A Dependable Time Series Analytic Framework for Cyber-Physical Systems of IoT-based Smart Grid

CHANG WANG[1*], Shanghai Jiao Tong University
YONGXIN ZHU[2*], Shanghai Jiao Tong University
WEIWEI SHI[3*], Shanghai Jiao Tong University
VICTOR CHANG[4#], Xi’an Jiaotong Liverpool University
P. VIJAYAKUMAR[5◇], University College of Engineering Tindivanam
BIN LIU[6*], Shanghai Jiao Tong University
YISHU MAO[7*], Shanghai Jiao Tong University

With the emergence of cyber-physical systems (CPS), we are now at the brink of next computing revolution. IoT (Internet of Things) based Smart Grid (SG) is one of the foundations of this CPS revolution and defined as a power grid integrated with a large network of smart objects. The volume of time series of SG equipment is tremendous and the raw time series are very likely to contain missing values because of undependable network transferring. The problem of storing a tremendous volume of raw time series thereby providing a solid support for precise time series analytics now becomes tricky. In this paper, we propose a dependable time series analytics (DTSA) framework for IoT-based SG. Our proposed DTSA framework is capable of proving a dependable data transforming from CPS to the target database with an extraction engine to preliminary refining raw data and further cleansing the data with a correction engine built on top of a sensor-network-regularization based matrix factorization (SnrMF) method. The experimental results reveal that our proposed DTSA framework is capable of effectively increasing the dependability of raw time series transforming between CPS and the target database system through the online light-weight extraction engine and the offline correction engine. Our proposed DTSA framework would be useful for other industrial big data practices.

CCS Concepts: • **Information systems** → **Extraction, transformation and loading**;

Additional Key Words and Phrases: IoT-based Smart Grid, Dependable Time Series Analytics, Cyber-Physical-Systems, Sensor-network-regularization based Matrix Factorization

ACM Reference format:
Chang Wang[1]*, Yongxin Zhu[2]*, Weiwei Shi[3]*, Victor Chang[4]#, P. Vijayakumar[5]◇, Bin Liu[6]*, and Yishu Mao[7]*. 2010. A Dependable Time Series Analytic Framework for Cyber-Physical Systems of IoT-based Smart Grid. *ACM Trans. Web* 9, 4, Article 39 (March 2010), 18 pages.
DOI: 0000001.0000001

Author’s addresses:
*Shanghai Jiao Tong University, 800 Dongchuan Road, Minhang District, Shagnhai 200240, P.R.C.
#Xi’an Jiaotong-Liverpool University, 111 Ren’ai Road, Suzhou Dushu Lake Science and Education Innovation District, Suzhou, P. R. C.
◇ University College of Engineering, Tindivanam Melpakkam, Tindivanam, Viluppuram District-604001, Tamilnadu, India.
Emails: {willy[1], zhuyongxin[2], iamshiweiwei[3], maoyishu[7]}@sjtu.edu.cn, Victor.Chang@xjtlu.edu.cn, vijibond2000@gmail.com[5], liupoem@126.com[6] .

DOI: 0000001.0000001

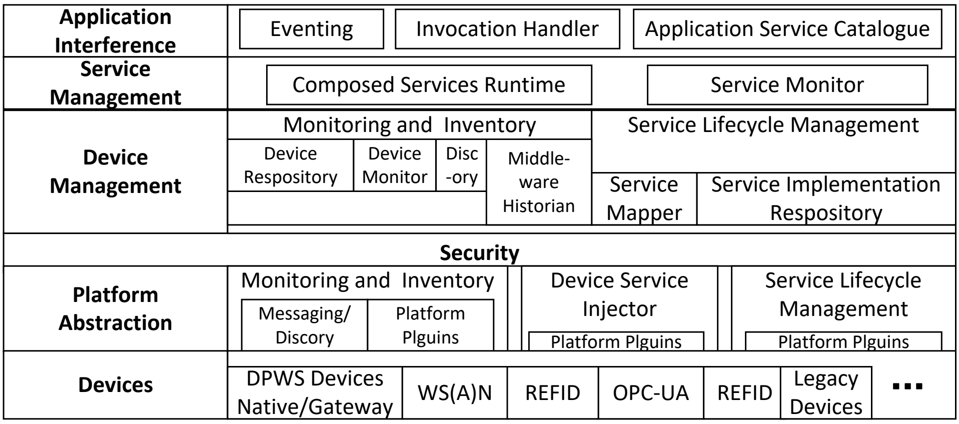


Fig. 1. The architecture of IoT [35].

1 INTRODUCTION

With the emergence of cyber-physical systems (CPS), we are now at the brink of next computing revolution. The CPS are based on the internet communication infrastructures, small and smart physical objects, technologies of big data. All physical devices, sensors in the CPS are connected and integrated into the one tremendous interconnected network where the small objects are monitored and manipulated by people or other devices [14]. The way people interacting with each other has been changed profoundly by the technologies and infrastructures based on the internet. Another tendency that the CPS will influence and reshape the measures how we cooperate and manipulate the small and smart objects around us is now coming [26]. As one of the foundations for this CPS revolution, IoT (Internet of Things) based Smart Grid (SG) is defined as a power grid integrated with a large network of smart objects.

In ten years, the scale of the Internet of Things (IoT) could be increased to a half billion [23]. Smart and small devices in IoT come from different application areas, where all these objects are connected through all kinds of communication infrastructures of Internet. Though there is no widely accepted model of IoT, the most representative structure of IoT is the “EPC Global Internet of Things" supported by Europe and America, and the Japanese Ubiquitous ID(UID) IoT system. We illustrate the integrated morphology of IoT in Figure 1 [35]. Generally, the architecture of IoT is composed of three different layers which are the data acquisition layer (DAL), the data service layer (DSL) and the application layer (AL)[12]. The major feature of the DAL is to recognize the connected devices in the network and collect real-time data; the DSL is responsible for network operation and ensure the data quality; the AL is responsible for providing application solutions for requirements from different fields.

Smart Grid (SG) based on IoT is defined as the power grid integrated with a large network of information and communication technology and combines billions of smart objects: smart appliances, smart meters, actuators and sensors etc [13]. Features of the SG are presented in [13][30], which can be summarized as that a SG is a powerful grid providing an interface between appliances of consumer and capable of supporting the integration of distributed power generation system into an optimized centralized power system.

There are two challenges to support dependable time series analytics for raw SG time series. One challenge is to provide a dependable way to store the raw data from CPS to the target database system. Because of the tremendous volume, the number of tables and relationships, relational databases are overwhelmed by the large scale of SG time series while NoSQL database could be considered as a feasible solution [19]. HBase, as a popular key-value store system, is capable of solving the storage problem[8]. HBase is open source software which runs on top of the Hadoop platform. Unlike a traditional relational database, HBase is a column-oriented database, which is designed and implemented based on google big table. The HBase is capable of providing the ability of to handle complex queries on distributed database clusters as well as tolerating component failures in the system [10] [36]. To provide an ability of processing complex quires on a tremendous amount of data on a computer cluster, HBase has shown a great advantage over traditional storage systems[32]. In the proposed DTSA framework, we use a unified event-driven storage model to store raw time series to better support later stage of data analytics. Hence we design an extraction engine (EE) to transforming data format and refines the raw data with a light-weight on-line method.

