https://doi.org/10.26599/IJCS.2022.9100007 International Journal of Crowd Science

Energy-Efficient Sensory Data Collection Based on Spatiotemporal Correlation in IoT Networks

Jine Tang[1], Shuang Wu[2], Lingxiao Wei[1], Weijing Liu[3] ✉, Taishan Qin[1], Zhangbing Zhou[2] , and Junhua Gu[1]

ABSTRACT

The Internet of Things (IoT) is currently in a stage of rapid development. Hundreds of millions of sensing nodes and intelligent terminals undertake the tasks of sensing and transmitting data. Data collection is the key to realizing data analysis and intelligent application of IoT. The life cycle of IoT is limited by the energy of the IoT nodes in the network. A complex computing model will bring serious or even unbearable burdens to IoT nodes. In this study, we use the data prediction method to explore time correlation data and adjust the appropriate spatial sampling rate on the basis of the spatial correlation of sensory data to further reduce data. Specifically, the improved and optimized DNA-binding protein (DBP) data prediction method can increase the time interval of sensing data to further reduce energy consumption. Based on the spatial characteristics of the sensing data, substituting the data of similar nodes can reduce the sampling rate. The probabilistic wake-up strategy is also adopted to adjust the spatial correlation of the sensing data. On the basis of node priority, an optimized greedy algorithm is proposed to select the appropriate dominating node for eliminating redundant nodes and improving network energy utilization. Experiments have proven that our scheme reduces network energy consumption under the premise of ensuring data reliability.

KEYWORDS
Internet of Things; data collection; spatiotemporal correlation; energy-saving

At present, the application of Internet of Things (IoT) technology has involved various industries and penetrated into all aspects of human life, such as the internet of cars[1,2], smart city[3], and smart home[4]. Data collection is the first step toward a great role for IoT and is key to effective data analysis. Data collection must respond to the different task requirements and is limited by the battery capacity of IoT nodes. It is one of the most basic challenging tasks in IoT. In many scenarios, users do not need strictly accurate data, and approximate data suffice. For example, the temperature perceived by node changes in adjacent time is low in environmental monitoring. In the case of no data mutation, the impact of small differences on production and life can be ignored. Large amounts of accurate and repetitive data not only result in unnecessary resource usage but also in unnecessary transmission energy consumption. A single node collects data continuously, and the collected data constitute a time series, so future data can be predicted from historical data. The sensor transmits data to the node only if the error between the predicted value and the actual measured value is greater than the specified threshold. This bilateral prediction method effectively reduces the number of data transmissions and has been widely used in IoT and wireless sensor networks (WSN)[5]. However, we find that these scenarios assume that the energy consumption of data collection and processing is significantly lower than that of communication. Many real-world applications require specific sensors whose sensing power consumption cannot be ignored. Therefore, reducing the number of IoT node activities is necessary to reduce (i) the amount of node sensing data through time correlation to predict data and (ii) the number of data transmissions. According to the first law of geography, geographical objects or properties that are distributed in space are related. Spatial correlation shows that the trends and values of the perceived data of different nodes are similar in the development process. Thus, the data among some nodes are redundant, and when IoT devices in adjacent locations transmit their sensing data without considering neighbor transmission, an unnecessary transmission may occur[6]. Many researchers use data similarity to reduce data transmission among similar data nodes[7,8]. The similarity measurement among node data sequences is important, and the classical methods include Euclidean distance and Pearson coefficient[9]. The similarity measurement method is proposed in Ref. [10]. However, these methods have some drawbacks. On the one hand, they are sensitive to subtle changes in serial data. On the other hand, the judgment of data similarity requires a large amount of data. When the amount of data is small, even a small difference will cause a great coefficient difference. However, if the amount of data is large, then additional data must be transmitted among nodes, which will further lead to energy consumption. Therefore, in the IoT environment, considering a method that can effectively judge (i) the similarity through spatial correlation for data collection and (ii) the correlation among data sequences in the case of a small amount of data is necessary.

To address the aforementioned issues, we thus propose an effective spatiotemporal correlation-based judgment strategy to collect remote sensing data. Specifically, we use the linear

1 School of Artificial Intelligence, Hebei University of Technology, Tianjin 300401, China
2 School of Information Engineering, China University of Geosciences, Beijing 100083, China
3 Tianjin Institute of Aerospace Mechanical and Electrical Equipment and Tianjin Key Laboratory of Aerospace Intelligent Equipment Technology, Tianjin 300301, China
Address correspondence to Weijing Liu, xiaodou7356@126.com

prediction model DNA-binding protein (DBP) to predict the perceived data of IoT. We also adopt a jump sensing strategy to reduce the amounts of sensed and transmitted data of IoT nodes. To further reduce the redundancy of transmitted data and the energy consumption of nodes, we try to use the slope of the DBP function as the standard to evaluate the data similarity among nodes. In addition, we select the appropriate dominant nodes by using a node priority method to improve the utilization rate of network energy. We also propose a probabilistic wake-up mechanism to determine whether the data correlation among nodes has changed and whether it must be recalculated. The main contributions of this paper are as follows:

• a leaping perception strategy,

• a new standard for evaluating the similarity of nodes, and

• a method for judging whether the data correlation of nodes changes.

The remainder of this paper is organized as follows: Section 1 provides a review of related studies. Section 2 introduces the time correlation data prediction. Section 3 presents the spatial correlation-based data collection. Section 4 provides the experimental results. Section 5 concludes.

1 Related Work

1.1 Network energy-saving data collection methods

In the IoT network with limited energy, data collection, processing, and transmission in an energy-saving way are the key to extending the network life cycle. Clustering is the most common and effective method to prolong the network life cycle and improve network management efficiency. LEACH[11] is the earliest widely used cluster-based routing protocol. It selects the cluster head randomly. However, because cluster heads undertake many tasks, they consume additional energy. The random selection of cluster heads makes it possible for nodes with low energy to serve as cluster heads, leading to the fast death of nodes and shortening of network life. To solve the problem that the fixed threshold cannot meet all conditions in the static research scheme, a fuzzy adaptive method is proposed in Ref. [12] to formulate the exponential membership function according to the energy level in the network for adjusting cluster head selection. However, the clustering method only reduces the transmission distance and does not deal with the redundant data among nodes. To reduce data redundancy and improve network life, a hybrid wake-up mechanism ETASA is proposed in Ref. [13] to achieve the load balancing of traffic and energy efficiency in heterogeneous WSN. In this method, the sensor nodes in the same application are paired together with a short distance within the communication range of the same cluster by using the pairing strategy. Then, the sleep-wake mechanism is used to make the paired nodes perform data sensing and transmission. However, selecting cluster head and awakening time is still the key issue. The selection strategy of cluster heads is related to whether the network load is balanced and whether the energy is used most effectively. The choice of wake-up time is related to whether the network can respond to environmental changes in time and reduce network delay.

