Zhu *et al. EURASIP Journal on Wireless Communications and Networking* (2019) 2019:132
https://doi.org/10.1186/s13638-019-1467-4

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Retrieving the relative kernel dataset from big sensory data for continuous queries in IoT systems

Tongxin Zhu[1], Jinbao Wang[1*], Siyao Cheng[1], Yingshu Li[2] and Jianzhong Li[1]

Abstract

Internet of Things (IoT) is rapidly developed and widely deployed in recent years, which makes the sensory data generated by IoT systems explode. The huge amount of sensory data generated by some IoT systems has already exceeded the storage, transmission, and computation capacities of IoT systems. However, the valuable sensory data which is highly related to a query in an IoT system is relatively small. The sensory data which is highly related to a query Q forms the relative kernel dataset of Q. Therefore, retrieving sensory data in the relative kernel dataset of a query instead of the raw sensory data will reduce the heavy burdens of an IoT system in terms of transmission and computation and then reduce the energy consumption of the IoT system. In this paper, we investigate the problem of retrieving relative kernel dataset from big sensory data for continuous queries in IoT systems. Two algorithms, relative kernel dataset retrieving algorithm and piecewise linear fitting-based relative kernel dataset retrieving algorithm, are proposed to retrieve the relative kernel dataset for continuous queries. Beside, algorithms for estimating the answers of continuous queries based on their relative kernel datasets are also proposed. Extensive simulation results are provided to verify the effectiveness and energy efficiency of the proposed algorithms.

Keywords: Internet of things, Big sensory data, Relative kernel dataset

1 Introduction

The Internet of Things (IoT) system, which refers to the network with a variety of intelligent things or objects around us, provides an efficient way to observe the complicated physical world and brings convenience to our life. With the repaid development of wireless telecommunications, embedded systems, and sensing techniques, the IoT systems are widely developed and generate mass sensory data. Gartner forecasts that there will be 20.8 billion connected things by 2020 and 100 billion by 2030 [1]. Such a large scale of IoT systems will generate mass sensory data. According to [2], the global climate data will reach to 100 petabytes in 2020. Other types of sensory data, such as monitoring data from a large-scale system (e.g., Large Hadron Collider) and GPS data, also increase rapidly and have exceeded terabytes or even petabytes currently.

*Correspondence: wangjinbao@hit.edu.cn
[1]School of Computer Science and Tech, Harbin Institute of Technology, Harbin, China
Full list of author information is available at the end of the article

The mass sensory data generated by IoT systems is called as big sensory data, which has four V characters.

- *Volume*. The volume of big sensory data is extremely huge that has already exceeded the storage, transmission and computation capacities of IoT systems.
- *Variety*. An IoT system always contains a wide variety of sensors to satisfy the complex applications of the IoT system.
- *Velocity*. To improve the accuracy of applications in an IoT system, the sampling frequency of each sensor is high. Therefore, the generating velocity of big sensory data is quite fast.
- *Value*. The value of big sensory data always concentrates on a relative small subset of the big sensory data on account of the existence of noises and redundancies in big sensory data.

The *Volume*, *Variety*, and *Velocity* characters of big sensory data make the data processing in IoT systems

more challenging. The mass sensory data may exceed the storage, transmission and computation capacities of IoT systems. Besides, the energy consumption for transmitting and computing big sensory data is quite high. Therefore, the existing sensory data acquisition, routing [3–6], data collection [7–9], and data computation [10, 11] techniques are no longer applicable for the big sensory data. The authors in [3] design cluster-based routing protocol for wireless sensor networks with nonuniform distribution of sensors. The works of [4–6] focus on constructing connected dominating sets for wireless sensor networks. However, the transmission of mass sensory data will exceed the communication capacity of wireless sensor networks and affect the effectiveness of these methods accordingly. The authors in [7] study the approximated holistic aggregation algorithms based on uniform sampling. And the authors in [8] propose aggregation scheduling algorithms for energy harvesting sensor networks. However, the mass sensory data will increase the aggregation latency and affect the performances of these algorithms. The work of [9] proposes algorithm to track quantiles and range countings in wireless sensor networks. The authors in [10] propose algorithm to support curve query processing in wireless sensor networks. The work in [11] studies the application-aware scheduling in wireless networks. When the amount of sensory data is huge, the computation complexity of these algorithms may exceed the computation capability of wireless sensor networks. Therefore, a series of new in-network processing algorithms with much lighter transmission and computation overloads should be considered.

The *Value* character of big sensory data reveals that the dataset of sensory data that is highly related to a given query is usually small. Therefore, for a query, only retrieving the dataset of sensory data that is highly related to the query will greatly reduce the transmission and computation overhead of an IoT system and then reduce the energy consumption of the IoT system.

There already exists some data reduction algorithms which reduce the amount of sensory data transmitted and computed in an IoT system and then reduce the energy consumption of the IoT system. Firstly, the simplest data reduction technique is based on sampling [12–16]. However, the sampling technique is only applicable to some simple statistic queries. Some valuable data in other types of queries may be missed by sampling technique on account of the limitation of sampling frequency. Besides, the sampled data may not be necessary for all queries. Secondly, another classical technique for in-network data reduction is compressed sensing [17–22]. However, compressed sensing only considers the temporal and spatial correlation between sensory data while neglecting the correlation between the queries conducted by users and the sensory data generated by different sensors. As a consequence, the data reduction for a given query is not enough in compressed sensing. Thirdly, the work proposed by Cheng et. al. is the first one to investigate data processing problem in big sensory data [23, 24]. The work can reduce sensory data by drawing dominant dataset from Big Sensory Data. Nevertheless, the dominant dataset defined in [23, 24] is irrelevant to any given query. As with the sampling technique and compressed sensing technique, the sensory data in the dominant dataset defined in [23, 24] may not be necessary for all queries.

