KBMP: Kubernetes-Orchestrated IoT Online Battery Monitoring Platform

Qingyang Chen, Yinghui He, *Member, IEEE*, Guanding Yu, *Senior Member, IEEE*, Chong Xu, Mingyang Liu, and Zhenming Li

***Abstract*—The rise in renewable energy has driven the widespread use of large-scale energy storage batteries, which makes the risk of overheating more threatening. To ensure battery safety, it is essential to build a monitoring system with a comprehensive evaluation of large quantities of batteries. However, existing battery management systems exhibit significant limitations in terms of monitoring scope, analytical precision, and transmission efficiency. As an applicable solution, cloud–edge technology is an advanced integrated method that provides low-latency data access, accurate analysis capabilities, and adjustable monitoring ranges. In this work, the Kubernetes-orchestrated battery monitoring platform (KBMP), which integrates Kubernetes and cloud–edge technology, is proposed to provide comprehensive battery management. Specifically, Kubernetes is used to ensure low latency in data transmission and analysis, while the *K*-Means clustering algorithm is applied to provide accurate thermal runaway (TR) warnings. To validate the performance of KBMP, four sets of real battery TR data are fed to test its accuracy and latency. The experimental findings reveal that KBMP is capable of providing battery TR warnings in advance within 30 min. Additionally, the platform concurrently decreases data transmission latency by up to 20% and reduces replica scaling latency by 50% compared to the platform without integrating Kubernetes.**

***Index Terms*—Battery monitoring platform, cloud–edge technology, *K*-Means clustering algorithm, Kubernetes, thermal runaway (TR) prediction.**

I. Introduction

As THE popularity of electric vehicles, renewable energy storage systems, and mobile devices increases, the application of large-scale energy storage battery systems becomes more widespread [1]. Its applications cover various fields, including transportation, energy storage, and mobile communications [2], [3]. However, with the widespread adoption of large-scale energy storage battery systems, the demand for battery thermal safety is increasing [4]. During charging, discharging, or prolonged use of battery, a large amount of heat can be generated [5]. Overheating, which is a very likely outcome, can lead to fires and explosions, posing a significant threat to human life and property [6], [7], [8]. In the past single year, there have been more than 30 energy storage station fires worldwide [9], each of which can lead to casualties and property losses up to more than U.S. $1.5 million [10]. Therefore, it is of paramount importance to develop a reliable battery monitoring system for thermal runaway (TR) warnings.

To date, most of the research on battery monitoring has focused on battery management systems (BMSs), which are deployed on batteries and use sensors to real-time monitor parameters of battery packs [11]. However, due to various limitations in practical application scenarios, BMS hardware resources are extremely limited [12]. As a result, research about combining BMSs with Internet of Things (IoT) technology has emerged, enabling the transmission of battery data to hardware devices with richer computing resources for analysis and processing. Friansa et al. [13] developed an IoT-based BMS that can upload battery data to cloud servers, and the transmission time is less than 20 s. Helmy et al. [14] developed a BMS with global message services (GMSs), which is capable of sending integrated information, such as battery condition, location, and time. Assad et al. [15] proposed a real-time BMS equipped with state-of-charge (SOC) estimation algorithm, and it uses the message queuing telemetry transport (MQTT) protocol for uploading battery data to the cloud server and therefore realized data persistence. However, the monitoring capabilities of the aforementioned IoT-based BMS are limited to individual battery cells or battery packs. When it comes to monitoring large quantities of batteries, the only solution currently available is replicating and stacking hardware devices, which is impractical for actual production processes.

Today, cloud–edge technologies have gradually matured, serving as an integrated technology that combining cloud computing, edge computing, and terminal computing [16], [17]. It extends the capabilities of computing, storage, and applications, providing faster data processing and decision-making capabilities in real time [18], [19], [20]. Therefore, incorporating cloud–edge technology into large-scale battery monitoring research would be an inspiring endeavor. Zhu et al. [21] proposed a cloud–end collaboration BMS framework to predict battery SOC using cloud computing and machine learning, and the analysis accuracy and framework flexibility of the method were validated. Li et al. [22] presented a cloud

Manuscript received 23 October 2023; revised 11 January 2024 and 1 April 2024; accepted 23 April 2024. Date of publication 30 April 2024; date of current version 9 July 2024. This work was supported by the Science and Technology Program from State Grid Corporation of China through "Research on Intelligent Monitoring Technology of Energy Storage Lithium-Ion Battery" under Grant 5500-202255364A-2-0-ZN. *(Corresponding author: Yinghui He.)*

Qingyang Chen, Yinghui He, and Guanding Yu are with the College of Information and Electronic Engineering, Zhejiang University, Hangzhou 310027, Zhejiang, China (e-mail: 22231156@zju.edu.cn; 2014hyh@zju.edu.cn; yuguanding@zju.edu.cn).

Chong Xu, Mingyang Liu, and Zhenming Li are with the Energy Storage and Novel Technology of Electrical Engineering Department, China Electric Power Research Institute Company Ltd., Beijing 100192, China (e-mail: xuchong@epri.sgcc.com.cn; liumingyang@epri.sgcc.com.cn; lizhenming@epri.sgcc.com.cn).

Digital Object Identifier 10.1109/JIOT.2024.3395501

TABLE I
KBMP COMPARED WITH OTHER CLOUD BMSS

Cloud BMS	Hardware Implementation	Large Monitoring Scope	Flexible Scalability	Platform Management	Battery Fault Protection
KBMP	✓	✓	✓	✓	✓
[21]	✓	✓	×	×	×
[22]	✓	✓	×	×	×
[23]	✓	✓	×	×	✓
[24]	×	✓	✓	×	✓

BMS for battery system which was capable of accurate SOC and state-of-health (SOH) estimation. Adhikaree et al. [23] proposed a small-scale cloud BMS simulator for SOC estimation using Raspberry pi boards and Google cloud, and therefore improved monitoring scope, computing efficiency of the battery energy storage systems. Yang et al. [24] proposed a conceptual BMS framework with edge–cloud architecture, using the Cyber Hierarchy and Interactional Network framework to achieve comprehensive data analysis and visualization capabilities.

However, the aforementioned works focused solely on transmitting battery data to the cloud for processing, without addressing the unified management of different data transmission nodes and data analysis modules. The monitoring was limited to SOC, neglecting battery safety and the potential congestion issue arising from the influx of large volumes of battery data. The challenges in effectively managing the continuous flow of extensive battery data include ensuring seamless, low-latency transmission, precise analysis, and streamlined management within the cloud–edge–end battery platform. To address these challenges, Kubernetes-orchestrated battery monitoring platform (KBMP) is first proposed in this article. By combining Kubernetes, an open-source container orchestration and management platform [25], with cloud–edge technologies, cross-level management and deployment of containerized applications can be achieved. KBMP is built with a four-layer architecture: 1) data receiving; 2) data storage; 3) data analysis; and 4) data visualization. Tailored solutions are proposed at each layer to ensure the high-performance implementation of their respective functionalities. As shown in Table I, among the above methods, KBMP has more comprehensive performance compared to the previous cloud BMSs, balancing multiple functions, such as large monitoring range, high scalability, convenient platform management, and accurate battery anomaly diagnosis.

The main contributions of this article are summarized as follows.

1) A Kubernetes-based platform for monitoring battery TR warnings, namely, KBMP, is built. KBMP enables fast data transmission and analysis, possesses high scalability, and facilitates comprehensive cluster management.
2) To achieve precise and timely battery TR warnings, a time-series-based K-Means clustering algorithm is proposed and deployed at the data analysis layer. Real-time data visualization is accomplished through the utilization of the InfluxDB database and Grafana dashboard within the platform.
3) Validation using four data sets of realistic TR data is performed. The results verify KBMP's exceptional precision in battery TR warnings, providing warnings 30 min in advance. Furthermore, the platform significantly reduces data transmission latency by up to 20% and replica scaling latency by 50% compared to the non-Kubernetes-integrated platform.

The subsequent sections of this article are organized as follows. In Section II, the specific design ideas and implementation plans of KBMP are introduced. Section III elaborates on the hardware and software deployment schemes of KBMP. Section IV uses real battery data for platform performance testing, including data transmission latency, data analysis latency, platform scalability, and TR accuracy. Finally, Section V concludes this article and highlights some future research directions.

II. KBMP DESIGN

In this section, a detailed introduction to the architecture and orchestration methods of KBMP is provided. The specific functions of each layer in KBMP are elaborated, and the applications implementing each function, as well as the deployment plans for these applications, are thoroughly presented.

A. KBMP Orchestration

To achieve accurate battery anomaly diagnosis, the BMS needs to have a high sampling frequency [26] and the capability to store historical battery data for a certain period of time. However, due to the limited hardware resources of the BMS, it lacks the capability to store a sufficient amount of battery data. Therefore, relying solely on the BMS to achieve accurate battery safety monitoring is unrealistic. Surplus battery data needs to be uploaded to a cloud platform for processing.