The energy-saving method based on mobile nodes plays an important role in improving energy consumption balance, data throughput, and network flexibility. An energy-saving dynamic routing algorithm based on relay nodes is proposed in Ref. [14]. An optimal relay node is selected as its next hop for the node in the routing request before data transmission, and the link reward and link cost are balanced with the adjustment factor. However, the method based on mobile nodes has high cost and low suitability for low-cost, large-scale networks. In Ref. [15], based on the data prediction method, a set of dynamic probability models is proposed to be obtained by training Markov dynamic models. The data can be mapped through a random process described by the probability density function. Then, unknown data can be predicted by observed data and probability density function. When the model is valid, the base station can respond to the user query only by obtaining a data expectation value according to the probability model, without communication among nodes. In Ref. [16], an intelligent terminal probability prediction algorithm based on an improved convolutional neural network (CNN) is adopted to reduce complexity and energy consumption and avoid the loss of important information. Data compression is the simplest way to reduce the amount of data for reducing storing and transmitting data costs. Compression technology can be divided into lossless and lossy compressions according to whether data information is lost[14]. Common methods of lossless compression include Huffman coding[17], LZW dictionary compression[18], and RLE travel coding compression[19]. Common methods of lossy compression include wavelet transform[20], principal component analysis[21], and compressive sensing[22]. Reducing the amount of data transferred through data compression is one method to reduce energy consumption.

1.2 Data collection method based on time correlation

The method based on time correlation is a way of time sequence-based data collection. If the data at a future time can be predicted, then the sampling and communication frequency of such data can be changed. In Ref. [23], based on the autoregressive time series model, without a great deal of training data and prior knowledge of the probability distribution of perceived data, historical data can be fit, data in future time slot can be predicted, communication cost can be greatly reduced, and loss rate can be improved. In Ref. [24], the holt exponential double smoothing method is employed to predict data on sensor nodes. The predicted values are obtained by the weighted sum of recent sensory data and historical observation data. The sampling interval is adjusted adaptively according to prediction accuracy. However, the exponential smoothing method can only specify one parameter as the weight value of the most recent sensory data and cannot adjust according to the change of data. In Ref. [25], the importance of each video frame is evaluated through the classification of CNN with a time dimension. Then, keyframes are extracted, and unimportant frames are discarded by using a decision algorithm, which preserves video quality and reduces multimedia data redundancy in the data center. Based on the fact that there should be no information or correlation in residuals when the model is applied to a given time series, a three-layer least-squares residual depth is proposed to support quantitative time-series data prediction to the machine[26]. If the number of training data transmissions can be further reduced or the sampling interval can be further increased to reduce the amount of sensory data, then it will be a good way to collect data.

1.3 Data collection method based on spatial correlation

A high degree of similarity among sensor sensing data is called spatial correlation. Common spatial correlation methods include clustering-based and sleep-wake-based methods. In Ref. [27], the absolute value of the difference in time series is proposed to judge the data correlation among nodes, divide the sensor network into different clusters, find the correlation among node data in the cluster at each cluster head node, upload the model parameters to the sink node by the cluster head, and collect the global approximate data at the sink node. A fine-grained spatial correlation model is put forward in Ref. [28], where the sensor can be divided into different clusters by using the probabilistic approximation method. One primary sensor is selected from each cluster, and the other sensors serve as subordinate sensors so that the primary sensor in the network sends all the sensing data to the receiver, whereas the subordinate sensors only need to send nonredundant data. In Ref. [29], Pearson coefficient, Spearman grade coefficient, and Kent grade coefficient are used to evaluate the correlation among data sequences, and the performance analysis is carried out. The result reveals that the correlation can be affected by the outliers in the sensory data. Therefore, the positive and negative values of the correlation coefficients are changed. The correlation among nodes is not invariable, so timely perceiving the change in correlation, making corresponding adjustments, and striking a balance between the quality of data collection and the energy consumption required for data collection are necessary.

2 Time Correlation Data Prediction

Due to the limited memory space, computing power, and node energy in IoT, data must be collected in an energy-efficient way. To solve this problem, the needs of IoT nodes with energy and computing power constraints can be met to a great extent by predicting the sensing data at the next moment and adjusting the prediction interval.

2.1 Prediction model

We adopt the linear prediction model DBP for data prediction. When a short-term linear behavior is observed in the data, the trend acquisition is more sustainable than the difference acquisition alone to predict data. Meanwhile, DBP can carry out long-term nonlinear trend prediction, and in this process, the model must be constantly updated.

Figure 1 provides a schematic of DBP. This model is based on a sliding learning window containing W data points. The sets of the first and last L data are called edges, and the size of W is greater than twice that of L. L_1 represents L edge points at the beginning of the learning window, and L_2 represents L edge points at the end of the learning window. The model parameter δ is the average of L_1 and L_2. This calculation is similar to the calculation of derivatives, so it is called derivative-based prediction, which is formulated as follows:

$$\delta = \frac{\left| \sum_{i=0}^{L_1} d_i - \sum_{j=1}^{L_2} d_j \right|}{L(W-L)} \tag{1}$$

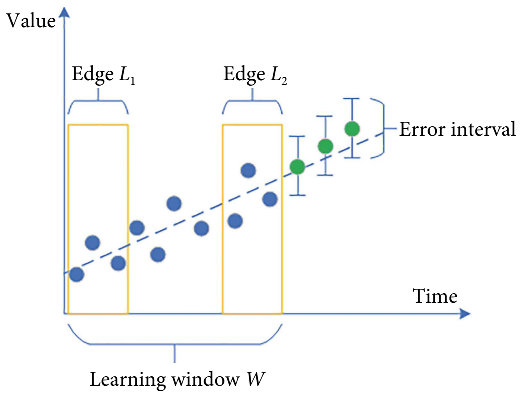

Fig. 1 DBP model.

In general, δ represents the average increment of inductive data relative to the previous learning window. Given the sensory data of a certain IoT node in the learning window as the initial data point, the sensory data at the next moment can be predicted by the following formula:

$$d_p[k+1] = d[k] + \delta \tag{2}$$

where d_p represents the forecast data, which are related to the data at the previous moment. If the forecast data at the previous moment are valid, then the forecast data at the next moment will be calculated. If the forecast data at the previous moment are invalid, then the sensory data will be used for calculation. The DBP data prediction model is found suitable for resource-constrained IoT nodes.

2.2 Data initialization model

In the beginning, when no data are available in the sensing and server nodes of IoT, the sensing nodes must upload the initial data to the server. In the previous section, we have introduced the details and advantages of the DBP model, but some shortcomings remain in the algorithmic model. Reducing the energy consumption of network initialization is our first step for algorithm optimization. In this study, a deformable part model (DPM)-based linear prediction algorithm is used for data initialization.

