

Received February 27, 2021, accepted March 17, 2021, date of publication March 24, 2021, date of current version April 5, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3068753

A Reliable Data Compression Scheme in Sensor-Cloud Systems Based on Edge Computing

SHAOFEI LU^{1,2}, QINHUA XIA^{1,2}, XIAOLIN TANG¹, XUYANG ZHANG¹,
YINGPING LU³, AND JINGKE SHE¹

¹College of Computer Science and Electronic Engineering, Hunan University, Changsha 410012, China

²Hunan Provincial Key Laboratory of Blockchain Infrastructure and Application, Hunan University, Changsha 410012, China

³Quantum Corporation, Mendota Heights, MN 55118, USA

Corresponding author: Shaofei Lu (sflu@hnu.edu.cn)

This work was supported in part by the Industrial Internet Innovation and Development Project of China under Grant TC19084DY, and in part by the Special Funds for Construction of Innovative Provinces in Hunan Province of China under Grant 2020SK2066 and Grant 2020GK2016.

ABSTRACT The rapid development of the IoT and cloud computing has spawned a new network structure — sensor-cloud system (SCS) where sensors, sensor networks, and cloud computing are integrated to perform data sensing, collection, transmission, and decision making. The large-scale deployment of sensors creates a massive amount of data, posing new challenges in data transmission and storage. As an intermediate platform between IoT and cloud platforms, edge computing provides IoT with data collection, processing, and scheduling services. This paper proposes a hybrid data compression scheme that incorporates lossy and lossless compression in SCS based on edge computing to address the increasing challenges. Moreover, we propose a new reliable lossy compression algorithm DFan, based on the simplified Fan algorithm with a high compression ratio (CR). By introducing the data tolerable deviation, DFan transforms single-factor decision-making into multi-factor decision-making, reducing the error of lossy compression. Through experiments on IntelLab and MIT-BIH datasets, the proposed hybrid data compression scheme achieves an overall CR of $4.21\times$ and $3.88\times$, respectively. The lossy CR of DFan is $6.42\times$ and $5.1\times$, respectively, and the Percentage RMS Difference (PRD) caused by lossy compression is 0.27% and 0.56%, respectively. The hybrid compression scheme, high compression ratio, and reliable data restoration make this scheme attractive to the data processing of sensors in SCS.


INDEX TERMS Sensor-cloud system (SCS), data compression, edge computing, Internet of Things (IoT).

I. INTRODUCTION

With the rapid development and broad applications of wireless sensor networks (WSN) coupled with massive deployment and wide reception of cloud computing, the integration of WSNs and cloud computing, aka Sensor-Cloud System (SCS) has received considerable attention in both academic and industry communities [1]. The emergence of SCS greatly extends the computing capacity, storage capacity, scalability, communication capacity and usability of traditional wireless sensor networks by leveraging the massive computing power, scalability, storage flexibility and ubiquitous access in cloud computing. Within a SCS, the underlying sensor network only needs to collect data while the data analysis, processing

and other operations are delegated to the cloud. This not only reduces the burden of sensor networks but also speeds up the data processing [2]. The SCS has been widely used, including everything from intelligent devices to smart home, industrial IoT, internet of vehicles, and precision agriculture [3]–[6].

However, the SCS also entails obvious constraints such as limited communication bandwidth between sensor network and the cloud, high latency due to the long-distance transmission of data from sensor network to the cloud, and security and privacy concerns due to the ubiquitous exposure of the cloud. In addition, data usability in the cloud also becomes a concern. In a typical SCS, the collected raw data are first uploaded from the underlying sensor network to the cloud, then processed and present to the end users. However, end users may not need all raw data, but certain useful data, and aggregated data. Therefore, it is desirable to ensure that

The associate editor coordinating the review of this manuscript and approving it for publication was Md Zakirul Alam Bhuiyan .

only effective data are received, stored and processed in the cloud. This can lead to lower storage requirements, faster data processing and better user experience. As an emerging technology, mobile edge computing [7] boasts with many advantages, including higher local processing power, wider geographical distribution and the support of mobility. It provides IoT with data collection, processing, and scheduling services. Within edge computing, each edge node can be used as a mobile sink. When the data in the sensor network is transmitted to an edge node, the data is first aggregated and compressed locally by the edge node, which can reduce the amount of data to be transmitted to a certain extent.

Data generated by sensor devices in SCS usually displays the following characteristics: large volume, high data sampling rate and multiple data types. On the other hand, sensor nodes in SCS generally consist of inexpensive electronic devices with limited communication bandwidth. With the rapid and continuous deployment of sensors and IoT devices, the massive amount of data is generated in the sensor networks, which poses significant challenges in data transmission and storage, and has mandated reliable, accurate and efficient compression techniques [8].

In terms of data usage, the equipment operating status and sensor monitoring data collected by IoT devices generally have two purposes in actual application scenarios. The first is to provide a reference for the operation of other equipment. This type of data demands extremely high real-time and data acquisition frequency. Thus, there is no need for additional data processing. Another purpose is to save the data to the database to provide data support for upper-level decision-making [9]. For this type of data, in actual production, there is no such high requirement. Because the total amount of data is too large, it will take up a lot of physical storage space and lead to a significant decline in the efficiency of retrieving historical data. We can classify this type of data into short-term storage data and permanent storage data. The short-term storage data is mainly used to monitor equipment operation status and environment. The data that fluctuates in a small range may not be the focus of monitoring. The monitoring personnel may only pay attention to some sharply changing inflection point data. In a few cases where all the original data is required, the original data can be restored by lazy loading. For data that needs to be stored permanently, considering the database's storage pressure, generally, the average value is stored. Considering the end usage of the data, compression processing will achieve better practical application effects.

In this paper, taking the actual application scenarios of data into consideration, we propose a novel hybrid data compression scheme based on edge computing to address the data transmission challenges in SCS. This scheme combines both lossy and lossless compression algorithms, realizing mixed lossless transmission and transmission rate selection. Moreover, we propose lossy compression algorithm (DFan) to process numeric data to be transmitted to the cloud. Through experiments with real data, it can be found that for edge sensor data, our proposed scheme achieves a good balance

among data fidelity, compression ratio, and processing efficiency.

