Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 9354273, 12 pages
https://doi.org/10.1155/2018/9354273

Research Article

DS-Harmonizer: A Harmonization Service on Spatiotemporal Data Stream in Edge Computing Environment

Weilong Ding[1,2] **and Zhuofeng Zhao**[1,2]

[1] *Data Engineering Institute, North China University of Technology, 100144 Beijing, China*
[2] *Beijing Key Laboratory on Integration and Analysis of Large-Scale Stream Data, Beijing 100144, China*

Correspondence should be addressed to Weilong Ding; dingweilong@ncut.edu.cn

Received 18 April 2018; Accepted 20 June 2018; Published 5 August 2018

Academic Editor: Shangguang Wang

Abundant sensors in various types are widely used in modern cities to comprehend the current situations in real time. The raw data in open conditions is always in low quality and is hard to employ directly due to its imperfect or missing records. Traditional data preprocessing methods focus on the offline historical data and remain a dilemma between the efficiency and the overhead. In this paper, a data harmonization service *DS-Harmonizer* is proposed on spatiotemporal data stream in the edge computing environment. Through the online cleaning and complementing steps of the hierarchical service instances, the records' validity and continuity can be guaranteed in an efficient way. On the simulated data in a practical project, the service shows high performance, low latency, and acceptable precision in extensive conditions.

1. Introduction

Various sensors of smart cities are adopted in practice nowadays, such as recognition cameras on the trunk roads, smart-card readers in the buses, inductive loops at the toll stations, and transducer in the power plants. Those sensors locating at the edge of the network gather their data hierarchically in a timely manner for further data analysis through either offline batch tasks or online real-time tasks [1]. The diversity and the adequacy of sensory data make it possible to comprehend the current urban situation and the future trends from different perspectives even in real time. In the emerging infrastructure of edge computing environment, leveraging resources closer to the edges would be more efficient for data analysis [2] than that of traditional ways at the data centers in the core network.

The raw data in such open conditions is always in low quality and hard to employ directly for further usage. In a general statistics, 15% of the data gathered directly from sensors is incomplete or contains errors [3]. Due to the data noise, communication error, or device failure, it is common for the processing to encounter critical data distortion or records missing. Take the data in transportation domain as an example. Williams B M et al. in their research [4] show that approximately 20% of the traffic flow data in the investigation were missing. In the data of seven years in Alberta, Canada, that proportion is more than 50% and even 90% at some periods. In the official statistic of Minnesota Department of Transportation, more than 40% of traffic flow data has missing values [3]. The missing data or errors among data are usually caused by three reasons: the data would be lost due to a broken communication link; the data would be abandoned actively by the backend system when the cache is overflow without compensation; the data would be dropped by the dispatching thread in an overloaded parallel or competing environment.

Against the imperfect or missing records, the preprocessing on sensory data is necessary but still remains inherent challenges in edge computing environment especially when the data volume and rate grow. On the one hand, low latency of preprocessing is hard to guarantee on continuous sensory data. Such data stream would be accumulated concurrently and fast from massive sensors at second-level frequency. After the data has been stored, traditional methods face the intrinsic shortage due to the physical limitation of disk

Table 1: The structure of traffic flow data.

Attribute	Notation	Type
Time_start	Start time of an interval	Time
Time_end	End time of an interval	Time
Exit_station	Identity of exit station	Space
Traffic_flow	Amount of exit vehicles	Aggregation

I/O. Although leveraging edge resources can alleviate costs associated with data processing, it tends to be constrained by the capabilities in edge servers. Accordingly, it has to balance the data quality and processing latency with computation overheads. On the other hand, the complementing on the missing values is always blamed on the accuracy. The missing records of the sensory data always lead to unreasonable zero values of business statistics. For example, the travel time being zero at given periods or on given road segments may come from missing records of ALPR (Automatic License Plate Recognition) data. In practice, it is ought to be recognized and repaired, because the aggregation on data stream reflects the continuous situation [5] and such zero or missing is not comprehensible. Moreover, no general standard exists yet due to the diversity of data or business. The spatiotemporal characteristics of sensory data have not paid enough attentions in current works, which must be the very key to improve the practical effects.

In this paper, the typical traffic flow data in highway domain is taken into consideration, and a data harmonization service DS-Harmonizer is proposed in edge computing environment. Through the steps of online cleaning and complementing in the hierarchical service instances, the records' validity and continuity can be guaranteed. Our contributions are listed as follows. (1) On spatiotemporal data stream, the latency of service can hold second-level even when fusing the auxiliary data; (2) through the business constraints and enhanced ARIMA model, the service can distinctly improve the accuracy for the data cleaning and complementing; (3) by the evaluation in a practical project, the benefits of service is convincing in extensive conditions. The rest of this paper is organized as follows: Section 2 shows the motivation and related works; Section 3 elaborates our data harmonization service including two key steps; Section 4 quantitatively demonstrates performance and effects in the experiments; Section 5 summarizes the conclusion.

2. Background

2.1. Motivation. The research in this paper is driven by *Highway Big Data Analysis System* in Henan, one of the provinces in China. The goal of this system is to improve the routine business analysis through Big Data technologies for public travelling and official management. Operated by *Henan Transport Department*, the system has been online since October 2017. On one billion records of the years 2016 and 2017, the analyses in the system involve 660 million vehicles, 37 highway lines, and 279 toll stations. In the system, the traffic flow defined as follows is one of the most significant business metrics.

Definition 1 (traffic flow of toll station(s)). On toll stations L_s, traffic flow is the volume of vehicles exiting L_s at a time period between the start time T_s and the end time T_e, where $|L_s| \geqslant 1$ and $T_e > T_s$. It is counted periodically with a frequency of interval length $\sigma = |T_e - T_s|$. In practice, σ is always 5 minutes, 15 minutes, or 30 minutes as short-time period.

In the system, a record of traffic flow data is typical spatiotemporal and contains 4 attributes as in Table 1. Here, the two temporal attributes present either the start or end of a time period; the spatial attribute is the identifier of a toll station where vehicles exit and pay the toll; the aggregative attribute is the traffic flow value at a given toll station and a given time period.

Such traffic flow data is generated continuously by inductive loops embedded in the ground at a toll station and would be transferred to hierarchical data centers regularly. The procedure of data transmission is illustrated as in Figure 1. The sensory data is first transferred from devices to a road side server through direct wires, and then to servers in section center, region center, and province center in batches step by step through private network. Typical tree topology is composed of those servers: a server connects only an upper one, while it could gather the data from various lower ones. Here, the road side servers and the section centers servers can be regarded as the edge servers in the hierarchy due to their limited computation capacity.