As depicted in Fig. 1, multiple lithium-ion batteries generate a substantial volume of data which is directly collected by the BMSs. Due to the excessive amount of data generated, which exceeds the processing capacity of the BMSs, the surplus data will be directly uploaded to the battery monitoring platform deployed at the cloud for further analysis. In order to avoid data accumulation, the cloud platform needs to quickly receive and store battery data. At the same time, it also needs to efficiently perform data analysis to prevent TR from happening and visualize the processing status.

In order to achieve the aforementioned multiple functionalities, it is inevitable to deploy multiple applications on the platform, which involves intensive data communication

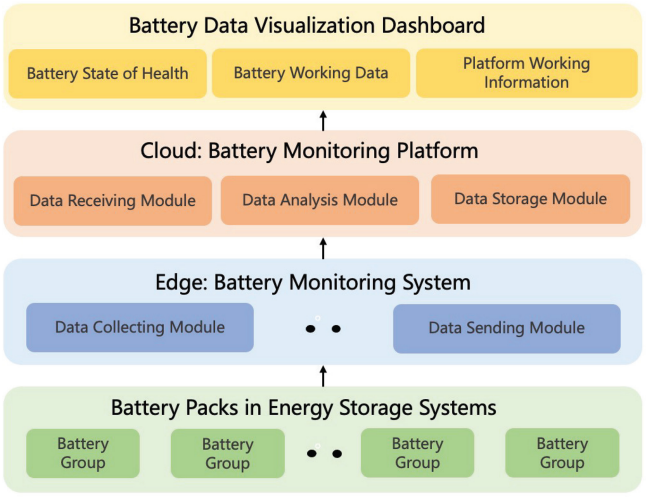


Fig. 1. Workflow of cloud–edge solution for battery monitoring.

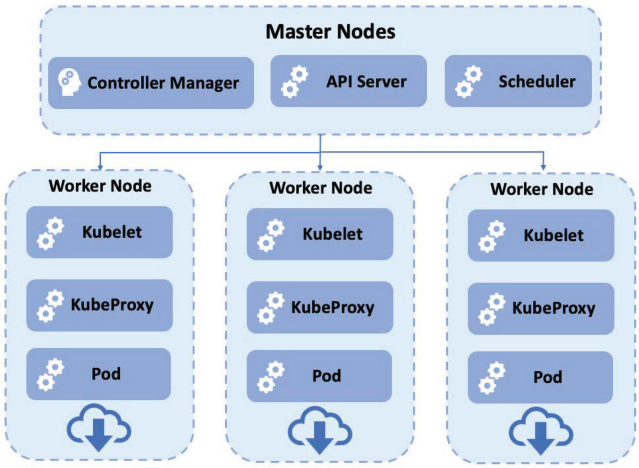


Fig. 2. Illustration of Kubernetes cluster.

between applications. Therefore, an efficient orchestration framework is needed to effectively manage and coordinate these applications. Therefore, Kubernetes becomes a potential solution, managing resource allocation and overall scheduling of multiple containers across the four layers.

Kubernetes is an open-source container orchestration platform that automates the deployment, scaling, and management of containerized applications. It allows users to easily manage and scale applications running in containers across clusters of hosts. A Kubernetes cluster comprises a collection of machines known as nodes. These nodes are categorized into master nodes and worker nodes. The master node is mainly responsible for cluster management and task allocation, while the worker node is mainly responsible for undertaking and implementing different tasks. As shown in Fig. 2, after setting up Kubernetes on the master node and the worker nodes, some applications are automatically deployed to complete cluster management and node communication. Specifically, the master node utilizes the control manager and scheduler to achieve cluster management, and the tasks are distributed to the worker nodes through the API Server. The Kubelet in worker nodes

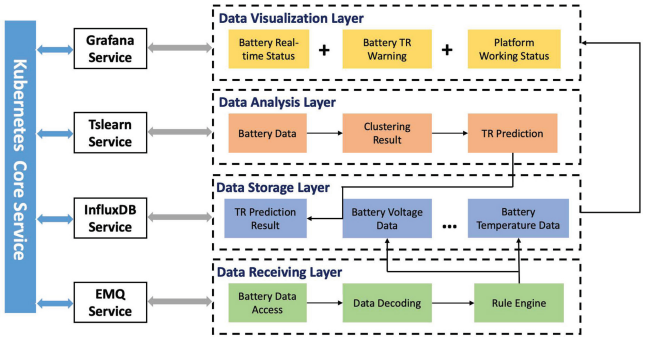


Fig. 3. Multilayer architecture of KBMP.

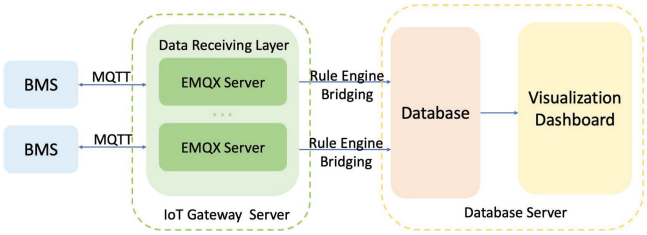


Fig. 4. Diagram of the data receiving layer workflow.

automatically downloads and executes the received tasks in Pods, while external access is facilitated through Kube-Proxy.

With the help of Kubernetes, servers responsible for different functions are gathered into a Kubernetes cluster, which can be uniformly monitored and managed, and communication between server nodes can be carried out in an orderly manner. This Kubernetes cluster is exactly the KBMP proposed in this article, which can achieve fast battery data reception and storage, battery TR warning, and battery visualization management functionalities. As shown in Fig. 3, KBMP can be categorized into four distinct layers: 1) the data reception layer; 2) the data storage layer; 3) the data analysis layer; and 4) the data visualization layer. After the BMS uploads battery data, the data is first received by the data reception layer, then processed through the data storage layer, and finally, the data analysis layer is responsible for implementing the TR warning. The results can be visualized through the data visualization layer. Each layer is equipped with suitable applications to fulfill their respective tasks and is distributed across different worker nodes within the Kubernetes cluster.

B. Data Receiving Layer

Receiving battery data uploaded by the BMS is the first step that KBMP needs to accomplish, and this step is performed by the data reception layer. Since battery data is always transmitted using a specific IoT communication protocol, the data reception application deployed on this layer needs to have the capability to efficiently decode the protocol.

In this work, EMQX is used to address the above problem. A lightweight IoT data access based on EMQX is established on the data reception layer. It supports multiple IoT protocols for data collection and parsing, including widely used ones like MQTT, CoAP [27], and JT808 [28]. Additionally, EMQX provides a custom rule engine based on the structured query language (SQL) for data republishing. As depicted in Fig. 4,

EMQX servers will be deployed on the IoT gateway servers (each IoT gateway server represents a worker node in the Kubernetes cluster), facilitating the real-time access, decoding, and forwarding of battery data via custom resource types.

C. Data Storage Layer

The data storage layer includes a database to ensure the persistent storage of battery data. As previously mentioned, battery state analysis necessitates access to not only the present data but also the historical data. Consequently, this database should be able to store battery data as a time series, making InfluxDB, an open-source temporal database, the preferred solution. Equipped with InfluxDB, the data storage layer has the capability to store and retain all historical data. Furthermore, it enables querying of data based on the timeline as a filtering condition.

Besides, the data storage layer is established on the database server, which represents a worker node in the Kubernetes cluster.

D. Data Analysis Layer

1) Layer Workflow: The data analysis layer, being the pivotal component of KBMP, implements the TR warning functionality for batteries. This layer comprises multiple worker nodes, each hosting several replicas of data analysis applications. The workflow of this layer involves retrieving the most recent battery data from the InfluxDB database, conducting battery data analysis to detect TR, and subsequently uploading the alert results back to the database.To accomplish the data storage function, the Python library InfluxDB-Client is utilized, enabling remote reading and uploading of data from the database.

For the data analysis, the K-Means clustering algorithm is employed, specifically tailored for time-series data, to achieve real-time detection of TR in batteries. The TR warning algorithm based on K-Means clustering offers a significant advantage in terms of its robust transferability. It eliminates the need for battery modeling or pretraining of neural networks. Consequently, this algorithm can be applied for TR warning across diverse battery models within an energy storage station. Furthermore, the algorithm demonstrates a notable level of fault tolerance. In the event of a disruption in battery data upload due to an incident, it maintains the accuracy of subsequent analysis and predictions without a significant impact.

2) K-Means Clustering Algorithm for Time-Series Data: Given an observation set $\{x_1, x_2, x_3, \ldots, x_n\}$ where each observation is a d-dimensional real vector, the K-Means clustering aims to partition these n observations into k clusters ($k \leq n$). Therefore, the sum of distances between each point and its corresponding cluster center is minimized. The process of the K-Means clustering algorithm is as follows [29].