All existing algorithms above are not energy efficient enough since they ignore the correlation between a given query and the sensory data. As a consequence, we concentrate on retrieving the relative kernel dataset for continuous queries from big sensory data in IoT systems in this paper. The relative kernel dataset of a query is the sensory data that is highly related to the query. The amount of sensory data in the relative kernel dataset of a query is usually small according to the *Value* character of big sensory data. Therefore, only transmitting and computing these sensory data will highly reduce the energy consumption of an IoT system. Besides, many applications in the IoT systems provide users to have access to temporal variation information of the monitored environment. That is, users can conduct continuous queries in IoT systems to monitor the physical world in real time, where a continuous query is a query which is conducted by a user continuously and frequently. Since continuous queries are frequent in an IoT system, reducing the energy consumption of continuous queries will highly reduce the energy consumption of an IoT system.

On account of the above reasons, we study the problem of retrieving the relative kernel dataset from big sensory data for continuous queries in IoT systems in this paper. Two algorithms, the relative kernel dataset retrieving (RKDR) algorithm and the piecewise linear fitting-based relative kernel dataset retrieving (PLF-RKDR) algorithm, are proposed firstly to retrieve the relative kernel dataset for continuous queries. Besides, two algorithms, the piecewise linear fitting-based answer estimating (PLF-AE) algorithm and temporal correlation-based answer estimating (TC-AE) algorithm, are proposed to estimate the answers of continuous queries based on their relative kernel datasets. The major contributions of this paper are as follows.

1. The definition of the relative kernel dataset of a continuous query is proposed firstly. Besides, the relative kernel dataset retrieving problem is formally defined in this paper.
2. Two algorithms, the RKDR algorithm and the PLF-RKDR algorithm, are proposed to retrieve the

relative kernel dataset from big sensory data for continuous queries in IoT systems.
3. Two algorithms, the PLF-AE algorithm and the TC-AE algorithm, are proposed to estimate the answers of continuous queries in an IoT system based on their relative kernel datasets.
4. Extensive simulations on both simulation dataset and real dataset are carried out to verify the accuracy of the proposed algorithms.

The rest of this paper is organized as follows. Section 2 summarizes the methods of this paper. Section 3 presents the problem definition. Section 4 elaborates the algorithms for the relative kernel dataset retrieving problem. Section 5 provides the algorithms for estimating the answers of continuous queries with their relative kernel datasets. Section 6 analyzes the performances of the proposed algorithms. Section 7 shows the simulation results and discussion. Section 8 discusses the related work. Finally, Section 9 concludes the whole paper.

2 Methods

In this paper, we study the problem of retrieving the relative kernel dataset from big sensory data for continuous queries in IoT systems. The Pearson correlation coefficient is applied to measure the linear correlation between sensory data and the continuous queries. Besides, the piecewise linear fitting method is applied to approximate the relationships between sensory data and the continuous queries. Two algorithms are proposed to retrieve the relative kernel dataset for continuous queries in IoT systems. Another two algorithms are proposed to estimate the answers of continuous queries with their relative kernel datasets. The absolute error and relative error are defined and evaluated on both simulation dataset and real dataset. The simulation results show that the proposed algorithms are effective and efficient.

3 Problem definition

3.1 The IoT system model

An IoT system is equipped with n categories of sensors, which can generate n types of sensory data from the monitored environment. Each type of sensory data is described as an attribute of the monitored environment. The attribute set of the monitored environment is denoted as $\mathcal{A} = \{x_1, x_2, \cdots, x_n\}$. For each continuous query Q from users at time slot t in an IoT system, the IoT system returns an answer to users according to the sensory data collected at time slot t. The answer of query Q is described as the target value of query Q, denoted as y_q. Apparently, the target value y_q is correlated with partial or all n attributes. Taking the healthcare monitoring IoT system as an example, there are a variety of sensors collecting data of heart sounds, electrical heart signals, blood oxygen saturation, respiratory sounds, blood pressure, body temperature, etc., which are attributes of this IoT system. When the user's query is the possibility of cardiac arrest, the IoT system returns a probability. In this application, the query Q is the possibility of cardiac arrest and the target value y_q is the returned probability.

3.2 The correlation model

To explore the correlations between attributes in $\mathcal{A}$ and query Q, a training set with m training examples is applied. Each training example is denoted as a column vector $S_l = [x_{1l}, \cdots, x_{nl}, y_{ql}]^T$, where T denotes the matrix transpose and x_{il} $(1 \le i \le n)$ is the value of attribute x_i and y_{ql} is the target value of query Q in training example S_l, $1 \le l \le m$. The training set can be denoted as a $(n+1) \times m$ matrix $\mathcal{S} = [S_1, S_2, \cdots, S_m]$.

In most applications of IoT systems, the sensory data can reflect the statement of the monitored physical world intuitively. That means the simple linear correlation can reveal the relationship between the sensory data and the query of an IoT system in most cases. The Pearson correlation coefficient is a common metric to measure the linear correlation between two variables in correlation analysis. The value of the Pearson correlation coefficient is in $[-1, 1]$, where -1 presents the total negative linear correlation, 0 presents no linear correlation, and 1 is the total positive linear correlation. Therefore, we apply the Pearson correlation coefficient to measure the correlation between attributes and the correlation between attributes and continuous queries.