Our major contributions are as follows:

- 1) Based on the simplified Fan algorithm, by introducing the data tolerable deviation, the DFan algorithm maintains the calculation accuracy and CR of the original Fan algorithm and is equivalent to the simplified Fan algorithm in data processing speed.
- 2) Using the DFan algorithm combined with the LZ4 compression algorithm, a new data hybrid compression scheme is proposed based on data purpose. Compared with other hybrid compression schemes, it has good scalability and low PRD. Experiments with real datasets demonstrate the algorithm's effectiveness which offer a new option for massive edge sensing data processing.
- 3) Our scheme improves the reliability of edge and cloud data transmission in SCS and reduces the possibility of data modification and eavesdropping. The scheme divides the original data into two parts, optimizes the complexity of obtaining the original data, and increases data transmission efficiency.

The rest of this paper is organized as follows. Section II discusses previous work in the area. Section III combines the actual application scenarios, from the edge to the cloud, and introduces our proposed hybrid compression scheme in detail. Section IV discusses the existing lossy compression algorithm; and proposes a new lossy compression algorithm on this basis. In Section V, the proposed scheme is verified by the IntelLab dataset and MIT-BIH arrhythmia database and compared with other data compression processing schemes. Section VI draws the conclusion and discusses the limitations and future work.

II. RELATED WORK

Data compression has been widely used to reduce the amount of data to be stored or transmitted. There are two categories of data compression, namely, lossless compression and lossy compression [10], [11]. Lossless compression algorithms such as LZW, Snappy, Deflate, LZ4 and LZO have a relatively low compression ratio (CR), which is about 1-3 times [12], [13]. Although the data fidelity is guaranteed, the compression effect is not ideal. On the other hand, lossy compression can obtain CR several times higher than lossless compression, but the price is the loss of fidelity. There are many lossy compression algorithms, such as transform-based methods including Wavelets Transform, Discrete Cosine Transform (DCT), Fourier Transforms and Cubic Hermitian, and neural network-based methods such as ANN-based lossy compression algorithm [14]–[17]. Besides, in order to neutralize the advantages and disadvantages of lossless compression and lossy compression, some scholars have proposed hybrid data compression algorithm. In [18], the original data is transformed by DCT, then the transformed data threshold is set to zero, and finally the run-length code (RLC) technology

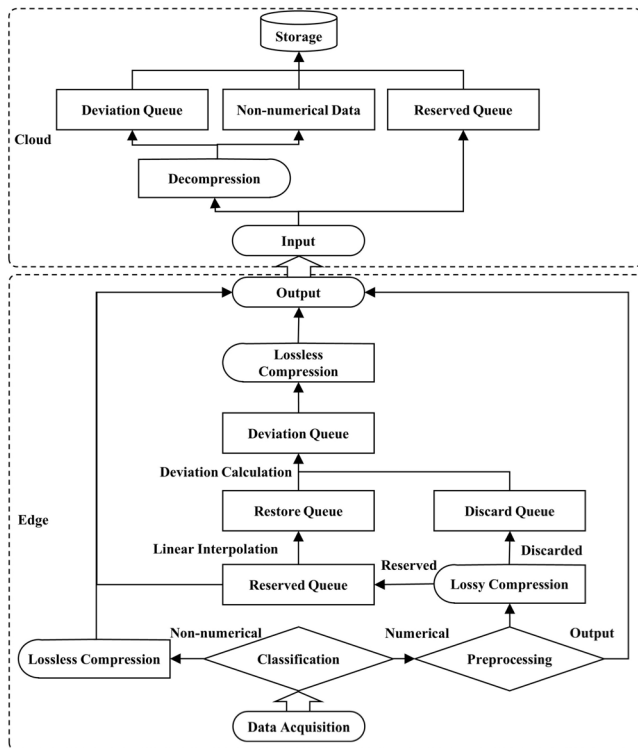


FIGURE 1. Data processing flowchart.

is used to encode the obtained data. This algorithm does not apply to the case where the data itself contains a value of 0, and the scope of application is limited. In [15], the original data is processed using wavelet transform and then combined with the SPIHT algorithm for encoding, which leads to a higher percentage RMS difference (PRD). In [19], through error judgment in the lossy compression stage, the former data is used to replace the latter data to reduce the total amount of data. In the lossless compression stage, the sample data is directly scaled down so that it is represented by fewer bits. However, the program cannot retain the details of data with small fluctuations. In [10], the AFD algorithm is mixed with the SS technique to obtain a higher CR at the expense of higher reconstruction error and implementation complexity. In [20], a simple recursive implementation is developed to simplify Fan geometric algorithm, reducing the need for computing resources. The residual error is processed in combination with the Huffman algorithm to achieve lossless data compression in the compression stage. If applying this scheme to an environment with more abundant computing resources, the overall processing efficiency can be further improved. The work in this paper distinguishes itself by focusing on data hybrid compression to promote data security and reliability in SCS.

III. PROPOSED DATA COMPRESSION SCHEME

Our scheme consists of sensors, edge, and cloud, as shown in Figure 1. Thousands of heterogeneous sensor data are gathered through the edge nodes. The first step of compression is lossy compression. After lossy compression, part of the

data is lost. Combined with the original data, the lost part of the data is calculated, followed by lossless compression. The lossy compression and lossy compression are used as the output of the edge. After the edge processing is completed, the data is sent to the cloud for decompression, and the decompressed data is summarized, calculated and stored.