The other challenge to support dependable time series analytics is that the raw time series are very likely to contain missing values which could seriously decrease the accuracy of data analytics. Based on our observations, multi-source time series in IoT-based SG are ubiquitous [6, 21]. Under situations that a collection of sensors are used to produce time series data, the generated time series are likely to share the same mission. The correlations of such time series could be positive enough thereby can be utilized to reconstruct the missing data. In the proposed DTSA framework, we design a correction engine (CE) to predict missing values for big data analytics with a sensor-network-regularization matrix factorization method (SnrMF). The SnrMF method takes advantage of the correlations among diverse sensors which are positively correlated. Through this way, the SnrMF method is capable of improving the performance of reconstructing missing data for a single time series. Moreover, we introduce similarity functions into the SnrMF method to determine whether two sensors are correlated, on the basis of which we employ different prediction models.

Our contributions of this paper are as follows.

- We design a dependable time series analytic framework to provide a dependable data transforming from CPS to target database with raw time series refining and cleansing. The proposed DTSA framework provides a solid support for precise time series analytics in IoT-based SG.
- We design a light-weight on-line preliminary data refining method between the buffering system and the extraction engine. The proposed method ensures a dependable data buffering and raw data format transformation to a proposed unified storage model.
- We design an off-line correction engine using a sensor-network-regularization based matrix factorization (SnrMF) method to further cleanse preliminary refining data. The correction engine is capable of predicting missing values more precisely, which lays a foundation for dependable data analytics.
- We verify our proposed DTSA framework on top of Hadoop and HBase platforms. The experiment results illustrate that our proposed DTSA framework is capable of effectively increasing the dependability of raw time series transforming between CPS and the target database system through an online light-weight extraction engine and an offline correction engine.

The organization of the rest of this paper is as follows. Section II introduces related works. Section III is a discussion of prerequisites. Section IV introduces our proposed dependable time series framework. Experimental setups and results are illustrated in section V. We make a conclusion in section VI.

2 RELATED WORKS

To the best of our knowledge, this is the first work which considers the dependability problem of data transforming from the CPS to the target database. In our proposed DTSA framework, the extraction engine is responsible for transforming data from different data formats to the target data format. The correction engine is responsible for cleansing the data which contain missing values. So we would like to introduce proposed works of the storage model on HBase and the methods of predicting missing values in time series in this section.

2.1 Storage model for Dependable Time Series Analytics on HBase

In CPS, technologies nurtured by big data researches have been widely accepted and applied. Hadoop and HBase are two popular storage infrastructures used to store a tremendous volume of time series in a typical modern data center. The major purpose of storing such a tremendous volume of data is to monitoring the statuses of SG equipments to conduct time series analytics thereby prevent the equipments from damages caused by abnormalities, which requires resolving two key issues.

One key issue is the performance challenge when retrieving data from the HBase database. A typical solution is to use Bloom Filter [32, 36] which is used to improve the data retrieving performance and has been integrated into HBase framework. However, the bloom filter only can improve the performance of a *get* operation. That is to say, the random access performance on HBase can be improved by the bloom filter while the performance of a *scan* operation cannot be accelerated with it.

Another key issue is the storage compatibility on HBase when it is applied in a scenario that applications require accommodating time series from several different data sources. When storing unstructured and heterogeneous data in HBase, a dependable and compatible storage model is necessary for times series analytics. One of the advantages of HBase is the huge capacity for CPS data, as well as the high performance of data queries. To effectively utilize the good support of huge storage capacity and the compatibility of storing structured/semi-structured data of Hbase for CPS data, the storage model implemented on HBase must consider both the compatibility and query performance. Only in this way, can HBase be used successfully for CPS time series analytics under a scenario of multiple data sources. Several works have been proposed to accommodate the time series from different data sources[17]. The purpose of the proposed model in [9] is to cleanse data from different data sources and then transform data into a unified data format such as JSON or XML . In [2], a hybrid model has proposed whose major contribution is that the proposed method try to improve the performance of put and delete operations in Hive.

However, all works mentioned above do not consider the compatibility problem which is considered from the beginning in our design. In our proposed DTSA framework, the storage model we adopted is capable of storing data from different data sources with different data formats.

2.2 Missing Data Reconstruction for CPS Time Series Analytics

A lot of models of data mining/statistics have been proposed to reconstruct the missing data in a single time series thereby provide a dependable big data analytics [33, 34]. SVM models [16] proposed by Frasconi et al. adopt a "seasonal kernel" to estimate the similarity between different time-series. Models proposed by Phong et. al. [25] attempt to "model gene expression profiles as simple linear and Gaussian dynamic systems". The author also uses the "Kalman filter" to predict missing values. The experimental results of in this work show that the model of simple linear dynamic systems for gene expression profiles is feasible for predicting missing values in the time series of gene expression matrix [33]. Grabocka et. al. [11] propose a method of matrix factorization

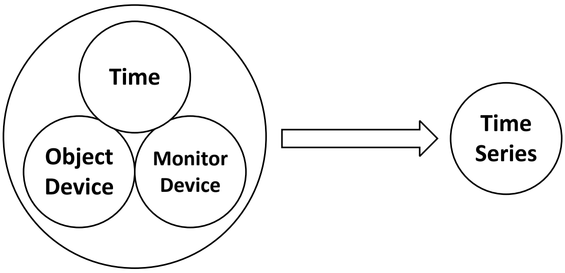
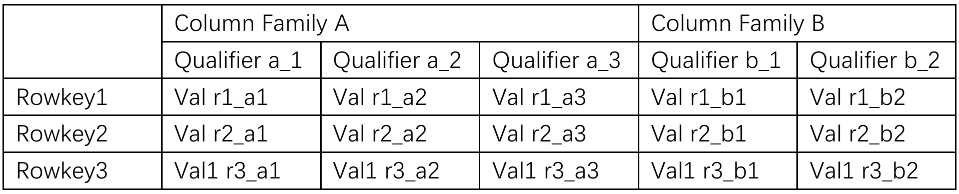


Fig. 2. The Event-driven Storage Model on HBase for IoT-based SG.



	Column Family A			Column Family B	
	Qualifier a_1	Qualifier a_2	Qualifier a_3	Qualifier b_1	Qualifier b_2
Rowkey1	Val r1_a1	Val r1_a2	Val r1_a3	Val r1_b1	Val r1_b2
Rowkey2	Val r2_a1	Val r2_a2	Val r2_a3	Val r2_b1	Val r2_b2
Rowkey3	Val1 r3_a1	Val1 r3_a2	Val1 r3_a3	Val1 r3_b1	Val1 r3_b2

Fig. 3. The structure of HBase table

to categorize diverse time series. The goal of their works is to extract latent factors based on observed entries. The seasonal autoregressive which corporates with "moving average model" and "Kalman filter" achieves the best performance for estimating missing values. Anava et al. also study the problem of time series prediction using the autoregressive (AR) model in the presence of missing data and proposed a new method for online learning problems [3]. Baraldi et al. [4] propose a fuzzy method for missing data prediction. The results of their work have illustrated the advantages to an auto-associative kernel regression method. Song et al. [31] adopt a method of matrix factorization to reconstruct traffic matrices. The experimental results indicate that their proposed method is better than traditional methods.