Firstly, the Pearson correlation coefficient of any two attributes x_i and x_j $(1 \le i \le n, 1 \le j \le n$ and $i \ne j)$ is presented by the following formula,

$$r_{ij} = \frac{\sum_{l=1}^{m}(x_{il} - \overline{x_i})(x_{jl} - \overline{x_j})}{\sqrt{\sum_{l=1}^{m}(x_{il} - \overline{x_i})^2}\sqrt{\sum_{l=1}^{m}(x_{jl} - \overline{x_j})^2}} \tag{1}$$

where x_{il} is the value of attribute x_i and x_{jl} is the value of attribute x_j in training example S_l. Besides, $\overline{x_i} = \frac{1}{m}\sum_{l=1}^{m} x_{il}$ is the average value of attribute x_i and $\overline{x_j} = \frac{1}{m}\sum_{l=1}^{m} x_{jl}$ is the average value of attribute x_j in the training set $\mathcal{S}$.

Secondly, the Pearson correlation coefficient of attribute x_i $(1 \le i \le n)$ and the target value y_q of a continuous query Q is presented by the following formula,

$$R_i^q = \frac{\sum_{l=1}^{m}(x_{il} - \overline{x_i})(y_{ql} - \overline{y_q})}{\sqrt{\sum_{l=1}^{m}(x_{il} - \overline{x_i})^2}\sqrt{\sum_{l=1}^{m}(y_{ql} - \overline{y_q})^2}} \tag{2}$$

where x_{il} denotes the value of attribute x_i and y_{ql} denotes the target value of query Q in training example S_l. Besides, $\overline{y_q} = \frac{1}{m}\sum_{l=1}^{m} y_{ql}$ is the average value of target values of query Q in the training set $\mathcal{S}$.

3.3 Problem statement

In this paper, we study the problem of retrieving the relative kernel dataset for continuous queries in IoT systems to reduce the transmission and computation overhead and energy consumption of the IoT systems. The formal definition of the relative kernel dataset $\mathcal{K}^q$ for a continuous query Q is described as follows.

Definition 1 *(β-compatible) Given a parameter β, where $0 < \beta < 1$, any two attributes x_i and x_j are β-compatible if their Pearson correlation coefficient r_{ij} satisfies that $|r_{ij}| \leq \beta$, where r_{ij} is calculated by formula (1).*

Definition 2 *(β-compatible set) Given a parameter β, where $0 < \beta < 1$, the subset $\mathcal{A}_S$ of attribute set $\mathcal{A}$ is a β-compatible set if any two attributes in $\mathcal{A}_S$ are β-compatible with each other, i.e., $\mathcal{A}_S \subseteq \mathcal{A}$ and $\forall x_i, x_j \in \mathcal{A}_S$, $|r_{ij}| \leq \beta$.*

Definition 3 *(the weight of a β-compatible set) Given a β-compatible set $\mathcal{A}_S$ and a continuous query Q, the weight of $\mathcal{A}_S$ is denoted as $w(\mathcal{A}_S)$. The definition of $w(\mathcal{A}_S)$ is $w(\mathcal{A}_S) = \sum_{x_i \in \mathcal{A}_S} |R_i^q|$, where R_i^q is the correlation coefficient between attribute x_i and query Q.*

Definition 4 *((k,β)-relative kernel dataset $\mathcal{K}^q$) Given the attribute set $\mathcal{A}$, query Q, the compatible parameter β, and the required size k of the relative kernel dataset, a training set $\mathcal{S}$. The (k,β)-relative kernel dataset of query Q is a subset of $\mathcal{A}$, denoted as $\mathcal{K}^q$, satisfying the following three conditions.*

(1) $\mathcal{K}^q$ is a β-compatible subset,

(2) the size of $\mathcal{K}^q$ is at most k, and

(3) $w(\mathcal{K}^q) > w(\mathcal{A}_S)$ for any attribute subset $\mathcal{A}_S$ satisfying conditions (1)(2).

Based on the definition of (k,β)-relative kernel dataset $\mathcal{K}^q$, we define the problem of retrieving the relative kernel dataset for continuous queries in IoT systems as follows.

Problem statement The problem of retrieving the relative kernel dataset for continuous queries in IoT systems is denoted as *r*elative *k*ernel *d*ataset *r*etrieving (RKDR) problem. The input and output of the RKDR problem are presented as follows.

Input:

1. The attribute set of an IoT system, $\mathcal{A} = \{x_1, x_2, \cdots, x_n\}$;
2. The continuous query, Q;
3. The training set, $\mathcal{S} = [S_1, S_2, \cdots, S_m]$;
4. The required size of the relative kernel dataset, k;
5. The compatible parameter, β.

