that targets the evaluation of trajectory data quality.
- We provide an extensive 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 small error margins.
- Extensive experiments offer insight into the effectiveness and efficiency of T-Assess, including its optimization.

2 PRELIMINARIES

2.1 Trajectory Data Models

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

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 $T = \langle p_1, p_2, \ldots \rangle$. 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 easy to obtain and is a crucial asset for detecting moving patterns in trajectory data [18, 26]. Therefore, if available, the road network is used as

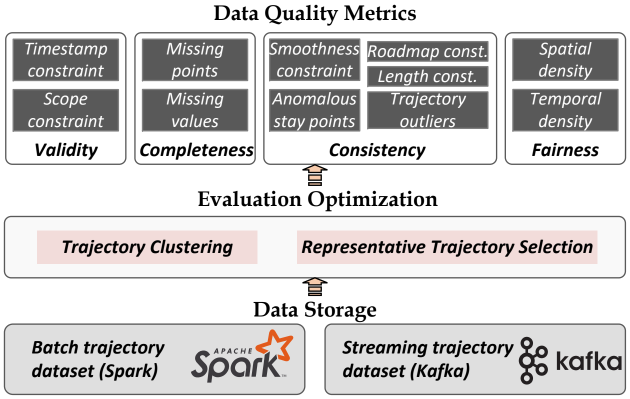


Figure 2: System architecture of T-Assess.

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 metrics (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 metrics.*

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 [19, 48], 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 metrics, 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 metrics.*

3 T-ASSESS ARCHITECTURE

To enable extensive, versatile, and efficient trajectory data quality assessment, the T-Assess automatic data quality assessment system adopts the architecture shown in Figure 2. T-Assess employs the distributed platforms Spark [2] and Kafka [1] for batch and streaming trajectory datasets, respectively. Both leverage parallel computing to facilitate high-performance processing.

Even with the advantages of distributed computing, efficiency challenges arise when intricate constraint checks are involved. For 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 representative trajectory selection. The details are covered in Section 5. This way, T-Assess performs constraint checks on trajectory representatives rather than the entire dataset, which enables a trade-off between correctness and efficiency.

In the subsequent layer, T-Assess exposes a user-facing API that enables the formulation of different categories of data quality metrics for trajectory data. Considering the unique characteristics of trajectories and potential downstream analyses, the proposed data quality metrics span four dimensions, to be detailed in Section 4. The metrics in T-Assess are encapsulated into user-defined functions (UDFs) for ease of use, and evaluations are performed through UDF calls. To modify existing metrics or add new metrics, users simply need to update or add the respective UDFs, making the system flexible and adaptable to new requirements.

4 DATA QUALITY EVALUATION

We first motivate and describe the detailed data quality dimensions, and then we present detailed offline and online evaluations.

4.1 Data Quality Dimensions

4.1.1 ***Motivation.*** We collected quality metrics based on three recent comprehensive surveys [26, 27, 39] of data quality issues in spatio-temporal data. The resulting quality metrics are grouped into four categories – *validity*, *completeness*, *consistency*, and *fairness* – which are commonly identified as essential in the surveys. To develop the quality metric system, we further reviewed recent studies and identified the data quality issues that impact data mining tasks the most to ensure that corresponding quality metrics are included in T-Assess. Due to the growing use of deep learning, we include the data quality dimension, *fairness*, which is often overlooked in existing surveys. Metrics in these dimensions consider both the trajectory characteristics and potential impacts on applications, addressing the "**failures**" 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 "failure"** 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 [16, 31, 36, 39] and can guide for data cleaning (*e.g.*, position imputation [28]). 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

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	$(p_i.x \neq \text{NULL}) \wedge (p_i.y \neq \text{NULL})$.
Dimension *Consistency*	
Smoothness constraint	Following the previous definition [51], 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 [25, 29], 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	OHMM [23] is employed to find the correspondence between each point to a road segment in the road network. Once the correspondence does not exist, the point and the corresponding trajectory violates the road network constraint.
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 number of GPS points in dataset, 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.