1) *Initialization:* Randomly select k observations from the data set as initial cluster centers.
2) *Assignment:* For each observation x_i and each cluster center μ_j, calculate the dynamic time warping (DTW) distance between them

$$d_{\min} = \min(\mathbf{DTW}(x_i, \mu_{j-1}), \mathbf{DTW}(x_{i-1}, \mu_j) \\ \mathbf{DTW}(x_{i-1}, \mu_{j-1}) + d(x_i, \mu_j)). \quad (1)$$

3) *Update:* For each cluster, calculate the average shape of all observations within that cluster and set it as the new cluster center

$$\mu_j = \text{mean}(x_i) \text{ for } x_i \textbf{ in } \text{Cluster } j. \quad (2)$$

4) Repeat steps 2) and 3) until the termination condition is met.
5) *Evaluation:* Evaluate the quality of the clustering using the silhouette coefficient. After completing the clustering, compute the silhouette coefficient for each vector in the clusters. Taking point i as an example, the average distance $a(i)$ from the vector to all other points within the cluster it belongs to should be calculated, as well as the average distance $b(i)$ from the vector to all points within different clusters that do not include it. Finally, its silhouette coefficient is calculated as

$$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}. \quad (3)$$

In step 2), calculating the DTW distance requires aligning two time series to find the best match. This involves computing a dynamic programming matrix to determine the minimum cost path between the two sequences. DTW distance takes into account the shape differences between time series and allows for stretching and compression along the time axis [30]. Overall, using DTW as the distance calculation method in the K-Means clustering algorithm is more suitable for clustering time-series data as it considers the shape variations between time series.

3) Battery Thermal Runaway Prediction Based on K-Means Clustering Algorithm for Time-Series Data: Battery TR behavior refers to a self-reinforcing and uncontrollable increase in temperature within a battery cell, leading to the release of energy, which can result in further temperature rise. This can lead to a destructive chain reaction within the battery. Feng et al. [31] characterized the TR process by three temperature. In particular, T_1 is the starting temperature for detecting the self heating of the battery, at which point the solid electrolyte interface (SEI) inside the battery begins to decompose, T_2 is the triggering temperature of TR, at which point the battery's internal separator failure triggers an internal short circuit (ISC), ultimately leading to TR, T_3 is the highest temperature for TR. Therefore, there is a strong correlation between battery temperature and TR behavior. If the abnormality of battery temperature can be detected promptly, it will greatly help predict TR.

The temperature of a battery continuously changes over time. Transmitting, storing, and analyzing the relevant thermal data of the battery in the form of time series is beneficial for predicting its future changes. The K-Means algorithm can be utilized to cluster the time-series temperature data of individual battery cells within a battery pack, and the clustering results can be used to characterize the battery's working state. There is a certain degree of similarity between

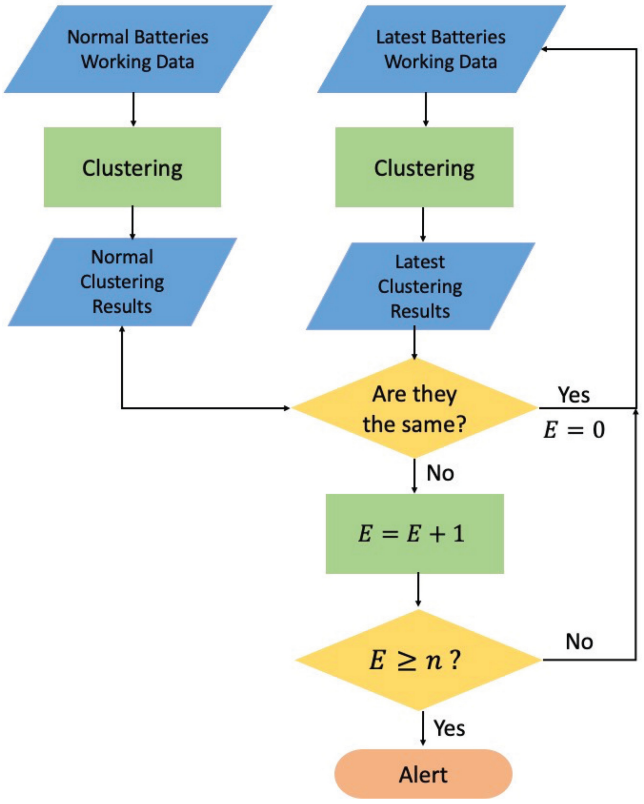


Fig. 5. Workflow of battery TR prediction based on the K-means algorithm.

some individual battery cells since high consistency can extend the batteries' service life and ensure the safety of the battery pack's operation [32]. Those battery cells or battery points also exhibit similar performance during normal charging and discharging processes. Therefore, when using the K-Means algorithm to process these battery data, the data produced by battery cells/points with similar performance will be grouped together into the same cluster. When one of the batteries/points experiences an abnormality, its performance will be different from the other batteries/points in the original cluster, and eventually an "outlier" behavior occurs, meaning that this battery/point no longer belongs to the previous cluster. In this experiment, K-Means is used to process battery temperature data. When temperature data is abnormal, TR is a highly likely result. Therefore, when the clustering results change, a TR warning will be triggered.

The algorithm flowchart is shown in Fig. 5, where E represents the number of times a battery is marked as abnormal. Assuming the experimental subject is an individual battery cell, the input to the algorithm consists of continuous temperature data from different surfaces of the battery cell. If the subject is a battery pack, the input data is continuous temperature data from different battery cells within the pack. First, the algorithm takes the offline normal resting, charging and discharging data of the battery as input. The K-Means clustering algorithm produces three corresponding clustering results, which can be recorded as the "normal clustering results." It is worth noting that the normal clustering results needs to be regularly updated using the latest battery data to address changes in battery performance due to aging [33], ensuring the accuracy of the algorithm. Then, real-time working data of the battery cell is input, and the clustering algorithm performs clustering on the temperature data from the past t seconds, updating the clustering result every second. Each time a clustering result is generated, it is compared to the normal clustering result. If it is consistent with the normal clustering result, the battery is considered normal. If there is a change compared to the normal clustering result, it is recorded as an anomaly. When the battery status is marked as abnormal for consecutive n times, it is diagnosed as a prelude to TR, and the platform issues an alert.

It is noteworthy that, despite exclusively employing temperature data as input in subsequent experiments, the proposed algorithm exhibits robust versatility. Other parameters associated with the battery, including voltage, current, internal gas concentration, etc., can equally serve as inputs, facilitating the assessment of potential abnormalities in the battery.

E. Data Visualization Layer

The data visualization layer provides a visual representation of the battery's operating status and alerts for TR, enhancing convenience and efficiency in safety management practices.

To fulfill this functionality, Grafana is commonly utilized as a complementary tool, forming a comprehensive visualization system. A Dashboard is built based on Grafana, enabling the data visualization layer to create, share, and explore data in a sophisticated and efficient manner. In the proposed platform, the data visualization layer dynamically retrieves the most recent operational data of the batteries and the results of TR alerts stored in the data storage layer. Subsequently, it presents this data to users in various chart and graph formats, tailored to their preferences.

Due to the high data interdependence between the data reception layer and the data visualization layer in their operational workflow, the data visualization layer is also deployed on the database server.

III. Platform Implementation

In this section, the practical implementation methods of KBMP will be discussed. Fig. 6(a) illustrates the platform workflow and the data processing workflow.

KBMP is built on a Kubernetes cluster architecture, consisting of a master node, two worker nodes, a database server, and multiple IoT broker servers. The integration of these edge nodes into a unified cluster is achieved through the utilization of the Kubernetes container orchestration system. The master node is responsible for running control plane components, such as the Kubernetes controller-manager, while the worker nodes deploy control components, including Kubelet, along with the necessary components for each layer of KBMP. These components engage in efficient data communication to enable timely battery TR alerting.