Output:

The (k,β)-relative kernel dataset $\mathcal{K}^q$ of the continuous query Q;

Theorem 1 *The RKDR problem is NP-hard.*

Proof 1 *The Maximum Weighted Budgeted Independent Set (MWBIS) problem is to find a maximum weighted independent set in a weighted graph $G(V,E)$ of cardinality at most k. The MWBIS problem has been proved to be NP-hard in [25]. The MWBIS problem can be converted to the RKDR problem in polynomial time. Each vertex i of the weighted graph $G(V,E)$ can be regarded as an attribute x_i of an IoT system, where the weight of vertex i can be regarded as the absolute value of the correlation coefficient between attribute x_i and query Q. For each edge $(i,j) \in E$ in the weighted graph $G(V,E)$, we regard that attributes x_i and x_j are not β-compatible. To find a maximum weighted independent set of cardinality at most k is to find the (k,β)-relative kernel dataset of query Q. Then the MWBIS problem is converted to the RKDR problem. Therefore, the RKDR problem is NP-hard.*

4 Algorithms for the relative kernel dataset retrieving problem

In this section, we propose two algorithms to retrieve the relative kernel dataset for continuous queries in IoT systems, which are the relative kernel dataset retrieving (RKDR) algorithm and the piecewise linear fitting-based relative kernel dataset retrieving (PLF-RKDR) algorithm. The detailed description of the RKDR algorithm is presented in Algorithm 1 and the detailed description of the PLF-RKDR algorithm is presented in Algorithm 3. The RKDR algorithm is suitable for the IoT systems where simple linear correlation can reveal the relationship between the sensory data and the continuous queries. And the PLF-RKDR algorithm is more suitable for the IoT systems where the correlation between an attribute and a continuous query is not merely linear correlation.

4.1 Relative kernel dataset retrieving algorithm

The relative kernel dataset retrieving (RKDR) algorithm firstly reduces redundancies in attributes by measuring the linear correlations between different attributes. Then, the Pearson correlation coefficient is applied to measure the correlations between attributes and the given continuous query. The RKDR algorithm contains the following two steps.

Step 1. Calculate the candidate relative kernel dataset through reducing the redundancies in attributes.

The candidate relative kernel dataset is denoted as $\mathcal{X}$. At first, the candidate relative kernel dataset is initialized as the attribute set, i.e., $\mathcal{X} = \mathcal{A} = \{x_1, \cdots, x_n\}$. Then, $\mathcal{X}$ is updated by the following two sub-steps.

Step 1.1. For each attribute x_i ($1 \le i \le n$) in $\mathcal{X}$, the Pearson correlation coefficient R_i^q between x_i and the target value y_q of query Q is calculated by formula (2).

Step 1.2 For any two attribute x_i and x_j in $\mathcal{X}$, the Pearson correlation coefficient r_{ij} of them is calculated by formula (1). If x_i and x_j are not β-compatible (i.e., $|r_{ij}| > \beta$), one of them is redundant in $\mathcal{X}$ and should be removed. If x_i has stronger linear correlation with query Q than x_j, i.e., $|R_i^q| > |R_j^q|$, remove x_j from $\mathcal{X}$. Otherwise, remove x_i from $\mathcal{X}$.

After the above two sub-steps, the candidate relative kernel dataset $\mathcal{X}$ is calculated through reducing redundancies in attributes.

Step 2. Retrieve the (k, β)-relative kernel dataset from the candidate relative kernel dataset.

Any two attributes in the candidate relative kernel dataset $\mathcal{X}$ are β-compatible according to Step 1.2. The Pearson correlation coefficient of each attribute in $\mathcal{X}$ and query Q is calculated in Step 1.1. The absolute values of these Pearson correlation coefficients are calculated. Therefore, we select the top-k absolute values of these Pearson correlation coefficients and their corresponding k attributes. The top-k absolute values of these Pearson correlation coefficients are denoted as $|R_{a_1}^q| \ge |R_{a_2}^q| \ge \cdots \ge |R_{a_k}^q|$ and the corresponding k attributes are $x_{a_1}, \cdots, x_{a_k}$. These k attributes are added to the (k, β)-relative kernel dataset of query Q, i.e., $\mathcal{K}^q = \{x_{a_1}, \cdots, x_{a_k}\}$.

After the above two steps, the (k, β)-relative kernel dataset for query Q is derived. The detailed operations are formally described in Algorithm 1. Algorithm 1 is proposed to retrieve the (k, β)-relative kernel dataset for continuous query Q in IoT systems by the linear correlation analysis.

Algorithm 1: Relative Kernel Dataset Retrieving Algorithm

Input: query Q, the attribute set $\mathcal{A}$, the training set $\mathcal{S} = [S_1, \cdots, S_m]$, the compatible parameter β, the required size k

Output: (k, β)-relative kernel dataset $\mathcal{K}^q$

$\mathcal{X} = \mathcal{A} = \{x_1, x_2, \cdots, x_n\}$;
for *each attribute* x_i *in* $\mathcal{X}$ **do**
  $R_i^q = \frac{\sum_{l=1}^{m}(x_{il}-\overline{x_i})(y_{ql}-\overline{y_q})}{\sqrt{\sum_{l=1}^{m}(x_{il}-\overline{x_i})^2}\sqrt{\sum_{l=1}^{m}(y_{ql}-\overline{y_q})^2}}$;
for *each pair of attributes* x_i *and* x_j *in* $\mathcal{X}$ **do**
  $r_{ij} = \frac{\sum_{l=1}^{m}(x_{il}-\overline{x_i})(x_{jl}-\overline{x_j})}{\sqrt{\sum_{l=1}^{m}(x_{il}-\overline{x_i})^2}\sqrt{\sum_{l=1}^{m}(x_{jl}-\overline{x_j})^2}}$;
  if $|r_{ij}| > \beta$ **then**
    if $|R_i^q| > |R_j^q|$ **then**
      remove x_j from $\mathcal{X}$;
    else
      Otherwise, remove x_i from $\mathcal{X}$;
Obtain the top-k correlations, denoted by $\{|R_{a_1}^q|, \cdots, |R_{a_k}^q|\}$;
$\mathcal{K}^q = \{x_{a_1}, \cdots, x_{a_k}\}$;
Return $\mathcal{K}^q$.