During the procedure above, the error or missing records exist in the records of raw data, which would be disseminated to the subsequent servers. On the one hand, the temporal attributes in the records sometimes are inconsistent. For example, both 2001-01-01 and 2015-05-31 appear in some continual records from the same device. It is always caused by the devices in troubles whose data is considered as untrustworthy. It is not easy to discriminate such false data if no business constraints or conditions are employed. On the other hand, the records of traffic flow are missing at given periods of some toll stations. Due to the high traffic density, the communication may be broken and the device may be in a malfunction, and the sensory data fails to transfer to the servers. In practice, such missing data has to be substituted by the reasonable values for reasonable statistics. In fact, the data preprocessing is difficult due to the scalability and time-sensitiveness guarantee especially when the data size or rate grows. Current methods to tackle imperfect records

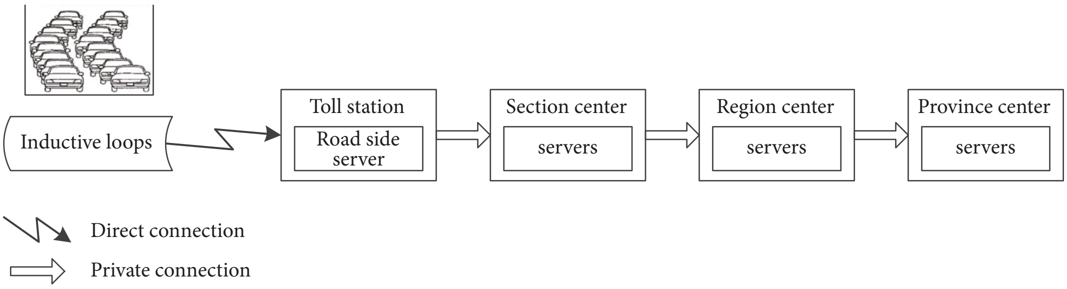


Figure 1: Traffic flow data transmission in a hierarchy.

are centralized and completed in data centers at the core network, which would consume heavy IO on GigaByte data and endures hour-level latency.

Therefore, it is required to handle the continuous spatiotemporal data before further usage in a low latency and cost-efficient way. That is just our original motivation.

2.2. Related Work. Data harmonization combines data from different sources with a comparable view and implies various intentions in different studies [6]. In this paper, this term is extended to the necessary preprocessing services working before the core business calculations. It includes two key steps: the data cleaning for imperfect records and data complementing for the missing ones. Accordingly, the related works can be classified in such two perspectives.

The first perspective is data cleaning. Data cleaning is the process to identify unreasonable data and fix possible errors [7] to improve data quality and keep the semantic consistency [8]. It always involves steps about identifying errors and repairing data. First, errors ought to be identified as the inconsistency, duplication, invalidness, and nonintegrity in the data. It is also the procedure to capture the violations based on business constraints [9] through the technologies like similarity join, clustering, and the rule-based fixing [10]. However, the spatiotemporal characteristics are not exploited enough especially for the continuous data stream [11]. An analogous work [12] concerns the problem about real-time data in wireless sensor network, but it only focuses on the redundant duplication. Second, data repairing aims to revise the records' errors after they have been identified. The heuristic repairing algorithms like functional dependency [13] or denial constraints [14] often employ confidence values [15] to alter possible errors or missing attributes. Commodity cleaning systems like NADEEF [16] even require consulting business professionals for more domain knowledge. However, in highway domain, the general principle for data cleaning still lacks. In this paper, on the continuous data stream, our method concerns temporal inconsistency and business constrains' violation according to typical spatiotemporal characteristics. The efficiency and scalability are guaranteed in edge computing environment.

The second perspective is data complementing. To improve the low quality from the missing records, the data complementing is the manipulation techniques to substitute the missing one [17] with the approximate value. Against the null or distorted value from the missing records, data complementing always refers to the domain experiences. First, an intuitive way is to substitute missing values through domain specific threshold. Such a method of highway domain is proposed using the upper and lower bound on traffic flow data [18]: the distorted value larger than the upper bound (or smaller than the lower bound) would be revised as that bound. It overemphasizes the value's validity but neglects the real rationality on spatial or temporal factors. Second, another typical idea is to generate values from the historical trends by offline processing. Such a method [19] is proposed for traffic flow data complementing from the values of the same period in the previous day and that of the previous period in the same day. The complementing value $y(t)$ $=a * y(t\text{-}1) + (1\text{-}a) * y^{(k-1)}(t)$, where a is a predefined weight for the current day, $y(t\text{-}1)$ is the traffic flow value of the previous period in the same day, and $y^{(k-1)}(t)$ is the value of the same period in the previous day. It can reduce the influence of the flow fluctuation, but the offline manner could not fit the continuous condition. Third, the value can be generated from the neighbors of the missing one by online or offline processing. Such a method [19] is used for traffic flow data complementing through the values of the adjacent periods in the same day. The complementing value $y(t) = [y(t\text{-}n) + y(t\text{-}n\text{+}1) + \ldots +y(t\text{-}1)]/n$, where n is the neighbor volume and $y(t\text{-}n)$ is the traffic flow of the nth previous period. That makes it possible to work on real-time data stream, but such arithmetic average cannot always achieve accurate value due to the outlier values. Fourth, the data complementing can be regarded as a short-term prediction problem on the recent data. The prediction idea is widely used in fields like QoS evaluation, process management, and service recommendation [20]. In highway domain, the traffic flow prediction is one of the hottest topics [21] and current work can be classified in different perspectives like predictive period, predictive range, and implemented technology. Such methods for data complementing can acquire more precise results than that of others, but they cost more due to the calculation complexity. Current technical mainstream is the Big Data [22] solutions on popular system like Hadoop, Storm, or Spark running at core data centers [2]; while in edge computing environment only limited computation capacities of edge servers are available. Accordingly, those methods cannot be applied directly for the data complementing in this paper. As the recent trends exploiting spatiotemporal characteristics, our method adopts enhanced prediction technology for the better accuracy with economy consumption.

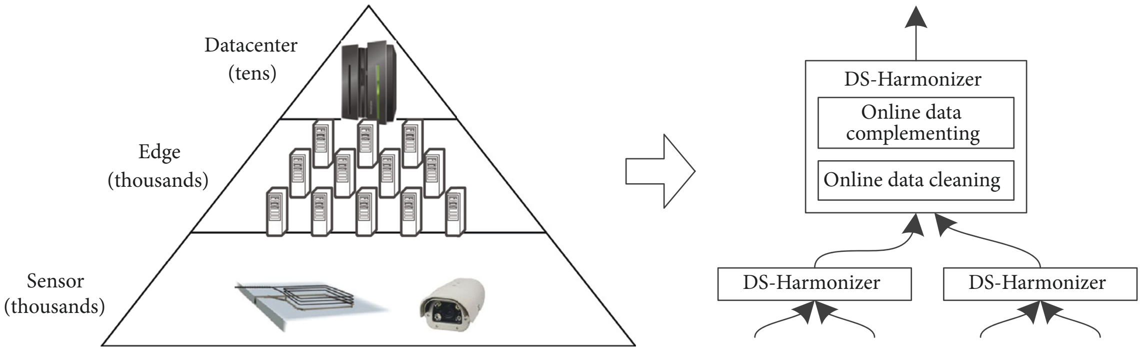


Figure 2: DS-Harmonizer in hierarchical edge environment.

In brief, on continuous data with spatiotemporal characteristics, current works still lack effective approaches to leverage edge resources for either data cleaning or data complementing. Taking the traffic flow data in highway domain as an example, we introduce our harmonization service on data stream against the imperfect or missing records.

3. Data Harmonization on Spatiotemporal Data Stream

3.1. Methodology. The DS-Harmonizer (Data Stream Harmonizer service) is a lightweight streaming processing service running in hierarchical edge servers as Figure 2.