However, methods mentioned above do not effectively resolve the problem of missing value prediction when facing multiple data sources. Some methods are capable of resolving the missing value prediction problem when handling a singular data source. The other methods try to resolve the problem of multiple data source missing value prediction but produce limited effects Our proposed SnrMF method is built on the basis of such an observation that correlated sensors in the networks generate similar time series, which can be used to predict missing values in a single time series. Based on whether the time series are correlated or uncorrelated, we adopt strategies with different similarity functions.

3 PREREQUISITES

Our proposed DTSA framework is designed and implemented on the basis of a proposed event-driven storage model [17], HBase platform and Redundant Arrays of Independent Disks (RAID). So in this section, we briefly introduce the storage structure of HBase, the event-driven storage model proposed by previous researchers and the basics of RAID system.

3.1 The Storage Structure of HBase

As a non-relational database and an open-source implementation of Google Big Table storage architecture, HBase is capable of managing structured and semi-structured data [10][1]. The built-in features of HBase also includes tolerating fault and recovering quickly from single server failures which provide a dependable storage platform for SG raw time series. In a traditional relational database, a row is an atomic unit of data which is retrieved and stored as a whole. While in a column-oriented database, the data in a column is stored together. Data in a column-oriented database is stored and retrieved in columns and thereby it is possible to only read relevant data if part of data is required.

An HBase cluster is usually composed of at least one master server and several slave servers storing data. In a logical view, a table in HBase is similar to a grid, where a cell can be located by a given row identifier and a column identifier. Row identifier is represented by a rowkey, and the column identifier is represented by a column family and column qualifier. One column family is possible to composed of many columns. The value in a cell can be referred to as the format (rowkey:column family:column qualifier:value). Figure 3 shows a logical view of a table in HBase.

3.2 Event-driven Storage Model for SG Data

In our proposed DTSA framework, we use a buffering system based on RAID10 and an extraction engine to transforming raw SG time series into a format suited for the unified event-driven storage model[17].

In this section, we introduce the event-driven storage model. In the SG system, events which can be monitored generate recordable data. In the event-driven storage model, a data record is defined as the instantaneous status value of a device recorded by a monitoring device. The behavior of generating a unique data record is called an event. One event produces one record in the database. An event can be determined uniquely at least by the monitored device, monitoring device and the time when the event occurred. Each event in the SG system corresponds to only one data record. The event-driven storage model is illustrated in Figure 2. In this storage model, the elements that distinguish different data records can be as less as possible, thereby improving the efficiency of data storage. The detailed design of the table structure of event-driven storage model is showed in Figure 4

3.2.1 Rowkey Structure in Event-driven Storage Model. As depicted in Figure 4, rowkey is composed of five width-fixed fields. Field PREFIX is two bytes long and used for load balance between different HBase servers. EVENTTYPE is used for distinguishing different sources of data, the length of which is two bytes. Field DEVICEID is encoded to 6 bytes and composed of two sub-fields: device type (DEVICE TYPE) and device number (DEVICE NUM). Field EVENTTIME is four bytes long and represents the time when the event happened. It is an integer which represents the number of seconds from current time to January 1st, 1970(1/1/1970). Field MONITOR DEVICEID is the same with field DEVICEID, the difference is that this field stands for a monitor device.

3.2.2 Column Family and Qualifier Structure in Event-driven Storage Model. As depicted in Figure 4, there is only one column family (CF, COLUMN FAMILY) in the table. The length of CF is one byte (ASCII character) as the name of column family must be printable characters. According to [7], the number of column family should be less than ten from the perspective of query performance. Too many column families will decrease the query performance seriously because it needs time to look for a specified column family. A virtual column family is added into the QUALIFIER and represents the EVENTTYPE. Field QUALIFIER is composed of a sub-field virtual column family (EVENTTYPE) and a sub-field column name (COLUMN NAME).

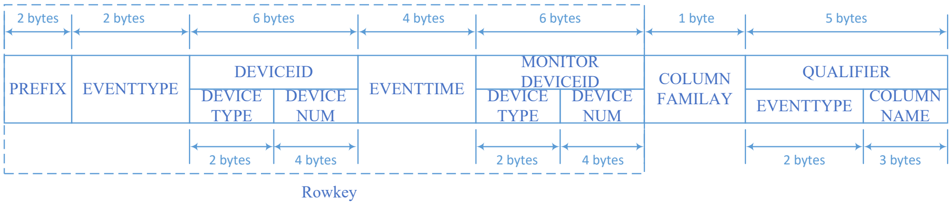


Fig. 4. Table Structure of Event-driven Storage Model [17].

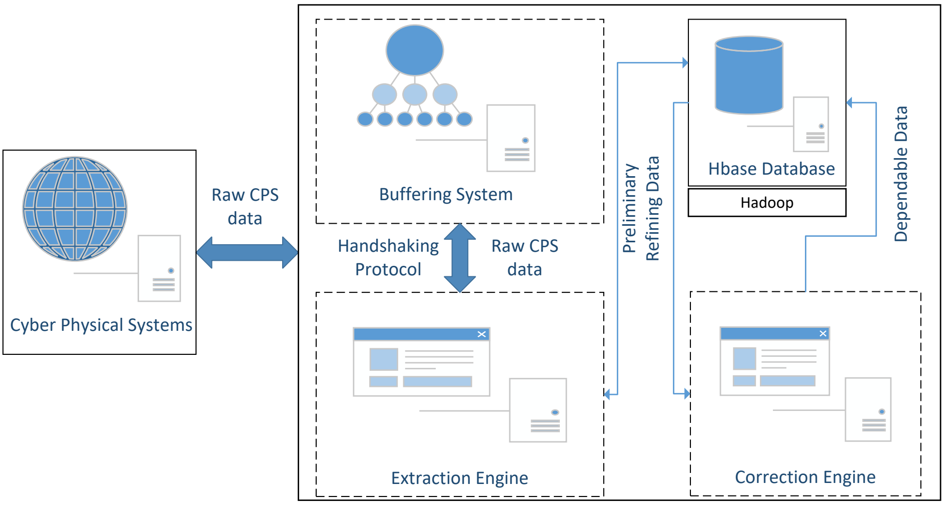


Fig. 5. The architecture of the proposed framework.

3.3 Redundant Arrays of Independent Disks

The buffering system adopted in our DTSA framework is based on the RAID10 system, which is capable of providing enough storage capacity and bandwidth for buffering raw CPS time series. RAID10 (Redundant Arrays of Independent Disks) is one of the most popular RAID systems used in industries, which combines a mirroring (RAID1) storage and a striping (RAID0) storage. The RAID10 system is fault-tolerant and has an I/O performance advantage comparing with other configurations. The RAID10 system is capable of tolerating double disk failures unless that both of the mirror-paired disks have a failure thereby it can provide a higher reliability [20].