4.2 Piecewise linear fitting-based relative kernel dataset retrieving algorithm

In some scenarios, linear correlation is not enough to describe the relationship between an attribute and a continuous query. For example, the correlation between an attribute and a continuous query could be exponential, quadric, or logarithmic. In this situation, the RKDR algorithm proposed in the last subsection may be no longer applicable. As a consequence, the piecewise linear fitting-based relative kernel dataset retrieving (PLF-RKDR) algorithm is proposed to improve the RKDR algorithm in this subsection.

Since it is almost impossible to know the actual correlation function between an attribute and a continuous query, it is difficult to measure the correlation coefficient between them. The PLF-RKDR algorithm applies the piecewise linear fitting method to approximate the actual correlation function between an attribute and a continuous query. Since each segment of the piecewise linear function is a linear function, the PLF-RKDR algorithm combines the Pearson correlation coefficients of all segments in the piecewise linear function to measure the correlation of each attribute and the continuous query. The PLF-RKDR algorithm contains the following three steps.

Step 1. Calculate the Pearson correlation coefficient of each attribute $x_i \in \mathcal{A}$ and the continuous query Q by piecewise linear fitting.

We apply the training set $\mathcal{S}$ to calculate the piecewise linear function between each attribute $x_i \in \mathcal{A}$ and the target value of the continuous query Q.

Firstly, for each training example S_l ($1 \le l \le m$), we extract the tuple (x_{il}, y_{ql}), where x_{il} is the value of attribute x_i and y_{ql} is the target value in S_l. Then, we sort the extracted tuples by the non-descending order of the values of attribute x_i and denote the ordered tuples as $\{(x_i^1, y_q^1), \cdots, (x_i^m, y_q^m)\}$, where $x_i^1 \le \cdots \le x_i^m$.

Secondly, the optimal segment points of the sorted tuples $\{(x_i^1, y_q^1), \cdots, (x_i^m, y_q^m)\}$ are calculated recursively by the method proposed in [26]. For one segment, $\{(x_i^s, y_q^s), (x_i^{s+1}, y_q^{s+1}), \cdots, (x_i^e, y_q^e)\}$, the linear fitting function of it is derived by the least-squares method and is denoted as $F_i^{(s,e)}(x)$. The error sum of squares of linear fitting function $F_i^{(s,e)}(x)$ is $E(s, e)$. The optimal segment point (x_i^j, y_q^j) of $\{(x_i^s, y_q^s), (x_i^{s+1}, y_q^{s+1}), \cdots, (x_i^e, y_q^e)\}$ should satisfy that $j = \arg\min_{s<l<e}(E(s, l) + E(l, e))$. This process is conducted recursively until $E(s, e) - (E(s, j) + E(j, e)) \le \gamma$. The formal description is presented in Algorithm 2. Algorithm 2 returns the set

Algorithm 2: Least-Squares Based Piecewise Linear Fitting (LS-PLF) Algorithm

Input: Two endpoints s and e, $E(s,e)$, the threshold γ
Output: The set of segment points $\mathcal{P}$

if $e-s>1$ **then**
  $j=\arg\min_{s<l<e}(E(s,l)+E(l,e))$;
  if $E(s,e)-(E(s,j)+E(j,e))>\gamma$ **then**
    $\mathcal{P}=\mathcal{P}\cup\{j\}$;
    LS-PLF$(s,j,E(s,j),\gamma)$;
    LS-PLF$(j,e,E(j,e),\gamma)$;
Return $\mathcal{P}$.

of segment points $\mathcal{P}_i=\{s_1,s_2,\cdots,s_{l_i}\}$ for segment $\{(x_i^s,y_q^s),(x_i^{s+1},y_q^{s+1}),\cdots,(x_i^e,y_q^e)\}$.

Finally, the sorted tuples $\{(x_i^1,y_q^1),\cdots,(x_i^m,y_q^m)\}$ are divided into l_i-1 subsets according to the calculated segment point set $\mathcal{P}_i$. Each subset is denoted as $D_j=\{(x_i^{s_j},y_q^{s_j}),\cdots,(x_i^{s_{j+1}},y_q^{s_{j+1}})\}$, where $s_j,s_{j+1}\in\mathcal{P}_i$, and $1\le j<l_i$. The Pearson correlation coefficient R_{ij}^q for subset D_j is calculated by the following formula,

$$R_{ij}^q=\frac{\sum_{l=s_j}^{s_{j+1}}(x_i^l-\overline{x_i}^j)(y_q^l-\overline{y_q}^j)}{\sqrt{\sum_{l=s_j}^{s_{j+1}}(x_i^l-\overline{x_i}^j)^2}\sqrt{\sum_{l=s_j}^{s_{j+1}}(y_q^l-\overline{y_q}^j)^2}} \tag{3}$$

where $\overline{x_i}^j=\frac{1}{s_{j+1}-s_j+1}\sum_{l=s_j}^{s_{j+1}}x_i^l$ and $\overline{y_q}^j=\frac{1}{s_{j+1}-s_j+1}\sum_{l=s_j}^{s_{j+1}}y_q^l$. As a consequence, the correlation coefficient of attribute x_i and the continuous query Q is the weighted average of the absolute values of all calculated Pearson correlation coefficients $\{R_{ij}^q|1\le j<l_i\}$, which is presented as the following formula.

$$R_i^q=\sum_{j=1}^{l_i-1}|R_{ij}^q|\times\frac{s_{j+1}-s_j}{x_i^m-x_i^1} \tag{4}$$

Step 2. Calculate the candidate relative kernel dataset $\mathcal{X}$ through reducing the redundancies in attributes.