In highway domain like the scenario of Section 2.1, the edge environment can be abstracted as three layers. (1) The bottom sensor layer maintains the thousands of sensors in multiple types. The raw sensory data generates from sensors and transfers to the edge layer. For example, at a toll station, the inductive loops embedded in the ground and the recognition cameras installed on the lane gantry would work collaboratively to yield traffic flow data. As the description in Table 1, the traffic flow data includes the time, station identifier, and an aggregative value. (2) The middle edge layer owns thousands of hierarchical edge servers. When gathering data from sensors, the layer would handle imperfect or missing records and then transfer the validated data to the data center layer. As mentioned before, those edge servers make up the typical tree hierarchy: one server could gather data from multiple lower servers and transfer its results to a certain upper one. Each edge server provides execution environment for a DS-Harmonizer instance with limited computation capacity like CPU, memory, storage, and bandwidth. (3) The top data center layer is a private Cloud for massive business calculations on sensory data. Compared with the servers in edge layer, the data center owns enough resources for Big Data analysis, such as *daily traffic flow of ETC vehicles*, *weekly mileage of MTC vehicles*, and *monthly proportion of vehicle types*.

The sensory data floats from bottom to the top, and it would be hierarchically handled and transferred in the edge layer through the instances of DS-Harmonizer service. The service is a lightweight streaming processing job including two consecutive steps: the data cleaning step against the imperfect records and the data complementing step for the missing ones. Each service instance runs independently in an edge server and collaborates with its direct neighbors through the message communication. Moreover, such service collaboration provides somewhat guarantee of data availability when an instance or an edge server crashes: the lost data in that case can be regarded as the missing records and would be complemented by DS-Harmonizer in an upper edge server. The two steps for data harmonization in the service would be elaborated in the following parts.

3.2. Online Data Cleaning. When the data is gathered from sensors or lower servers, DS-Harmonizer in the target edge server would carry out the online cleaning first to handle the imperfect records in the data. We take the scenario in Section 2.1 as an example and illustrate the procedure by Figure 3. The input here is the continuous sensory data in two different types, and the output is the cleaned data stream of traffic flow. The goal of this step is to revise some errors in attributes and calibrate the semantic inconsistency among data.

As in the left part of Figure 3, it is a typical streaming processing procedure. Each record of traffic flow data in the stream is read at a time and then its attributes are extracted for verification. If the station does not exist or neither of temporal attributes is on the current day, the record is regarded as meaningless and would be abandoned directly, because no hints are available for the repair. Otherwise, some temporal errors can be corrected: if the start time is less than the end one, these twos would be replaced by each other; the false date in either temporal attribute can be substituted by the current day. More business constraints can be adopted here. After an optional calibration procedure with the data in other types, a revised valid record would be emitted to the downstream.

The right part of Figure 3 demonstrates how the data in different types would fuse for the data cleaning. At a toll station in highway, besides the inductive loops, the recognition cameras are deployed, whose data could be used for the traffic flow calibration. The ALPR (Automatic License Plate Recognition) data generated by those cameras has the structure in Table 2, which includes several temporal, spatial, and entitative attributes. Compared with the traffic flow data, the ALPR data contains more information about vehicles

Table 2: The structure of ALPR (automatic license plate recognition) data.

Attribute	Notation	Type
Time	Timestamp when vehicle is passing	Time
Exit_station	Identity of exit station	Space
Exit_lane	Lane number of exit station	Space
Vehicle_license	Vehicle identity	Entity
Vehicle_type	Vehicle type	Entity
Card_id	vehicle passing card identity	Entity
ETC_id	vehicle ETC card identity	Entity
ETC_cpu_id	ETC card chip identity	Entity

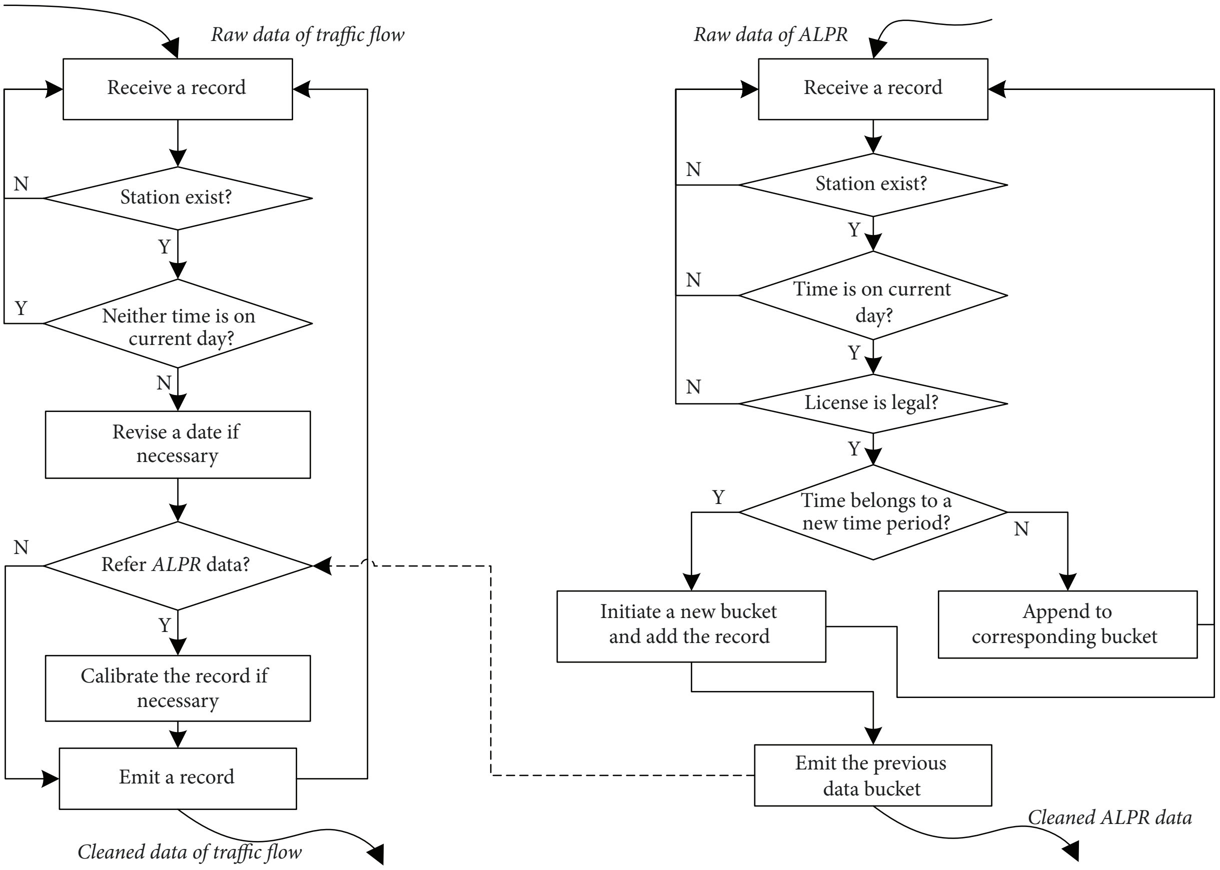

Raw data of traffic flow
Receive a record
Station exist?
N
Y
Neither time is on current day?
Y
N
Revise a date if necessary
Refer ALPR data?
N
Y
Calibrate the record if necessary
Emit a record
Cleaned data of traffic flow
Raw data of ALPR
Receive a record
Station exist?
N
Y
Time is on current day?
N
Y
License is legal?
N
Y
Time belongs to a new time period?
Y
N
Initiate a new bucket and add the record
Append to corresponding bucket
Emit the previous data bucket
Cleaned ALPR data