A. DATA PRE-PROCESSING

According to instructions from the upper node, the terminal equipment pre-processes and returns the data at the edge end. Data pre-processing includes data classification and data filtering. The purpose of data compression is to remove redundant information in the data. General data compression algorithms rarely consider the connection between data but simply compress it from bytes or characters' perspectives. For complex and diverse data, research and analysis according to the data's characteristics can facilitate the selection or design of suitable compression algorithms to improve the efficiency of compression processing. Different data types have different data compression algorithms. The actual investigation found that numerical data accounts for the vast majority, so this type of data should be considered in data compression. When we design the scheme, we also consider the characteristics of data fluctuation. For other types of data, the amount of data itself is not large. If making further subdivision, it will increase the computational complexity, thus it is directly classified as non-numerical data. The data filtering mainly considers the high real-time property of certain data. The data that need to be processed in real time is transmitted directly. After pre-processing, the data will continue to be compressed later.

B. DATA COMPRESSION PROCESSING

The data compression process is carried out in batches. Firstly, the current batch of data is lossy compressed, and the compressed data is stored in the reserved queue. Lossy compression leads to partial data loss, and the discarded data is put into the discard queue. Then, the reserved queue data is decompressed to obtain the decompressed data, which is stored in the restore queue. There is still a deviation between the original data and the data in the restore queue. The deviation can be obtained by calculating the difference between data generated in the restore queue and the corresponding data in discard queue. The result is then stored in the deviation queue. The deviation queue is lossless compressed and used as the batch's compression result together with the reserved queue data. After processing, the data is divided into two parts. In order to reconstruct the original data, it needs to obtain the output of two parts simultaneously and the corresponding relationship between the two parts of the data. It is evident that our scheme improves the reliability of data.

In theory, our scheme can be used with any lossy and lossless compression/decompression algorithms. However, the implementation complexity highly depends on the choice of lossy compression and lossless compression schemes because the lossy compressed data must be decompressed at

TABLE 1. Comparison of lossless compression algorithms.

Method	Source Data Size (byte)	Size After Compression (byte)	Compression Time Consuming (ms)	CR	Decompression Time Consuming (ms)	Processing Efficiency (M/s)
LZ4	31407	14225	0.603	2.21	0.517	27.3847
Snappy	31407	13915	0.786	2.26	0.515	23.5749
GZIP	31407	9713	2.754	3.23	5.506	3.7132
Deflater	31407	11041	18.965	2.84	19.537	0.7966
LZO	31407	13702	0.923	2.29	0.645	19.5605
Huffman	31407	16866	1277.37	1.86	17.798	0.0237

the edge node to calculate the error caused by lossy compression, so the implementation complexity of lossy compression and lossless compression schemes should be as low as possible.

Lossless compression is used uniformly for non-numeric data. It helps reduce the computational complexity of edge nodes and reduces the bandwidth pressure of data transmission. In addition, the deviation queue calculated from the difference between the original and reconstructed data has a shallow dynamic range. The use of lossless compression helps to minimize the bit rate. Commonly used fast lossless compression algorithms include LZ4, Snappy, GZIP, Deflater, LZO [21]–[23]. Traditional lossless compression algorithms include Huffman [24] and RLC [25]. A benchmark test was performed on each lossless compression algorithm, and the results are shown in Table 1. From the perspective of the compression ratio, GZIP and Deflater have higher compression ratios. The Huffman algorithm has the lowest compression ratio, and the compression ratios of several other algorithms are concentrated around $2.2\times$. The LZ4 algorithm is the highest in terms of processing efficiency, with 27M data processed per second. From the view of the data's characteristics to be compressed and the processing speed requirements, the LZ4 algorithm is more suitable. Due to the low implementation complexity of the Huffman algorithm, it has advantages in the case of limited computing resources.

C. DATA DECOMPRESSION PROCESSING

The reserved queue data, deviation queue data, and non-numerical queue data are aggregated in the cloud through the edge node. The edge node involves data compression, so it needs to be decompressed in the cloud. For non-numerical data and deviation queue data, lossless compression is adopted at the edge node, and the data before compression can be obtained by directly applying the corresponding decompression method in the cloud. The deviation queue data is used to restore the original numerical data. For the original data, the reserved queue data can be combined with the linear interpolation method to obtain the reconstructed data. There could be an error between the reconstructed data and the original data. In the process of edge node processing, the error has been stored in the deviation queue. The original numerical type data can be obtained by arithmetic operation between the reconstructed data and the decompressed

deviation queue data. From the perspective of data usage, the focus is on inflection point data. Small range precision loss can be tolerated, and reserved queue data can be directly used. When there is a demand for original numerical data, the deviation queue can be combined to restore the data. It can be seen that the scheme we propose has good scalability. The reconstruction of the original data requires the coordination of the two parts of the reserved queue and the compressed deviation queue. This improves the security of the data.

The deviation between the reserved queue and the original data is not large from the perspective of data storage. From the experiment in Section V, it can be found that the PRD is 0.27% - 0.56%. In a specific time interval, the calculated mean deviation value is smaller, and the calculation can be simplified by retaining the queue calculation mean value for permanent storage.

IV. DFAN: A LOSSY DATA COMPRESSION ALGORITHM

For lossy compression, the compressed data is required to have better data fidelity. Several lossy compression techniques are reviewed in [26]. A uniform sub-sampling technique (such as random extraction) can achieve a higher compression ratio, but it cannot ensure data fidelity. Compared with uniform sub-sampling methods, non-uniform sub-sampling techniques, such as Fan compression algorithm, are particularly prominent in data fidelity. Compared with techniques based on domain transformation, the Fan algorithm is simpler in complexity, and the data processed is still readable without being decompressed. The feature meets the needs of specific application scenarios.

Fan algorithm is an adaptive sub-sampling technique. Under the premise that the error after reconstruction of the intermediate sample is less than the maximum specified error, the largest possible data deviation area is constructed between the starting sample and the ending sample. The processing process of the Fan algorithm is shown in Figure 2.