Initially, all attributes are added to $\mathcal{X}$, i.e., $\mathcal{X}=\mathcal{A}=\{x_1,\cdots,x_n\}$. Then, for each two attribute $x_i,x_j\in\mathcal{X}$, we compute the Pearson correlation coefficient r_{ij} of them by formula (1). When $|r_{ij}|>\beta$, meaning that x_i and x_j are not β-compatible, x_i and x_j are redundant in $\mathcal{X}$ and one of them should be removed. $|R_i^q|$ and $|R_j^q|$ are compared, where R_i^q and R_j^q are calculated by formula (4). If $|R_i^q|>|R_j^q|$, remove x_j from $\mathcal{X}$. Otherwise, x_i is removed from $\mathcal{X}$.

Step 3. Retrieve the relative kernel dataset from the candidate relative kernel dataset for the continuous query Q.

After Step 2, all attributes in candidate relative kernel dataset $\mathcal{X}$ are β-compatible. Then, we can obtain the top-k correlation coefficients of all attributes in $\mathcal{X}$ calculated in Step 1 and the corresponding k attributes. The top-k correlation coefficients are denoted as $R_{a_1}^q\ge R_{a_2}^q\ge\cdots\ge R_{a_k}^q$, and the corresponding k attributes are denoted as $x_{a_1},x_{a_2},\cdots,x_{a_k}$. Finally, the (k,β)-relative kernel dataset of query Q is $\mathcal{K}^q=\{x_{a_1},x_{a_2},\cdots,x_{a_k}\}$.

After the above three steps, the (k,β)-relative kernel dataset of query Q is calculated. The detailed operations are formally described in Algorithm 3. Algorithm 3 is proposed to retrieve the (k,β)-relative kernel dataset of continuous query Q in IoT systems by the piecewise linear fitting method. And Algorithm 2 is called by Algorithm 3 to return the set of segment points for piecewise linear fitting.

Algorithm 3: PLF-Relative Kernel Dataset Retrieving Algorithm

Input: query Q, the attribute set $\mathcal{A}$, the training set $\mathcal{S}=[S_1,\cdots,S_m]$, the compatible parameter β, the required size k, threshold γ
Output: (k,β)-relative kernel dataset $\mathcal{K}^q$

$\mathcal{X}=\mathcal{A}=\{x_1,x_2,\cdots,x_n\}$;
for *each attribute* x_i *in* $\mathcal{X}$ **do**
  Extract the tuples (x_{il},y_{ql}) from training example S_l $(1\le l\le m)$;
  Sort the extracted tuples by non-descending order of the values of attribute x_i, i.e., $\{(x_i^1,y_q^1),\cdots,(x_i^m,y_q^m)\}$, where $x_i^1\le\cdots\le x_i^m$;
  $\mathcal{P}_i=\{1,m\}\cup$**LS-PLF**$(1,m,\infty,\gamma)$;
  $\{(x_i^1,y_q^1),\cdots,(x_i^m,y_q^m)\}$ are divided into l_i-1 subsets $D_1,\cdots,D_{l_i-1}$;
  for *each subset* $D_j=\{(x_i^{s_j},y_q^{s_j}),\cdots,(x_i^{s_{j+1}},y_q^{s_{j+1}})\}$ **do**
    Calculate the Pearson correlation coefficient R_{ij}^q by formula (3);
  $R_i^q=\sum_{j=1}^{l_i-1}|R_{ij}^q|\times\frac{s_{j+1}-s_j}{x_i^m-x_i^1}$;
for *each pair of attributes* x_i *and* x_j *in* $\mathcal{X}$ **do**
  $r_{ij}=\frac{\sum_{l=1}^m(x_{il}-\overline{x_i})(x_{jl}-\overline{x_j})}{\sqrt{\sum_{l=1}^m(x_{il}-\overline{x_i})^2}\sqrt{\sum_{l=1}^m(x_{jl}-\overline{x_j})^2}}$;
  if $|r_{ij}|>\beta$ **then**
    if $|R_i^q|>|R_j^q|$ **then**
      remove x_j from $\mathcal{X}$;
    else
      Otherwise, remove x_i from $\mathcal{X}$;
Obtain the top-k correlations, denoted by $\{|R_{a_1}^q|,\cdots,|R_{a_k}^q|\}$;
$\mathcal{K}^q=\{x_{a_1},\cdots,x_{a_k}\}$;
Return $\mathcal{K}^q$.

5 Algorithms for estimating the answer of a continuous query with its relative kernel dataset

Once a continuous query Q is issued by users in an IoT system, sensory data of attributes in (k,β)-relative kernel dataset of query Q are transmitted to the sink node of the IoT system. After the (k,β)-relative kernel dataset of Q is retrieved by the RKDR algorithm (Algorithm 1) or the PLF-RKDR algorithm (Algorithm 3), the

IoT system should return an answer of Q to the users. In this section, we propose two algorithms to estimate the answers of continuous queries with their relative kernel datasets, which are the piecewise linear fitting-based answer estimating (PLF-AE) algorithm and the temporal correlation-based answer estimating (TC-AE) algorithm. The detailed descriptions of the PLF-AE algorithm and the TC-AE algorithm are presented in Algorithm 4 and Algorithm 5, respectively. The PLF-AE algorithm is proposed to estimate the answers for continuous queries with their (k, β)-relative kernel datasets by piecewise linear fitting method. And the TC-AE algorithm is proposed to estimate the answers for continuous queries with less energy consumption by applying the temporal correlations of sensor data.