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 "failure"**), *length constraint* and *trajectory outliers* (addressing the **second "failure"**), *anomalous stay points* and *road network constraint* (addressing the **third "failure"**). 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, 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, T_3 has a set of stay points (*e.g.*, p_4) 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. Detailed evaluations of these data quality metrics are provided in the extended technical report [5]. 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. 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 evaluations. Moreover, to accommodate dynamic traffic conditions (*e.g.*, road closures), T-Assess can be extended to integrate traffic-aware techniques.

5 EVALUATION OPTIMIZATION

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

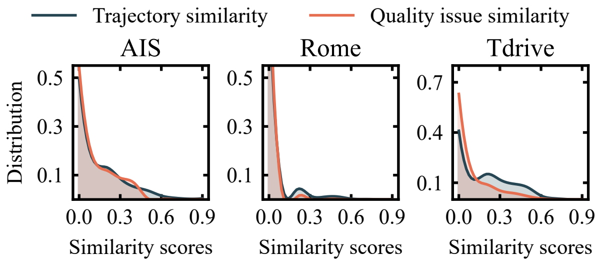


Figure 3: The score distribution of pairwise trajectory and data quality issue similarity.

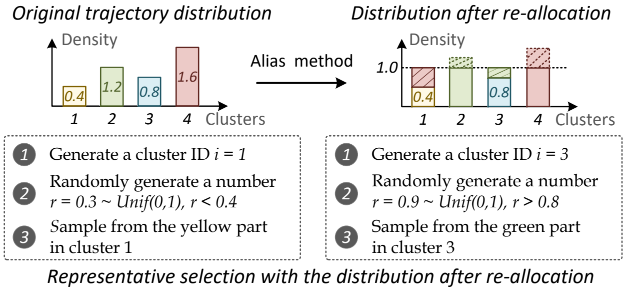

Original trajectory distribution
Distribution after re-allocation
Density
0.4
1.2
0.8
1.6
1 2 3 4 Clusters
Alias method
Density
1.0
0.4
0.8
1 2 3 4 Clusters
1 Generate a cluster ID i = 1
2 Randomly generate a number r = 0.3 ~ Unif(0,1), r < 0.4
3 Sample from the yellow part in cluster 1
1 Generate a cluster ID i = 3
2 Randomly generate a number r = 0.9 ~ Unif(0,1), r > 0.8
3 Sample from the green part in cluster 3
Representative selection with the distribution after re-allocation

Figure 4: Toy example illustrating the representative trajectory selection strategy.

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 issues.* To support this intuition, we compute trajectory similarity and the similarity of data quality issues for all trajectory pairs in the three datasets and visualize the distributions of the two types of similarity scores (detailed settings can be found in the technical report [5]). Figure 3 shows that the distributions of pairwise trajectory similarity and quality issue similarity are similar, thus supporting our intuition. The core challenges are to *identify spatially similar trajectories* and to *select 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 [24]. 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, making them computationally expensive [37]. Consequently, we propose a more efficient approach to generating trajectory clusters that avoids complex feature engineering.

5.2.2 ***Design.*** To improve the efficiency of trajectory clustering, we convert free space trajectories into grid space-represented trajectories. Specifically, given a two-dimensional space, we construct a grid G that partitions the space into $2^\theta \times 2^\theta$ equal-sized grid cells, where 2^θ is 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. Given a grid-based trajectory dataset, to eliminate the impact of the visiting order, we first sort the trajectories in descending order of their length. Then we initialize an empty list that stores the centers of each cluster (we regard the longest trajectory in a 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 in common with cluster center c, we add it to the corresponding cluster 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. The pseudocode can be found in the technical report [5].

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

5.3 Representative Trajectory 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 improve efficiency considerably. However, randomly selecting trajectories from clusters would severely distort the trajectory distribution of the dataset, rendering the quality assessment ineffective. Instead, we propose a representative trajectory selection strategy that can preserve the trajectory distribution as well as ensure high performance of the selection process. We provide a visualization of how the proposed selection strategy preserve the trajectory distribution as compared to other strategies in the accompanying technical report [5].