DPM algorithm only needs two measurement values to establish the prediction model. In the beginning, the IoT node collects the measured values $d[0]$ and $d[1]$ of the first two moments. Each time the sink receives a new measurement, it stores the value of the measurement received in memory and uploads them to the server node. Then, the difference Δd between the two measurements is calculated for the IoT node and the server node at the same time, as shown in the following formula:

$$\Delta d = d[1] - d[0] \tag{3}$$

To predict the data at time k, IoT and server nodes add Δd to the data at time $k-1$ to obtain the predicted value at time k.

$$d_p[k] = d[k-1] + \Delta d \tag{4}$$

The IoT nodes then compare predicted value $d[k]$ with the actual perceived value. If the difference between them is not greater than the previously set error threshold TH, then the truly sensed data will not be transmitted to the server node. Conversely, if the predicted value is not within the error threshold, then the IoT nodes will delete it and transmit the real sensing data to the server node. They simultaneously update difference Δd and start the new data prediction and transmission.

2.3 Jump strategy

The energy consumption of IoT nodes mainly comes from two aspects: data transmission and data sensing. Regardless of the bilateral prediction of the DBP algorithm or the use of the DPM algorithm for data initialization, the aim is to minimize the energy consumption in data transmission, so the improvement and optimization of data sensing comprise the second step of DBP optimization.

Through the analysis, we can see that DBP has good prediction performance in the case of short- and medium-term predictions. Therefore, IoT nodes do not need to transmit data to server nodes in continuous multiple timeslots. The excellent prediction performance of DBP makes it possible to further reduce data sensing and network energy consumption. We change the sensing time interval of IoT nodes to reduce the amount of sensory data, that is, the “jump” strategy.

When predictive functions are good at predicting sensory data, we do not need to sense real data all the time. Therefore, we propose a method to further save energy. As shown in Fig. 2, at a certain moment when the difference between predicted and real data is within the scope of the set error threshold, the time interval of sensory data can be increased. To avoid an unlimited increase, the time interval can be set to a maximum value. If the predicted data value cannot meet the requirements of the data error threshold at the perceived moment, the sensed data interval is restored to the initial value.

The time interval is defined as $S = 1$, representing the time interval from the current perceived moment to the next perceived moment. If the predicted data are accurate, then S should be added by 1; otherwise, S will be restored to 1. At each time point, the IoT nodes make a judgment through time interval S. If time interval $S = 0$, then the current moment is the perceived moment. If the time interval is greater than 0, then the sensing data are saved as node data, and S is reduced by 1. The node data formula is as follows:

$$d[k]=\begin{cases}x_p[k], & S>0\&\&|d_s[k]-d_p[k]|<TH;\\ x_s[k], & S=0\|\,|d_s[k]-d_p[k]|\geqslant TH\end{cases}\tag{5}$$

where $d[k]$ represents the data stored in IoT nodes and servers at time k, $x_p[k]$ is the predicted data that will be saved at time k, and $x_s[k]$ represents the sensing data that will be saved at time k. When time interval S is greater than 0 and the difference between predicted data $d_p[k]$ and sensing data $d_s[k]$ is less than error threshold TH, the predicted data will be saved. When time interval S is equal to 0 or the difference between predicted data $d_p[k]$ and sensing data $d_s[k]$ is greater than or equal to error threshold TH, the sensing data uploaded by sensor nodes will be saved.

Algorithm 1 describes the final data forecast. Given the first two sensing data, data error threshold and sliding window learning parameters, such as length and time interval, the INIT_PRE algorithm initializes W sensory data, and then the DBP algorithm calculates the average slope of sliding learning window δ. Finally, the prediction data in the next time point are calculated according to the average slope (lines 1–3). When the value of the interval indicator variable is greater than 0, it means entering the nonperception stage. The prediction data are directly stored in PRE, and interval indicator variable s is reduced by 1. Subsequently, the next prediction is made until the value of s is 0 (lines 4–9). The next stage is to sense data, and the sensory data will be saved to SD (line 10). Then, whether the difference between the current sensory data and the predicted model data exceeds the range of error threshold is determined. If it does not exceed, then the forecast data will be saved in PRE, and then whether the current time interval can be increased is determined. If the current time interval can be increased, then the time interval will be added 1 in the nonperception stage, and the current time interval is recorded with indicator variable (lines 11–17). If the gap exceeds the error threshold, then PRE must save the current sensory data and transmit it to the server node. Next, the model goes to line 2 to retrain the model and update the parameters during prediction (lines 18–24). The time complexity of Algorithm 1 is $O(Nu+L\times Sn)$, where Nu stands for the amount of data saved in the PRE data sequence, L represents the size of the learning edge in the sliding learning window, and Sn refers to the time of updating the prediction model during transmission.

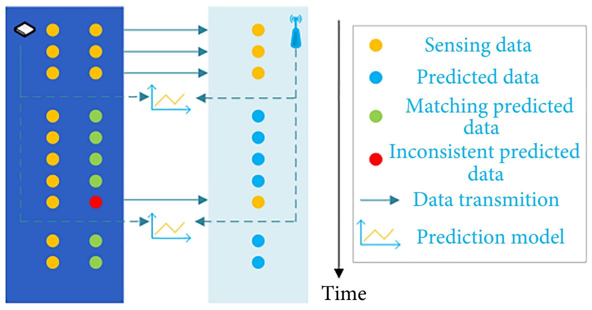

Fig. 2 Jump strategy diagram.

Algorithm 1 DSM

```
Input: $TH$: error threshold between the predicted value and the true value
  $MAX\_S$: maximum time interval
  $d[0]$: first sensory data
  $d[1]$: second sensory data
  $S$: time interval, and initialized to 1
  $W$: length of learn window
  $L$: learn window edge length
  $k$: W+1, the next moment of the training window
Output: $PRE$: the data sequence
1:  INIT_PRE(): $s \leftarrow 0$
2:  $\delta \leftarrow$ DBP($PRE.splice(-W)$)
3:  $d[k] \leftarrow PRE[k-1]+\delta$
4:  for all $s>0$ do
5:      $PRE$.push($d[k]$)
6:      $s \leftarrow s-1$
7:      $k \leftarrow k+1$
8:      $d[k] \leftarrow PRE[k-1]+\delta$
9:  end for
10: $SD(k) \leftarrow getSensor(k)$
11: if $|d[k]-SD(k)|<TH$ then
12:     $PRE$.push($d[k]$)
13:     if $S<MAX\_S$ then
14:         $S \leftarrow S+1$
15:         $s \leftarrow S$
16:     end if
17: end if
18: if $|d[k]-SD(k)|>TH$ then
19:     $PRE$.push($SD(k)$)
20:     $S \leftarrow 1$
21:     $s \leftarrow 0$
22:     data are transmitted to the server node
23:     goto 2
24: end if
```

2.4 Data synchronization

The ultimate goal of data collection is to have data available on the server nodes. On the basis of the data prediction model, the server and IoT nodes thus run the same prediction model at the same time to ensure data synchronization on both sides.

