Lossy Compression on IoT Big Data by Exploiting Spatiotemporal Correlation

Aekyeung Moon[1] Jaeyoung Kim[1] Jialing Zhang[2] Seung Woo Son[2]
[1]*Daegu-Gyeongbuk Research Center*
Electronics and Telecommunications Research Institute
Daegu, South Korea
Email:{akmoon,jaeyoung}@etri.re.kr
[2]*Electrical & Computer Engineering Department*
University of Massachusetts Lowell
Lowell, Massachusetts, 01854 USA
Email: jialing_zhang@student.uml.edu, seungwoo_son@uml.edu

***Abstract*—As the volume of data generated by various deployed IoT devices increases, storing and processing IoT big data becomes a huge challenge. While compression, especially lossy ones, can drastically reduce data volume, finding an optimal balance between the volume reduction and the information loss is not an easy task given that the data collected by diverse sensors exhibit different characteristics. Motivated by this, we present a feasibility analysis of lossy compression on agricultural sensor data by comparing fidelity of reconstructed data from various signal processing algorithms and temporal difference encoding. Specifically, we evaluated five real-world sensor data from weather stations as one of major IoT applications. Our experimental results indicate that Discrete Cosine Transform (DCT) and Fast Walsh-Hadamard Transform (FWHT) generate higher compression ratios than others. In terms of information loss, Lossy Delta Encoding (LDE) significantly outperforms others nonetheless. We also observe that, as compression factor is increased, error rates for all compression algorithms also increase. However, the impact of introduced error is much severe in DCT and FWHT while LDE was able to maintain a relatively lower error rate than other methods.**

***Index Terms*—Smart farm, lossy compression, IoT, spatiotemporal, signal processing**

I. Introduction

Rapid advances in IoT big data platforms are changing the current knowledge discovery processes in various domains [1], [2]. It supports a wide range of applications for discovering actionable knowledge from raw data. For example, advanced data analytics in IoT agricultural applications provide a new insight for a precise weather forecast, thus enabling improvement of crop yield and reduction of unnecessary cost related to harvests such as a use of unnecessary herbicide and fertilizers [3], [4].

In developing IoT applications, an efficient management of big data is one of the challenging problems [5]. Transmitting a large volume of data is one of the major causes of high energy consumption and high communication bandwidth usage. This problem can be mitigated by applying data compression techniques such that the storage and communication overheads are reduced [6]. Various compression algorithms have been proposed to meet different application needs, and many of them considered lossy algorithms. Lossy compression [7] can help reduce the data size significantly, but error rates are not easy to bound. To address this problem, several recent approaches have proposed techniques to isolate the error introduced by applying lossy compression methods [8]–[10]. For example, Sustika and Sugiarto [9] proposed a compressive sensing algorithm on weather data. Their approach reduces the number of data samples required to reconstruct from the data in frequency domain, thereby reducing the energy consumption of the sensor in sampling and transmission periods.

In the era of big data, time-based weather data gathering is particularly essential in discovering insights because such data describes changes of weather conditions such as temperature, pressure, humidity, rainfall, solar radiation, wind direction/speed, etc. at one location over time that can be used to evaluate climate patterns and long-term forecasts. However, what lacks is a comparative evaluation that establishes how various compression algorithms, particularly lossy ones, impact on storing weather sensor data efficiently. In [9], lossy compression techniques are used to evaluate temperature and humidity data only.

In this paper, we evaluate several lossy compression algorithms for efficiently storing weather sensor data based on the encoding of temporal changes [8] and three signal transformation algorithms on spatial data [10]–[12]. Specifically, we evaluate the fidelity of reconstructed weather sensor data using Discrete Cosine Transform (DCT), Fast Walsh-Hadamard Transform (FWHT), Discrete Wavelet Transform (DWT), and Lossy Delta Encoding (LDE). Our objective is to provide useful information for minimizing data reconstruction errors, and more importantly, make sure they are within a tolerable range.