5.3.2 ***Design.*** It is computationally expensive to sample trajectories w.r.t. a particular non-uniform distribution ($O(KN_s)$ computation for N_s samples). To overcome this, we leverage the alias method [43] to convert sampling trajectories w.r.t. a non-uniform distribution into constant-time sampling from a uniform distribution. The core idea is to equalize the cluster sizes through trajectory reallocation without altering the actual trajectory distribution.

Suppose that k clusters, $C = \{C_1, C_2, \cdots, C_K\}$, are created through trajectory clustering, and that an identifier is assigned to each trajectory that indicates the trajectory it belongs to. For instance, Figure 4 shows four trajectory clusters, with the trajectories in the yellow cluster assigned ID 1 and those in the red cluster assigned ID 4.

Step 1. Normalization of cluster sizes. The sizes of each cluster is normalized using the formula $\frac{|C_i|K}{\sum_{j=1}^{K}|C_j|}$, so the average size becomes 1. In Figure 4, the normalized sizes of the clusters are 0.4, 1.2, 0.8, 1.6, respectively, with an overall average of 1.

Step 2. Reallocation of excess trajectories. Trajectories from larger clusters are reallocated to smaller clusters to obtain equal cluster sizes. Figure 4 provides an example of this reallocation process: Cluster 4 (the red block) "donates" trajectories from its excess portion (the dashed red section) to cluster 1 (the yellow block). To track the reallocation and support the following selective sampling, we use a dictionary to record "donor-recipient" relationships (*i.e.*, the IDs of donors as keys and the IDs of recipients as values), a queue to store IDs of clusters whose normalized sizes are greater than 1 (*i.e.*, donors), and another queue to store the IDs of smaller clusters (*i.e.*, recipients).

Step 3. Selective sampling. The sampling process involves two steps: (1) First, a cluster is selected randomly based on its ID; (2) Then, whether a trajectory is selected from the original or "donated" section of the cluster depends on if the random number is less than the normalized cluster size. This redirection maintains the overall trajectory distribution. Figure 4 illustrates two cases: in the first, cluster 1 is chosen, and a random number $r = 0.3 < 0.4$ leads to selecting a trajectory from its original section (*i.e.*, the yellow part); in the second case, cluster 3 is randomly selected, and a random number $r = 0.9 > 0.8$ leads to the selection from the reallocated portion in the cluster 3 (*i.e.*, the green part of cluster 3).

The pseudocode for the trajectory reallocation and sampling process is present in the technical report [5].

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 $Pr(S_{N_s} \geq \epsilon) \leq \frac{(N-N_s)(1-O/|T|)}{\epsilon}$.*

The proof can be found in the extended version [5].

6 EXPERIMENTS

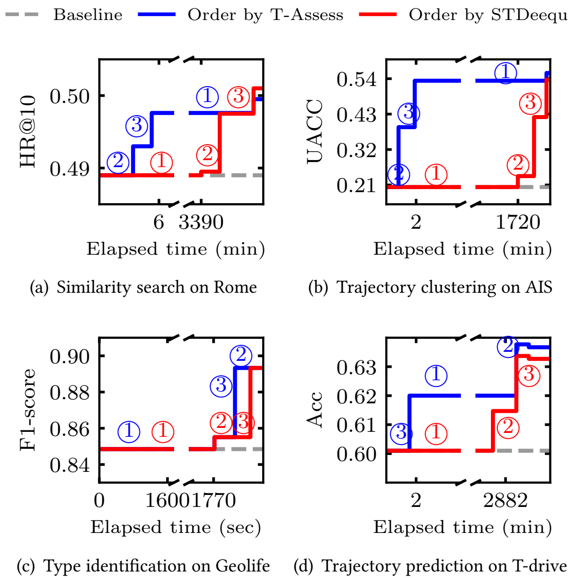
In this section, we conduct experiments with the aim of answering the following research questions:

RQ1 How does T-Assess assist downstream data cleaning and subsequent data mining tasks?
RQ2 Does T-Assess outperform the off-the-shelf competitors?
RQ3 How do parameter settings affect T-Assess?