The algorithm is initialized by directly retaining the first sample I_1 and storing it in the sample set as the first origin. The second sample I_2 is used to calculate the two boundaries of the current sample set (upper bound is $I_2 + \varepsilon$, lower bound is $I_2 - \varepsilon$, the size of ε impacts the data compression, which is named CF in this paper). Two slopes (U_1 and L_1) are drawn by connecting the origin with the upper bound, the origin, and the lower bound. This constitutes the data deviation region. With the arrival of the next sample I_3 , if I_3 falls in the data

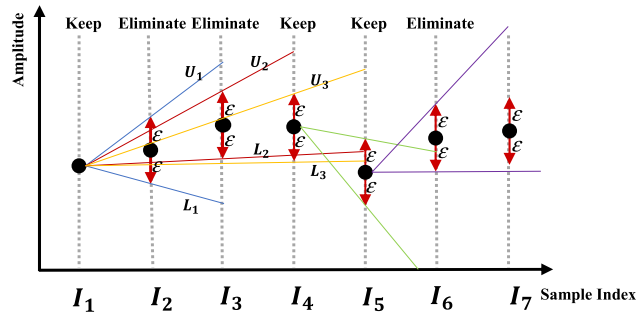


FIGURE 2. Fan algorithm illustration.

deviation area, that is, the sample value of I_3 is between U_1 and L_1 , which means that within the allowable error range, I_2 sample has little fluctuation and belongs to redundant data, so it should be discarded and continue to process the next data. Continue to calculate the upper and lower error boundary ($I_3 + \epsilon$) of I_3 . It can be seen from Figure 2 that the intersection is the region formed by (L_2, U_2) , and sample I_4 is in this region, so I_3 is discarded. Continue to calculate the upper and lower error boundary ($I_4 + \epsilon$). It can be seen from Figure 2 that the intersection is the region composed of (L_2, U_3) , but sample I_5 is not in this region. This shows that the data fluctuates greatly at I_4 , and I_4 should be retained and stored in the sample set. Replace I_4 with the new origin and repeat the above process.

In the Fan algorithm, the calculation of intersection is more complicated. To simplify the calculation of the Fan algorithm and reduce the demand for computing resources, paper [20] developed a simple recursive geometry implementation method. The area determined by the upper and lower boundaries of the current processed sample is used as the intersection in the original Fan algorithm to simplify the calculation and reduce the complexity of the implementation.

The simplified processing method ignores the impact of discarded samples on the intersection, resulting in the enlarged intersection. If the trend of multiple adjacent data changes is linear, the simplified Fan algorithm will discard many data samples with large fluctuations, which will lead to large data reconstruction errors. We have made improvements in response to this problem.

Figure 3 shows all possible fluctuations in two consecutive samples. If the situation of Figure 3 (a) and (c) occurs, the impact of this processing on data accuracy will not be too significant. However, if the data in Figure 3 (b) and (d) shows a linear trend and if the simplified Fan algorithm continues to be used, the sample data that is seriously deviated from the origin (as shown in sample I_5 in Figure 4) will be discarded. This leads to a substantial impact on the accuracy of the data. For such cases, we introduce the data tolerable deviation from single-factor judgment to multi-factor judgment to reduce the impact of intersection on the current sample, thereby reducing reconstruction errors.

For the sample to be discarded, if its fluctuation is the same as that of the next sample, then calculate and judge whether the difference between it and the next sample is

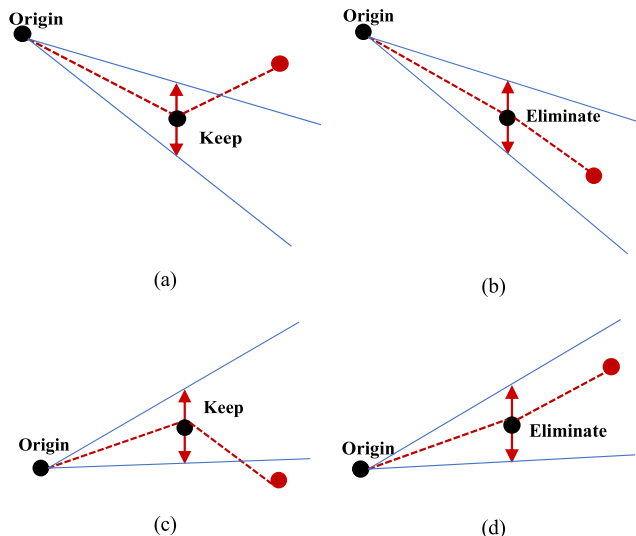


FIGURE 3. The effect diagram of data selection without introducing tolerance deviation.

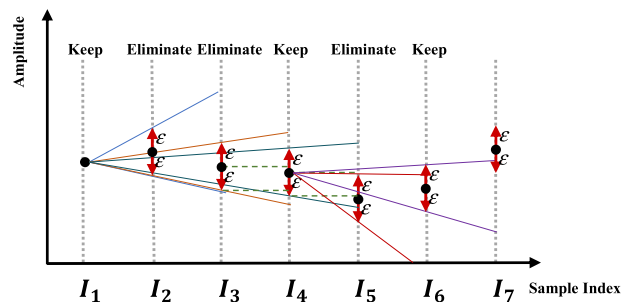


FIGURE 4. DFan algorithm illustration.

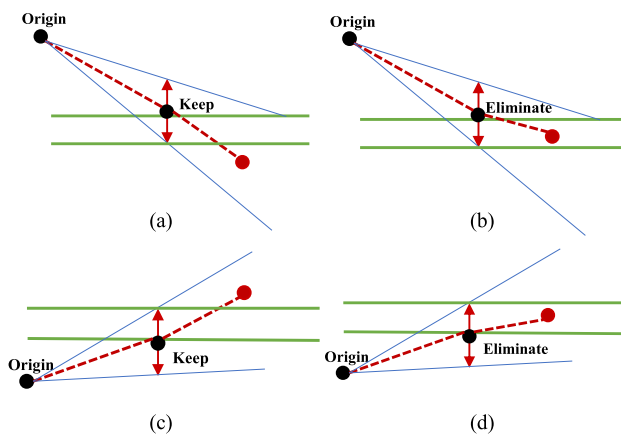


FIGURE 5. The effect diagram of data selection after introducing data tolerance deviation.

less than the given deviation, and discard if it is less than (Figure 5 (b) and (d)), reserve if greater than or equal to (Figure 5 (a) and (c)). Through the above processing, the problem of large data reconstruction errors can be solved. To show the difference, we name the improved algorithm DFan algorithm. Figure 4 shows the data of samples I_3 , I_4 , and I_5 continue to

decrease with a small trend. If we follow the simplified Fan’s processing strategy, sample I_4 will be discarded, causing a large error to the data’s overall trend.