The data synchronization policy applied to each node comprises the following steps. The IoT nodes construct a data prediction model on the basis of the initial and observed values. Next, the initial value is delivered to the server nodes, which construct an identical data prediction model at the same time. Meanwhile, the IoT nodes also use the model to predict their own sensory data and compare the predicted value with the actual observed value. If the difference between them is within the allowable margin of error, then no further action is required. Otherwise, the IoT nodes will transmit the real sensory data of the wrong time to the server nodes, and both sides will rebuild a new model to collect data.

To implement this strategy, applications running on server and IoT nodes must allow a fault-tolerant mechanism for the accuracy of reported data. When both nodes have the same prediction model at the same time, a point is sampled, and the sensory data are compared with the value predicted by DBP according to the current model. If the absolute value of the difference is less than error threshold TH, then the value is an acceptable approximation of the real value. Otherwise, the current data and prediction model are considered not reliable enough, and the real data are sent to the server nodes to recalculate the data prediction model.

3 Data Collection Based on Spatial Correlation

This section demonstrates how to use the spatial correlation of nodes in IoT to select appropriate sampling nodes. To minimize network energy consumption and meet data reliability constraints during the data collection process, we propose a data collection strategy on the basis of the spatial correlation among nodes. This strategy is divided into two steps. First, a network correlation map is constructed according to the spatial correlation among nodes, and the dominant nodes are selected by using the greedy strategy according to the priority of nodes. Second, the network dominance node is adjusted according to the dynamic changes in data after the dominance node has been selected.

3.1 Minimum domination set selection

Intuitively, the correlation among data reported by IoT nodes can greatly reduce the spatial sampling rate of IoT nodes. If the data of two IoT nodes A and B are similar in the past period, then we can infer that the sensing data of A and B are similar in the next moment; that is, the sensing data of one IoT node can replace the sensing data of the other node. Given time series $A=\{a_1, a_2, \ldots, a_W\}$ and time series $B=\{b_1, b_2, \ldots, b_W\}$, they are relevant if the following formula is satisfied:

$$|\delta_a-\delta_b|<\theta \tag{6}$$

where δ is the average increment of node prediction, which represents the variation trend of time series data to a certain extent. If the current parameter is still valid, then the server node can directly calculate the node similarity. If the current parameter does not meet the requirements, then the IoT nodes only need to transmit one data to the server node to meet the requirements, thus greatly reducing the number of data transmissions among nodes and the energy consumption of data transmission. θ is a similarity parameter set for different application scenarios. Formula (6) shows that when the absolute value of the difference in average increment δ between two nodes is less than the preset parameter, the two nodes meet the requirement of similarity and have a similarity. Otherwise, we consider that no similarity is observed between the two nodes. The correlation among node data sequences is stored in matrix ADJ as follows:

$$ADJ=\begin{pmatrix}1 & 0 & \cdots & 1 & 0\\ 0 & 1 & \cdots & 0 & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots\\ 1 & 0 & \cdots & 1 & 0\\ 0 & 0 & \cdots & 0 & 1\end{pmatrix}$$

where ADJ has N rows and N columns. $ADJ[i][j]$ indicates whether a correlation exists between the ith and jth nodes. Value 1 indicates that the two nodes are correlated, whereas value 0 suggests the opposite. ADJ must be a symmetric matrix because there should be a correlation between nodes and themselves, and the correlation between nodes i and j is equal to the correlation between nodes j and i.

In the IoT scenario, the remaining energy of the selected node must also be considered, including the energy consumed by the node for data transmission. Therefore, we propose node priority Eq. (7) to describe the priority order in which a node is selected as a node of the dominant set.

$$priority_i=\alpha\times\frac{NS_{i.eng}}{\sum\limits_{j\in C_k}NS_{j.eng}}-\beta\times\frac{dis(NS_i,SE)}{\sum\limits_{j\in C_k}dis(NS_j,SE)}+\gamma\times\frac{D_i}{\sum\limits_{j\in C_k}D_j} \tag{7}$$

where $priority_i$ represents the priority of node i whose value is related to the remaining energy of the node, the distance from the node to the server node, and the node degree. $NS_{i.eng}$ is the remaining energy of node i. $dis(NS_i,SE)$ is the distance between node i and server node. D_i is the degree of node i, which refers to the number of edges associated with it. α, β, and γ are priority parameters, representing the weight of residual energy, the weight of distance to the server node, and the weight of node degree, respectively. C_k represents the k-th cluster, where node i resides. The priority of a node is the weighted sum of its remaining energy, the distance from the node to the server node, and the node degree.

In Algorithm 2, the correlation between the two nodes is determined through correlation parameters. If the correlation exists, then the degree between two nodes will be added 1. Correlation matrix elements $ADJ[i][j]$ and $ADJ[j][i]$ are set as 1 (lines 1–8). Then, the remaining energy, the sum of the distance from the node to the server node, and the sum of node degree are calculated for all nodes (lines 9–13). Finally, according to the remaining energy of each node, the distance and degree are calculated using the node priority formula and stored into the PRI set (lines 14–17). The time complexity of Algorithm 2 is $O(N\times N)$.

3.2 Greedy algorithm

We transform the selection of nonredundant IoT nodes into the solution of the minimum domination set problem, which is an NP–hard problem, and propose an optimized greedy algorithm to solve it. The optimizing greedy algorithm consists of selecting and optimizing a minimum domination node set.

3.2.1 Selecting the minimum domination node Set.

The nodes are selected in ascending order of priority. Other nodes associated with the current node are saved in the TMP collection. If undominated nodes are seen, then the nodes dominated by the current node are not dominated by other nodes, so they should be added to the minimum domination node set. If no undominated node exists, then this node has no contribution to the minimum dominated node set and is thus excluded. The TMP_ALL set

Algorithm 2 CalculNodeCorPri

```
Input: $NS$: set of IoT node
    $\delta$: average incremental set of nodes
    $\theta$: correlation parameter
Output: $D$: set of node degrees
    $ADJ$: correlation matrix
    $PRI$: set of node priorities
1:  for all $\delta_i \in \delta$ do
2:    for all $\delta_j \in \delta$ & $i \neq j$ do
3:      if $|\delta_i - \delta_j| < \theta$ then
4:        $D_i \leftarrow D_i+1$
5:        $D_j \leftarrow D_j+1$
6:        $ADJ[i][j] \leftarrow 1$
7:        $ADJ[j][i] \leftarrow 1$
8:      end if
9:    end for
10: end for
11: for all $NS_i \in NS$ do
12:   $E\_SUM \leftarrow E\_SUM + NS_i.\text{eng}$
13:   $DIS\_SUM \leftarrow DIS\_SUM + \text{distance}(NS_i, SE)$
14:   $D_{SUM} \leftarrow D_{SUM} + D_i$
15: end for
16: for all $NS_i \in NS$ do
17:   $pi \leftarrow \text{priority}(NS_i)$
18:   $PRI.\text{push}(pi)$
19: end for
```

includes all nodes dominated by the current DM. If the nodes in TMP_ALL are the same as the NS node set, then the current DM is already a dominant set of the undirected network graph.