4 THE PROPOSED DTSA FRAMEWORK FOR DEPENDABLE SG TIME SERIES ANALYTICS

4.1 Design Overview

Our proposed dependable time series analytics (DTSA) framework is mainly composed of an extraction engine and a correction engine. The target of our work is to design a dependable time series extraction engine engine and correction engine for the IoT-based SG data. In the extraction process, the time series should be extracted and transformed to suit our proposed event-driven data storage model on HBase. Also, the extraction must be dependable, which means the time series from CPS could not be lost or put into the target database with the wrong format. In the correction

ALGORITHM 1: Buffering System Scheduling 1: Sending Data Batch

```
Input: Processing window: $PW$, batch counter: $bc$.
Lock($bc$);
while ($bc \leq PW$) do
    Unlock($bc$);
    Send a batch $batch_i$ to EE;
    Wait for an acknowledge $rack_i$ from EE of $batch_i$;
    if ($rack_i$ is positive) then
        Mark batch $batch_i$ as received;
        Lock($bc$);
        $bc = bc + 1$;
        Unlock(bc);
    end
end
```

ALGORITHM 2: Buffering System Scheduling 2: Confirming Data Batch

```
Input: Processing window: $PW$, batch counter: $bc$.
while (true) do
    Wait for an acknowledge $pack_i$ from EE of $batch_i$ ;
    if ($pack_i$ is positive) then
        Drop $batch_i$ from BS;
        Lock($bc$);
        $bc = bc - 1$;
        Unlock(bc);
    else
        Resend $batch_i$;
    end
end
```

process, we resolve the problem of missing value prediction. The purpose is to improve the data quality and provide a solid support for dependable data analytics.

As has depicted in Figure 5, the extraction engine (EE) is responsible for raw SG time series transforming between CPS and the target database system while the correction engine (CE) is responsible for missing values prediction. The EE receives raw time series from the CPS. To ensure a dependable data transforming, we design a buffering system communicating with the EE with a handshaking protocol. The CE works offline due to the consideration of time overhead. In the following sections, we introduce our proposed DTSA in detail.

4.2 Extraction Engine: Preliminary Time Series Refining in the Proposed DTSA Framework

The extraction engine (EE) is composed of two parts, a buffering system and an extraction system. To construct a dependable EE, there are two major challenges. One challenge is to ensure the buffering system dependable. We uses a RAID10 storage system as the platform of the buffering system, which makes the process of buffering raw CPS data dependable. The other challenge is that the interaction between the buffering system and the extraction system must be dependable. In the interaction process, the batches are sent and received between these two systems. But the load of preliminary processing in the extraction system can be dynamically changed over time. When a batch sent by the buffering system cannot be processed in time because of the full load on extraction system, this bath will be dropped. Measures must be taken to ensure the dropped batches can be resent. By this way, we can ensure the whole EE is dependable.

4.2.1 Dependable Data Buffering System. The buffering system is on the basis of a RAID10 storage system which is capable of providing a dependable data buffering. When the extraction engine (EE) has a failure, the buffering system will keep receiving CPS time series from the network and no data will be discarded. The capacity and the bandwidth of the buffering system could be sufficient with a cheap price because the RAID systems are very mature products.

Besides the RAID system, it is required a pair of scheduling algorithms to negotiate between the buffering system and the extraction system because the buffering system must ensure that every data batch transformed to the extraction system is really put into the target database. If the extraction process fails, the buffering system must have the ability to resend the data batch. Therefore, we adopt a handshaking protocol between the buffering system and the extraction system. The scheduling algorithms running on the buffering system are shown in Algorithm 1 and Algorithm 2.

The Algorithm 1 is responsible for sending a data batch to the extraction system. The processing window *PW* is used by the buffering system to control the rate of sending batches to the extraction system (Algorithm 1, line 2). If the the sent batch was successfully received by the extraction system, a positive acknowledge message will sent to the buffering system and then this batch is marked as successfully received but not processed (Algorithm 1, line 6-11). The Algorithm 2 is responsible for receiving the acknowledge message from the extraction system and confirming the data batch is processed. If the acknowledge message is positive which means the data batch is successfully processed, then this data batch will be dropped from the buffering system (Algorithm 2, line 2-4). Otherwise, if the acknowledge message of the sent data batch is negative which means the extraction system fails to process it, the data batch will be stored in the buffering system and resent by the buffering system(Algorithm 2, line 9).

4.2.2 Dependable Data Extraction System. One of the two jobs of the extraction system is to negotiate with the buffering system. If the extraction system successfully process a data batch sent by the buffering system, it will send an positve acknowledge to the buffering system. If the extraction process failed, the extraction system will resend a data request to the buffering system for the failed data batch. The scheduling algorithms running the extraction system are showed in the Algorithm 3 and Algorithm 4.

In the data extraction system, a data queue is used to store the received data batch. The Algorithm 3 is responsible for receive the data batch sent by the buffering system (Algorithm 3, line 2). If the data batch was successfully put into the data queue, a positive acknowledge message will be sent to the buffering system (Algorithm 3, line 4-8). Otherwise, if the received data bath is failed to put into the data queue, a negative message will be sent to the buffering system and the data batch will be dropped (Algorithm 3, line 8-12). The Algorithm 4 is responsible for processing the received data batch. If a data batch was successfully processed, a positive acknowledge message will be sent to the buffering system or a negative message will be sent (Algorithm 4, line 6-10) .

The other job of the extraction system is to extract raw time series and preliminary refine the data. The target database in our proposed DTSA framework is HBase on top of a Hadoop platform. To extract the time series, we use a proposed storage model: event-driven storage model [17]. The preliminary refining method is to recreate the rowkey of the raw time series from the CPS based

ALGORITHM 3: Extraction Engine Scheduling 1: Sending Acknowledge of Successfully Receiving A Data Batch

```
Input: A batch data FIFO: $F_{bd}$, the maximum FIFO length: $L_{max}$.
while (true) do
    Wait for a data batch $batch_i$;
    Lock($F_{bd}$);
    if (The length of $F_{bd} \leq L_{max}$) then
        Receive $batch_i$ and put $batch_i$ into $F_{bd}$;
        Unlock($F_{bd}$);
        Send a positive acknowledge $rack_i$ to BS for successfully receiving $batch_i$;
    else
        Drop $batch_i$;
        Unlock($F_{bd}$);
        Send a negative acknowledge $rack_i$ to BS for rejecting $batch_i$;
    end
end
```

ALGORITHM 4: Extraction Engine Scheduling 2: Sending Acknowledge of Successfully Processing A Data Batch

```
Input: A batch data FIFO: $F_{bd}$.
while ($F_{bd}$ is not empty) do
    Lock($F_{bd}$);
    Get a data batch $batch_i$ from $F_{bd}$;
    Unlock($F_{bd}$);
    Process $batch_i$;
    if (process successfully) then
        Send a positve acknowledge $pack_i$ to BS for $batch_i$;
    else
        Send a negtive acknowledge $pack_i$ to BS for $batch_i$;
    end
end
```

on structure showed in Figure 4. When the raw time series are extracted and put into the target database, the first stage finishes.

4.3 Correction Engine: Missing Values Prediction of Time Series in the Proposed DTSA Framework

To support dependable time series analytics, in our proposed DTSA framework, we design a sensor-network-regularization based matrix factorization (SnrMF) method to cleanse preliminary refining time series stored in the target database. Because of the overhead, the proposed SnrMF method in the correction engine works offline. The goal of the proposed SnrMF method is to reconstruct the missing data in the time series based on the correlations among positively correlated time series thereby make the time series more accurate. We first introduce the low-rank matrix factorization used in the correction engine. Then on the basis of low-rank matrix factorization, we incorporate two regularization terms into the optimization problem respectively and design two corresponding models: CSbR and USbR. Finally, we introduce the similarity functions used in our method.