6.1 Experimental Setup

Datasets. The experiments are conducted on four real-world datasets: T-drive [50], Rome [4], AIS [3], and Geolife [52]. The statistics of datasets are available in the technical report [5].

Baselines. Deequ [36] and TsFile [41] 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.



(a) Similarity search on Rome

(b) Trajectory clustering on AIS

(c) Type identification on Geolife

(d) Trajectory prediction on T-drive

Figure 5: Case study of how T-Assess assists trajectory data cleaning and trajectory data mining tasks.

Parameter settings. The implementation details of T-Assess are presented in an extended technical report [5].

Evaluation 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 datasets. To improve legibility, abbreviations are used for 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 Case Study: As an Advisor (RQ1)

Datasets usually contain quality issues of varying severity, reflected in different violation counts for each quality metric. It is more efficient to target the most severe issues first rather than applying all cleaning operations indiscriminately. The type of quality assessment enabled by T-Assess helps identify the most severe quality issues are most critical enabling efficient data cleaning strategies.

In this case study, we apply three trajectory cleaning operations on the datasets: ① *data imputation* [14] to impute missing data, ② *trajectory calibration* [40] to correct trajectory outliers, and ③ *trajectory segmentation* [15] to remove location shifts. The execution order of these cleaning operations is determined by the violation counts provided by T-Assess, thereby prioritizing operations that address the most severe quality issues. We measure both the runtimes of the data cleaning and the resulting improvements

Table 2: Runtime of components in T-Assess in milliseconds.

Datasets		**T-drive**	**Rome**	**AIS**	**Geolife**
Full data scan		3825.82	162.95	1264.98	231.28
Optimization	Clustering	21.62	5.30	9.65	2.05
	Selection	0.0013	0.0023	0.0129	0.0018
Evaluation	Distributed T-Assess	417.40	25.57	136.35	31.90
	Online T-Assess	0.43	0.35	0.06	0.21

in downstream model performance. The downstream trajectory data mining methods include: ST2Vec [18] for similarity search, E2DTC [17] for trajectory clustering, SECA [12] for transportation type identification, and MetaPTP [47] for trajectory prediction.

In Figure 5, the numbers (i.e., ①, ②, ③) indicate the data cleaning operation used. Compared to STDeequ, it takes less time to achieve the same performance improvements on a dataset cleaned in the order suggested by T-Assess. This is because T-Assess considers a broader range of quality metrics, enabling it to effectively determine the most impactful data cleaning sequence. For example, in Figure 5a, T-Assess recommends prioritizing trajectory calibration due to the large number of location shifts in the dataset, while STDeequ suggests performing data imputation first, as it cannot detect location shifts. We observe that trajectory calibration, as recommended by T-Assess, results in a higher-quality dataset in 4 minutes, allowing ST2Vec to performs better, while the data imputation suggested by STDeequ is still ongoing.

6.3 System Design Comparisons (RQ2)

Time breakdown analysis. Table 2 provides a detailed runtime analysis of key optimization components (including trajectory clustering and representative selection) and other system components. The results indicate that the overhead introduced by these optimizations of clustering and selection is minimal compared to the overall evaluation time. Further, the total runtime of T-Assess is approximately 9× shorter than that of a full data scan, demonstrating the efficiency of our approach.

Evaluation optimization comparison. For the component of trajectory clustering, we compare our method with three methods:

- Dynamic variant (DV). A variable grid-based method, where lower-density grids are merged into larger cells.
- TrajStore (TS). The trajectory clustering component used in the TrajStore system [11].
- E2DTC. A deep learning based trajectory clustering method [17].