Besides, to avoid continuous small changes in the data and severe deviation from the origin for the same collection point, the new sample will be directly replaced as the origin after processing a certain number of samples. The specific number is determined according to the sample collection frequency.

V. SCHEME VALIDATION AND RESULT ANALYSIS

A. EXPERIMENTAL DATA SOURCES

We verify the effectiveness of the data processing scheme on the IntelLab dataset. The IntelLab dataset was based on the monitoring of 54 devices deployed in the Intel Berkeley Laboratory at University of Oxford [27]. The data included time-stamped information on humidity, temperature, light and voltage, with a time span of 35 days and a total of 2.3 million data items. In the experiment, we split the dataset at intervals of dates and randomly selected five groups of temperature data on March 1, 6, 13, 17, and 20 as the experimental data. After removing abnormal data, the total number of samples in each group was about 80,000.

The sample fluctuations of the IntelLab dataset are relatively gentle. In the comparative experiment, we also used 48 sets of MLI recorded data from the MIT-BIH arrhythmia database [28]. The samples in this dataset fluctuated violently to verify the applicability of the proposed scheme. In this dataset, the total number of data samples per record is 21,600.

B. EVALUATION METRICS

The compression effect of data compression is related to the size of the compressed data. The commonly used compression ratio (CR) indicates the degree of data compression:

$$CR = \frac{\text{No. of uncompressed bits}}{\text{No. of compressed bits}} \tag{1}$$

There are two leading indicators for evaluating reconstruction quality: diagnostic index and mathematical index. For example, the mean score is a diagnostic index, but it is not widely used because of its high computational complexity and time complexity [26]. The mathematical index is widely used to evaluate the energy difference between the original and reconstructed data because of the small amount of calculation. This paper uses the following popular mathematical indexes:

- 1) The Root Mean Square Error (RMSE) defined in the following formula is used to measure the difference between the original data $x(n)$ with a total N and the reconstructed data $x'(n)$. The smaller the root mean square error, the better it indicates that the reconstructed data is close to the original data.

$$RMSE = \sqrt{\frac{\sum_n (x(n) - x'(n))^2}{N}} \tag{2}$$

- 2) Quality Score (QS) reflects the relationship between CR and reconstruction quality. The larger the value,

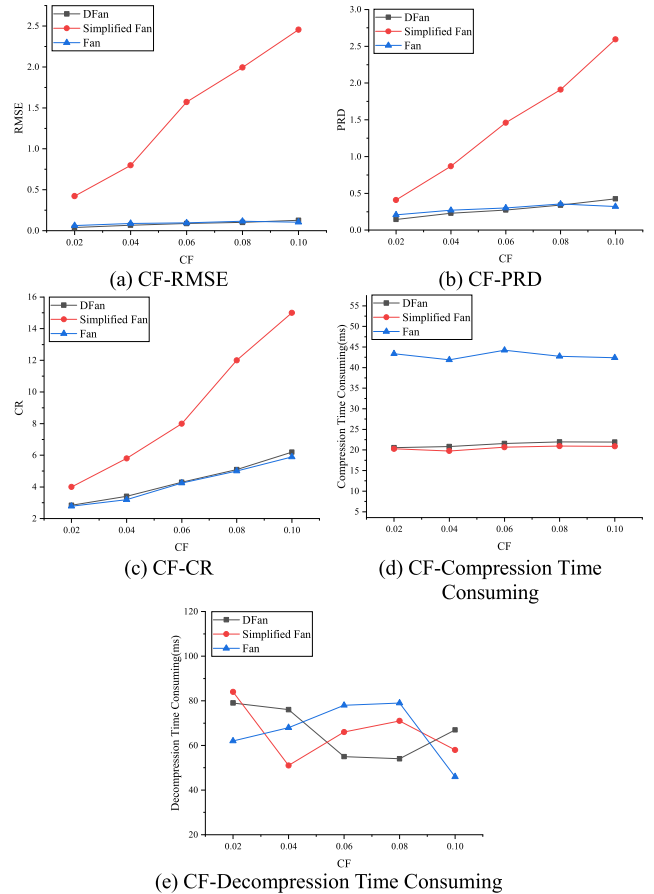


FIGURE 6. Comparison chart of multi-dimensional effect of DFan, Fan and simplified Fan.

the better the compression effect [17].

$$QS = CR/PRD \tag{3}$$

- 3) Compared with RMSE, the Percentage RMS Difference (PRD) defined in the following formula can better reflect the reconstruction quality [20].

$$PRD = \sqrt{\frac{\sum_n (x(n) - x'(n))^2}{\sum_n (x(n))^2}} \times 100\% \tag{4}$$

C. PERFORMANCE IMPROVEMENT EVALUATION

In our experiment, the lossless compression algorithm was LZ4, and the lossy compression algorithm was DFan. The data tolerable deviation value was similar to CF’s change trend, and the amplitude was close. Therefore, in our subsequent experiments, the data tolerable deviation value and CF were considered as equivalent values.

Regarding the improvement of DFan, under different CFs, five metrics of RMSE, PRD, CR, compression time, and decompression time were selected and compared with the original Fan algorithm and the simplified Fan algorithm. The experiments were performed on five test data groups in the IntelLab dataset, and the mean values under different CF were calculated. The experimental results are shown in Figure 6.