4.3.1 Matrix Factorization in Method SnrMF . First, we use the matrix TS^{N*M} to represent the time series received from N different data sources. The problem of low-rank matrix factorization attempts to approximate a matrix TS^{N*M} with multiplying a L-rank factor. The ${TS^{N*M}}_{ij}$ (where i is in range $\{1, 2, ..., N\}$ and j is in range $\{1, 2, , ..., M\}$) stands for the j_{th} element in the time series generated by the i_{th} data source. Our purpose is to factorize the corresponding entities in TS^{N*M}. The reason is that there could be too many missing values in the TS^{N*M} including many irrelative entities. Thereby, an optimization problem on Singular Value Decomposition (SVD) emerges.

$$\min_{S,V} \frac{1}{2} \|W \diamond (TS^{N*M} - SV^T)\|_F^2 + \frac{\lambda_1}{2} \|S\|_F^2 + \frac{\lambda_2}{2} \|V\|_F^2, \tag{1}$$

In Equation 1, the W represents an indicator matrix and $\diamond$ are denoted as the Hadamard product. We also have that $S \in \mathbb{R}^{N*L}$, $V \in \mathbb{R}^{M*L}$ $(s.t.\ L < min(N, M))$ and both λ_1 and $\lambda_2 > 0$. To resolve the problem of overfitting in the optimization process, we introduce two regularization terms: $\|S\|_F^2$ and $\|V\|_F^2$. Based on [18], we also apply gradient strategies to find a minimum.

4.3.2 Model CSbR: Regularization of Correlated Sensors. In our DTSA framework, we denote network sensors as correlated if the sensors share a common mission such as a scenario that there are many diverse sensors in the same building, we could reasonably infer that the humidity sensors could have a strong bond with the temperature sensors because the humidity might go up with the temperature[33].

The CSbR model is proposed for predicting missing value on the basis of a latent sensor matrix S. Because among correlated sensors, there might strong correlations, the issue of missing value prediction can be regarded as an optimization problem on the basis of matrix factorization illustrated in Equation 2.

$$\begin{aligned} \min_{S,V} \mathcal{L}(S, V, TS^{N*M}) = \frac{1}{2} \|W \diamond (TS^{N*M} - SV^T)\|_F^2 + \frac{\lambda_1}{2} \|S\|_F^2 + \frac{\lambda_2}{2} \|V\|_F^2 + \\ \frac{\alpha}{2} \sum_{i=1}^{N} \|S_i - \frac{\sum\limits_{c \in C(i)} H(i, c) * \rho_{i,c} S_c}{\sum\limits_{c \in C(i)} H(i, c)} \|_F^2. \end{aligned} \tag{2}$$

In Equation 2, the similarity between the i_{th} sensor and the c_{th} sensor is measured by the function $H(i, c)$. We denote the correlated sensors of the t_{th} sensor and i_{th} sensor as $C(i)$. The $|C(i)|$ is denoted as the number of correlated sensors.

In the CSbR model, regularization terms can be introduced into the sensor network such as a regularization term of correlated sensors. Therefore, if we have the correlated sensors which is denoted as $C(i)$, we can reasonably infer that there is a positive correlation between the i_{th} sensor and the sensors set $C(i)$. In Equation 2, we use a scaling factor $\rho_{i,c}$ to match the scale difference between the i_{th} sensor and the c_{th} sensor. We also have that α is a penalty factor ($\alpha > 0$) for the equation. On the basis of the regularization items, we can conclude that the value of similarity function $H(i, c)$ could be large enough if that there is a strong correlation between sensor c_{th} and sensor i_{th}.

4.3.3 Model USbR: Regularization of Uncorrelated Sensors. On one hand, the proposed CSbR model is built on the basis that network sensor could be strongly correlated thereby the correlations can be used to guide the matrix factorization. On the other hand, we are aware of that the CSbR model is not suited for missing value prediction for uncorrelated network sensors. Therefore, we propose the USbR model to tackle the problem of missing value prediction for weak-/un- correlated network sensors. In our proposed USbR model, we denote the two sensors uncorrelated if there is a weak correlation between the two sensors.

On the basis of that uncorrelated network sensors share a weak relation, we introduce a term to constrain the distance maximization. More precisely, the maximum distance between sensor i_{th} and C'_i (the uncorrelated sensors set) should be limited. After introducing a new regularization term of the sensor network, we change the form of the optimization problem from Equation 2 to Equation 3. In Equation 3, we also have that α' is the penalty factor and $\alpha' > 0$.

$$\min_{S,V} \mathcal{L}'(TS^{N*M}, S, V) = \frac{1}{2}\|W \diamond (TS^{N*M} - SV^T)\|_F^2 + \frac{\lambda_1}{2}\|S\|_F^2 + \frac{\lambda_2}{2}\|V\|_F^2 - \frac{\alpha'}{2}\sum_{i=1}^{N}\|S_i - \frac{\sum_{c' \in C'(i)} H(i,c') * \rho_{i,c'} S_{c'}}{\sum_{c' \in C'(i)} H(i,c')}\|_F^2. \tag{3}$$

4.3.4 Similarity Functions in the Regularization Models. In our proposed regularization models CSbR and USbR, the similarity function plays an important role because this function directly determines whether two network sensors are correlated. Based on the decision, we adopt different regularization model to factorize the matrix. There are many alternatives for the function H in both Equation 2 and Equation 3. In this section, we discuss five different similarity functions to more comprehensively evaluate our method.

- "Vector Space Similarity (VSS) " can be used to determine how similar between two different sensors i and c. The definition of VSS is shown in Equation 4. If the function $H_{VSS}(i,c)$ produce a large enough value, then we can say that sensor i and sensor c are correlated. In Equation 4, $\mathbf{o}_i$ and $\mathbf{o}_c$ is the subset of $\mathbf{x}_i$ and $\mathbf{x}_c$, which are observed from the sensor network.

$$H_{VSS}(i,c) = \frac{\sum_{j \in \mathbf{o}_i \cap \mathbf{o}_c} {TS^{N*M}}_{ij} \cdot {TS^{N*M}}_{cj}}{\sqrt{\sum_{j \in \mathbf{o}_i \cap \mathbf{o}_c} {TS^{N*M}}^2_{ij}}\sqrt{\sum_{j \in O_i \cap O_c} {TS^{N*M}}^2_{cj}}} \tag{4}$$

- "Gaussian Kernel (GK) " is also a typical measure to determine whether two sensors i and c are correlated, which is showed in Equation 5. Also, we have that if this function produces a larger value, the two sensors are more correlated.

$$H_{GK}(i,c) = exp(-\frac{\sum_{j \in \mathbf{o}_i \cap \mathbf{o}_c} ({TS^{N*M}}_{ij} - {TS^{N*M}}_{cj})^2}{2\sigma^2}). \tag{5}$$

- "Pearson Correlation Coefficient (PCC) " also can be used in our proposed regularization models to determine whether two sensors are correlated. A detailed explanation of method PCC can be found in research [29].
- "Dynamic Time Warping (DTW) " is also used to compare two different time series. The key to apply the DTW function is to make the warping cost minimum [33]. So we need to find a "warping path" to for the DTW function. Detail explanation of DTW can be found in [24].