5.1 Piecewise linear fitting-based answer estimating algorithm

The piecewise linear fitting-based answer estimating (PLF-AE) algorithm applies the piecewise linear fitting method to estimate the answer of a continuous query with sensory data of attributes in its (k, β)-relative kernel dataset. Once a continuous query Q is issued by users in an IoT system at time slot t, only sensory data of attributes in the (k, β)-relative kernel dataset $\mathcal{K}^q$ of Q are transmitted to the sink node. For each attribute, the sensory data generated at time slot t is denoted as $x_i(t)$. That is, the sink node collects sensory data in $\{x_i(t)|x_i \in \mathcal{K}^q\}$ to estimate the answer of Q. Firstly, the piecewise linear function of each attribute in $\mathcal{K}^q$ and the target value of query Q is calculated by the piecewise linear fitting method. Then, each piecewise linear function is assigned a weight. Finally, the target value of Q at time slot t is estimated by the sensory data of attributes in $\mathcal{K}^q$ according to the weighted piecewise linear functions. The PLF-AE algorithm consists of the following three steps.

Step 1. Calculate the piecewise linear functions of query Q and attributes in $\mathcal{K}^q$.

For each attribute $x_i \in \mathcal{K}^q$, the piecewise linear function of x_i and query Q is denoted as f_i. The process of calculating f_i consists of the following three sub-steps.

Step 1.1. Preprocess the training set $\mathcal{S} = \{S_1, \cdots, S_m\}$.

For each training example S_l $(1 \le l \le m)$, the value of attribute x_i and the target value y_{ql} are extracted and denoted as a tuple (x_{il}, y_{ql}). All extracted tuples are sorted by the non-descending order of the values of attribute x_i. The sorted tuples are denoted as $\{(x_i^1, y_q^1), \cdots, (x_i^m, y_q^m)\}$, where $x_i^1 \le \cdots \le x_i^m$.

Step 1.2. Calculate the optimal segment points of piecewise linear function f_i.

The optimal segment points of f_i is calculated by Algorithm 2. The returned segment point set is denoted as $\mathcal{P}_i = \{s_1, s_2, \cdots, s_{l_i}\}$. As a consequence, the sorted tuples $\{(x_i^1, y_q^1), \cdots, (x_i^m, y_q^m)\}$ are divided into $l_i - 1$ subsets, where each subset D_j $(1 \le j < l_i)$ is denoted as $D_j = \{(x_i^{s_j}, y_q^{s_j}), \cdots, (x_i^{s_{j+1}}, y_q^{s_{j+1}})\}$.

Step 1.3. Calculate the linear functions for all subsets.

For each subset $D_j = \{(x_i^{s_j}, y_q^{s_j}), \cdots, (x_i^{s_{j+1}}, y_q^{s_{j+1}})\}$, the linear function is calculated by the least-squares method and is denoted as $f_{i,j}$. The Pearson correlation coefficient R_{ij}^q for subset D_j is calculated by formula (3). As a consequence, the piecewise linear function f_i of attribute x_i and the continuous query Q is as follows.

$$f_i = \begin{cases} f_{i,1}, & x_i \le x_i^{s_2} \\ f_{i,2}, & x_i^{s_2} < x_i \le x_i^{s_3} \\ \cdots, & \cdots \\ f_{i,l_i-1}, & x_i > x_i^{s_{l_i-1}} \end{cases} \tag{5}$$

Step 2. Assign weights for the calculated piecewise linear functions.

For each attribute $x_i \in \mathcal{K}^q$, the generated sensory data at time slot t is $x_i(t)$. If $x_i(t)$ satisfies that $x_i^{s_j} < x_i(t) \le x_i^{s_{j+1}}$ $(s_j, s_{j+1} \in \mathcal{P}_i$ and $1 \le j < l_i)$, the piecewise linear function f_i is assigned a weight with $w_i = |R_{ij}^q|$, where R_{ij}^q is the Pearson correlation coefficient calculated in Step 1.

Step 3. Estimate the target value of query Q at time slot t.

Based on the sensory data generated at time slot t, i.e., $\{x_i(t)|x_i \in \mathcal{K}^q\}$, the estimated target value of Q at time slot t is calculated by the following formula.

$$\widehat{y_q}(t) = \sum_{x_i \in \mathcal{K}^q} \frac{w_i}{\sum_{x_i \in \mathcal{K}^q} w_i} f_i \tag{6}$$

The detailed operations for the PLF-AE algorithm is described in Algorithm 4. Algorithm 4 is proposed to estimate the answers of continuous queries with their (k, β)-relative kernel datasets by piecewise linear fitting method.

5.2 Temporal correlation-based answer estimating algorithm

It has been revealed by many research works that the monitored physical environment always varies continuously [27, 28]. That means the difference between the sensory data of an attribute in two continuous time slots may be small. Consequently, the temporal correlation-based answer estimating (TC-AE) algorithm utilizes the temporal correlation between sensory data of an attribute to estimate the answer of a continuous query with less energy consumption.

In the PLF-AE algorithm, each time a continuous query Q is issued by users, k sensory data of attributes in the (k, β)-relative kernel dataset of Q are transmitted and computed to estimate the answer of Q. The basic idea of TC-AE algorithm is to further reduce the sensory data transmitted and computed in the IoT system based on the temporal correlations of sensory data.