For trajectory selection, we compare our alias method (abbr. A) with random selection (abbr. R). This results in a total of 8 combinations (4 clustering methods × 2 selection methods). We use the runtime to evaluate the efficiency, and we employ the average 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:

$$Error\ rate^* = \frac{|E^*_{fds}(\mathcal{T}) - E^*_{N_s,\theta}(\mathcal{T})|}{E^*_{fds}(\mathcal{T})} \tag{1}$$

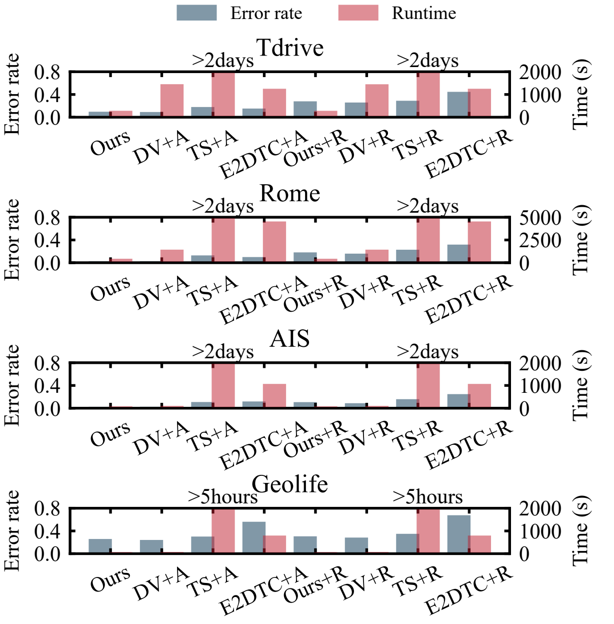


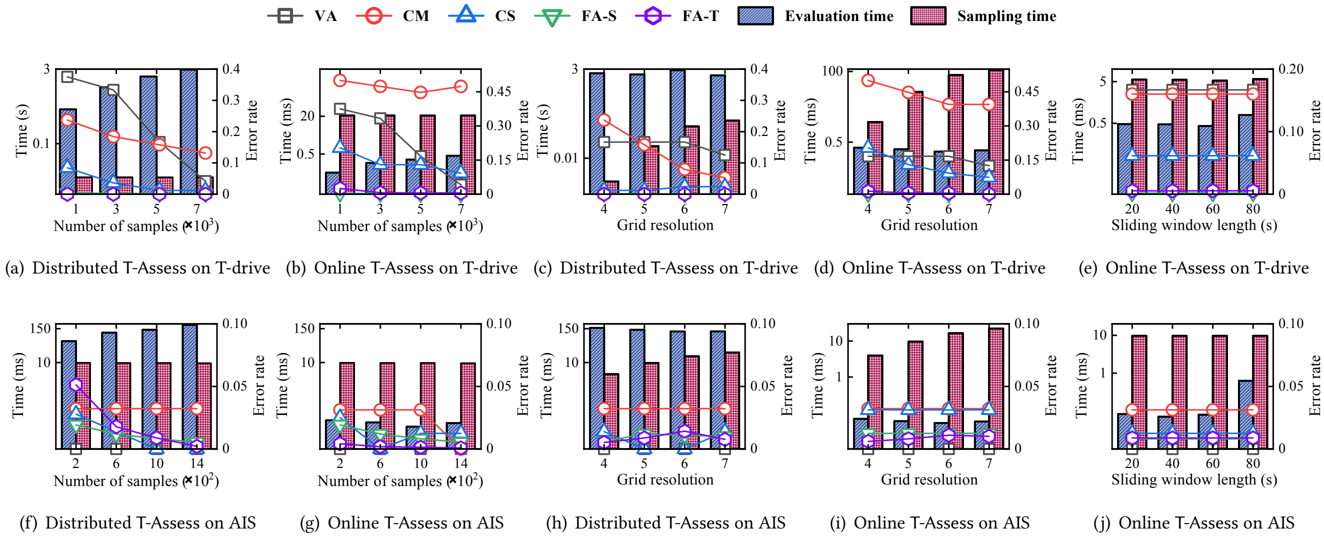
Figure 6: Comparison of trajectory clustering and selection.

Figure 6 shows that our approach outperforms the competitors in terms of both assessment accuracy and efficiency. While the dynamic variant of our method achieves lower error rates, it requires more computation time, particularly with large datasets like Tdrive and Rome. The clustering method in TrajStore performs poorly due to its reliance on a continuous trajectory similarity threshold, which is challenging to fine-tune given its extremely large search space. Additionally, it has high computational cost (*e.g.*, taking over 2 days on Tdrive). E2DTC is worse than non-learning clustering approaches since it is trained on labeled data annotated by a non-learning clustering approach. Further, its process of representing trajectories by vectors incurs high computational overhead as the dataset size grows. Compared to the alias method used in T-Assess, random trajectory selection sacrifices assessment accuracy for a marginal reduction in runtime.