The IoT broker servers, which are lightweight servers based on ARM architecture, also deploy components like Kubelet. Their primary function is to facilitate the distributed

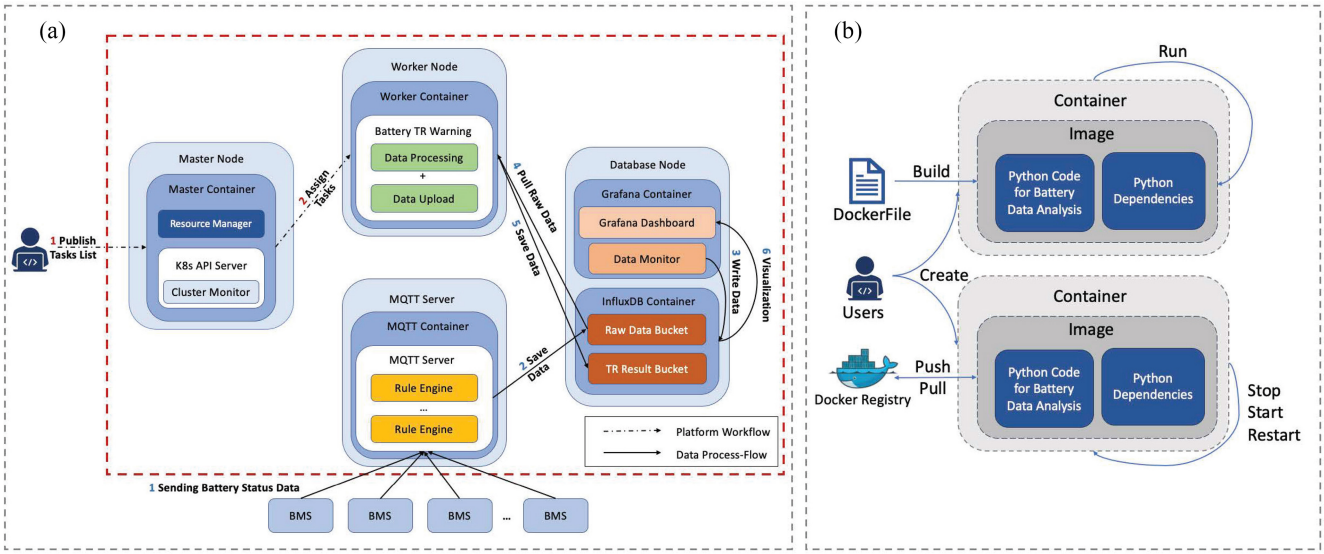


Fig. 6. Diagram of KBMP implementation. (a) Illustration of KBMP's workflow and the data processing workflow. (b) Illustration of the process for building and running images with docker.

deployment of the real-time data access layer for IoT. They directly handle real-time IoT application protocol data reported by end devices, specifically the latest working state data of the batteries.

The database server serves as the host for the InfluxDB time-series database and Grafana, which, respectively, provide data storage and data visualization services for KBMP. Furthermore, the master node can establish communication with cloud servers in Kubernetes controller mode and continuously upload the working status of each worker node to the cloud.

TABLE II
NODE INFORMATION OF KBMP

Name	Status	Roles	Version
IoT-server1	Ready	Worker	v1.27.2+k3s1
IoT-server2	Ready	Worker	v1.27.2+k3s1
k3s-node1	Ready	Worker	v1.27.2+k3s1
k3s-node2	Ready	Worker	v1.27.2+k3s1
database-server	Ready	Worker	v1.27.2+k3s1
k3s-master	Ready	Control-Plane, Master	v1.27.2+k3s1

A. Software Implementation

Considering the substantial population of batteries under surveillance, it is imperative to deploy multiple replicas of the application within each layer to facilitate parallel data processing. Consequently, it becomes crucial to mitigate interference among diverse applications spanning across different layers. As mentioned in the previous section, Kubernetes is an orchestration framework used to manage containerized applications. Therefore, before deploying the software into the Kubernetes cluster, they need to be packaged in containers. Docker, an open-source application container engine, is utilized to address this challenge. Docker empowers developers to encapsulate applications and their dependencies into an image, facilitating their execution within isolated containers. Fig. 6(b) demonstrates how to create the image of the data analysis application using Docker, which includes a primary Python program along with relevant dependencies.

Containerized deployment obviates concerns regarding the environment and guarantees autonomous operation of containers, thereby enhancing the convenience of application management [34]. Therefore, all subsequent applications are deployed and run within containers created by the Docker.

Additionally, KBMP's applications and components are orchestrated using the edge-native engine K3s, which is a lightweight Kubernetes distribution with a memory footprint of just half and a binary size under 100 MB, making it suitable for edge computing scenarios. K3s allows for easy deployment of Kubernetes clusters on both ARM architecture-based Raspberry Pi devices and X86 architecture-based servers, and it features an efficient command-line interface.

In this architecture, the K3s server component is deployed on the master node, while the K3s agent component is deployed on worker nodes and IoT broker server nodes. The Kubernetes deployment type is employed for distributing components across layers, and Kubernetes services enable seamless data transmission among components.

Table II illustrates node details within the Kubernetes cluster, and Table III depicts the operation of the aforementioned applications within each layers. Notably, all the applications exhibit normal operating status. The term "k3s-node" signifies working nodes, "Database server" represents database server nodes, and "IoT-server" designates lightweight Raspberry Pi IoT gateway server nodes. The EMQX component operates as Kubernetes Deployment objects within the K3s cluster, and intercommunication among distributed points is realized through direct access to domain names.

TABLE III
APPLICATION STATUS OF KBMP

Name	Ready	Status	Restarts	Age	IP	Node
emq-0	1/1	Running	1(3d ago)	5d	10.42.4.13	IoT-server1
emq-1	1/1	Running	1(3d ago)	5d	10.42.4.11	IoT-server2
mqtt-receiver-0	1/1	Running	3(3d ago)	5d	10.42.4.23	IoT-server1
mqtt-receiver-1	1/1	Running	0	5d	10.42.4.21	IoT-server2
tr-prediction-84ddbf5794-k3xtn	1/1	Running	2(1d5h ago)	9d	10.42.2.0	k3s-node1
tr-prediction-2kn8eked32-r2knd	1/1	Running	2(1d5h ago)	9d	10.42.2.9	k3s-node1
tr-prediction-76dbcc684c-nkq7n	1/1	Running	2(1d5h ago)	9d	10.42.2.7	k3s-node2
tr-prediction-892hde9110-l39j5	1/1	Running	2(1d5h ago)	9d	10.42.2.6	k3s-node2
indluxdb-978c885c-937fg	1/1	Running	3(1h ago)	10d	10.42.0.4	database-server
grafana-684b55c65c-9qcj5	1/1	Running	3(1h ago)	10d	10.42.0.5	database-server

TABLE IV
HARDWARE ENVIRONMENT OF KBMP

Server Name	IoT Server	Worker Node	Master Node	Database Server
Hardware	Raspberry 4B	X86 Server	X86 Server	X86 Server
CPU	Core Number: 4 Thread Number: 4 Architecture: ARM64 V8 Processor: Cortex-A72 Soc: 64-bit, 1.5GHz BogoMIPS: 108 Max MHZ: 1500	Core Number: 4 Thread Number: 4 Architecture: ARM64 V8 Processor: Cortex-A72 Soc: 64-bit, 1.5GHz BogoMIPS: 108 Max MHZ: 1500	Core Number: 4 Thread Number: 4 Architecture: ARM64 V8 Processor: Cortex-A72 Soc: 64-bit, 1.5GHz BogoMIPS: 108 Max MHZ: 1500	Core Number: 6 Thread Number: 6 Processor: Intel Core i5 OS Type: 64 BogoMIPS: 6000 Max MHZ: 4100
Storage	Memory Space: 4GB Storage Space: 32GB	Memory Space: 7.62GB Storage Space: 250GB	Memory Space: 7.62GB Storage Space: 129GB	Memory Space: 4GB Storage Space: 180GB
Operating System	Ubuntu 22.04.2 LTS	Ubuntu 22.04.2 LTS	Ubuntu 22.04.2 LTS	Ubuntu 22.04.2 LTS

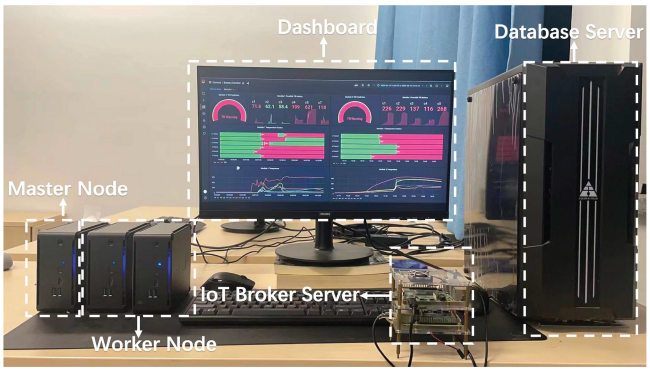


Fig. 7. Hardware deployment of KBMP.