$$H_{DTW}(i,c) = \frac{1}{DTW(\mathbf{o}_i, \mathbf{o}_c)}. \tag{6}$$

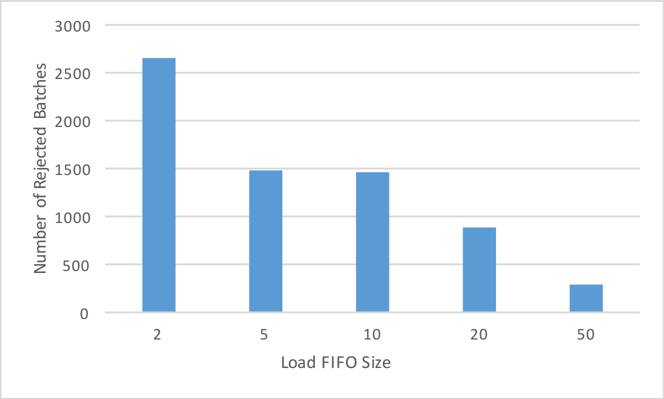


Fig. 6. The total number of rejected batches decreases with the increase of the size of load FIFO. In each condition, the rejected batches are all successfully resent by the buffering system and successfully put into the database.

- A constant function is used to show it is necessary to introduce a similarity function. The constant function is constructed as $H_{CF}(i,c) = C$. This constant function is also used a baseline function for the evaluation of different methods.

5 EXPERIMENTAL RESULTS AND ANALYSES

5.1 Experimental Setups

For purpose of illustrating the effectiveness of our proposed method SnrMF which is used in the DTSA framework, we use two typical data sets: the Motes data set (MDS) [28] and the Diagnostic Gases data set (DGDS) [29] in our simulation. For a more comprehensive evaluation, we run the simulation with different ratios of missing value. The purpose is to observe the tendency of prediction accuracy in different methods.

We use the RMSE (root mean squared error) to evaluate the quality of missing value prediction in comparing methods just as it is used in [29]. The definition of RMSE is showed in Equation 7 where ${TS^{N*M}}_{ij}$ is denoted as the observed value. We denote ${TS^{\hat{N}*M}}_{ij}$ as the predicted value corresponding to ${TS^{N*M}}_{ij}$. The W is denoted as an indicator matrix. We choose representative comparing methods in our simulation, which include method LI (linear interpolation), method NMF (non-negative matrix factorization) '[15], method PMF (probabilistic matrix factorization) [22], method MPMF (bayesian PMF) [27], method SVM (support vector machine) [29] and method SMSNR (simplified MSNR), which is extracted from the proposed method of SnrMF with $\alpha = 0$.

$$RMSE = \sqrt{\frac{\sum\limits_{i,j}(1-W_{ij})({TS^{N*M}}_{ij} - {TS^{\hat{N}*M}}_{ij})^2}{\sum\limits_{i,j}(1-W_{ij})}}, \tag{7}$$

We denote our proposed method SnrMF with different regularization models as SnrMF$_{CSbR}$ and SnrMF$_{USbR}$, respectively. We set parameter C as 1. The reason is based on the observation of Equation 2 and Equation 3 that parameter C is a constant value and has no influence on the final value of both equations. The parameter $C(i)$ and $C'(i)$ represent the set of correlated sensors. We

Table 1. Performance evaluation of our proposed SnrMF method and comparing methods.

	θ	LI	NMF	PMF	BPMF	SVM	SM	SnrMF$_{CSbR}$	SnrMF$_{USbR}$
Motes data set	0.1	84.1	80.6	85.8	78.8	63.8	34.5	16.78	25.9
	0.4	85.6	81.0	90.6	80.1	66.91	89.8	21.3	33.2
	0.6	86.1	83.9	92.14	81.0	66.9	97.3	49.3	60.9
	0.7	86.4	85.0	96.64	83.8	67.5	87.7	53.6	61.6
	0.8	86.8	87.1	112.22	85.5	72.4	81.1	49.9	60.3
	0.9	92.9	90.1	152.2	87.0	99.8	67.85	46.8	57.5
Diagnostic Gases data set	0.1	5.6	5.2	5.6	17.1	4.8	3.0	2.8	2.4
	0.4	10.3	9.3	18.1	17.2	5.9	2.7	2.6	2.5
	0.6	12.5	11.5	59.5	17.3	9.9	3.2	2.9	2.8
	0.7	13.4	12.4	82.4	17.3	12.1	3.2	3.2	3.0
	0.8	14.2	13.3	87.9	17.4	14.1	3.2	3.1	3.0
	0.9	14.9	14.0	99.6	17.4	15.4	3.2	3.2	3.0

denote $|C(i)|$ ($|C'(i)|$) as the number of the set. Besides, the parameters λ_1 and λ_2 are both set as the same value of λ in our simulation.

5.2 Experimental Results and Analyses

The experimental results are illustrated in Table 1 including our proposed SnrMF method and comparing methods. All numbers in Table 1 are the logarithm of RMSE.

For the SnrMF$_{CSbR}$ model, based on the observation of the results, we can conclude that our proposed SnrMF$_{CSbR}$ method outperforms the comparing methods using all different types of similarity functions. For example, when handling with the MDS and the ratio of missing value θ equals to 0.4, SnrMF$_{CSbR}$ outperforms the PMF method with an 85% lower RMSE. When the ratio of missing value is greater than 60%, the RMSE of our proposed SnrMF$_{CSbR}$ is still relatively lower than comparing methods. From the Table 1, we can observe the same tendency that our proposed SnrMF$_{CSbR}$ method outperforms comparing methods on the DGDS. The result reveals that our proposed SnrMF$_{CSbR}$ method is capable of predicting missing values more accuracy for both data sets and has a stable performance.

As for the proposed SnrMF$_{USbR}$ method, based on the observation of the results, we can conclude that the value of RMSE of method SnrMF$_{USbR}$ is lower than method SnrMF$_{CSbR}$ on the MDS. While on the DGDS, the prediction accuracy of method SnrMF$_{USbR}$ is worse than method SnrMF$_{USbR}$. The reason is that the MDS is produced from 54 different sensors thereby there is a much higher chance that it contains correlated time series while the DGDS is only generated from five different sensors. The results also reveal that our proposed SnrMF (SnrMF$_{USbR}$ and SnrMF$_{USbR}$) methods predict better than comparing methods on both data sets thus are feasible in both situations.

The experimental results demonstrate that our proposed SnrMF method substantially provides a support for dependable time series analytics by involving a missing value prediction.

5.3 Discussions on Similarity Functions and Parameters

In our proposed regularization models CSbR and USbR, the similarity function plays an important role because this function directly determines whether two network sensors are correlated. We

Table 2. Performance evaluation of SnrMF$_{CSbR}$ with different missing ratio θ and similarity functions.