II. Compression Algorithms

Our motivation in evaluating several lossy compression techniques on IoT sensor data is as follows. First, naively applying lossy compression seldom gives a reasonably high compression ratio because of the inherent randomness exhibited in many sensor datasets. Second, although individual data values show some randomness, their overall patterns are smooth spatially as well as temporally. Because of this, techniques that combine data transformation with compression can be more effective as the transformed data usually reveal the correlation of the data explicitly. For example, let us consider the temperature data (shown in Fig. 1a) we evaluated in this paper. Fig. 1b shows the result of applying DCT on

978-1-5386-3472-1/17/$31.00 ©2017 IEEE

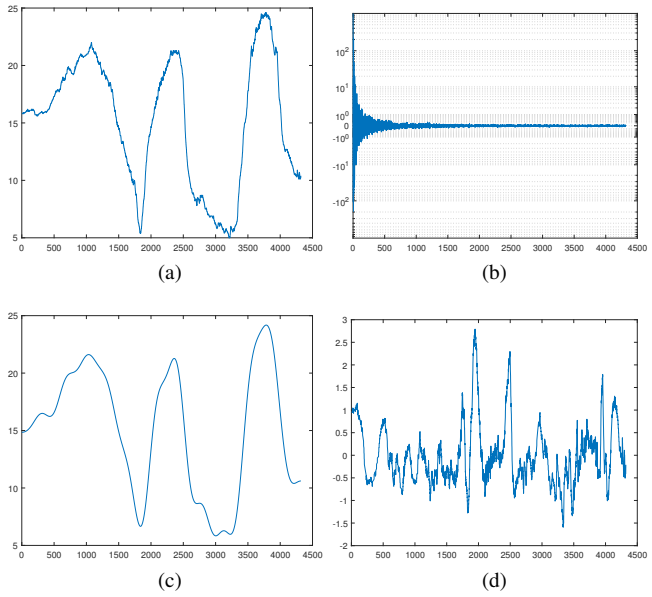


Fig. 1. The variation of "temperature" values. (a) Original data. (b) DCT coefficients. (c) Reconstructed data from the inverse DCT of the coefficients containing 99.9% of the energy of the original data, which accounts for only 3.7% (16 out of 4,320) of the entire data points. All other coefficients are set to zero. (d) Difference between the original data and the reconstructed data.

the temperature where each value is transformed to DCT coefficients. Once the data is represented in frequency domain, we can easily find the relationship between the percentage of informative DCT coefficients and the amount of energy carried by them. To demonstrate the effect of this relationship on data compression, we chose the DCT coefficients containing 99.9% of the energy attained by the original data, which accounts for only 3.7% of the entire data points. We then apply inverse DCT on these selected coefficients to evaluate the difference between the reconstruct data and the original data. As shown in Fig. 1c and Fig. 1d, the relative error is small, and the data can be reconstructed close to the original by maintaining a very small amount of DCT coefficients. It should be noted that this error can be reduced even further if a proper quantization method is applied.

In the theory of signal processing, transforming an original signal to another domain or basis allows us to represent a signal in a more concise format. The outcome of such transformation is used to reduce storage data transmission overheads. If the number of non-zero coefficients from data transformation is sufficiently small, high compression ratios can be achieved [10]. Many transformation methods for representing signals have been proposed and provide a mechanism of reconstructing the signal [13]. This section describes four lossy algorithms, three based on spatial data characteristics and one based on temporal data characteristics, that we evaluate in our study as data transformation methods. To describe each transformation in detail, let us consider a one-dimensional discrete-time data x of length N, which is denoted as $N \times 1$ column vector with elements $x[n]$, where $n = 1, 2, \ldots, N$.

A. Discrete Cosine Transform (DCT)

We first consider discrete cosine transformation (DCT). DCT transforms data from spatial domain into frequency domain. Specifically, it represents a signal as a sum of varying magnitude and frequency, and has been used in lossy compression of audio and images [11]. DCT is defined as:

$$y(k) = w(k)\sum_{n=1}^{N} x(n) cos(\tfrac{\pi(2n-1)(k-1)}{2N}), k = 1, 2, \ldots, N,$$

$$w(k) = \begin{cases} \frac{1}{\sqrt{N}}, & k = 1 \\ \sqrt{\frac{2}{N}}, & 2 \le k \le N, \end{cases}$$

where $x(n)$ is the original data, $y(k)$ is the transformed data, and N is the length of x.

B. Fast Walsh-Hadamard Transform (FWHT)

FWHT is a faster version of Walsh-Hadamard Transform (WHT). A naive implementation of WHT would have a time complexity of $O(N^2)$, but FWHT requires only $N \log N$ additions or subtractions [12]. The FWHT for a signal $x(n)$ of length N is denoted as:

$$y_n = \frac{1}{N}\sum_{i=1}^{N} x_i WAL(n, i),$$

where $i = 1, 2, \ldots, N$ and WAL(n, i) are Walsh functions.