Figure 3: Online data cleaning procedure.

and roads, so that it can be employed for wider analysis in more perspectives. As the streaming processing procedure, each record of ALPR data in the stream is read at a time. Besides the verification for the spatial and temporal attributes like that of traffic flow data, the vehicle licenses would be inspected for its legality. For some occasions, a vehicle cannot be recognized correctly in its plate, type, or payment mode. As the key of ALPR data, the license is a vehicle's identifier and its illegality would make a record invalid. The licenses are examined by the filtering on regular expressions. For example, the expression in Figure 4 is used to examine four aspects: plate color, belonging provincial region, license number, and license type. The business constrains on those factors are beyond this paper and would not be elaborated here. After that, each cleaned record would be cached in a bucket structure according to the time attribute. Besides being transferred continuously to the next DS-Harmonizer instance, the result as bucket can be referred by traffic flow data for the calibration: the value of traffic flow during the same period has to be consistent with the count of valid ALPR records.

Therefore, DS-Harmonizer can fuse the continuous sensory data in different types to complete the online data cleaning. The cleaning procedure for any type of data above is an independent streaming processing job and works collaboratively with the others by the message communication in the same edge server. Here, the output is the continuous data stream of the traffic flow data and ALPR data: the former

$$\underbrace{\text{([蓝白黄黑])?\{1\}}}_{\textit{Plate color}}\underbrace{\text{[京津冀沪渝豫云辽黑湘皖鲁新苏浙赣鄂桂甘晋蒙陕吉闽贵粤青藏川宁琼使领]\{1\}}}_{\textit{Belonging to provincial region}}\underbrace{\text{[A-Z]\{1\}[A-Z0-9]\{4\}}}_{\textit{License number}}\underbrace{\text{[A-Z0-9挂学试警港澳]\{1\}}}_{\textit{License type}}$$

Figure 4: Regular expression for a valid vehicle license.

would be the input of the data complementing of the same service instance; the latter would be the input of other service instance in an upper edge server. With the help of data cleaning, the data quality is improved in a certain extend because the imperfect records cannot disseminate further.

3.3. Online Data Complementing. The output data stream of the data cleaning triggers the online data complementing, which becomes more feasible when some irreparable records are abandoned. Although the traffic flow seems to be fluctuating randomly, it appears to be spatiotemporal correlation [19] because an ongoing vehicle on a specific line has certain speed limitation and imperative distances between others. That makes the traffic flow predictable in short term. Moreover, such prediction to complement the missing data in edge environment has to be low latency in cost-efficient manner, because only limited computation capacity is available in edge servers to balance the effect and cost. Accordingly, we extend the ARIMA (AutoRegressive Integrated Moving Average) model on spatiotemporal data stream to achieve preferable accuracy with comprehensible parameters. The procedure is illustrated by Figure 5.

As the left part of Figure 5, the input is the cleaned traffic flow from data cleaning, and the output is the complemented data stream. It is illustrated as follows, where the "missing" refers to a lost record of traffic flow data at station s of time interval l.

(i) A record at s of time interval l+1 in the traffic flow stream is read.

(ii) The records at the same station are handled by the same hierarchical DS-Harmonizer instances, and their temporal sequence are guaranteed. A record at s of the previous interval l is regarded as a missing one, if it has not been read from the input stream yet. Otherwise, go to (iv).

(iii) The missing record at s of l is substituted by a cached one, which is predicted during the previous time interval l-1. After the record complementing and result emitting, the very record of l+1 would be cached further.

(iv) The cache updates: the record of l+1 would append to cache when the record of l is missing or replaces the one in the cache otherwise.

(v) With this record and the ones at s of recent K intervals in the cache, the predictive value at s of l+2 would be calculated through our enhanced ARIMA model. It would be elaborated in the right part of Figure 5.

(vi) After the predictive record of l+2 is cached, the record of l+1 would be emitted to the output stream.

The cache is maintained in the memory of edge server where the service instance is resident. ARIMA model is employed in the (v) step to predict the traffic flow of the next interval as a potential complement. As a classical short-term prediction model on offline time-series data, ARIMA (p, d, q) process can be expressed as $(1-\sum_{i=1}^{p}\phi_i L^i)(1-L)^d X_t = (1+\sum_{i=1}^{q}\theta_i L^i)\varepsilon_t$, where L is the lag operator, X_t is the traffic flow value at time t, and ε_t is the white noise at time t. It can be regarded as three steps with a respective algorithmic parameter as autoregressive (abbr. AR(p)), differencing (abbr. I(d)), and moving average (abbr. MA(q)). We extend it on spatiotemporal data stream like the right part of Figure 5, where the record is just read and the recent K ones are used as the input.

(a) When the record is just read and the previous K ones at s are ready, the stationary of time-series would be verified. To guarantee low latency of the verification, the ADF (Augmented Dickey-Fuller test) [23] is employed: if the hypothesis is rejected by test, those values are proved stationary and step(c) is triggered with the parameter d=0.

(b) Otherwise, on those nonstationary traffic flow values, differencing operation I(d) would be evaluated iteratively d times until ADF test rejects the hypothesis (i.e., their differential values are stationary). Considering the latency for data stream processing and the fast convergence for the hypothesis testing, the differencing parameter d is restricted not larger than 5 according to the domain experience. In fact, it is always not greater than 3 in practice. Therefore, the d value is found by I(d) and ADF test here.

(c) On the stationary differential values of traffic flow, the AR(p) and MA(q) steps are evaluated. Traditionally, it is a long-term iterative calculation through Least Squares method, while a trick is adopted here for the online processing. Parameter q is not more than 3 according to the domain experience, and parameter $p <= K+1$ (K is a build-time parameter above) implies the sample size. Due to the finite combination of p and q, the minimum of AIC (Akaike Information Criterion) [24] can be found by the enumeration instead of the iteration with the time complexity O($p*q$). For example, it is sufficient for short-term prediction when K=200, because the recent data lies in more than 16 hours (200∗5=1000 minutes) even if the minimal interval length of 5 minutes is used. Therefore, the p and q values are found by the minimal AIC metric.