3.2.2 Optimizing the minimum domination node Set

Figure 3 shows the domination set of an undirected graph. The blue nodes represent the dominant nodes of the network, and the number above the nodes represents the priority of the nodes. As displayed in Fig. 3, the blue and white nodes related to them cover all the nodes in the figure, so the blue node set {10, 9, 8, 6} is a network dominance set. However, all the dominant nodes with priority 8 are included in the domination set of other nodes. The node with priority 8 is a redundant node in the domination set. Therefore, further optimization can be carried out to remove the redundant nodes that may exist in the dominant set. Considering the node priority, we tend to remove the low priority nodes and preserve the high priority nodes. If the remaining nodes can dominate the whole network, then the nodes will be deleted from the domination node set.

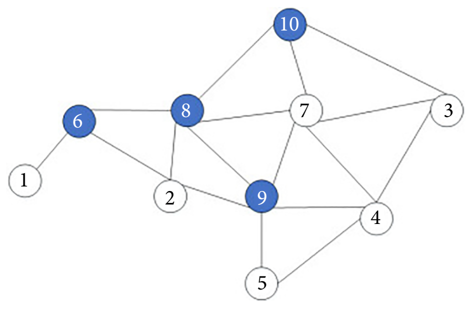

Fig. 3 The domination set of an undirected graph.

Algorithm 3 shows the greedy algorithm of the minimum domination node set. Given the nodes, priority set, and correlation matrix, each node is traversed, and a temporary set is created for storing other nodes related to this node for each traversed node (lines 1–5). TMP_ALL stores the nonrepeating node set that has been dominated, and all the nodes controlled by this node are determined whether they are already in TMP_ALL. If nodes that are not dominated by other nodes exist, then the addition of these nodes contributes to the complete coverage of the entire network node. Subsequently, these nodes are added to the domination node set, and the TMP_ALL set is updated. Meanwhile, the current domination node set TMP is saved (lines 6–10). If the current domination node has covered all nodes in the network, then the node traversal cycle ends (lines 11–13). Next, the nodes of the domination set are arranged in descending order of priority, and each node is traversed from low to high. Each traversed node is temporarily removed from the domination set to form a new domination set (lines 15–17). The new domination set should be traversed, and all nodes dominated by it must be saved (lines 18–20). If the dominant nodes of the new dominant set can cover the entire network, then the deleted nodes are redundant in

Algorithm 3 Greedy algorithm

```
Input: $NS$: set of IoT nodes
    $PRI$: set of node priorities
    $ADJ$: correlation matrix
Output: $DM$: minimum domination node set
1:  for all $priority_i \in PRI$ do
2:    $TMP \leftarrow$ new Set($NS_i$)
3:    for all $priority_j \in PRI$ & $ADJ[i][j] = 1$ do
4:      $TMP$.add($NS_j$)
5:    end for
6:    if $TMP$ - ($TMP_{ALL} \cap TMP$ is not empty then
7:      $DM \leftarrow DM \cup NS_i$
8:      $TMP_{ALL} \leftarrow TMP_{ALL} \cup TMP$
9:      $GS.NS_i \leftarrow TMP$
10:   end if
11:   if $TMP_{ALL} = NS$ then
12:     break
13:   end if
14: end for
15: the $DM$ is arranged in reverse order of $PRI$ size
16: for all $NS_i \in DM$ do
17:   $TMP2 \leftarrow DM - NS_i$
18:   for all $NS_j \in TMP2$ do
19:     $TMP3 \leftarrow TMP \cup GS.NS_j$
20:   end for
21:   if $TM3 = NS$ then
22:     $DM \leftarrow TMP2$
23:   end if
24: end for
```

the domination set and can be formally removed from it (lines 21–24). The time complexity of Algorithm 3 is $O(N \times N)$.

3.3 Probabilistic wake-up strategy

In practice, the correlation among nodes may change over time. To judge such correlation, we must know the sensory data of each node. Therefore, we propose a probability-based wake-up method, which not only can keep the validity of spatial correlation but also can detect the change in correlation among nodes.

For each moment, the probability of the controlled IoT node entering the working state is set as p, which is called the wake-up probability. The size of p is related to the size of the domination node set, and the probability changes with the size of the domination node set. A large domination node set should have a small wake-up probability, and a small domination node set must have a large wake-up probability to achieve a balance between the validity of spatial correlation and the similarity of detection nodes. Each dominated node has only one corresponding dominant node. If the number of nodes controlled by the dominant node is pn, then the wake-up probability of nodes under the dominant node is $p = PN / pn$, where PN is a probability constant and pn is the number of nodes dominated by a node.

The larger the pn, the larger the set of the dominant nodes, and each of the dominated nodes has a small wake-up probability. Otherwise, the wake-up probability is high. Then, the system generates a random number between 0 and 1. If the random number is less than p, then the dominated node is awakened; otherwise, it will not be awakened. If the dominated node is awakened, then it will sense the data and judge whether any correlation exists among the nodes dominating it currently.

We can control the wake-up probability of nodes by controlling the probability constant PN. When the wake-up probability of each node is 1, it is equivalent to uploading data directly without judgment. When the wake-up probability of each node is 0, it is equivalent to not uploading data, and the validity of spatial correlation cannot be maintained. When the wake-up probability of nodes gradually increases between 0 and 1, the effectiveness of spatial correlation becomes increasingly strong. At the same time, the consumption in uploading data becomes increasingly large, but it is still less than that in uploading data directly.

4 Implementation and Evaluation

We use python for the simulation experiments. The operating system is 64-bit Windows 10, the processor is 1.7 GHz Intel i5-4 210 CPU, and the memory is 4 GB. Our simulations use the real dataset provided by Intel's Berkeley Research Lab. In this dataset, 54 Mica2Dot sensors with weather panels collect humidity, temperature, illumination, and voltage values every 31 seconds. On the premise of losing generality, we take 500 data from each node for experiments and carry out ten experiments to obtain the average result.