C. Discrete Wavelet Transform (DWT)

Wavelet transforms have been used in compression algorithms because of its high-energy compaction properties [10]. Fig. 2 shows an example implementation of one-level DWT where, starting from a signal x of length N, two sets of coefficients are computed: approximated coefficients (C_1) and detailed coefficients (D_1). These two vectors are obtained by convolving x with the low-pass filter h_0 for C_1 and with the high-pass filter h_1 for D_1. $\downarrow 2$ represents the downsampling operator by a factor of 2.

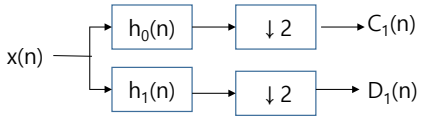


Fig. 2. 1-level DWT.

D. Lossy Delta Encoding (LDE)

Lastly, we evaluate the lossy delta encoding proposed in [8], which is based on the relative changes (called "change ratio" or "temporal change") defined as follows:

$$\Delta x(n_{i,j}) = \frac{x(n_{i,j}) - x(n_{i-1,j})}{x(n_{i-1,j})}, \quad (1)$$

where $x_{i,j}$ and $x_{i-1,j}$ are j-th data values in i-th and $i-1$-th data sampling steps, respectively. x_0 is the first sampling step, which is stored as it is. Once the deltas (in terms of change ratios) have been calculated, it computes the distributions of changes and then encodes them into an index.

978-1-5386-3472-1/17/$31.00 ©2017 IEEE

This encoded data is much preferable for higher compression ratio. Compared with other lossy compression algorithms, this technique allows users to guarantee a user-specified tolerable error rate, δ. It needs to be emphasized that, unlike previous three transformations, LDE approximates data on temporal characteristics. Because of this, δ in LDE differs from that in the previous three transformations. In other words, δ in LDE represents an error bound while δ in other three transformations represents an amount of retained energy. Although the exact meaning of δ differs among different transformations, it will serve our purpose of evaluating the impact of information losses because higher δ essentially entails higher data fidelity.

III. Experimental Evaluation

A. Datasets

We evaluate the effectiveness of compression algorithms discussed in Section II on weather sensor data. For evaluation, we used a real-world dataset from the wireless weather stations located in a small orchard in Youngcheon, South Korea. We chose following five most important variables from the weather data collected during October 2015 in the deployed weather station: temperature, humidity, solar radiation, wind direction and wind speed. The data is continuously monitored and stored every minute. In our evaluation, we used 4,320 data points, which corresponds to about 3 days of sampling. The original data samples are shown in Fig. 3. Table I shows statistical properties of the original data, such as standard deviation (STD), normalized standard deviation, skewness, and kurtosis. The normalized standard deviation is calculated as $\frac{STD(x)}{Mean(x)}$. Skewness is a measure of data asymmetry around the mean value. Negative skewness means that more data are scattered to the left of the mean whereas positive skewness means more data are scattered to the right. Normal distribution, which is symmetric about its mean, gives zero skewness. Measures of kurtosis in Table I indicate how outlier-prone a distribution is. As the kurtosis of any normal distribution is 3, distributions with kurtosis higher than 3 are more outlier-prone. The distributions of kurtosis lower than 3, on the other hand, are less outlier-prone. Solar radiation and wind speed show higher STD than other datasets. In the case of skewness, solar radiation and wind speed have positive value only. As shown in Table I, kurtosis values are not significant.

TABLE I
THE EVALUATED DATASET AND ITS CHARACTERISTICS.

	Standard Deviation	Normalized Standard Deviation	Skewness	Kurtosis
Temperature	5.4873	0.3612	-0.3023	1.9045
Humidity	22.0031	0.2812	-0.7571	2.1204
Solar Radiation	7.3987	1.1454	0.7715	1.9296
Wind Direction	108.9125	0.5421	-0.812	2.1136
Wind Speed	1.8422	0.9836	0.9113	2.7560

We use the following two metrics to evaluate the performance of each compression algorithm: compression ratio and the Normalized version of Root Mean Square Error (NRMSE).