	θ	VSS	GK	PCC	DTW	CF
Diagnostic Gases data set	0.1	40.47	24.66	**16.72**	24.93	56.45
	0.4	52.02	51.16	**21.40**	72.26	54.02
	0.6	62.20	85.82	**56.29**	92.23	56.44
	0.7	55.51	49.92	63.19	**33.40**	74.21
	0.8	80.84	99.06	91.03	**58.66**	91.01
	0.9	63.47	46.84	99.34	**42.50**	62.58
Motes data set	0.1	2.48	2.49	**2.41**	2.48	2.50
	0.4	2.61	2.62	**2.55**	2.62	2.59
	0.6	2.94	2.93	**2.90**	2.97	2.93
	0.7	3.08	3.01	2.98	**2.95**	3.09
	0.8	3.07	3.13	2.97	**2.86**	3.06
	0.9	3.06	3.16	2.98	**2.96**	3.09

would like to discuss the effects of different similarity functions. We have observed a similar tendency with using SnrMF$_{CSbR}$ and SnrMF$_{USbR}$ combining with different similarity functions. For the purpose of brief, here we only discuss the tendency of prediction using SnrMF$_{CSbR}$ method. As we have elaborated in precious sections, the purpose of the utilization of H function in our proposed methods is to determine whether two given network sensors are correlated. More precisely, the aim of function H is to find a correlated sensor set $C(i)$ or an uncorrelated sensor set $C'(i)$.

The performance evaluation of SnrMF$_{CSbR}$ with different similarity functions is showed in Table 2. Obviously, the performance of DTW outperforms other similarity functions when the value of missing ratio θ is greater than 0.6 while the method of PCC achieves a lower RMSE value when the value of missing ratio is least than 0.6. Based on this observation, we can make a reasonable conclusion that PCC considers different scales of various sensors which might contribute this tendency and when the missing ratio is high, DTW is capable of measuring the similarity better for two different time series. The results of constant function CF reveal that without an appropriate similarity function, the prediction cannot be good thereby indicate it is necessary to employ an appropriate similarity function.

Grid search is one of the traditional ways of performing hyper parameter optimization [5]. The performance tendency of SnrMF$_{CSbR}$ method with various parameters is showed in Figure 7 and 8. As illustrated, the number of the correlated sensors corresponding to sensor i_{th} is denoted as $|C(i)|$. As we can observe from Figure 7, when the value of $|C(i)|$ is 4, the value of RMSE produced by SnrMF$_{CSbR}$ method is 2.76 while when the value of $|C(i)|$ is set as 11, the SnrMF$_{CSbR}$ method produces a lower value 2.38 of RMSE on the MDS . On the DGDS (Figure 8), we observe a 58% lower RMSE value than the worst case, which is the best value of RMSE. Such observation enlighten us that a smaller $|C(i)|$ could be insufficient to guide the matrix factorization while a much bigger $|C(i)|$ could cause more noise thereby produce a worse prediction.

The influence of the matrix dimension L is also discussed. There is a totally different tendency in Figure 7 and Figure 8 when L varies. From a general perspective, for the MDS, a suited value of L is 4 while for the DGDS, a suited value of L is 1.

Parameter α determines the scale of the optimization problem. More precisely, the value of α indicates the volume of sensor network information which is involved in the problem. As has

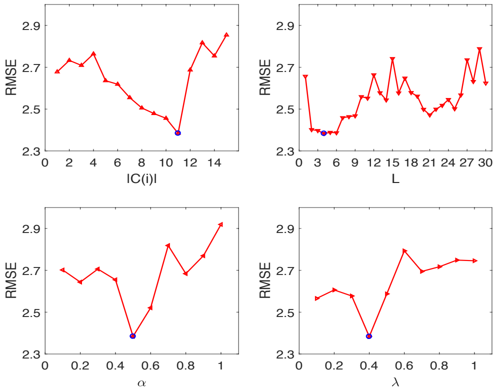


Fig. 7. Prediction Tendency with Different Parameters on the MDS (N=54, M=14000).

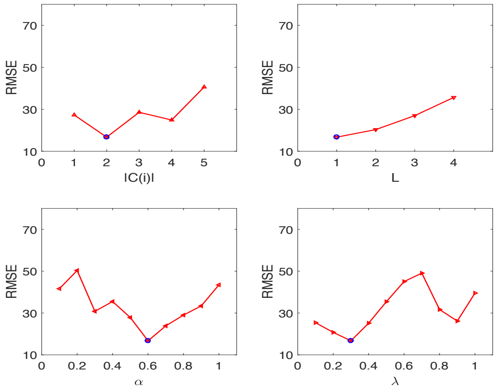


Fig. 8. Prediction Tendency with Different Parameters on the DGDS (N=5, M=500).

illustrated in Figure **??**, the RMSE is relative stable when the values of α vary. The best cases are observed when α equals to 0.5 on the MDS and 0.6 on the DGDS. Such observations indicate that a smaller α could deprecate the effect of the sensor regularization term while a larger α could cause a domination of the sensor regularization term thereby conversely damage the learning processes.

The influence of the penalty coefficient λ is also illustrated in Figure **??**. The experimental results demonstrate that appropriate values of λ can be set as 0.4 and 0.3 for the MDS and DGDS, respectively. Some variances are shown in these figures, but in general, it turns out the hyper parameters do not significantly affect the performance of the proposed method.

6 CONCLUSION

In this paper, we propose a dependable time series analytics (DTSA) framework for IoT-based SG. The proposed DTSA framework is composed of an extraction engine (EE) and a correction engine (CE). The EE is responsible for receiving raw time series from the CPS and preliminary refining the time series. To ensure the dependability of data transforming, we design a buffering system cooperating with the EE with a handshaking protocol. The CE is responsible for missing value prediction in the time series where we design a sensor-network-regularization based matrix factorization (SnrMF) method. By incorporating two different regulation models (CSbR and USbR) in SnrMF, the process of missing value prediction is obviously optimized. We verify our proposed DTSA framework on top of Hadoop and HBase platforms. The experiment results illustrate that our proposed DTSA framework is capable of effectively increasing the dependability of raw time series transforming between CPS and the target database system through an online light-weight extraction engine and an offline correction engine. Our proposed DTSA framework is feasible to support a dependable time series analytics for CPS.

ACKNOWLEDGMENTS

This research project is funded by the National Key Research and Development Program of China (2016YFE0100600), the National Natural Science Foundation of China (No.61373032), the National High Technology Research and Development Program of China (863 Program, 2015AA050204), the State Grid Science and Technology Project (520626140020, 14H100000552), State Grid Corporation of China, and the National Research Foundation Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme.

REFERENCES

[1] 2012. Apache hbase reference guide. (2012), 04–04 pages. https://wiki.apache.org/hadoop/Hbase/HbaseArchitecture

[2] Azza Abouzeid, Kamil Bajda-Pawlikowski, Daniel Abadi, Avi Silberschatz, and Alexander Rasin. 2009. HadoopDB: an architectural hybrid of MapReduce and DBMS technologies for analytical workloads. *Proceedings of the VLDB Endowment* 2, 1 (2009), 922–933.

[3] Oren Anava, Elad Hazan, and Assaf Zeevi. 2015. Online Time Series Prediction with Missing Data. In *Proceedings of the 32nd International Conference on Machine Learning (ICML '15)*, David Blei and Francis Bach (Eds.). JMLR Workshop and Conference Proceedings, 2191–2199. http://jmlr.org/proceedings/papers/v37/anava15.pdf

[4] P. Baraldi, F. Di Maio, D. Genini, and E. Zio. 215. Reconstruction of missing data in multidimensional time series by fuzzy similarity. *Applied Soft Computing* 26 (215), 1 –9. DOI: https://doi.org/10.1016/j.asoc.2014.09.038

[5] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for Hyper-parameter Optimization. In *Proceedings of the 24th International Conference on Neural Information Processing Systems (NIPS'11)*. 2546–2554.