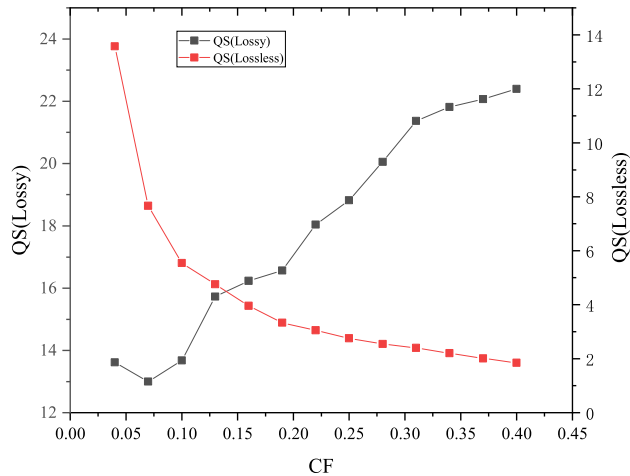


FIGURE 7. The relationship between QS and CF.

From the perspective of RMSE, the DFan algorithm is very close to the original Fan algorithm, and the RMSE is about 0.1. The RMSE of the simplified Fan algorithm increases with the increase of CF, and the maximum value is close to 2.5. The reason for this result is that the simplified Fan algorithm discards many inflection point data. From the perspective of PRD, the PRD value of the DFan algorithm is very close to the original Fan, and the value is relatively small. While the PRD value of simplified Fan algorithm fluctuates wildly, and the maximum value is close to 7.5%. From the perspective of CR, the simplified Fan algorithm has a good compression effect. When the CF value is 0.1, its CR is as high as 29 \times , and the price of this result is the loss of data accuracy. From the point of view of compression time, because the DFan algorithm is based on the simplified Fan algorithm, it simplifies the calculation of the intersection of the original Fan algorithm. The time consumption of the DFan algorithm is about half of the original Fan algorithm. Compared with the simplified Fan algorithm, its time consumption is slightly higher than the simplified Fan algorithm. This is because the DFan performs additional calculation processing when processing the discard queue. Since the three algorithms' decompression processes are similar, they have equivalent effects in the decompression time. Overall, the DFan maintains the calculation accuracy and CR of the original Fan and is equivalent to the simplified Fan in data processing speed.

D. COMPARISON WITH OTHER HYBRID COMPRESSION SCHEMES

To find the proposed scheme's best performance, we analyzed the overall compression performance for different CFs (corresponding to 0.1–1% of the maximum dynamic range of the collected data. The data tolerable deviation is considered equivalent to the CF). We conducted experiments using five sets of data in the IntelLab dataset (the data range is [0-40]). Calculate the average of five groups of results of the same CF. To comprehensively evaluate the impact of different CFs on

TABLE 2. Average compression performance of the proposed algorithm with MIT/BIH and IntelLab database.

Database	Avg. RMSE	Avg. PRD	Avg. Lossy CR	Avg. Lossless CR	Avg. CR	Avg. CR (Non-numerical Data)
MIT-BIH	0.04	0.56	5.12	3.27	3.88	2.51
IntelLab	0.09	0.27	6.42	3.45	4.21	2.33

lossy compression and lossless compression, under the same CF, the entire scheme's PRD was used to calculate the QS of lossy compression and lossless compression. The result is shown in Figure 7. When the CF is small for lossy compression, CR grows slowly, and there is a negative correlation. As the CF increases, the relationship between CR and CF is positively correlated. This is because more samples were discarded, making CR grow faster than PRD. For lossless compression, the lossless compression ratio in the numerical queue gradually slows down as the CF increases. Because more and more similar data is discarded, CR decreases, and PRD continues to increase. The higher the QS value, the better the compression effect. To obtain the best performance, from the figure, the CF and the data tolerable deviation value should be limited between 0.12-0.24, that is, between 0.3-0.6% of the maximum fluctuation range of the collected data. This result is consistent with the results of literature.

We apply the DFan algorithm and the LZ4 algorithm to the proposed scheme and conduct comparative experiments with other existing similar schemes. In the comparative experiment, the same dataset was used to ensure the experiment's fairness, as the other schemes were used; that is, 48 records in the MIT-BIH arrhythmia database were used for the experiment. Since this dataset only contained numerical data, 1,000 pieces of non-numerical data were manually added to each set of sample data for simulation experiments to meet the prerequisites set by the scheme. Combined with the previous conclusions, the data tolerable deviation and CF should be limited to 0.3-0.6% of the collected data samples' maximum fluctuation range. In the comparative experiment, we chose the CF and the data tolerable deviation value as 0.006 (the dynamic data range in this dataset is $[-1,1]$, and 0.006 is 0.3% of the maximum fluctuation range). After the experiment, we calculated the average of the 48 sets of data obtained, and the results are shown in Table 2. Our proposed scheme achieves an overall compression ratio of 3.88 \times , an average lossy compression ratio of 5.12 \times , an average PRD of 0.56%, and an average RMSE of 0.09 in the test data. The average lossless compression ratio for numerical data (deviation queue) is 3.27 \times , and the average lossless compression ratio for non-numerical data is 2.51 \times . The reason for this difference in lossless compression is that the deviation queue has a short fluctuation range.

Moreover, we also conducted experiments on the IntelLab dataset with relatively smooth data fluctuations. Since the dataset only contained numerical data, 1,000 non-numerical

TABLE 3. Performance comparison with other techniques.

Method	Lossy CR	Lossless CR	PRD	Ref
Fourier Decomposition	17-44	-	0.8-2	[10]
Cubic Hermitian	2-6.6	-	0.9-9.6	[11]
Wavelet with Quality Control	11.10	-	2.99	[14]
SPIHT	8.40	-	6.58	[15]
Neural Networks	18.27	-	1.17	[17]
Fan & Huffman	7.8	2.11	0.51	[20]
DCT Compression	27.90	-	2.93	[29]
Proposed Method	5.12	3.27	0.56	-

data were also manually added before the experiment. The data tolerable deviation of CF is 0.12 (0.3% of the maximum fluctuation range). Experiments were performed on five data sets, and the average was calculated. Compared with the experimental results of the MIT-BIH arrhythmia database, the effect of lossless compression is similar. According to the four indexes of lossy compression, overall compression ratio, PRD, and RMSE, the proposed scheme's application effect in the MIT-BIH arrhythmia database with severe data fluctuations is not ideal.