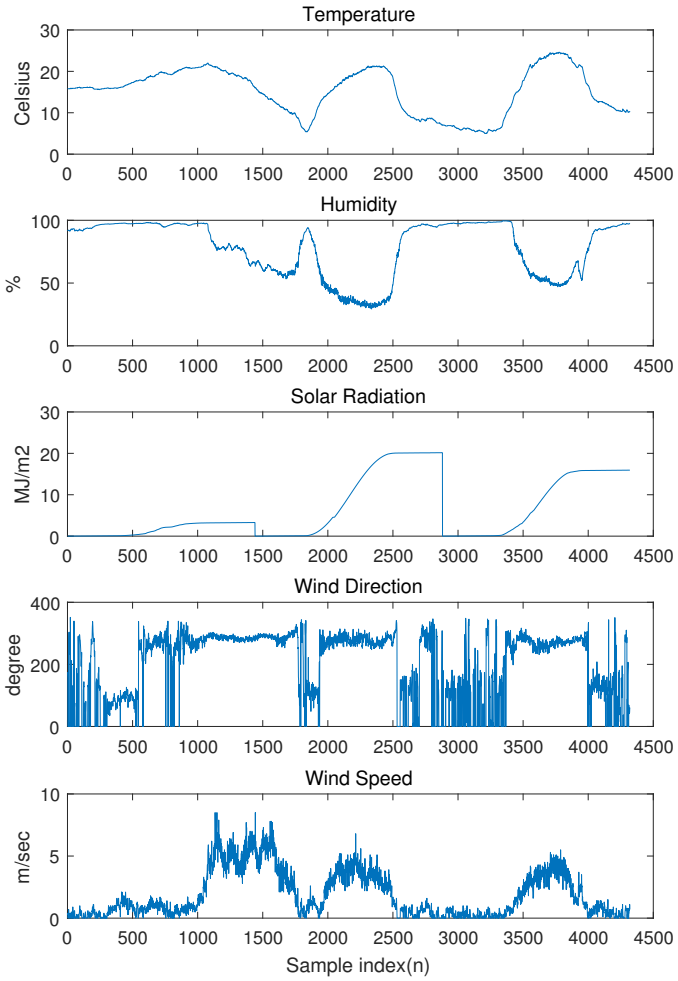


Fig. 3. Value variations exhibited in the original dataset (temperature, humidity, solar radiation, wind direction, and wind speed) during the entire sampling period.

B. Compression Ratio

This section describes how we compress the evaluated data based on DCT, DWT and FWHT transformations. For DWT wavelet transform, we used Daubechnies d4 wavelet (or db4 wavelet). Specific steps for calculating compression ratio for each algorithm are as follows:

1) Decompose the original data into DCT, DWT, FTWH basis vectors. The coefficient vector measures how much energy is stored in each component. Any signal x can be represented in terms of a basis of $N \times 1$ vectors $\{\Psi_i\}_{i=1}^{N}$.

$$x = \sum S_i \Psi_i \quad \text{or} \quad x = \Psi S, \tag{2}$$

where S is $N \times 1$ column vector of coefficients.
2) Sort the coefficient vector S in descending order of coefficient values. The sorted coefficient vector is denoted as: $SS = SS_1, SS_2, ...SS_n$.
3) Find k which determines how many coefficients represent δ of the energy in the signal.

978-1-5386-3472-1/17/$31.00 ©2017 IEEE

TABLE II
COMPARISON OF COMPRESSION RATIOS.

Algorithm	Threshold (δ)	Temperature	Humidity	Solar Radiation	Wind Direction	Wind Speed
DCT	0.95	99.9537	99.9769	99.8843	99.722	99.8148
	0.99	99.8611	99.8611	99.6528	87.222	94.9094
	0.999	99.6296	99.5602	99.5278	47.8241	53.491
DWT	0.95	66.5973	63.9159	85.8067	70.0416	83.333
	0.99	56.7499	55.1780	80.7675	61.5118	72.0989
	0.999	50.9015	50.6472	71.6597	52.9820	52.2423
FWHT	0.95	99.9634	99.9390	99.7437	99.5728	99.6826
	0.99	99.7681	99.6704	99.0112	84.3506	92.9688
	0.999	98.7671	98.0347	97.4854	43.8721	50.6470
LDE	0.95	80.52	80.52	80.52	80.52	80.52
	0.99	80.52	80.52	80.52	80.52	80.52
	0.999	80.52	80.52	80.52	80.52	80.52

TABLE III
COMPARISON OF ERROR RATES.