B. Hardware Implementation

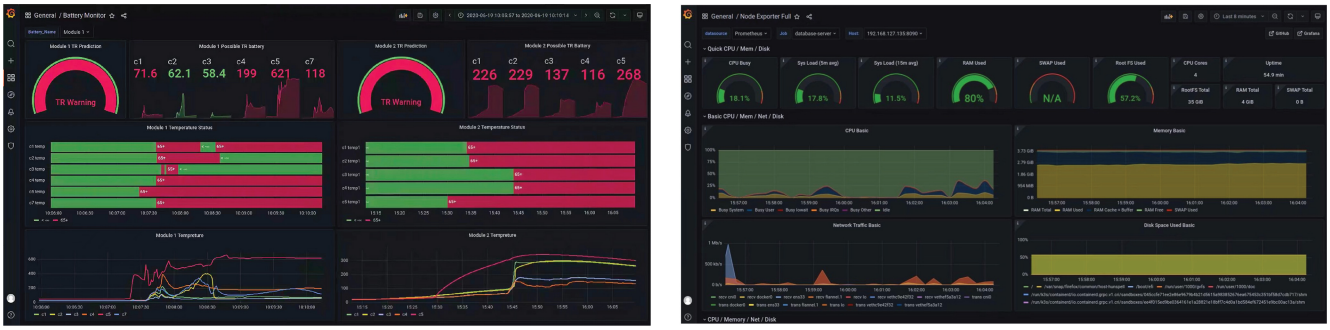
The specific hardware deployment of KBMP is shown in Fig. 7. The master node, worker nodes, and database server in the platform cluster are all 64-bit host servers based on the x86 architecture. On the other hand, the IoT broker servers are lightweight Raspberry Pi single-board computer servers based on the ARM architecture. The servers are interconnected through a 1-Gb/s Ethernet connection within the local area network, which effectively reduces communication latency between nodes and prevents interruptions in data stream processing caused by long communication delays. Table IV presents the hardware and operating system information for each server component in the edge computing cluster.

IV. EXPERIMENTS

In this section, real battery TR data are utilized to test the comprehensive performance of KBMP. Simulated MQTT data sources are employed to transmit real-time battery status data to KBMP. This approach aims to evaluate the platform's efficacy in terms of data visualization, data transmission and analysis latency, scalability, as well as the precision of TR alerts.

A. Data Source

The data used in this article are obtained during TR assessments conducted by a battery manufacturing company, including battery temperature, voltage, and current. The experimental objects encompass two lithium-ion battery cells and



(a) (b)

Fig. 8. Data visualization in KBMP. (a) Customized dashboard displays battery temperature and TR analysis results. (b) Prometheus assists KBMP to monitor detailed information about the Kubernetes cluster.

two lithium-ion battery packs. In the case of battery cells, temperature sensors are strategically positioned at the anode, cathode, and three surfaces. For battery packs, temperature sensors are placed on the exteriors of 8–10 battery cells within each pack.

In these TR assessments, the initiation of TR is facilitated through overcharging and controlled external heating. The experimental procedure unfolds as follows, wherein step 4) is exclusively executed for the battery cell configuration.

1) The battery cells/battery packs were subjected to an initial charging process.
2) A flat or rod-shaped heating device, with its surface covered in ceramic, was used. The heating device was assembled with the battery cells, directly in contact with them. The size of the heating device did not exceed the heated surface of the test object. Temperature sensors were installed, and the temperature monitoring points were placed on the side far from heat conduction, i.e., on the opposite side of the heating device. The sampling interval for temperature data was 0.5 s, with an accuracy of ± 2 °C. The diameter of the temperature sensor's tip was less than 1 mm.
3) After fully charging the battery cells using the standard charging method, the cells were continuously charged with a current of 1 Coulomb.
4) After overcharging, the heating device was immediately activated, applying its maximum power to continuously heat the test object. The triggering was stopped when TR occurred or when the temperature at the monitoring point (opposite the heating surface) reached 300 °C. The heating device was then turned off.

The criteria for determining TR are as follows.

A) The test object experiences a voltage drop.
B) The temperature at the monitoring point reaches the battery's protection operating temperature, which is 60 °C.
C) The temperature rising rate at the monitoring point, dT/dt, is greater than or equal to 1 °C/s.

When both A) and C) or B) and C) occur, it is considered as a TR happens. The test will be terminated when a fire or explosion occurs during the heating process or within 1 h after heating.

Subsequent to the acquisition of the aforementioned battery data set, real-time transmission of battery data is simulated employing the Paho framework. Paho offers open-source MQTT client implementations in diverse programming languages [35]. In this experiment, Paho is used to continuously publish battery data with a specific topic, including the temperature of the battery and the corresponding time. The EMQX server deployed on the Raspberry Pi only needs to subscribe to this topic to receive the data.

B. Data Visualization

Following the successful reception of battery data by KBMP, the InfluxDB database, situated on the database server, undertakes the continuous storage of both battery data and TR analysis results. Directly sourcing the latest data from InfluxDB, Grafana enables the visualization of battery data on the monitoring dashboard.

As depicted in Fig. 8(a), the battery monitor dashboard encompasses a dynamic display of battery temperature fluctuations, complete with annotations denoting temperature anomalies, historical peak values, and the outcomes of TR alerts. Each chart is underpinned by pertinent database query codes, ensuring real-time presentation of battery data results upon database alterations. This feature grants users a highly intuitive grasp of the operational state of an individual battery. Located in the top-right corner, a selection form empowers users to designate the battery group under observation, facilitating the retrieval of the corresponding data set.

Furthermore, KBMP offers comprehensive visualization and oversight of cluster nodes. The built-in K3s dashboard can exhibit a brief showcase of node information. For further cluster information, users can deploy the Prometheus monitoring system on KBMP. Prometheus offers a universal data model and convenient data collection, storage, and querying interfaces. The main program of Prometheus is installed on the master node, and the node-exporter plugins are installed as listeners on the worker nodes. The configured Prometheus server is then set as the data source for Grafana, and visualizations are created as needed. As shown in Fig. 8(b), this dashboard provides an intuitive display of server hardware configurations, CPU utilization, memory utilization, and other

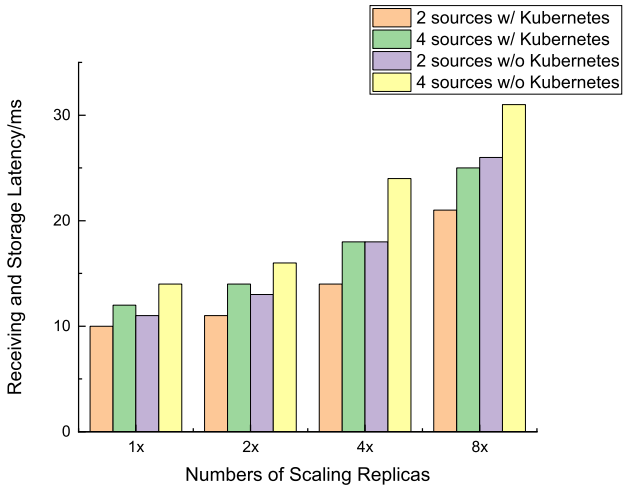


Fig. 9. Latency required for data reception and storage.

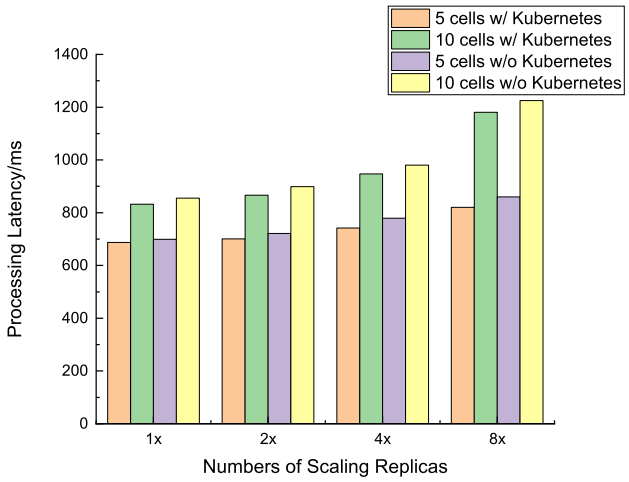


Fig. 10. Latency required for data analysis.

critical information, making it more conducive to managing large-scale clusters.

Leveraging its robust community backing and extensive ecosystem, Kubernetes simplifies the administration of expansive clusters, thereby augmenting and enriching our battery monitoring platform.

C. Data Transmission and Analysis Latency

With the platform operating seamlessly, the latency required for receiving and storing battery data is first tested. For comparison, the latency required for data reception and storage without utilizing Kubernetes orchestration is also tested. As illustrated in Fig. 9, the measured delays consistently fall within the millisecond scale, significantly shorter than data reception intervals, thereby obviating the possibility of data accumulation. Moreover, the increase in the number of data sources simultaneously received by each EMQX server results in a slight increase in data reception latency. This underscores the efficacy of parallel data reception in enhancing efficiency.

Furthermore, the employment of Kubernetes orchestration in data reception yields a reduction in data transmission latency. This advantage is more prominent as the number of replicas of the data receiving application increases.

A comparison is performed between data processing delays for TR warning applications with and without Kubernetes orchestration. As depicted in Fig. 10, it is evident that clustering ten battery data samples simultaneously within the TR warning application would result in significantly higher latency compared to clustering five samples. This finding underscores that increased data processing parallelism does not necessarily lead to improved efficiency. Nonetheless, irrespective of the number of samples clustered simultaneously, groups with Kubernetes orchestration consistently require less time.