4.1 Experimental setup

Table 1 describes the settings of the experimental environment. The simulated environment is a 100 m × 100 m IoT network with 54 IoT nodes. The server node is located (50 m, 50 m) away from the center point, and other nodes are distributed with different skewness. Skewnessed sd represents the degree of the uneven distribution of nodes in IoT, and the calculation formula is as follows:

$$sd = \frac{dn - sn}{N} \times 100\% \tag{8}$$

Table 1 Experimental parameters and their settings.

Parameters	Value
Network region	100 m × 100 m
Number of IoT nodes	54
Skewness degree	0%, 10%, 20%, 40%, 60%, 80%
Server node location	(50 m, 50 m)
Transmission radius	100 m
Propagation energy constant	50 nJ
Energy consumption constant ε	0.1 nJ
Power supply voltage V	2.7 V
Electricity required for induction activity I	25 mA
Duration of sensed node T	0.5 ms
Packet size	64 bit

where dn and sn are the numbers of IoT nodes in dense and sparse subregions, respectively. $(dn - sn)$ indicates that the number of IoT nodes in dense subareas is greater than that in sparse subareas, and N indicates the number of IoT nodes in the IoT network area. A skewness of 0% means that IoT nodes are evenly and randomly distributed in the IoT network area. The transmission radius of IoT nodes is set as 100 m, two nodes can communicate with each other within this radius, and each IoT node has the same perception ability. To ensure network connectivity, we assume that any two nodes in the network are within the communication range.

4.2 Evaluation factors

4.2.1 Different probability constants

By setting different probability constants, the influence of probability constants on the experimental effect is tested. The value can be $PN = 0$, $PN = 1$, $PN = 10$, $PN = 20$, $PN = 30$ or $PN = pn$.

4.2.2 Different data prediction models on nodes

We use DBP to refer to the method in Ref. [30], which is the bilateral data prediction method by DBP prediction algorithm. DSM refers to the previously mentioned method of adding "jump" and initialization optimization algorithms to DBP algorithms. EDSAS refers to the data prediction collection method employed in Ref. [24], which adopts the quadratic exponential smoothing method for data prediction. ADC is the method used in Ref. [10], which adopts linear prediction function and spatial correlation for data collection. DLM refers to the method employed in Ref. [23], which uses the autoregressive model to fit historical data and predict the data in future time slots. TSD is the data prediction collection method combining the temporal and spatial correlations proposed in our study. By comparing DBP, DSM, and TSD methods, we determine the influence of the time correlation optimization method on the results.

4.2.3 Various error thresholds and skewness of node distribution

Error threshold indicates that the deviation between the predicted real data within a certain range is acceptable. In our experiment, three different thresholds are set: $TH = 0.1$, $TH = 0.15$, and $TH = 0.2$. At the same time, the IoT node distribution will affect the data transmission distance among IoT nodes; thus, the total energy consumption of the network will be affected. To determine the influence of IoT node distribution on our method, we set the skewness as 0%, 10%, 20%, 40%, 60%, and 80%.

4.3 Experimental evaluation

4.3.1 Influence of probability constant

Figure 4 shows the different network energy consumption and mean absolute errors of data brought by different probability constants in the IoT, where the network skewness is 0%, and the error threshold is 0.1. The blue bar chart represents the network energy consumption (EC), and the red broken line chart represents the mean absolute error (MAE) of data. A probability constant of 0 means that our probabilistic wake-up strategy is not adopted in this network; that is, the dominated node is not awakened during the entire network life cycle; thus, the energy consumption of node sensing and data transmission is saved. When the probability constant is 0, the energy consumption is the lowest. However, the data error is far beyond the data error threshold 0.1 set in the experiment. In addition, as the probability constant increases from 1 to 30, the network energy consumption increases gradually. As the probability constant increases, the awakening of the dominated node increases, and the number of sensing and data transmissions of each node increases, so the energy consumption also increases.

4.3.2 Influence of different data collection methods

Figure 5 shows that for three different data error thresholds, the TSD method has the lowest network energy consumption, followed by DSM and DBP. TSD makes use of not only time correlation data but also spatial correlation and jump perception data. DSM uses and optimizes time correlation data, whereas DBP only uses time correlation data. Therefore, by comparison, the TSD method consumes less energy and is better than the two other methods.

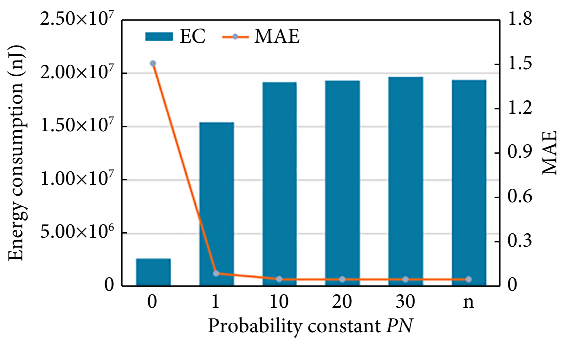

Fig. 4 Comparison of energy consumption and error under different probability constants.

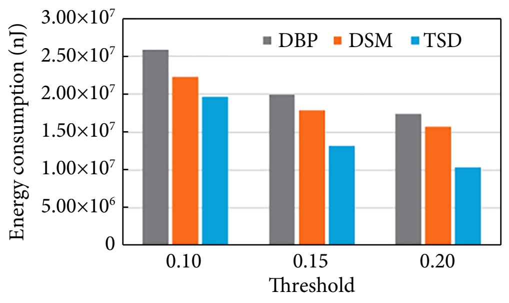

Fig. 5 Comparison of energy consumption in different data collection and prediction methods under different error thresholds.

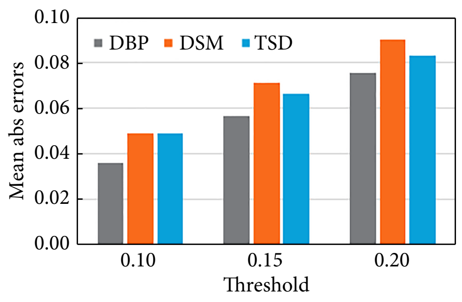

Fig. 6 Comparison of mean absolute errors of different data collection and prediction methods under different error thresholds.

Figure 6 shows that for three different data error thresholds, the network mean absolute error of the DSM method is the largest, followed by TSD and DBP. DSM and TSD adopt the interval data predictive collection method, which means that the IoT nodes do not carry out data sensing at some time points but only perform approximate data replacement at the last moment. Therefore, the average absolute error of DSM and TSD is slightly larger than that of DBP. However, the error value of TSD is smaller than that of DSM, and the error in each case is within the error threshold, so the data collected by TSD are considered valid.