(d) Model is learned to predict the complementing value. Through the certain algorithmic parameters p, d, and

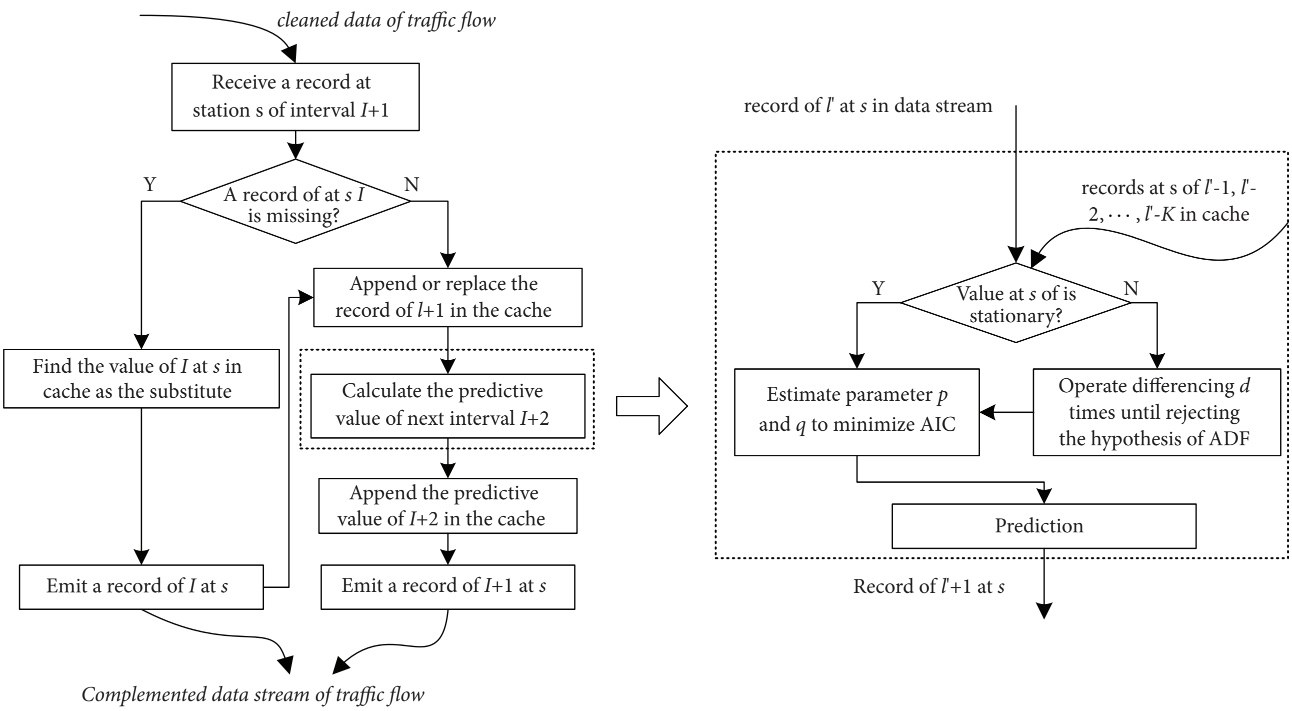

cleaned data of traffic flow
Receive a record at station s of interval I+1
A record of at s I is missing?
Y
N
Find the value of I at s in cache as the substitute
Append or replace the record of l+1 in the cache
Calculate the predictive value of next interval I+2
Append the predictive value of I+2 in the cache
Emit a record of I at s
Emit a record of I+1 at s
Complemented data stream of traffic flow
record of l' at s in data stream
records at s of l'-1, l'-2, ··· , l'-K in cache
Value at s of is stationary?
Y
N
Estimate parameter p and q to minimize AIC
Operate differencing d times until rejecting the hypothesis of ADF
Prediction
Record of l'+1 at s

Figure 5: Online data complementing procedure.

q, the ARIMA model can be determined by learning the optimized parameters of model as the traditional way. The traffic flow value at the same station of next time interval can be predicted by that model.

Accordingly, DS-Harmonizer realizes the online data complementing through continuous prediction on data stream. The trade-off of the low latency stream processing is considered in the limited computation capacity of edge servers. Common domain experience is adopted to speed the parameter tuning to avoid the heavy iterations. The output is the traffic flow of current period as data stream and would transfer to the service instance in an upper edge server. Their predictive value of next period is cached for potential data complementing in the future. It makes a cost-efficient solution.

4. Evaluation

4.1. Experiment Setting. In the project mentioned in Section 2.1, our service is evaluated by extensive experiments. For the data center layer, tens of Acer AR580 F2 rack servers via Citrix XenServer 6.2 are utilized to build a private Cloud, each of which owns 8 processors (Intel Xeon E5-4607 2.20GHz), 64 GB RAM, and 80 TB storage. For the edge layer, tens of personal computers are used as edge servers, each of which owns 2-core CPU, 4 GB RAM, and 500 GB storage installing CentOS 6.6 x86_64 operating system. For the sensor layer, the data stream is simulated through our dedicated data generator [25, 26]. The data imported were generated in Henan province since Feb 1st 2017 to Apr 30th 2017. The concurrency and velocity of the simulated stream could be configured by defined scripts and settings in the data generator. By default, simulated traffic flow data is generated from virtual inductive loops with the rate of 1 record per 5 minutes in a loop; each toll station contains 10 loops for 200 toll stations (i.e., concurrency is 10∗200=2000); simulated ALPR data is generated from recognition cameras with the rate of 1 record per second in a camera; each toll station contains 10 cameras for 200 toll stations (i.e., concurrency is identical to loops); the parameter K (sample size of each station) is set as 200. That is, the number of the inductive loops is identical to that of cameras and equals that of bottom DS-Harmonizer instances in a hierarchy.

In any edge server, several tools are deployed. (1) A tailored *Apache Storm* 0.9.4 is installed as a single-machine cluster. All daemons (e.g., *nimbus* and *supervisor*) run in one machine, message acknowledgement is disabled, and some components are removed to reserve more resource for data harmonization. (2) The *ZeroMQ* 2.1.4 is employed as a lightweight message service in front of Storm. (3) DS-Harmonizer is implemented as a *topology* as Figure 6, in which the data cleaning and complementing is realized as a respective *bolt*. Two types of sensory data are brought into the topology by a respective *spout*: ALPR_Spout is for ALRP data and TF_Spout is for traffic flow data. The data transmission between spout and bolt is *shuffle grouping*, which dispatches the data uniformly to each task of Cleaning_Blot; the data transmission between bolts is *field grouping* by the station identifier, which ensures the records of the same station would be dispatched to the same task of Complementing_Bolt. Moreover, the *worker* number is 1,

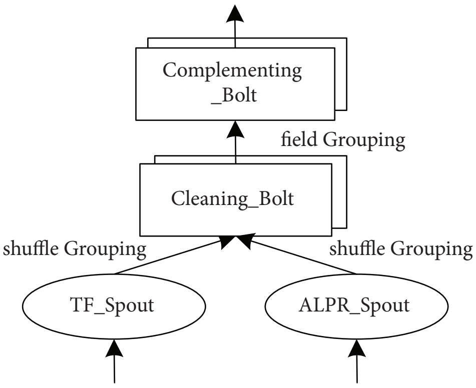


Figure 6: A Storm topology of a DS-Harmonizer instance.

the parallelism (*executor* number) of any bolt is 2, and the parallelism of any spout is 1, all of which can be tuned in Storm. The service instance is the worker of that topology running in an edge server, and the message communication among instances is just like hierarchy of Figure 2.

4.2. Service Performance. First, the performance of DS-Harmonizer is the focus, and its interrelation among the factors of time (i.e., interval length), space (i.e., sensors volume), and edge condition (i.e., service hierarchy) is evaluated. The first two can be configured by the velocity and concurrencies in the data generator and the last one is tuned by service instances' architecture.