Kubernetes, although not inherently tailored for data transfer or processing optimization, offers features and capabilities that indirectly enhance these aspects. Particularly for the battery monitoring platform, effective resource management is pivotal, given the concurrent handling of multiple tasks, such as

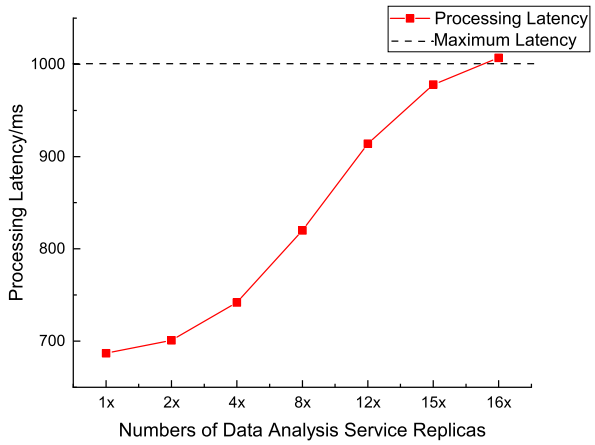


Fig. 11. Maximum of 15 data analysis services can concurrently operate on a single worker node.

data collection, analysis, and real-time monitoring. Kubernetes efficiently allocates and manages resources according to priority and requirements, ensuring equitable distribution among tasks. Strategically distributing containers based on resource constraints reduces network congestion, consequently enhancing data transfer rates. Simultaneously, Kubernetes prioritizes computationally intensive layers, like data analysis, further refining data processing efficiency.

Numbers of processing service replicas are continuously raised to determine the load capacity of KBMP. As shown in Fig. 11, a maximum of 15 data analysis services can concurrently operate on a single worker node, with each service handling data from five batteries simultaneously. With two worker nodes, the KBMP can effectively monitor up to 150 batteries while ensuring that the cumulative processing latency does not exceed 1 s.

D. Platform Scalability

This study extends its investigation to assess the scalability of the platform using Kubernetes as the foundational distributed operating system. A comparative analysis focused on the average scaling latency of the TR warning component within the edge IoT battery monitoring platform is made.

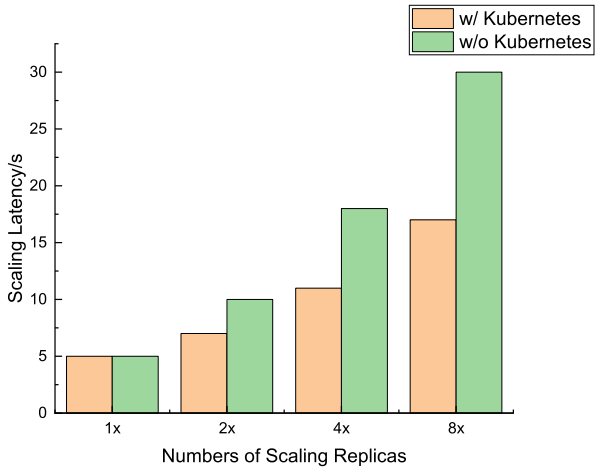


Fig. 12. Latency required for replicas scaling.

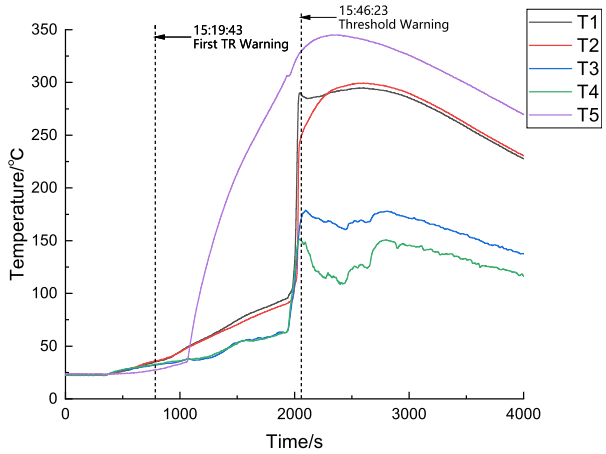


Fig. 13. Temperature variation and TR prediction of battery cell No.1.

TABLE V
CLUSTERING RESULT OF BATTERY CELLS

Cluster Number	Normal Result No.1	Abnormal Result No.1	Normal Result No.2	Abnormal Result No.2
1	T1, T2	T1	T1, T2	T1, T2
2	T3, T4	T2, T3	T3, T4	T3
3	T5	T4	T5	T4
4	/	T5	/	T5

As shown in Fig. 12, the Kubernetes-orchestrated deployment approach exhibited notably reduced scaling time compared to the default component deployment mode.

The rationale behind this phenomenon stems from Kubernetes' intrinsic capability for automated application scaling. Through the modification of configuration files, users can adjust the quantity of replicas for distributed components, prompting the Kubernetes system to concurrently scale the necessary Pod replicas for applications. This extends to the parallel upgrading of older Pods to newer versions, significantly improving the efficiency of application scalability. The pronounced scalability of Kubernetes empowers the battery monitoring platform to adapt its monitoring scope and quantity with heightened flexibility.

E. Battery Thermal Runaway Prediction

1) Battery Cell No. 1: The prediction of battery TR is accomplished by the data processing layer within KBMP. The *K*-Means clustering algorithm deployed in this layer continuously processes the most recent 60 s of battery data and obtains the clustering results under normal operating conditions, as shown in Table V (line 1), which remain unchanged before battery overcharging begins. Until 15:19:43, 70 s after the battery begins to overcharge, the battery clustering situation changes as shown in Table V (line 2), triggering the battery TR warning, which occurs approximately 27 min before the actual battery TR at 15:46:23.

Subsequently, with the continuous overcharging of the battery and the introduction of external heat sources, the battery temperature clustering situation undergoes frequent changes, triggering multiple TR alarms. The specific temperature variations and warning results are shown in Fig. 13.

2) Battery Cell No. 2: The *K*-Means clustering algorithm deployed in this layer continuously processes the most recent 60 s of battery data and obtains the clustering results under normal operating conditions, as shown in Table V (line 3), which remains unchanged before the battery overcharging begins. Until 9:03:28, 50 s after the battery begins to overcharge, the battery clustering situation changes as shown in Table V (line 4), triggering another battery TR warning, which occurs approximately 28 min before the actual battery TR at 9:32:37.

Subsequently, with the continuous overcharging of the battery and the introduction of external heat sources, the battery temperature clustering situation undergoes frequent changes, triggering multiple TR alarms.

To establish the superiority of the proposed algorithm, identical data sets were employed for the training of long short-term memory (LSTM) neural networks [36]. In particular, the data pertaining to battery cell No. 1 was designated as the training set, while the data corresponding to battery cell No. 2 was allocated as the validation set.

However, owing to the exacting prerequisites concerning both the quantity and quality of data, coupled with its limited versatility, the LSTM neural network exhibits diminished accuracy when predicting the temperature of battery cell No. 2. The LSTM algorithm is provided with more training data and costs more time on training and fine-tuning. However, its prediction accuracy remains lower than that of the *K*-Means. Consequently, to establish a minimum threshold for prediction accuracy within the LSTM neural network, the prediction horizon is constrained to a 3-min timeframe. Ultimately, an alarm is triggered 202 s prior to the alarm generated by the threshold method. In contrast, the *K*-Means algorithm, unburdened by the need to precisely predict future temperatures, issues an alert with a 29-min lead time by discerning alterations in clustering outcomes. The intricate details pertaining to temperature variations and the corresponding warning results are illustrated in Fig. 14.

3) Battery Group No. 1: The *K*-Means clustering algorithm deployed in this layer continuously processes the most recent 60 s of battery data and obtains the clustering results under

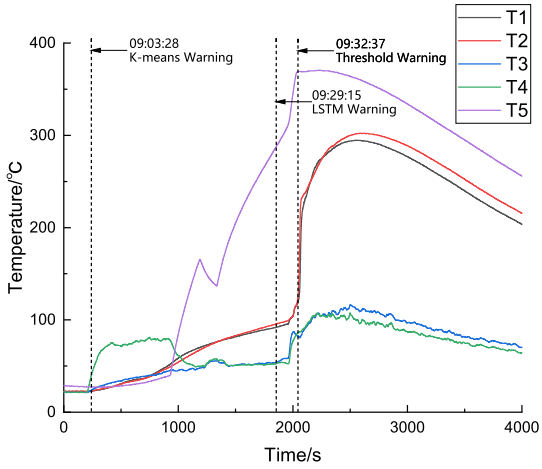


Fig. 14. Temperature variation and TR prediction of battery cell No.2.