4.3.3 Influence of different error thresholds

Figure 7 displays the energy consumption of TSD, DLM, ADC, and EDSAS under different error thresholds. As the error threshold increases, their network energy consumption shows a trend of gradual decline. The larger the error threshold, the more opportunities the nodes have to use the predicted data. Therefore, the sensing and transmission times gradually decrease, so does the network energy consumption. At the same time, the network energy consumption of TSD is far less than that of EDSAS and DLM under three different error thresholds. TSD adds a spatially correlated data predictive collection method, which reduces the energy consumed in the network. Meanwhile, the network energy consumption of TSD is far less than that of ADC. On the one hand, TSD adopts different time prediction methods and adaptively adjusts the perceived node interval. On the other hand, TSD adopts a probabilistic wake-up strategy instead of a polling wake-up strategy, which greatly reduces network energy consumption.

Figure 8 shows the mean absolute errors of TSD, DLM, ADC, and EDSAS under different error thresholds. As the error threshold increases, their mean absolute errors exhibit a trend of gradual increase. The larger the error threshold, the more times the nodes use the predicted data, which means there exists a large gap between the predicted and real data. Therefore, the mean absolute error of the whole network will gradually rise. At the same time, the mean absolute error of TSD is less than those of

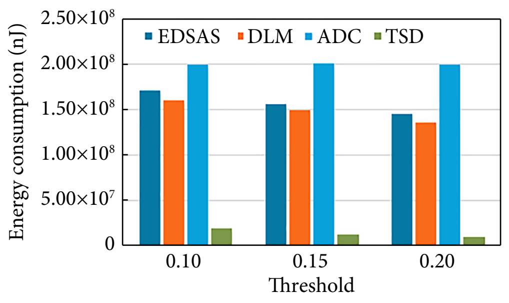

Fig. 7 Comparison of network energy consumption under different error thresholds.

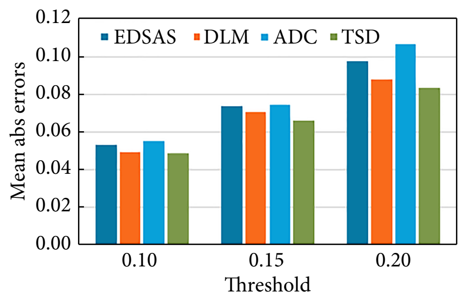

Fig. 8 Comparison of mean absolute errors under different error thresholds.

EDSAS and DLM under three different error thresholds. TSD not only applies the time correlation prediction method but also uses the spatial correlation data prediction collection method and probability wake-up strategy. During the process, the error between the predicted and real data is reduced. Thus, the mean absolute error of network nodes is reduced. Meanwhile, the mean absolute error of TSD is smaller than that of ADC under three different thresholds. TSD in probability wake-up strategy can timely adjust the distribution of the dominating node set. Therefore, the performance of spatial correlation based on temporal correlation is better than that of temporal correlation only. These methods are within the error threshold, indicating that they ensure the quality of the collected data. The increase of the mean absolute error of TSD is less than that of the three other methods with the growth of threshold, suggesting that TSD has better stability than other methods.

4.3.4 Influence of different skewness

Figure 9 illustrates the different network energy consumption levels of ADC and TSD in different IoT with network skewness ranging from 10% to 80% when the error threshold is 0.1. As the skewness increases, energy consumption increases, although its value may be quite small. Then, the energy consumption decreases as the skewness increases and then increases again. During the process of the network skewness changing from 10% to 80%, the network energy consumption of TSD has been far less than that of ADC, which suggests that the layout of the network node will have an impact on the network data collection method but is small. Under the condition of different network skewness, the network energy performance of TSD is better than that of ADC.

Figure 10 shows the different mean absolute errors of ADC and TSD in different IoT with network skewness ranging from 10% to 80% when the error threshold is 0.1. When the network skewness changes from 10% to 80%, the mean absolute errors of TSD and ADC have the same trend as the deviation degree increases, which first increases and then decreases. As the network skewness increases, the correlation among nodes in sparse areas becomes

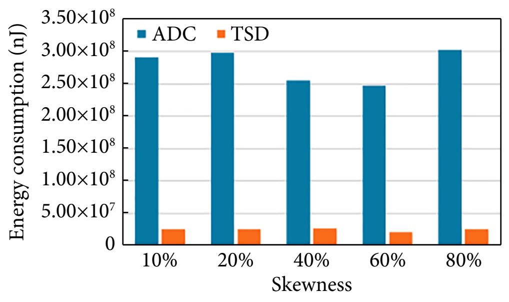

Fig. 9 Comparison of energy consumption under different network skewness.

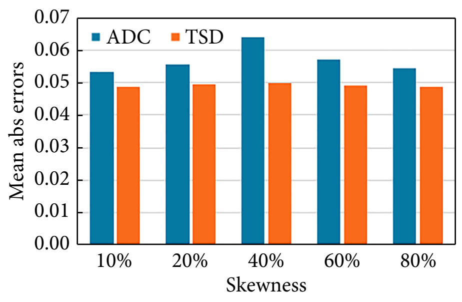

Fig. 10 Comparison of average absolute errors under different network skewness.

increasingly small, which leads to the increasing error of prediction data. When skewness increases to a certain degree, in the TSD method, the number of nodes in sparse areas becomes increasingly less, and the probability of node wake-up in this area becomes increasingly large, which leads to the error of prediction data gradually decreasing. However, in the ADC method, the number of nodes in sparse areas and that of nodes with large errors become increasingly less, which leads to the gradual reduction of the error of prediction data.

5 Conclusion

Due to the limitations of energy, computation, and storage capacity of nodes in IoT, we propose a spatiotemporal data correlation-based data collection method to ensure effective data and reduce network energy consumption. First, the data prediction model DBP is improved and optimized. The time interval of sensing data can be increased during the data collection to further reduce energy consumption. Second, we define the correlation and priority among nodes and optimize the greedy algorithm to reduce the number of redundant nodes and network energy consumption. The experimental results reveal that our method can reduce energy consumption in the network and achieve the purpose of saving energy while ensuring the validity of data.

Acknowledgment

This work was supported in part by the Open Project Program of Tianjin Key Laboratory of Aerospace Intelligent Equipment Technology, Tianjin Institute of Aerospace Mechanical and Electrical Equipment (NO. ZNZB-2021-01) and in part by the National Natural Science Foundation of China (NO. 61 702 232).

Dates

Received: 3 January 2022; Revised: 5 March 2022; Accepted: 5 March 2022

References

[1] T. D. T. Nguyen, V. Nguyen, V. N. Pham, L. N. T. Huynh, M. D. Hossain, and E. N. Huh, Modeling data redundancy and cost-aware task allocation in MEC-enabled internet-of-vehicles applications, *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1687–1701, 2021.
[2] H. Vasudev, V. Deshpande, D. Das, and S. K. Das, A lightweight mutual authentication protocol for V2V communication in internet of vehicles, *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6709–6717, 2020.
[3] F. Cirillo, D. Gómez, L. Diez, I. Elicegui Maestro, T. B. J. Gilbert, and R. Akhavan, Smart city IoT services creation through large-scale collaboration, *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5267–5275,

2020.