Table 3 shows the results of the performance indexes comparison between the existing scheme and our proposed scheme. Our proposed scheme has low implementation complexity, so we do not intend to compare the compression performance with the existing lossy/lossless compression technologies. As shown in Table 3, our algorithm's PRD is 0.56 when the lossy CR is 5.12 \times . Also, the lossless CR of 3.27 \times can be achieved using the deviation queue data for complete reconstruction. Existing algorithms that use transform domain techniques like Wavelets, DCT, Fourier transforms obtain higher lossy CR at the expense of higher reconstruction error and implementation complexity. Similarly, the methods based on the neural network can achieve higher lossy CR but at higher PRD. Using SPIHT and Cubic Hermitian technique can also lead to higher PRD.

VI. CONCLUSION

Massive data collected in sensor networks makes the data transmission to the cloud particularly challenging due to the limited communication bandwidth in sensor devices. To address this, this paper proposes a reliable sensor data processing scheme for sensor-cloud systems based on edge computing. The scheme realizes the lossy and lossless mixed compression of data to improve the CR and processing efficiency as much as possible while obtaining lossless data. The concept of the data tolerable deviation has been introduced and used our proposed compression algorithm DFan to reduce the error of lossy compression. Through experiments on IntelLab and MIT-BIH datasets, the proposed hybrid data compression scheme achieves an overall CR of 4.21 \times and 3.88 \times , respectively. The lossy CR of DFan is 6.42 \times and 5.1 \times , respectively, and the PRD caused by lossy compression is 0.27% and 0.56%, respectively. This hybrid compression

scheme in conjunction with high compression ratio, and reliable data restoration offers an attractive option for the data processing of sensors in SCS.

It should be noted that the scheme also has limitations. Experiments conducted on the MIT-BIH arrhythmia database showed that the highest lossy CR was 10.38 \times , and the lowest lossy CR was 2.71 \times . It showed large fluctuations compared to the average value of 5.12 \times . After inspection, it is found that the dataset numbered 107 corresponding to the lowest lossy CR fluctuates very sharply. Therefore, it can be concluded that the compression effect of the proposed scheme is not ideal for the situation of sharply volatile data. We will try to find a more suitable method to deal with the sharply volatile data in future.

REFERENCES

- [1] M. Z. A. Bhuiyan, G. Wang, J. Cao, and J. Wu, "Deploying wireless sensor networks with fault-tolerance for structural health monitoring," *IEEE Trans. Comput.*, vol. 64, no. 2, pp. 382–395, Feb. 2015.
- [2] J. Zeng, T. Wang, W. Jia, S. Peng, and G. Wang, "A survey on sensor-cloud," *J. Comput. Res. Develop.*, vol. 54, no. 5, pp. 925–939, 2017.
- [3] G. D. O'Mahony, P. J. Harris, and C. C. Murphy, "Detecting interference in wireless sensor network received samples: A machine learning approach," in *Proc. IEEE 6th World Forum Internet Things (WF-IoT)*, New Orleans, LA, USA, Jun. 2020, pp. 1–6.
- [4] P. Wang, F. Ye, and X. Chen, "A smart home gateway platform for data collection and awareness," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 87–93, Sep. 2018.
- [5] Y. Liang, Y. Mei, Y. Yang, Y. Ma, W. Jia, and W. Tian, "A low-coupling method in sensor-cloud systems based on edge computing," *J. Comput. Res. Develop.*, vol. 57, no. 3, pp. 639–648, 2020.
- [6] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," *IEEE Trans. Services Comput.*, vol. 13, no. 2, pp. 289–300, Nov. 2020.
- [7] T. Wang, L. Qiu, A. K. Sangaiah, A. Liu, M. Z. A. Bhuiyan, and Y. Ma, "Edge-computing-based trustworthy data collection model in the Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4218–4227, May 2020.
- [8] Y. Xu, Q. Zeng, G. Wang, C. Zhang, J. Ren, and Y. Zhang, "An efficient privacy-enhanced attribute-based access control mechanism," *Concurrency Comput., Pract. Exper.*, vol. 32, no. 5, pp. 1–10, Mar. 2020.
- [9] M. M. Rashid, J. Kamruzzaman, M. M. Hassan, S. Shahriar Shafin, and M. Z. A. Bhuiyan, "A survey on behavioral pattern mining from sensor data in Internet of Things," *IEEE Access*, vol. 8, pp. 33318–33341, 2020.
- [10] J. Ma, T. Zhang, and M. Dong, "A novel ECG data compression method using adaptive Fourier decomposition with security guarantee in E-health applications," *IEEE J. Biomed. Health Informat.*, vol. 19, no. 3, pp. 986–994, May 2015.
- [11] T. Marisa, T. Niederhauser, A. Haerberlin, R. A. Wildhaber, R. Vogel, M. Jacomet, and J. Goette, "Bufferless compression of asynchronously sampled ECG signals in cubic hermitian vector space," *IEEE Trans. Biomed. Eng.*, vol. 62, no. 12, pp. 2878–2887, Dec. 2015.
- [12] W. Liu, F. Mei, C. Wang, M. O'Neill, and E. E. Swartzlander, "Data compression device based on modified LZ4 algorithm," *IEEE Trans. Consum. Electron.*, vol. 64, no. 1, pp. 110–117, Feb. 2018.
- [13] A. Gopinath and M. Ravisanakar, "Comparison of lossless data compression techniques," in *Proc. Int. Conf. Inventive Comput. Technol. (ICICT)*, Coimbatore, India, Feb. 2020, pp. 628–633.
- [14] C.-T. Ku, K.-C. Hung, T.-C. Wu, and H.-S. Wang, "Wavelet-based ECG data compression system with linear quality control scheme," *IEEE Trans. Biomed. Eng.*, vol. 57, no. 6, pp. 1399–1409, Jun. 2010.
- [15] A. Alesanco and J. E. Garcia, "Automatic real-time ECG coding methodology guaranteeing signal interpretation quality," *IEEE Trans. Biomed. Eng.*, vol. 55, no. 11, pp. 2519–2527, Nov. 2008.
- [16] C. Tan, L. Zhang, and H.-T. Wu, "A novel Blaschke unwinding adaptive-Fourier-decomposition-based signal compression algorithm with application on ECG signals," *IEEE J. Biomed. Health Informat.*, vol. 23, no. 2, pp. 672–682, Mar. 2019.