Algorithm	Threshold (δ)	Temperature	Humidity	Solar Radiation	Wind Direction	Wind Speed
DCT	0.95	0.2882	0.2812	0.3761	0.3550	0.4297
	0.99	0.1315	0.1433	0.2141	0.1604	0.1978
	0.999	0.0447	0.0452	0.0678	0.0509	0.0627
DWT	0.95	0.3320	0.3245	0.4733	0.3544	0.4370
	0.99	0.1498	0.1465	0.2145	0.1602	0.1976
	0.999	0.0470	0.0458	0.0676	0.0507	0.0627
FWHT	0.95	0.2889	0.2687	0.3994	0.3490	0.4317
	0.99	0.1167	0.1085	0.1866	0.1476	0.1936
	0.999	0.0313	0.0390	0.0558	0.0392	0.0567
LDE	0.95	0.0041	0.0075	0.0027	0.0035	0.0181
	0.99	0.0034	0.0023	0.0025	0.0032	0.00
	0.999	0.00	0.00	0.0003	0.0001	0.00

$$\frac{\sum_{i=1}^{k} SS_i^2}{\sum_{i=1}^{n} SS_i^2} < \delta. \tag{3}$$

4) In case of DCT, FWHT, and DWT, coefficients smaller than threshold value δ are set to zero. In other words, we do not apply any quantization.

$$SS_i = 0, \text{ if } k+1 \leq i \leq n \tag{4}$$

The compression ratio achievable by each compression method, R_M, is given by:

$$R_M = \frac{|D| - |D'|}{|D|} \times 100\%, \tag{5}$$

where $|D|$ of the size of D and $|D'|$ is the reduced size, and M is individual compression method. Consequently, in case of DCT, DWT, FWHT transformations, we compute k, which is reduced to $(n-k)/n$.

In case of LDE [8], the specific error rate needs to be set beforehand. Let γ denote incompressible ratio and B denote the number of bits to store the indexes. Then, compression ratio can be defined as [8]:

$$|D'| = (1-\gamma) \times \frac{B}{64} + \gamma \times |D| + (2^B - 1) \times 64, \tag{6}$$

where $(1-\gamma) \times \frac{B}{64}$ represents storage requirements for compressible portion (encoded in indexes), $\gamma \times |D|$ represents storage requirements for incompressible portion (as it is), and $(2^B - 1) \times 64$ represents storage requirements for an approximated value of bins or clusters.

Table II shows the compression ratio when δ = 0.99 (99%) in equation (3). In case of LDE, we calculate the compression ratio using common incompressible rate obtained from the five sample data. As shown in Table II, DCT and FWHT show better compression ratios than others. DWT shows the lowest compression ratios because we used a 1-level wavelet transformation, and therefore, k in equation (3) is increased. In case of DCT and FWHT, there are variances in the compression ratio depending on the characteristics of data. The compression rates of temperature data in DCT and FWHT are about 1.1 times and 1.2 times higher than those of wind direction respectively.

C. Error Rate

To measure how much the reconstructed data deviate from the original data, we measure the normalized version of RMSE (NRMSE), a frequently used distortion estimate. Let $x = x_1, x_2, x_3, ...x_n$ be the original data and $\hat{x} = \hat{x}_1, \hat{x}_2, \hat{x}_3, ...\hat{x}_n$ be the reconstructed data. Then, NRMSE for each compression method M can be defined as:

$$NRMSE_M = \frac{RMSE_M}{Mean(x)} = \frac{1}{\bar{x}} \sqrt{\frac{\sum_{n=1}^{N} (x(n) - \hat{x}(n))^2}{N}},$$