6.4 Sensitivity Study (RQ3)

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. Due to the space limitation, only T-drive and AIS results are presented here and more can be found in our technical report [5].

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, 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 aligns with the $O(1)$ complexity of the representative trajectory selection strategy.



(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 AIS (g) Online T-Assess on AIS (h) Distributed T-Assess on AIS (i) Online T-Assess on AIS (j) 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.

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 show the corresponding results. 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 accurate with a larger resolution, resulting in a higher trajectory clustering accuracy but also a longer clustering time. The evaluation time remains constant regardless of the resolution. This is because the number of trajectories used for quality evaluations remains the same regardless of the resolution.

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 when varying the length of the sliding window W from 20s to 80s. 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 in the number of data points.

7 RELATED WORK

Data Quality Assessment. Task-agnostic data quality assessment [10, 31, 35, 36, 38, 41] uses data characteristics (*e.g.*, data distributions) as a means of quantifying different quality aspects of the data. However, Those works only offer basic metrics for data quality assessments that do not take into account the specifics of the type of data considered, and fall short in providing sufficiently informative the quality assessments. Task-aware data quality assessment studies [8, 21, 22, 34] allow data quality to be considered in the context of particular tasks, typically machine learning tasks. Although task-aware data quality assessment is more indicative of how data quality affects analysis tasks, this type of assessment is designed for specific tasks and is not generally applicable.

Trajectory Data Management System. Several trajectory management systems exist, including TrajStore [11], UlTraMan [13], and Torch [45], which are designed to support efficient trajectory query processing. A key component of trajectory data management systems is trajectory cleaning, which involves operations such as trajectory segmentation [7], trajectory calibration [40], and trajectory enrichment [6]. Such operations are essential for noise removal but tend to focus on specific types of noise, lacking a holistic approach to assess and improve overall data quality comprehensively.

Trajectory Data Applications. A range of applications involve trajectory data, as highlighted in recent surveys [9, 44]. Most prior work on common trajectory applications, like travel time estimation [49], and traffic prediction [20], focuses on innovating in model architectures. In contrast, our study takes a data-centric approach, focusing on detecting and addressing data quality issues within trajectory data to improve downstream application performance.

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.

ACKNOWLEDGMENTS

This work was supported in part by the NSFC under Grants No. (62472377, 62025206, 62402422, U23A20296), and Zhejiang Province's "Lingyan" R&D Project under Grant No. 2024C01259. Lu Chen is the corresponding author of the work.

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] 2024. T-Assess: full version. https://github.com/ZJU-DAILY/T-Assess/blob/main/technical_report.pdf.

[6] Luis Otávio Alvares, Vania Bogorny, Bart Kuijpers, José Antônio Fernandes de Macêdo, Bart Moelans, and Alejandro A. Vaisman. 2007. A model for enriching trajectories with semantic geographical information. In *SIGSPATIAL*. 22:1–22:8.

[7] Maike Buchin, Anne Driemel, Marc J. van Kreveld, and Vera Sacristán. 2011. Segmenting trajectories: A framework and algorithms using spatiotemporal criteria. *J. Spatial Inf. Sci.* 3, 1 (2011), 33–63.

[8] 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.

[9] Wei Chen, Yuxuan Liang, Yuanshao Zhu, Yanchuan Chang, Kang Luo, Haomin Wen, Lei Li, Yanwei Yu, Qingsong Wen, Chao Chen, Kai Zheng, Yunjun Gao, Xiaofang Zhou, and Yu Zheng. 2024. Deep Learning for Trajectory Data Management and Mining: A Survey and Beyond. *CoRR* abs/2403.14151 (2024).