[4] A. Sivanathan, D. Sherratt, H. H. Gharakheili, V. Sivaraman, and A. Vishwanath, Low-cost flow-based security solutions for smart-home IoT devices, in *Proc. 2016 IEEE Int. Conf. Advanced Networks and Telecommunications Systems*, Bangalore, India, 2016, pp. 1–6.

[5] M. B. Attia, K. K. Nguyen, and M. Cheriet, Dynamic QoS-aware scheduling for concurrent traffic in smart home, *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5412–5425, 2020.

[6] C. Alippi, G. Anastasi, M. Di Francesco, and M. Roveri, An adaptive sampling algorithm for effective energy management in wireless sensor networks with energy-hungry sensors, *IEEE Trans. Instrum. Meas.*, vol. 59, no. 2, pp. 335–344, 2010.

[7] H. Ko and S. Pack, Neighbor-aware energy-efficient monitoring system for energy harvesting internet of things, *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5745–5752, 2019.

[8] L. Wang, D. Zhang, Y. Wang, C. Chen, X. Han, and A. M'hamed, Sparse mobile crowdsensing: Challenges and opportunities, *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 161–167, 2016.

[9] S. Yoon and C. Shahabi, The clustered AGgregation (CAG) technique leveraging spatial and temporal correlations in wireless sensor networks, *ACM Trans. Sens. Netw.*, vol. 3, no. 1, p. 3, 2007.

[10] L. A. Villas, A. Boukerche, H. A. B. F. de Oliveira, R. B. de Araujo, and A. A. F. Loureiro, A spatial correlation aware algorithm to perform efficient data collection in wireless sensor networks, *Ad Hoc Netw.*, vol. 12, pp. 69–85, 2014.

[11] M. M. Afsar and M. H. Tayarani-N, Clustering in sensor networks: A literature survey, *J. Netw. Comput. Appl.*, vol. 46, pp. 198–226, 2014.

[12] N. Choudhury, R. Matam, M. Mukherjee, J. Lloret, and E. Kalaimannan, NCHR: A nonthreshold-based cluster-head rotation scheme for IEEE 802.15.4 cluster-tree networks, *IEEE Internet Things J.*, vol. 8, no. 1, pp. 168–178, 2021.

[13] G. Anastasi, M. Conti, and M. Di Francesco, Extending the lifetime of wireless sensor networks through adaptive sleep, *IEEE Trans. Industr. Inform.*, vol. 5, no. 3, pp. 351–365, 2009.

[14] Y. Zeng, R. Zhang, and T. J. Lim, Throughput maximization for UAV-enabled mobile relaying systems, *IEEE Trans. Commun.*, vol. 64, no. 12, pp. 4983–4996, 2016.

[15] W. Wang and M. Zhang, Tensor deep learning model for heterogeneous data fusion in internet of things, *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 4, no. 1, pp. 32–41, 2020.

[16] H. Cheng, Z. Xie, Y. Shi, and N. Xiong, Multi-step data prediction in wireless sensor networks based on one-dimensional CNN and bidirectional LSTM, *IEEE Access*, vol. 7, pp. 117883–117896, 2019.

[17] C. Hu, Y. Pu, F. Yang, R. Zhao, A. Alrawais, and T. Xiang, Secure and efficient data collection and storage of IoT in smart ocean, *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9980–9994, 2020.

[18] D. Mechta and S. Harous, HC-LEACH: Huffman coding-based energy-efficient LEACH protocol for WSN, in *Proc. 11th IEEE Ann. Ubiquitous Computing, Electronics & Mobile Communication Conf.*, New York, NY, USA, 2020, pp. 932–938.

[19] H. M. Al-Kadhim and H. S. Al-Raweshidy, Energy efficient data compression in cloud based IoT, *IEEE Sensors J.*, vol. 21, no. 10, pp. 12212–12219, 2021.

[20] A. K. M. Al-Qurabat, C. Abou Jaoude, and A. K. Idrees, Two tier data reduction technique for reducing data transmission in IoT sensors, in *Proc. 15th Int. Wireless Communications & Mobile Computing Conf.*, Tangier, Morocco, 2019, pp. 168–173.

[21] W. Chen, F. Cui, and K. K. L. Wong, Data collection mechanism based on wavelet multi-resolution for opportunistic social networks, *IEEE Access*, vol. 9, pp. 21357–21366, 2021.

[22] A. Burrello, A. Marchioni, D. Brunelli, S. Benatti, M. Mangia, and L. Benini, Embedded streaming principal components analysis for network load reduction in structural health monitoring, *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4433–4447, 2021.

[23] H. Zhang, X. Zhang, and D. K. Sung, Lightweight self-adapting linear prediction algorithms for wireless sensor networks, *IEEE Sens. J.*, vol. 15, no. 5, pp. 3050–3058, 2015.

[24] D. Tulone and S. Madden, PAQ: Time series forecasting for approximate query answering in sensor networks, in *Proc. 3rd European Workshop Wireless Sensor Networks*, Zurich, Switzerland, 2006, pp. 21–37.

[25] Z. Han, J. Zhao, H. Leung, K. Ma, and W. Wang, A review of deep learning models for time series prediction, *IEEE Sens. J.*, vol. 21, no. 6, pp. 7833–7848, 2021.

[26] C. Xu, K. Wang, Y. Sun, S. Guo, and A. Y. Zomaya, Redundancy avoidance for big data in data centers: A conventional neural network approach, *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 1, pp. 104–114, 2020.

[27] L. A. Villas, A. Boukerche, H. A. B. F. de Oliveira, R. B. de Araujo, and A. A. F. Loureiro, A spatial correlation aware algorithm to perform efficient data collection in wireless sensor networks, *Ad Hoc Netw.*, vol. 12, pp. 69–85, 2014.

[28] C. Wang, H. Ma, Y. He, and S. Xiong, Adaptive approximate data collection for wireless sensor networks, *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 6, pp. 1004–1016, 2012.

[29] Z. Guo, J. Peng, W. Xu, W. Liang, W. Wu, Z. Xu, B. Guo, and Y. Ivan Wu, Minimizing redundant sensing data transmissions in energy-harvesting sensor networks via exploring spatial data correlations, *IEEE Internet Things J.*, vol. 8, no. 1, pp. 512–527, 2021.

[30] A. Jain, E. Y. Chang, and Y. F. Wang, Adaptive stream resource management using Kalman Filters, in *Proc. 2004 ACM SIGMOD Int. Conf. Management of Data*, Paris, France, 2006, pp. 11–22.