Experiment 1. DS-Harmonizer is applied to harmonize the traffic flow data (abbr. TF) in two ways with (*fusion* way) or without (*solo* way) auxiliary ALPR data. All the service instances are in the same bottom level, and the factor concerned here is the number of inductive loops (i.e., hierarchical width). In both ways, the interval length of TF is, respectively, set as 5, 15, and 30 minutes, by configuring TF rate. In the fusion way, the rate of ALPR is kept as 1 record/second per camera. In each test around, the concurrency of TF and ALPR data is from 1000 to 5000 (i.e., 5~25 sensors in each of the 200 stations); the average latency of a record from a certain loop to the data center is counted after the services having run smoothly. The result is shown in Figure 7(a).

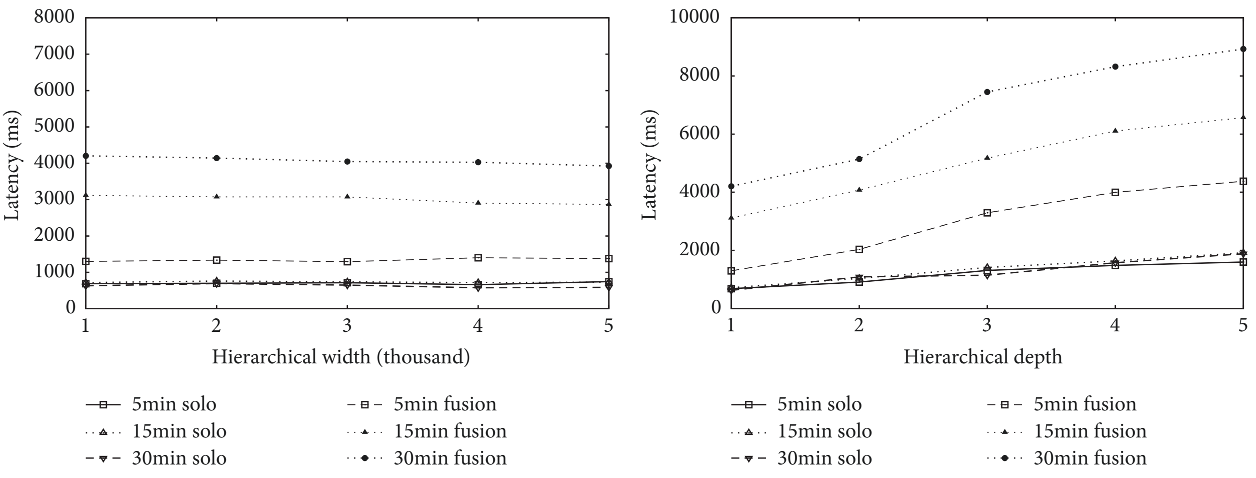
Experiment 2. Like two ways above, the factor concerned here is the edge level (i.e., hierarchical depth), and the levels of service instances are deployed from 1 to 5. The concurrency of either TF or ALPR data is kept as 1000 (i.e., 5 sensors in each of the 200 stations). In both ways, the interval length of TF is, respectively, set as 5, 15, and 30 minutes, by configuring TF rate. In the fusion way, the rate of ALPR is kept as 1 record/second per camera. In each test around, a new level is deployed between the top level and the data center by importing 5 service instances (i.e., each instance connects a lower one); the average latency of a record from a certain loop to the data center is counted after the services having run smoothly. The result is shown in Figure 7(b).

We found these consequences during the service execution. (1) DS-Harmonizer has well horizontal scalability in concurrency, and the latency is related to the interval length. As Figure 7(a), the latency stably holds second-level in both ways under different interval lengths. In the solo way when only TF data is harmonized, the latency reflects the processing capacity regardless of the concurrency of TF data. As a result, three interval lengths make the latency no different in the solo way. While in the fusion way collaborated with the ALPR data, the longer the interval length is, the higher the latency would be in any hierarchical width. The velocity of ALPR data is kept as a constant, longer interval length implies larger ones to handle during the data cleaning step, which undoubtedly requires much time. (2) The latency is related to the hierarchical depth of edges. As Figure 7(b), in both ways under different interval length, the latency increases progressively when more levels are introduced. More levels bring longer path of data transmission, which inevitably delays the data from the sensor to the data center. In the solo way, the latency of three interval lengths is still identical and grows slightly in almost the same extent. While in the fusion way, the latency of three intervals rises obviously, and the longer the interval length brings higher latency in any hierarchical depth. The reason also comes from the ALPR data volume discussed before. (3) The latency is also related to the data fusion. In either Figure 7(a) or 7(b), the fusion way under the same interval consumes longer time than that of solo way. It is an intuitive that more resources are required to handle the data in another type. The fusion way seems heavier than the solo one, but it is worthy due to its higher accuracy as discussed later.

4.3. Harmonization Effect. Next, we introduce artificial dirt into the records and evaluate the harmonization effect during the online data cleaning.

Experiment 3. From the real data of a certain day, some records are selected randomly. The proportion of the selected ones in that day is 5%, 10%, 15%, 20%, or 25%, respectively. For such selected records, the artificial dirt in temporal attributes would be introduced as follows: in a half of the selected records, a decimal digit of a timestamp (*time_start* or *time_end*) is altered randomly; in the other half, a decimal number of both timestamps would be altered randomly. Analogously, the artificial dirt in spatial attribute is introduced as follows: in the selected records, the station identifier (*exit_station*) would be substituted with an inexistent one. Then, the stream is relayed through the data generator on the data in that day, where the concurrency of TF data is kept as 1000 (i.e., 5 loops in each of the 200 stations) and the interval length of TF is set, respectively, as 5 minutes by configuring TF rate. DS-Harmonizer is applied as the solo way, and 5 service instances are in the same bottom level. In each test round, after the services having run smoothly, the proportion of the correct records in the output of online cleaning is counted after the comparison with the raw data before the artificial dirt is introduced. The result is showed as in Figure 8.

The feasibility of online data cleaning is proved when the dirty records are introduced. The proportion of correct



(a) (b)

Figure 7: The latency of DS-Harmonizer.

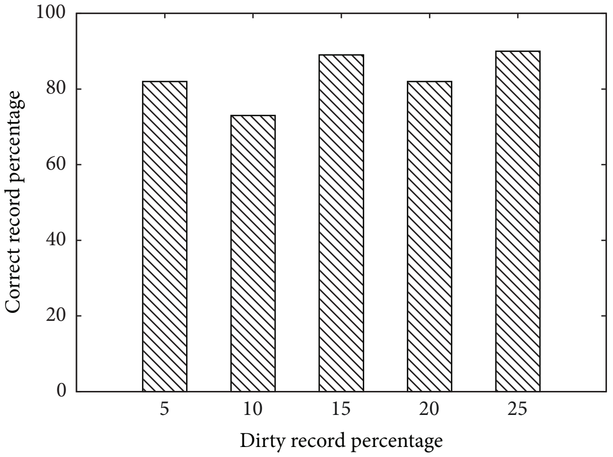


Figure 8: Data cleaning against artificial dirty records.

Table 3: The evaluation for the prediction on a workday.

Item	ARIMA	ARIMA+
MAPE (%)	6.87	5.79
MDAPE (%)	4.57	3.64
Latency(ms)	3725	1940

records in output is more than 70% and remains high even when dirty record scales. In intuition, the result ought to decline when more such altered records have to be handled; while in our online method, it is not related to the volume of the dirty records because any record would be examined once. Although some of dirty records are not distinguished here due to the limited business constraints (more can be introduced then), the online data cleaning still shows the advantage about accuracy.