- [17] C. M. Fira and L. Goras, "An ECG signals compression method and its validation using NNs," *IEEE Trans. Biomed. Eng.*, vol. 55, no. 4, pp. 1319–1326, Apr. 2008.
- [18] M. Alsenwi, M. Saeed, T. Ismail, H. Mostafa, and S. Gabran, "Hybrid compression technique with data segmentation for electroencephalography data," in *Proc. 29th Int. Conf. Microelectron. (ICM)*, Beirut, Lebanon, Dec. 2017, pp. 1–4.
- [19] K. Nemati and K. Ramakrishnan, "Hybrid lossless and lossy compression technique for ECG signals," in *Proc. 3rd Int. Conf. Sens., Signal Process. Secur. (ICSSS)*, Chennai, India, May 2017, pp. 450–455.
- [20] C. J. Deepu, C.-H. Heng, and Y. Lian, "A hybrid data compression scheme for power reduction in wireless sensors for IoT," *IEEE Trans. Biomed. Circuits Syst.*, vol. 11, no. 2, pp. 245–254, Apr. 2017.
- [21] K. Rattanaopas and S. Kaewkeeree, "Improving Hadoop MapReduce performance with data compression: A study using wordcount job," in *Proc. 14th Int. Conf. Electr. Eng./Electron., Comput., Telecommun. Inf. Technol. (ECTI-CON)*, Phuket, Thailand, Jun. 2017, pp. 564–567.
- [22] S. Oswal, A. Singh, and K. Kumari, "Deflate compression algorithm," *Int. J. Eng. Res. Gen. Sci.*, vol. 4, no. 1, pp. 430–436, Jan. 2016.
- [23] D. Harnik, E. Khaizim, D. Sotnikov, and S. Taharlev, "A fast implementation of deflate," in *Proc. Data Compress. Conf.*, Snowbird, UT, USA, Mar. 2014, pp. 223–232.
- [24] A. Basheer and K. Sha, "Cluster-based quality-aware adaptive data compression for streaming data," *J. Data Inf. Qual.*, vol. 9, no. 1, pp. 1–33, Oct. 2017.
- [25] K. Radhika and D. Mohana Geetha, "Augmented recurrence hopping based run-length coding for test data compression applications," *Wireless Pers. Commun.*, vol. 102, no. 4, pp. 3361–3374, Oct. 2018.
- [26] Y. Zigei, A. Cohen, and A. Katz, "The weighted diagnostic distortion (WDD) measure for ECG signal compression," *IEEE Trans. Biomed. Eng.*, vol. 47, no. 11, pp. 1422–1430, Nov. 2000.
- [27] N. Jain, A. Gupta, and V. A. Bohara, "PCI-MDR: Missing data recovery in wireless sensor networks using partial canonical identity matrix," *IEEE Wireless Commun. Lett.*, vol. 8, no. 3, pp. 673–676, Jun. 2019.
- [28] G. Xu, "IoT-assisted ECG monitoring framework with secure data transmission for health care applications," *IEEE Access*, vol. 8, pp. 74586–74594, 2020.
- [29] S. Lee, J. Kim, and M. Lee, "A real-time ECG data compression and transmission algorithm for an E-health device," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 9, pp. 2448–2455, Sep. 2011.



XIAOLIN TANG was born in Shaoyang, Hunan, China, in 1996. He received the bachelor's degree in software engineering from Jishou University, China, in 2018. He is currently pursuing the master's degree in computer technology from Hunan University, China. His research interests include the industrial IoT condition monitoring and machine learning.



XUYANG ZHANG was born in Guilin, Guangxi, China, in 1998. He received the B.E. degree in network engineering from Guangxi University, in 2020. He is currently pursuing the M.S. degree in software engineering with Hunan University, China.

His research interests include blockchain infrastructure technology and the IoT.



YINGPING LU received the B.S. and M.S. degrees in computer science from Hunan University, Changsha, China, in 1985 and 1988, respectively, and the Ph.D. degree in computer science from the University of Minnesota, USA, in 2005. He currently works with Quantum Corporation. His research interests include cloud computing, edge computing, and mass storage systems.



SHAOFEI LU received the B.S. degree in computer application technology from the National University of Defense Technology, China, in 2001, the M.S. degree in computer software and theory from Hunan University, China, in 2005, and the Ph.D. degree in computer application from Central South University, China, in 2012. He is currently an Associate Professor with the College of Computer Science and Electronic Engineering, Hunan University. His main research interests include

blockchain, the IoT, and edge computing.



QINHUA XIA was born in Yongzhou, Hunan, China, in 1996. He received the B.S. degree from the Guilin University of Technology, Guangxi, China, in 2018. He is currently pursuing the M.S. degree in software engineering with Hunan University, China.

His research interests include the data processing, the IoT communication, and the data quality assurance and quality control.



JINGKE SHE received the B.Eng. degree in nuclear engineering and technology from Tsinghua University, China, in 2000, the M.Eng. degree in computer engineering from the Memorial University of Newfoundland, Canada, in 2006, and the Ph.D. degree in electrical and computer engineering from The University of Western Ontario, Canada, in 2012. He is currently serving as an Associate Professor with the College of Computer Science and Electronic Engineering,

Hunan University. His main research interests include intelligent control and edge computing.

...