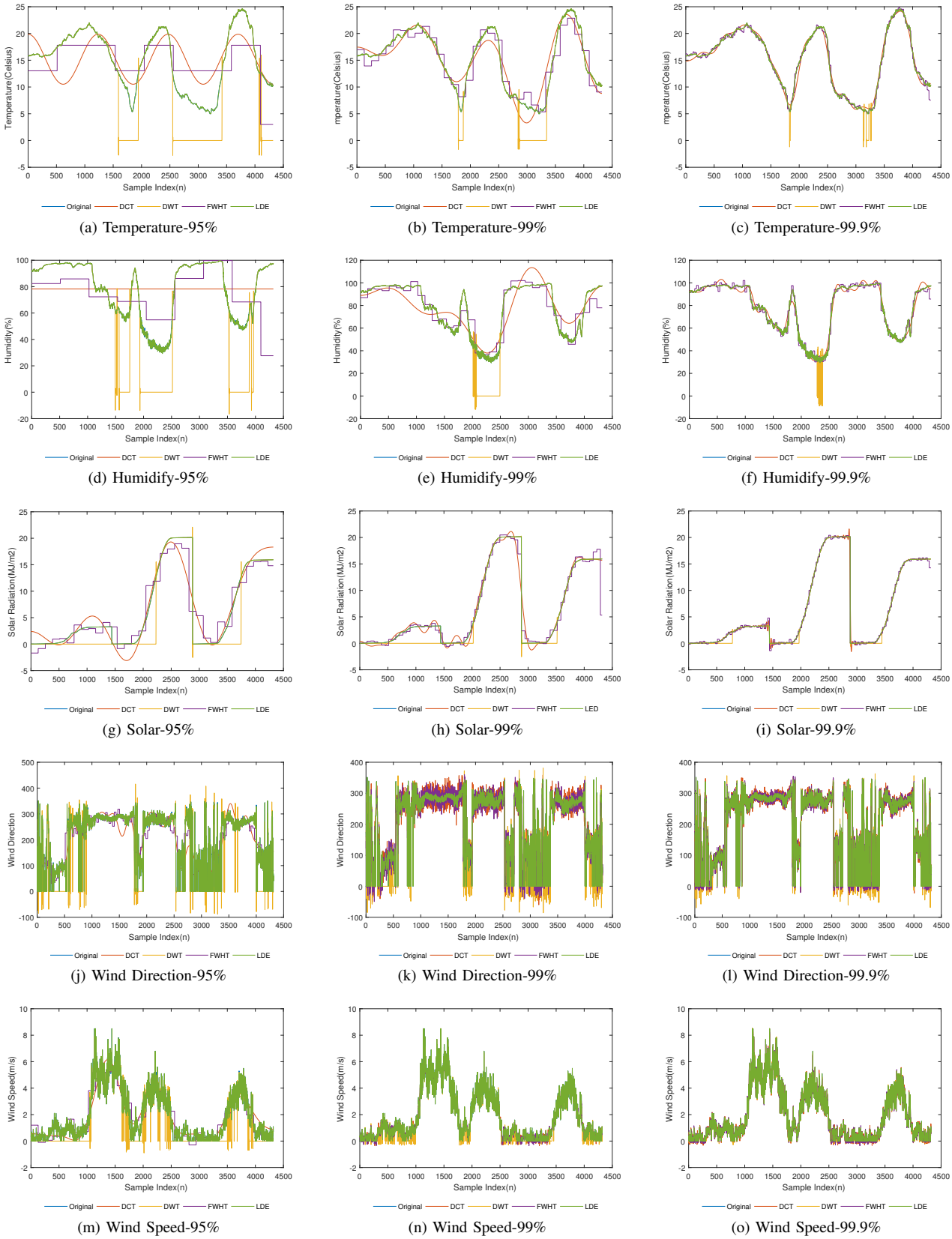
where N is the number of data and $\bar{x}$ is mean of original data x, and $\hat{x}$ is the reconstructed value of x.

The recovered data from our reconstruction method is shown in Fig. 4. Overall, the error threshold of 95% shows slightly higher error rate than that of 99% (or 0.99). DWT shows higher variances than other algorithms although it is similar to other recovered data. For temperature, the reconstructed data using LDE shows the best performance, and FWHT is the worst. LDE also performs the best on the humidity data. DWT performs the worst on this humidity data. On the solar radiation data, LDE also gives the best performance, and DWT and FWHT are the worst.

From these graphs, we can see that when comparing the reconstructed data with the original data, the reconstructed data of DCT and LDE almost coincide with the original data. These results suggest that we can recover data with a small number of measurement or sampling rate. The amount of tolerable error rate depends on application's needs, the weather data in this paper. In the case of FWHT, almost all values are set to zero. Thus as shown in Fig. 4, the threshold value of 95% is unable to reconstruct the humidity values as close as the original values. In other words, it has poor reconstruction capability.