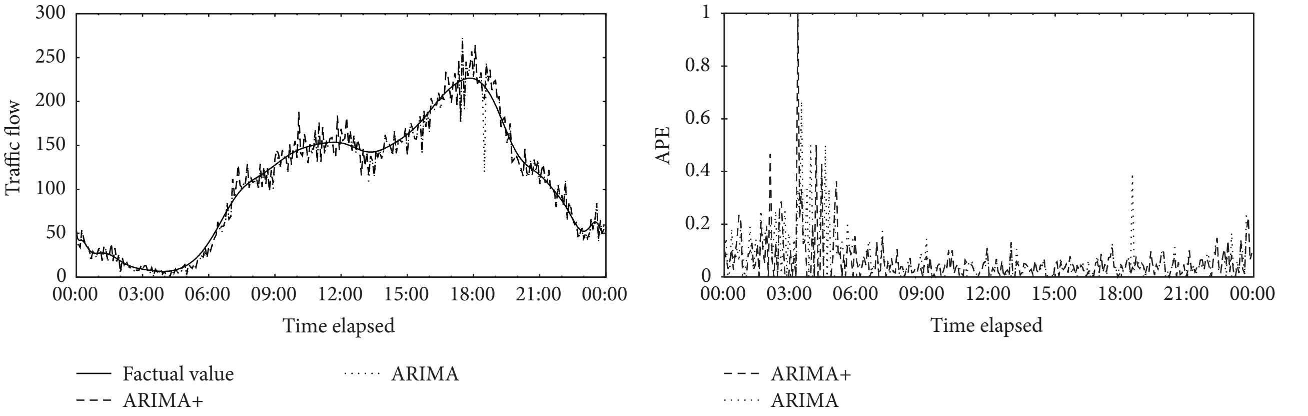
Then, we analyze the effect of the online data complementing on the data of two different days, because the traffic flow of highway shows distinct trends on workday or holiday. The short-term traffic flow prediction is adopted in DS-Harmonizer to complement missing records, so the predictive error could be the indicator for the evaluation. In this section, three common metrics are used to evaluate errors for prediction. The first is the absolute percentage error (abbr. APE) defined as (1); the second is the mean absolute percentage error (abbr. MAPE) by the definition of (2); the third is the median absolute percentage error (abbr. MDAPE) which is the median of APE. Here, at the beginning time of an interval t, x_t is the factual value of traffic flow; x_t' is the predictive value calculated at the beginning time of the previous interval t-1.

$$\mathrm{APE} = \frac{\left|x_t - x_t'\right|}{x_t} * 100\% \tag{1}$$

$$\mathrm{MAPE} = \frac{1}{N}\sum_{1}^{N}\frac{\left|x_t - x_t'\right|}{x_t} * 100\% \tag{2}$$

Experiment 4. The data in a workday, Apr 20th 2017, is imported to replay as data stream. In the data generator, a record of traffic flow data is generated every 5 minutes (i.e., interval length is 5 minutes) from one virtual inductive loop. A dedicated instance of DS-Harmonizer is deployed as the solo way in a single bottom level to receive the simulated data. For comparison, both traditional ARIMA (ARIMA) and our enhanced model (ARIMA+) are implemented in DS-Harmonizer. After the service having run smoothly, the output of online complementing and the latencies during each data complementing are noted. After the end of the replay, calculate three error metrics of both models with the real data in that day. The average latency of a record from the loop to the data center is also counted. For a heavy traffic station *ZhengzhouNan*, the result is illustrated as in Figure 9 and Table 3.

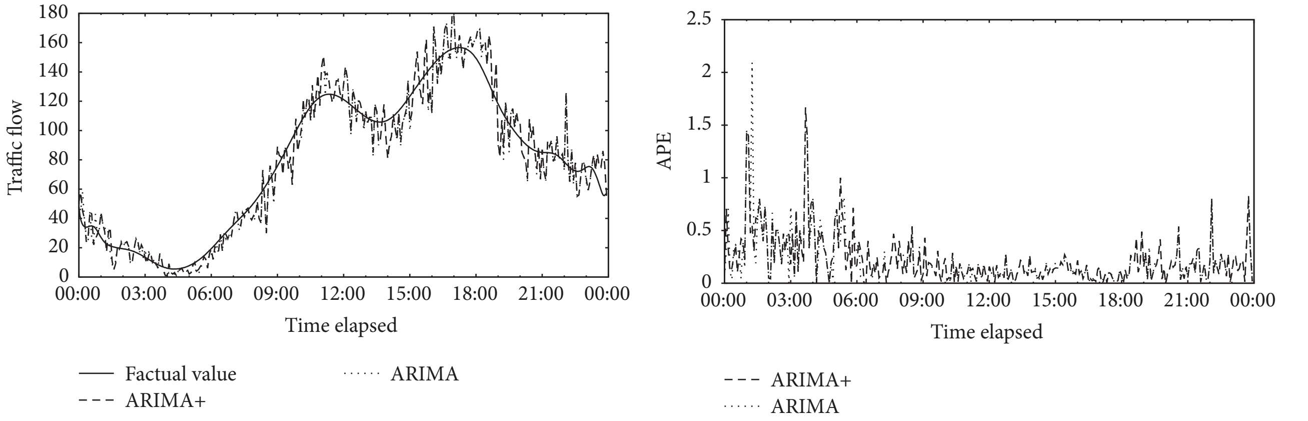
Experiment 5. The data on a holiday, Feb 11th 2017, is imported to replay as data stream. The day is the Lantern Festival on the Saturday just after the Chinese New Year.



(a)

(b)

Figure 9: Prediction at a certain station on a workday.



(a)

(b)

Figure 10: Prediction at a certain station on a holiday.

Other settings are the same as that of the last experiment. After the end of the replay, calculate three error metrics of both models with the real data in that day. The average latency of a record from the loop to the data center is also counted. For the same station *ZhengzhouNan*, the result is illustrated as in Figure 10 and Table 4.

Based on the results of two experiments above, we found that DS-Harmonizer holds the high accuracy with low latency as the discussion below. (1) Both models can approximate the factual trends on either workday or holiday. As in Figure 9(a) and Figure 10(a), two evident spikes appear on either day when the traffic is busy, but on holiday first spike is postponed about 30 minutes and the second one advances 30 minutes. As in Figure 9(b) with Table 3 and Figure 10(b) with Table 4, any of the three metrics in both models is acceptable. (2) Both models own the different precision in workday with that of holiday. Comparing MAPE in Tables 3 and 4, both models fit better on workday, because the stationary of traffic flow on holiday is not outstanding relatively due to more unexpected factors. (3) Our enhanced ARIMA+ model presents the better effects. On the one hand, ARIMA+ shows lower error than the traditional ARIMA, especially in MDAPE. Our model is tailed with the business constrains, which avoids the overfitting in certain extents. On the other hand, our model has advantage in the latency on the data stream. In Tables 3 and 4, ARIMA+ cuts down 35~45% of the average executed time. Compared with the traditional one, our model dramatically reduces the time due to the algorithmic parameters tuning, in which three key parameters d, q, and p have been restricted by their upper bound according to the business constrains.

In brief, DS-Harmonizer proved its high performance and low latency with acceptable accuracy in extensive conditions.