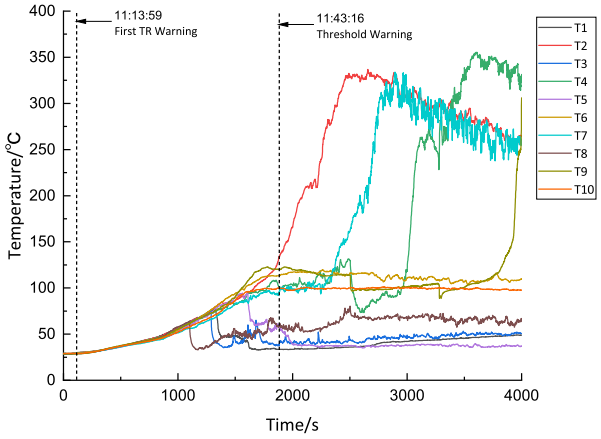


Fig. 15. Temperature variation and TR prediction of battery group No.1.

TABLE VI
CLUSTERING RESULT OF BATTERY GROUPS

Cluster Number	Normal Result No.1	Abnormal Result No.1	Normal Result No.2	Abnormal Result No.2
1	T1, T2, T3, T4, T5, T6	T1	T1	T1, T3, T5, T7
2	T7, T10	T2, T3, T4, T5, T6	T2, T4, T6, T8	T2, T6
3	T8, T9	T7, T8, T9, T10	T3, T5, T7	T4, T8

normal operating conditions, as shown in Table VI (line 1), which remains unchanged before battery overcharging begins. Until 11:13:59, 97 s after the battery began to overcharge, the battery clustering situation changed as shown in Table VI (line 2), triggering the battery TR warning, which occurred approximately 29 min before the actual battery TR at 11:43:16.

Subsequently, with the continuous overcharging of the battery and the introduction of external heat sources, the battery temperature clustering situation undergoes frequent changes, triggering multiple TR alarms. The specific temperature and battery voltage variations at both ends are shown in Fig. 15.

4) Battery Group No. 2: The *K*-Means clustering algorithm deployed in this layer continuously processes the most recent 60 s of battery data and obtains the clustering results under normal operating conditions, as shown in Table VI (line 3), which remains unchanged before battery overcharging begins. Until 15:32:59, 35 s after the battery began to overcharge, the battery clustering situation changed as shown in Table VI (line 4), triggering the battery TR warning, which occurred approximately 28 min before the actual battery TR at 16:00:52.

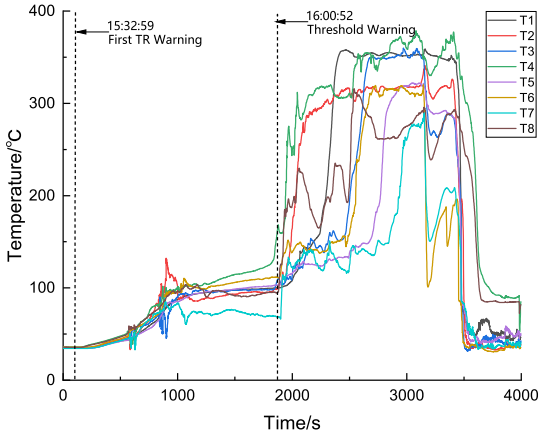


Fig. 16. Temperature variation and TR prediction of battery group No.2.

As shown in Fig. 16, between 1000 and 2000 s after the start of the experiment, overcharging has stopped, and the temperature of some battery cells gradually decreases. In reality, several battery cell structures have already been damaged during the previous overcharging step, and the battery temperature has not cooled down to room temperature, resulting in a continuous deterioration of the internal battery structure until complete TR. If the traditional method is used for judgment, the battery condition during this period is likely to be classified as safe, while this platform can repeatedly indicate that the clustering situation does not match the normal working condition, indicating the presence of TR risk.

From the above four sets of experiments, it can be observed that the proposed battery monitoring platform has performed excellently in both individual battery TR and group battery TR warnings. It can issue TR warnings approximately 30 min before TR occurs (around 1 min after overcharging starts). Sufficient warning lead time ensures that battery management personnel can take timely protective measures to prevent further danger from occurring.

V. CONCLUSION

This study leverages Kubernetes and cloud–edge technology to establish KBMP, a battery monitoring platform characterized by low-latency, high-precision, and exceptional scalability. KBMP encompasses four hierarchical layers: 1) data reception; 2) storage; 3) analysis; and 4) visualization. Specifically, the data reception layer employs the EMQX, the InfluxDB is deployed on the storage layer, the analysis layer's TR prediction algorithm is based on the *K*-Means clustering, and the Grafana facilitates data visualization.

In the experimental section, KBMP underwent testing using genuine TR data from both individual battery cells and battery packs. The *K*-Means algorithm enables KBMP to successfully issue TR alerts approximately 30 min ahead with lower training and inference cost, underscoring the accuracy of its predictive capabilities. Moreover, compared to non-Kubernetes orchestration scenarios, KBMP exhibits reduced time for data

transmission, processing, and expansion within the battery monitoring platform, emphasizing its inherent advantages.

Future endeavors include integrating model-driven and data-driven predictive modes to more effectively align with battery state changes during working processes, enabling real-time updates of algorithm parameters to enhance the battery TR warning accuracy. Additionally, KBMP's data scheduling mechanism will be optimized to prioritize the analysis and processing of batteries at risk of TR, thus ensuring robust battery safety.

REFERENCES

[1] J. Wen, D. Zhao, and C. Zhang, "An overview of electricity powered vehicles: Lithium-ion battery energy storage density and energy conversion efficiency," *Renew. Energy*, vol. 162, pp. 1629–1648, Dec. 2020.

[2] L. Kong, C. Tang, H.-J. Peng, J.-Q. Huang, and Q. Zhang, "Advanced energy materials for flexible batteries in energy storage: A review," *SmartMat*, vol. 1, no. 1, pp. 1–35, Dec. 2020.

[3] S. Wang, L. Sun, and S. Iqbal, "Green financing role on renewable energy dependence and energy transition in E7 economies," *Renew. Energy*, vol. 200, pp. 1561–1572, Nov. 2022.

[4] B. E. Lebrouhi, Y. Khattari, B. Lamrani, M. Maaroufi, Y. Zeraouli, and T. Kousksou, "Key challenges for a large-scale development of battery electric vehicles: A comprehensive review," *J. Energy Storage*, vol. 44, Dec. 2021, Art. no. 103273.

[5] Z. Liao, S. Zhang, K. Li, G. Zhang, and T. G. Habetler, "A survey of methods for monitoring and detecting thermal runaway of lithium-ion batteries," *J. Power Sources*, vol. 436, Oct. 2019, Art. no. 226879.

[6] F. Gao et al., "A review on materials for flame retarding and improving the thermal stability of lithium ion batteries," *Int. J. Electrochem. Sci.*, vol. 15, no. 2, pp. 1391–1411, Feb. 2020.

[7] S. Sripad, A. Bills, and V. Viswanathan, "A review of safety considerations for batteries in aircraft with electric propulsion," *MRS Bull.*, vol. 46, no. 5, pp. 435–442, May 2021.

[8] F. Larsson, P. Andersson, P. Blomqvist, and B.-E. Mellander, "Toxic fluoride gas emissions from lithium-ion battery fires," *Sci. Rep.*, vol. 7, no. 1, Aug. 2017, Art. no. 10018.

[9] Z. Zhang, S. Liang, and C. Yan, "Research and suggestions on the safe and orderly development of electrochemical energy storage under the background of carbon peak and carbon neutrality," *China Eng. Consult.*, vol. 257, pp. 41–45, Oct. 2021.

[10] J. Conzen, S. Lakshmipathy, A. Kapahi, S. Kraft, and M. DiDomizio, "Lithium ion battery energy storage systems (BESS) hazards," *J. Loss Prev. Process Ind.*, vol. 81, Feb. 2023, Art. no. 104932.

[11] H. Gabbar, A. Othman, and M. Abdussami, "Review of battery management systems (BMS) development and industrial standards," *Technologies*, vol. 9, no. 2, p. 28, Apr. 2021.

[12] J. K. Thomas, H. R. Crasta, K. Kausthubha, C. Gowda, and A. Rao, "Battery monitoring system using machine learning," *J. Energy Storage*, vol. 40, Aug. 2021, Art. no. 102741.

[13] K. Friansa, I. N. Haq, B. M. Santi, D. Kurniadi, E. Leksono, and B. Yuliarto, "Development of battery monitoring system in smart microgrid based on Internet of Things (IoT)," *Procedia Eng.*, vol. 170, pp. 482–487, Mar. 2017.

[14] M. Helmy et al., "IoT-based battery monitoring system for electric vehicle," *Int. J. Eng. Technol.*, vol. 7, pp. 505–510, Jan. 2018.