We observe that the reconstruction error increases as the compression ratio gets higher. We also observe that the increase is influenced by user-defined error rates. In Table III, the reconstructed data by LDE almost coincide with the original data for all datasets we evaluated. In the case of DCT and FWHT, the effect of error varied depending on the type of data. Specifically, the DCT increases the error rate of wind speed by 67% compared to the error rate of temperature. However,

978-1-5386-3472-1/17/$31.00 ©2017 IEEE



(a) Temperature-95% (b) Temperature-99% (c) Temperature-99.9%

(d) Humidify-95% (e) Humidify-99% (f) Humidify-99.9%

(g) Solar-95% (h) Solar-99% (i) Solar-99.9%

(j) Wind Direction-95% (k) Wind Direction-99% (l) Wind Direction-99.9%

(m) Wind Speed-95% (n) Wind Speed-99% (o) Wind Speed-99.9%

Fig. 4. Comparison of reconstructed "temperature", "humidify", "solar", "wind direction", and "wind speed" values against the original values. All lossy compression algorithms used 5% error rate ((a), (d), (g) ,(j), (m)), 1% error rate ((b), (e), (h), (k), (n)) and 0.1% error rate ((c), (f), (i), (l), (o)) respectively.

978-1-5386-3472-1/17/$31.00 ©2017 IEEE

TABLE IV
COMPARISON OF ORIGINAL AND RECONSTRUCTED DATA.

Algorithm	Threshold (δ)	Temperature	Humidity	Solar Radiation	Wind Direction	Wind Speed
DCT	0.95	9.3790	48.9450	11.6392	271.3640	4.9796
	0.99	6.4200	31.0900	10.6220	250.0010	2.7305
	0.999	2.7920	11.6660	9.6430	48.7300	0.4486
DWT	0.95	15.2651	77.6400	12.7000	350.7000	3.3710
	0.99	9.1343	55.2450	3.9300	184.2570	1.2553
	0.999	6.7116	40.1575	2.1700	73.2000	0.5060
FWHT	0.95	9.704	70.0280	13.9552	287.3230	4.69087
	0.99	6.97	37.8650	10.6069	202.3950	2.6067
	0.999	3.0490	15.4400	1.6850	1.6850	0.3903
LDE	0.95	0.2000	3.4000	0.0600	15.1000	0.3000
	0.99	0.2000	0.9000	0.0600	3.1000	0.00
	0.999	0.00	0.00	0.0200	0.3000	0.00

what is more surprising is that the magnitude of the simple error is very large in the case of DWT as shown in Table IV. Table IV shows the maximum value of the error between the original data and reconstructed data. In the temperature data, the maximum error is 15.2631 with 95% threshold, which is 76 times larger than that of LDE.

IV. Related Work

Bose et al. [6] compared four existing lossy compression techniques based on signal characteristics of time-series sensor data to identify which compression technique achieves higher compression ratio with minimal error rates.

Numerous algorithms have been proposed for domain-specific applications as well as broad time-series sensor data. In many cases, the data is converted into a different basis such that energy is accumulated into a small number of coefficients in the converted domain. [11] and [14] discussed several latest compression techniques for wireless sensor networks.

Variations of several signal processing algorithms such as DFT, Discrete WHT, DCT and Wavelet have been proposed to effectively compress various datasets from different applications. In [13], those transformation algorithms are applied on Electrocardiogram (ECG) dataset to compare their applicability. In [9], DCT, DWT, and WHT are compared for weather data. Their simulation results show that using DCT-transformed data as a basis has a better performance on weather data recovery compared with other transformation methods such as WHT and DWT. But, they showed that compressed signal is reconstructed with some error [13]. Bicer et al. [15] proposed an online compression algorithm for climate data by exploiting spatial and temporal characteristics exhibited in climate data, thereby improving data retrieval performance.

ISOBAR [16] is a data compression framework designed to improve I/O performance at scale. ISOBAR partitions the original data into two segments, compressible and incompressible, and apply lossless compression to the compressible portion. Chen et al. [8] showed a higher compression ratio than ISABELA [17] in the majority of datasets.

Our approach is different from these prior studies in that we compare the impact of various data transformation techniques on lossy compressions without any complex quantization techniques. Our extensive experimental results using various real-world weather data provides a guideline for IoT applications with regards to choosing the most effective compression algorithm depending on their inherent data characteristics.