[6] Yongjie Cai, Hanghang Tong, Wei Fan, and Ping Ji. Fast Mining of a Network of Coevolving Time Series. In *Proceedings of the 2015 SIAM International Conference on Data Mining*. 298–306. DOI: https://doi.org/10.1137/1.9781611974010.34 arXiv:http://epubs.siam.org/doi/pdf/10.1137/1.9781611974010.34

[7] Dorin Carstoiu, Elena Lepadatu, and Mihai Gaspar. 2010. Hbase-non sql database, performances evaluation. In *in Computer Science (1986), Master in Computer Science (1990), and PhD in Computer Science*. Citeseer.

[8] Xiaoying Chen, Chong Zhang, Bin Ge, and Weidong Xiao. 2015. Spatio-temporal queries in HBase. In *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*. 1929–1937. DOI: https://doi.org/10.1109/BigData.2015.7363970

[9] Mohamed Y. Eltabakh, Fatma Özcan, Yannis Sismanis, Peter J. Haas, Hamid Pirahesh, and Jan Vondrak. 2013. Eagle-eyed elephant. In *Proceedings of the 16th International Conference on Extending Database Technology - EDBT '13*. 89. DOI: https://doi.org/10.1145/2452376.2452388

[10] Lars George. 2011. *HBase the Definitive Guide*. 556 pages.

[11] Josif Grabocka, Alexandros Nanopoulos, and Lars Schmidt-Thieme. 2012. Classification of Sparse Time Series via Supervised Matrix Factorization. (2012).

[12] François Jammes and Harm Smit. 2005. Service-oriented paradigms in industrial automation. *IEEE Transactions on Industrial Informatics* 1, 1 (2005), 62–70.

[13] Maninder Kaur and Sheetal Kalra. 2016. A Review on IOT Based Smart Grid. *International Journal of Energy, Information and Communications* 7, 3 (2016), 11–22.

[14] Kyoung-Dae Kim and Panganamala R Kumar. 2012. Cyber–physical systems: A perspective at the centennial. *Proc. IEEE* 100, Special Centennial Issue (2012), 1287–1308.

[15] Daniel D. Lee and H. Sebastian Seung. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature* 401 (1999), 788–791.

[16] M. Lippi, M. Bertini, and P. Frasconi. 2013. Short-Term Traffic Flow Forecasting: An Experimental Comparison of Time-Series Analysis and Supervised Learning. *IEEE Transactions on Intelligent Transportation Systems* 14, 2 (2013), 871–882.

[17] Bin Liu, Yongxin Zhu, Chang Wang, Yufeng Chen, Tian Huang, Weiwei Shi, Mengjun Li, and Yishu Mao. 2016. A Versatile Event-Driven Data Model in HBase Database for Multi-source Data of Power Grid. In *Smart Cloud (SmartCloud), IEEE International Conference on*. IEEE, 208–213.

[18] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender Systems with Social Regularization. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining (WSDM '11)*. ACM, 287–296. DOI:https://doi.org/10.1145/1935826.1935877

[19] Kun Ma and Bo Yang. 2016. Column Access-aware In-stream Data Cache with Stream Processing Framework. *Journal of Signal Processing Systems (JSPS)* (2016), 1–15.

[20] Fumio Machida, Jianwen Xiang, Kumiko Tadano, Yoshiharu Maeno, and Takashi Horikawa. 2013. Performability analysis of RAID10 versus RAID6. In *Proceedings of the International Conference on Dependable Systems and Networks*. 1–2. DOI:https://doi.org/10.1109/DSN.2013.6575339

[21] Nicolas Méger, Christophe Rigotti, and Catherine Pothier. 2015. Swap Randomization of Bases of Sequences for Mining Satellite Image Times Series. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*. Springer, 190–205.

[22] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic Matrix Factorization. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 1257–1264.

[23] Olivier Monnier. 2014. *A smart grid with the internet of things*. Technical Report. http://www.ti.com/lit/ml/slyb214/slyb214.pdf

[24] Meinard Müller. 2007. Dynamic time warping. *Information retrieval for music and motion* (2007), 69–84.

[25] Connie Phong and Raul Singh. 2008. Missing Value Estimation for Time Series Microarray Data Using Linear Dynamical Systems Modeling. In *AINAW 2008*. IEEE, 814–819.

[26] R. Rajkumar, Insup Lee, Lui Sha, and J. Stankovic. 2010. Cyber-physical systems: The next computing revolution. *47th ACM/IEEE Design Automation Conference (DAC)* (2010), 731–736. DOI:https://doi.org/10.1145/1837274.1837461

[27] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian Probabilistic Matrix Factorization Using Markov Chain Monte Carlo. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. ACM, 880–887. DOI:https://doi.org/10.1145/1390156.1390267

[28] Madden Samuel. Intel Lab Data. http://db.csail.mit.edu. (????).

[29] Weiwei Shi, Yongxin Zhu, Jinkui Zhang, Xiang Tao, Gehao Sheng, Yong Lian, Guoxing Wang, and Yufeng Chen. 2015. Improving Power Grid Monitoring Data Quality: An Efficient Machine Learning Framework for Missing Data Prediction. In *HPCC*. IEEE, 417–422.

[30] Wu Shu-wen. 2011. Research on the key technologies of IOT applied on smart grid. In *Electronics, Communications and Control (ICECC), 2011 International Conference on*. IEEE, 2809–2812.

[31] Yunlong Song, Min Liu, Shaojie Tang, and Xufei Mao. 2012. Time series matrix factorization prediction of internet traffic matrices. In *2012 IEEE 37th Conference on Local Computer Networks (LCN)*. 284–287. DOI:https://doi.org/10.1109/LCN.2012.6423629

[32] Giorgio Luigi Valentini, Walter Lassonde, Samee Ullah Khan, Nasro Min-Allah, Sajjad A. Madani, Juan Li, Limin Zhang, Lizhe Wang, Nasir Ghani, Joanna Kolodziej, Hongxiang Li, Albert Y. Zomaya, Cheng-Zhong Xu, Pavan Balaji, Abhinav Vishnu, Fredric Pinel, Johnatan E. Pecero, Dzmitry Kliazovich, and Pascal Bouvry. 2013. An overview of energy efficiency techniques in cluster computing systems. *Cluster Computing* 16, 1 (2013), 3–15. DOI:https://doi.org/10.1007/s10586-011-0171-x

[33] Xiao Pan et el Weiwei Shi, Yongxin Zhu. 2016. Missing Data Prediction in Multi-source Time Series with Sensor Network Regularization. https://aaltd16.irisa.fr/files/2016/08/AALTD16_paper_2.pdf

[34] Shin-Fu Wu, Chia-Yung Chang, and Shie-Jue Lee. 2015. Time series forecasting with missing values. In *2015 1st International Conference on Industrial Networks and Intelligent Systems (INISCom)*. 151–156.

[35] Miao Yun and Bu Yuxin. 2010. Research on the architecture and key technology of Internet of Things (IoT) applied on smart grid. In *2010 International Conference on Advances in Energy Engineering, ICAEE 2010*. 69–72. DOI:https://doi.org/10.1109/ICAEE.2010.5557611

[36] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: cluster computing with working sets. *HotCloud* 10 (2010), 10–10.