[10] Gao Cong, Wenfei Fan, Floris Geerts, Xibei Jia, and Shuai Ma. 2007. Improving Data Quality: Consistency and Accuracy. In *VLDB*. 315–326.

[11] Philippe Cudré-Mauroux, Eugene Wu, and Samuel Madden. 2010. TrajStore: An adaptive storage system for very large trajectory data sets. In *ICDE*. 109–120.

[12] Sina Dabiri, Chang-Tien Lu, Kevin P. Heaslip, and Chandan K. Reddy. 2020. Semi-Supervised Deep Learning Approach for Transportation Mode Identification Using GPS Trajectory Data. *IEEE Trans. Knowl. Data Eng.* 32, 5 (2020), 1010–1023.

[13] Xin Ding, Lu Chen, Yunjun Gao, Christian S. Jensen, and Hujun Bao. 2018. UlTraMan: A Unified Platform for Big Trajectory Data Management and Analytics. *Proc. VLDB Endow.* 11, 7 (2018), 787–799.

[14] Mohamed M. Elshrif, Keivin Isufaj, and Mohamed F. Mokbel. 2022. Network-less trajectory imputation. In *SIGSPATIAL*. 8:1–8:10.

[15] Mohammad Etemad. 2020. TrajSeg. https://github.com/metemaad/TrajSeg.

[16] 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.

[17] Ziquan Fang, Yuntao Du, Lu Chen, Yujia Hu, Yunjun Gao, and Gang Chen. 2021. E^{2}DTC: An End to End Deep Trajectory Clustering Framework via Self-Training. In *ICDE*. 696–707.

[18] 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.

[19] 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.

[20] Ziquan Fang, Lu Pan, Lu Chen, Yuntao Du, and Yunjun Gao. 2021. MDTP: A Multi-source Deep Traffic Prediction Framework over Spatio-Temporal Trajectory Data. *Proc. VLDB Endow.* 14, 8 (2021), 1289–1297.

[21] Daniele Foroni, Matteo Lissandrini, and Yannis Velegrakis. 2021. Estimating the extent of the effects of Data Quality through Observations. In *ICDE*. 1913–1918.

[22] Daniele Foroni, Matteo Lissandrini, and Yannis Velegrakis. 2021. The F4U System for Understanding the Effects of Data Quality. In *ICDE*. 2717–2720.

[23] 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.

[24] 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.

[25] Jae-Gil Lee, Jiawei Han, and Xiaolei Li. 2008. Trajectory Outlier Detection: A Partition-and-Detect Framework. In *ICDE*. 140–149.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

[38] 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.

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

[40] Han Su, Kai Zheng, Haozhou Wang, Jiamin Huang, and Xiaofang Zhou. 2013. Calibrating trajectory data for similarity-based analysis. In *SIGMOD*. 833–844.

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

[42] 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.

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

[44] Sheng Wang, Zhifeng Bao, J. Shane Culpepper, and Gao Cong. 2022. A Survey on Trajectory Data Management, Analytics, and Learning. *ACM Comput. Surv.* 54, 2 (2022), 39:1–39:36.

[45] Sheng Wang, Zhifeng Bao, J. Shane Culpepper, Zizhe Xie, Qizhi Liu, and Xiaolin Qin. 2018. Torch: A Search Engine for Trajectory Data. In *SIGIR*. ACM, 535–544.

[46] 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.

[47] Yuan Xu, Jiajie Xu, Jing Zhao, Kai Zheng, An Liu, Lei Zhao, and Xiaofang Zhou. 2022. MetaPTP: An Adaptive Meta-optimized Model for Personalized Spatial Trajectory Prediction. In *SIGKDD*. 2151–2159.

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

[49] Haitao Yuan, Guoliang Li, Zhifeng Bao, and Ling Feng. 2020. Effective Travel Time Estimation: When Historical Trajectories over Road Networks Matter. In *SIGMOD*. 2135–2149.

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

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

[52] Yu Zheng, Xing Xie, and Wei-Ying Ma. 2010. GeoLife: A Collaborative Social Networking Service among User, Location and Trajectory. *IEEE Data Eng. Bull.* 33, 2 (2010), 32–39.