V. Conclusion

The emerging IoT smart farm produces a large volume of diverse data, which need to be stored efficiently. In this paper, we evaluated five most important variables in the real weather data as an exemplar of IoT application and compared the performance of reconstructed data using DCT, FWHT, DWT, and LDE to evaluate the feasibility of applying lossy compression on IoT big data. Our experimental results show that the compression ratio of DCT and FWHT is higher than other approaches. However, in the case of NRMSE, LDE significantly outperforms other lossy compression methods such as DCT. We also observe that the reconstructed data by LDE almost coincide with the original data for all five datasets we evaluate. The impact of the error for individual dataset showed high variances in case of DCT and FWHT. Specifically, DCT increases the error rate of wind speed by 67% as compared to the error rate of temperature. Therefore, it is important to select compression coefficients within the range tolerable by the application.

References

[1] R. D. A. Ludena and A. Ahrary, "A Big Data Approach for a New ICT Agriculture Application Development," in *2013 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Oct 2013, pp. 140–143.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[3] M. R. Bendre, R. C. Thool, and V. R. Thool, "Big data in precision agriculture: Weather forecasting for future farming," in *2015 1st International Conference on Next Generation Computing Technologies (NGCT)*, Sept 2015, pp. 744–750.

[4] "Precision agriculture: Using predictive weather analytics to feed future generations," http://www.research.ibm.com/articles/precision_agriculture.shtml.

[5] A. Ukil, S. Bandyopadhyay, and A. Pal, "IoT Data Compression: Sensor-Agnostic Approach," in *2015 Data Compression Conference*, April 2015, pp. 303–312.

[6] T. Bose, S. Bandyopadhyay, S. Kumar, A. Bhattacharyya, and A. Pal, "Signal Characteristics on Sensor Data Compression in IoT – An Investigation," in *13th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, June 2016, pp. 1–6.

[7] J. J. Chou and L. A. Piegl, "Data reduction using cubic rational B-splines," *IEEE Computer Graphics and Applications*, vol. 12, no. 3, pp. 60–68, May 1992.

[8] Z. Chen, S. W. Son, W. Hendrix, A. Agrawal, W.-k. Liao, and A. Choudhary, "NUMARCK: Machine Learning Algorithm for Resiliency and Checkpointing," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2014, pp. 733–744.

[9] R. Sustika and B. Sugiarto, "Compressive Sensing Algorithm for Data Compression on Weather Monitoring System," *TELKOMNIKA Telecommunication, Computing, Electronics and Control*, pp. 974–980, 2016.

[10] M. M. Abo-Zahhad, A. I. Hussein, and A. M. Mohamed, "Compressive Sensing Algorithms for Signal Processing Applications: A Survey," *International Journal of Communications, Network and System Sciences*, vol. 8, no. 6, pp. 197–216, 2015.

[11] M. A. Razzaque, C. J. Bleakley, and S. Dobson, "Compression in wireless sensor networks: A survey and comparative evaluation," *ACM Transactions on Sensor Networks*, vol. 10, no. 1, p. 5, 2013.

978-1-5386-3472-1/17/$31.00 ©2017 IEEE

[12] B. J. Fino and V. R. Algazi, "Unified matrix treatment of the fast walsh-hadamard transform," *IEEE Trans. Comput.*, vol. 25, no. 11, pp. 1142–1146, Nov. 1976.
[13] R. Chaturvedi and Y. Yadav, "A Survey on Compression Techniques for ECG Signals," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 9, 2013.
[14] Y.-C. Wang, "Data Compression Techniques inWireless Sensor Networks," *Pervasive Computing*, 2012.
[15] T. Bicer, J. Yin, D. Chiu, G. Agrawal, and K. Schuchardt, "Integrating Online Compression to Accelerate Large-Scale Data Analytics Applications," in *27th IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*, 2013, pp. 1205–1216.
[16] E. R. Schendel, Y. Jin, N. Shah, J. Chen, C. S. Chang, S. H. Ku, S. Ethier, S. Klasky, R. Latham, R. Ross, and N. F. Samatova, "ISOBAR Preconditioner for Effective and High-throughput Lossless Data Compression," in *IEEE 28th International Conference on Data Engineering*, April 2012, pp. 138–149.
[17] S. Lakshminarasimhan, N. Shah, S. Ethier, S. Klasky, R. Latham, R. Ross, and N. F. Samatova, "Compressing the Incompressible with ISABELA: In-situ Reduction of Spatio-temporal Data," in *Proceedings of the 17th International Conference on Parallel Processing - Volume Part I*, 2011, pp. 366–379.

978-1-5386-3472-1/17/$31.00 ©2017 IEEE