[15] M. Asaad, F. Ahmad, M. S. Alam, and Y. Rafat, "IoT enabled monitoring of an optimized electric vehicle's battery system," *Mob. Netw. Appl.*, vol. 23, no. 4, pp. 994–1005, Aug. 2018.

[16] H. Xue, B. Huang, M. Qin, H. Zhou, and H. Yang, "Edge computing for Internet of Things: A survey," in *Proc. Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Soc. Comput. (CPSCom) IEEE Smart Data (SmartData) IEEE Congr. Cybermatics (Cybermatics)*, 2020, pp. 755–760.

[17] C. Yang, Y. Wang, S. Lan, L. Wang, W. Shen, and G. Q. Huang, "Cloud-edge-device collaboration mechanisms of deep learning models for smart robots in mass personalization," *Rob. Comput. Integr. Manuf.*, vol. 77, Oct. 2022, Art. no. 102351.

[18] K. Cao, Y. Liu, G. Meng, and Q. Sun, "An overview on edge computing research," *IEEE Access*, vol. 8, pp. 85714–85728, 2020.

[19] C. Yang et al., "Cloud-edge-device collaboration mechanisms of cloud manufacturing for customized and personalized products," in *Proc. 25th Int. Conf. Comput. Support. Cooperative Work Des. (CSCWD)*, 2022, pp. 1517–1522.

[20] B. Chen, J. Wan, A. Celesti, D. Li, H. Abbas, and Q. Zhang, "Edge computing in IoT-based manufacturing," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 103–109, Sep. 2018.

[21] W. Zhu, X. Zhou, M. Cao, Y. Wang, and T. Zhang, "The cloud-end collaboration battery management system with accurate state-of-charge estimation for large-scale lithium-ion battery system," in *Proc. Int. Conf. Big Data Inf. Anal. (BigDIA)*, 2022, pp. 199–204.

[22] W. Li, M. Rentemeister, J. Badeda, D. Jöst, D. Schulte, and D. U. Sauer, "Digital twin for battery systems: Cloud battery management system with online state-of-charge and state-of-health estimation," *J. Energy Storage*, vol. 30, Aug. 2020, Art. no. 101557.

[23] A. Adhikaree, T. Kim, J. Vagdoda, A. Ochoa, P. J. Hernandez, and Y. Lee, "Cloud-based battery condition monitoring platform for large-scale lithium-ion battery energy storage systems using Internet-of-Things (IoT)," in *Proc. IEEE Energy Convers. Congr. Expo. (ECCE)*, 2017, pp. 1004–1009.

[24] S. Yang et al., "Implementation for a cloud battery management system based on the CHAIN framework," *Energy AI*, vol. 5, Sep. 2021, Art. no. 100088.

[25] V. Medel, O. Rana, J. Á. Bañares, and U. Arronategui, "Modelling performance & resource management in Kubernetes," in *Proc. IEEE/ACM Int. Conf. Util. Cloud Comput. (UCC)*, 2016, pp. 257–262.

[26] A. Chen, W. Zhang, C. Zhang, Z. Wang, and X. Fan, "A novel AlCu internal short circuit detection method for lithium-ion batteries based on on-board signal processing," *J. Energy Storage*, vol. 52, Aug. 2022, Art. no. 104748.

[27] C. Bormann, A. P. Castellani, and Z. Shelby, "COAP: An application protocol for billions of tiny internet nodes," *IEEE Internet Comput.*, vol. 16, no. 2, pp. 62–67, Mar./Apr. 2012.

[28] H. Bai and L. Wang, "Performance analysis of t-box internet of vehicle communication," *Solid State Technol.*, vol. 63, pp. 2274–2279, Mar. 2020.

[29] J. Wu, "Cluster analysis and K-means clustering: An introduction," in *Advances in K-means Clustering*. Berlin, Germany: Springer, 2012, pp. 1–16.

[30] T. Giorgino, "Computing and visualizing dynamic time warping alignments in R: The dtw package," *J. Stat. Softw.*, vol. 31, no. 7, pp. 1–24, Aug. 2009.

[31] X. Feng et al., "Key characteristics for thermal runaway of Li-ion batteries," *Energy Procedia*, vol. 158, pp. 4684–4689, Feb. 2019.

[32] Q. Wang, Z. Wang, L. Zhang, P. Liu, and Z. Zhang, "A novel consistency evaluation method for series-connected battery systems based on real-world operation data," *IEEE Trans. Transp. Electrif.*, vol. 7, no. 2, pp. 437–451, Jun. 2021.

[33] W. Yuan, D. Liang, Y. Chu, and Q. Wang, "Aging effect delays overcharge-induced thermal runaway of lithium-ion batteries," *J. Loss Prev. Process Ind.*, vol. 79, Oct. 2022, Art. no. 104830.

[34] S. Singh and N. Singh, "Containers & docker: Emerging roles & future of Cloud technology," in *Proc. Int. Conf. Appl. Theor. Comput. Commun. Technol. (iCATccT)*, 2016, pp. 804–807.

[35] M. Bender, E. Kirdan, M.-O. Pahl, and G. Carle, "Open-source MQTT evaluation," in *Proc. IEEE Consum. Commun. Netw. Conf.*, 2021, pp. 1–4.

[36] A. Z. Hinchi and M. Tkiouat, "A deep long-short-term-memory neural network for lithium-ion battery prognostics," in *Proc. Int. Conf. Ind. Eng. Syst. Manag.*, 2018, pp. 2162–2168.

Qingyang Chen received the B.E. degree in information engineering from the School of Electronic and Information Engineering, Zhejiang University, Hangzhou, China, in 2022, where she is currently pursuing the M.S. degree with the College of Information Science and Electronic Engineering.

Her current research interests mainly include battery thermal runaway protection, data mining, and Internet of Things.

Yinghui He (Member, IEEE) received the B.E. degree in information engineering and the Ph.D. degree in information and communication engineering from Zhejiang University, Hangzhou, China, in 2018 and 2023, respectively.

His research interests mainly include mobile-edge computing, device-to-device communications, and integrated sensing and communications.

Guanding Yu (Senior Member, IEEE) received the B.E. and Ph.D. degrees in communication engineering from Zhejiang University, Hangzhou, China, in 2001 and 2006, respectively.

He joined Zhejiang University in 2006, where he is currently a Professor with the College of Information and Electronic Engineering. From 2013 to 2015, he was also a Visiting Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA. His research interests include integrated sensing and communications, mobile-edge computing/learning, and machine learning for wireless networks.

Dr. Yu received the 2016 IEEE ComSoc Asia–Pacific Outstanding Young Researcher Award. He has served as a Guest Editor for *IEEE Communications Magazine* special issue on Full-Duplex Communications, an Editor for IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS Series on Green Communications and Networking, and Series on Machine Learning in Communications and Networks, an Editor for IEEE WIRELESS COMMUNICATIONS LETTERS, a Lead Guest Editor for *IEEE Wireless Communications Magazine* special issue on LTE in Unlicensed Spectrum, and an Editor for IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING and IEEE ACCESS. He is currently serving as an Editor for IEEE TRANSACTIONS ON MACHINE LEARNING IN COMMUNICATIONS AND NETWORKING. He regularly sits on the technical program committee boards of prominent IEEE conferences, such as ICC, GLOBECOM, and VTC. He also serves as a Symposium Co-Chair for IEEE Globecom 2019 and a Track Chair for IEEE VTC 2019'Fall.

Chong Xu received the B.S. degree in electrical engineering and automation from Southwest Jiaotong University, Chengdu, China, in 2011, and the M.S. degree in optoelectronic technology from the University of Tsukuba, Tsukuba, Japan, in 2014.

He is currently an Engineer with the Energy Storage and Electrotechnics Department, China Electric Power Research Institute Company Ltd., Beijing, China. His research interests include fiber-optic sensing technology and intelligent monitoring technology for lithium-ion batteries.

Mingyang Liu received the B.S. degree in communication engineering from Lanzhou University, Lanzhou, China, in 2018, and the M.S. degree in microelectronics and solid-state electronics from the University of Chinese Academy of Sciences, Beijing, China, in 2021.

She is currently an Engineer with the Energy Storage and Electrotechnics Department, China Electric Power Research Institute Company Ltd., Beijing. Her research interests include smart grid measurements, flexible sensing systems, and analog/mixed-signal integrated circuits with a particular focus on sensor interfaces for miniaturized systems.

Zhenming Li received the B.S. degree in safety engineering from Henan Polytechnic University, Jiaozuo, China, in 2005, and the Ph.D. degree in refrigeration and cryogenic engineering from the University of Chinese Academy of Sciences, Beijing, China, in 2010.

He is currently a Professor with the Energy Storage and Electrotechnics Department, China Electric Power Research Institute Company Ltd., Beijing. His current research interests include smart power grid measurements, flexible miniaturized sensors, and their application in batteries for energy storage.