5. Conclusions

In edge computing environment, a data harmonization service *DS-Harmonizer* is proposed to handle imperfect and missing records among the spatiotemporal data stream. By the online data cleaning and data complementing of

Table 4: The evaluation for the prediction on a holiday.

Item	ARIMA	ARIMA+
MAPE (%)	22.52	22.28
MDAPE (%)	6.63	4.00
Latency (ms)	2337	1529

hierarchical services, the records' validity and continuity can be guaranteed in an efficient way. The service shows minute-level latency with horizontal scalability, and it can achieve better precision guarantee during either step in extensive conditions.

Data Availability

The TF (traffic flow) data and ALPR (Automatic License Plate Recognition) data used to support the findings of this study have not been made available because the data were supplied by local management Henan Transport Department under license with certain confidentiality level and so cannot be made freely available. Requests for access to these data should be made to the corresponding author for an application of joint research.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors are grateful to the graduate student Xuefei Wang who helped in improving the quality of this paper. This work was supported by the Youth Program of National Natural Science Foundation of China (nos. 61702014, 61602437), Beijing Natural Science Foundation (no. 4162021), Youth Innovation Foundation of North China University of Technology (no. 1473-1743028), and the Ministry of Transportation, Institute of Highway Science Key Projects (no. 2015-9024).

References

[1] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A Survey on Service Migration in Mobile Edge Computing," *IEEE Access*, vol. 6, pp. 23511–23528, 2018.

[2] E. Renart, D. Balouek-Thomert, X. Hu, J. Gong, and M. Parashar, "Online Decision-Making Using Edge Resources for Content-Driven Stream Processing," in *Proceedings of the 2017 IEEE 13th International Conference on e-Science (e-Science)*, pp. 384–392, October 2017.

[3] M. Zhong, P. Lingras, and S. Sharma, "Estimation of missing traffic counts using factor, genetic, neural, and regression techniques," *Transportation Research Part C: Emerging Technologies*, vol. 12, no. 2, pp. 139–166, 2004.

[4] B. M. Williams, P. K. Durvasula, and D. E. Brown, "Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models," *Transportation Research Record*, no. 1644, pp. 132–141, 1998.

[5] D. Sun, G. Zhang, W. Zheng, and K. Li, "Key Technologies for Big Data Stream Computing," in *Big Data: Algorithms, Analytics, and Applications*, CRC Press, Taylor & Francis Group, USA, 2014.

[6] DSDR, "Data Harmonization" https://www.icpsr.umich.edu/icpsrweb/content/DSDR/harmonization.html.

[7] N. Tang, "Big Data Cleaning," in *Proceedings of the Web Technologies and Applications: 16th Asia-Pacific Web Conference, APWeb 2014*, Proceedings., L. Chen, Y. Jia, T. Sellis, and., and G. Liu, Eds., pp. 13–24, Springer International Publishing, Changsha, China, 2014.

[8] M. Chen, S. Mao, and Y. Liu, "Big data: a survey," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.

[9] W. Fan, F. Geerts, N. Tang, and W. Yu, "Inferring data currency and consistency for conflict resolution," in *Proceedings of the 2013 29th IEEE International Conference on Data Engineering (ICDE 2013)*, pp. 470–481, Brisbane, Australia, April 2013.

[10] J. Wang and N. Tang, "Towards dependable data repairing with fixing rules," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, SIGMOD 2014*, pp. 457–468, Snowbird, Utah, USA, June 2014.

[11] W. Ding, S. Zhang, and Z. Zhao, "A collaborative calculation on real-time stream in smart cities," *Simulation Modelling Practice and Theory*, vol. 73, pp. 72–82, 2017.

[12] L. Wang, L. D. Xu, Z. Bi, and Y. Xu, "Data cleaning for RFID and WSN integration," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 408–418, 2014.

[13] G. Beskales, I. F. Ilyas, and L. Golab, "Sampling the repairs of functional dependency violations under hard constraints," in *Proceedings of the VLDB Endowment*, vol. 3, pp. 197–207, 2010.

[14] X. Chu, I. F. Ilyas, and P. Papotti, "Holistic data cleaning: Putting violations into context," in *Proceedings of the 29th International Conference on Data Engineering, ICDE 2013*, pp. 458–469, Australia, April 2013.

[15] W. Fan, J. Li, S. Ma, N. Tang, and W. Yu, "Towards certain fixes with editing rules and master data," *The VLDB Journal*, vol. 21, no. 2, pp. 213–238, 2012.

[16] M. Dallachiesat, A. Ebaid, A. Eldawy et al., "NADEEF: A commodity data cleaning system," in *Proceedings of the 2013 ACM SIGMOD Conference on Management of Data, SIGMOD 2013*, pp. 541–552, New York, NY, USA, June 2013.

[17] L. de S. Ribeiro, R. R. Goldschmidt, and M. C. Cavalcanti, *Complementing Data in the ETL Process*, Springer Berlin Heidelberg, Berlin, Germany, 2011.

[18] M. Kim, P. Jinsoo, O. Jaeyoung, C. Hakjin, and K. Yoonkee, "Study on network architecture for traffic information collection systems based on RFID technology," in *Proceedings of the 3rd IEEE Asia-Pacific Services Computing Conference, APSCC 2008*, pp. 63–68, Yilan, Taiwan, December 2008.

[19] T. Wang, *Research of the Short-term Traffic Flow Prediction Based on Spark Platform (in Chinese)*, South China University of Technology, Guangzhou, China, 2016.

[20] S. Wang, Y. Zhao, L. Huang, J. Xu, and C.-H. Hsu, "QoS prediction for service recommendations in mobile edge computing," *Journal of Parallel and Distributed Computing*, 2017.

[21] J. Guo, W. Huang, and B. M. Williams, "Real time traffic flow outlier detection using short-term traffic conditional variance prediction," *Transportation Research Part C: Emerging Technologies*, vol. 50, pp. 160–172, 2015.

[22] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, “Traffic flow prediction with big data: a deep learning approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[23] Wikipedia, “Augmented Dickey–Fuller test,” https://en.wikipedia.org/wiki/Augmented_Dickey%E2%80%93Fuller_test.

[24] Wikipedia, “Akaike information criterion,” https://en.wikipedia.org/wiki/Akaike_information_criterion.

[25] W. Ding, Y. Han, J. Wang, and Z. Zhao, “Feature-based high-availability mechanism for quantile tasks in real-time data stream processing,” *Software: Practice and Experience*, vol. 44, no. 7, pp. 855–871, 2014.

[26] W. Ding, Z. Zhao, and Y. Han, “A Framework to Improve the Availability of Stream Computing,” in *Proceedings of the 2016 23rd IEEE International Conference on Web Services (ICWS 2016)*, pp. 594–601, IEEE, San Francisco, CA, USA, June 2016.

Journal of Engineering
The Scientific World Journal
International Journal of Rotating Machinery
Journal of Sensors
Advances in Multimedia
Advances in Civil Engineering
Journal of Control Science and Engineering
Journal of Robotics
Journal of Electrical and Computer Engineering
Advances in OptoElectronics
VLSI Design
International Journal of Navigation and Observation
Modelling & Simulation in Engineering
International Journal of Aerospace Engineering
International Journal of Chemical Engineering
International Journal of Antennas and Propagation
Active and Passive Electronic Components
Shock and Vibration
Advances in Acoustics and Vibration