Algorithm 4: Piecewise Linear Fitting Based Answer Estimating Algorithm

Input: the training set $\mathcal{S} = [S_1, \cdots, S_m]$, (k, β)-relative kernel dataset $\mathcal{K}^q$ of query Q, the sensory data collected at time slot t, $\{x_i(t) | x_i \in \mathcal{K}^q\}$

. **Output**: the estimated answer $\widehat{y_q}(t)$

for *each attribute* x_i *in* $\mathcal{K}^q$ **do**

  Extract the tuples (x_{il}, y_{ql}) from training example S_l $(1 \le l \le m)$;

  Sort the extracted tuples by non-descending order of the values of attribute x_i, i.e., $\{(x_i^1, y_q^1), \cdots, (x_i^m, y_q^m)\}$, where $x_i^1 \le \cdots \le x_i^m$;

  $\mathcal{P}_i = \{1, m\} \cup$ **LS-PLF**$(1, m, \infty, \gamma)$;

  $\{(x_i^1, y_q^1), \cdots, (x_i^m, y_q^m)\}$ are divided into $l_i - 1$ subsets $D_1, \cdots, D_{l_i - 1}$;

  for *each subset* $D_j = \{(x_i^{s_j}, y_q^{s_j}), \cdots, (x_i^{s_{j+1}}, y_q^{s_{j+1}})\}$ **do**

    Calculate the Pearson correlation coefficient R_{ij}^q by formula (3);

    Calculate piecewise linear function $f_{i,j}$ by the least-squares method;

  The piecewise linear function f_i is calculated by formula (5);

  if $x_i^{s_j} < x_i(t) \le x_i^{s_{j+1}}$ **then**

    $w_i = |R_{ij}^q|$;

$\widehat{y_q}(t) = \sum_{x_i \in \mathcal{K}^q} \frac{w_i}{\sum_{x_i \in \mathcal{K}^q} w_i} f_i$;

Return $\widehat{y_q}(t)$.

Algorithm 5: Temporal Correlation Based Answer Estimating Algorithm

Input: the training set $\mathcal{S} = [S_1, \cdots, S_m]$, (k, β)-relative kernel dataset $\mathcal{K}^q$ of query Q, the stored sensory data $\{b_i | x_i \in \mathcal{A}\}$, the parameter α, the sensory data generated at time slot t, $\{x_i(t) | x_i \in \mathcal{K}^q\}$

. **Output**: the estimated answer $\widehat{y_q}(t)$

for *each attribute* x_i *in* $\mathcal{K}^q$ **do**

  if $|x_i(t) - b_i| \ge \alpha \times (q_i - p_i)$ **then**

    $b_i = x_i(t)$;

  Extract the tuples (x_{il}, y_{ql}) from training example S_l $(1 \le l \le m)$;

  Sort the extracted tuples by non-descending order of the values of attribute x_i, i.e., $\{(x_i^1, y_q^1), \cdots, (x_i^m, y_q^m)\}$, where $x_i^1 \le \cdots \le x_i^m$;

  $\mathcal{P}_i = \{1, m\} \cup$ **LS-PLF**$(1, m, \infty, \gamma)$;

  $\{(x_i^1, y_q^1), \cdots, (x_i^m, y_q^m)\}$ are divided into $l_i - 1$ subsets $D_1, \cdots, D_{l_i - 1}$;

  for *each subset* $D_j = \{(x_i^{s_j}, y_q^{s_j}), \cdots, (x_i^{s_{j+1}}, y_q^{s_{j+1}})\}$ **do**

    Calculate the Pearson correlation coefficient R_{ij}^q by formula (3);

    Calculate piecewise linear function $f_{i,j}$ by the least-squares method;

  The piecewise linear function f_i is calculated by formula (5);

  if $x_i^{s_j} < b_i \le x_i^{s_{j+1}}$ **then**

    $w_i = |R_{ij}^q|$;

$\widehat{y_q}(t) = \sum_{x_i \in \mathcal{K}^q} \frac{w_i}{\sum_{x_i \in \mathcal{K}^q} w_i} f_i$;

Return $\widehat{y_q}(t)$.

The sink node of an IoT system stores the latest received sensory data of all attributes, denoted as $\{b_1, b_1, \cdots, b_n\}$. Besides, for each attribute $x_i \in \mathcal{A}$, the corresponding sensor also stores its latest transmitted sensory data b_i. For each attribute $x_i \in \mathcal{A}$, there exists a value range $[p_i, q_i]$ for its sensory data, *i.e.*, $\forall t, p_i \le x_i(t) \le q_i$. When a continuous query Q is conducted by users, only sensory data in a subset of $\{x_i(t) | x_i \in \mathcal{K}^q\}$ are transmitted to the sink node. For each attribute $x_i \in \mathcal{K}^q$, $x_i(t)$ is transmitted to the sink node only if $x_i(t)$ satisfies that $|x_i(t) - b_i| \ge \alpha \times (q_i - p_i)$, where $0 \le \alpha \le 1$ is a user-defined adjustable constant. Applying this method, the energy consumption of the IoT system can be further reduced with a little sacrifice of the accuracy for estimating the answer of a continuous query. Through adjusting α, we can balance the tradeoff between the energy consumption and the estimation accuracy.

If the sensory data $x_i(t)$ satisfies that $|x_i(t) - b_i| \ge \alpha \times (q_i - p_i)$, the stored sensory data in the corresponding sensor is updated to $b_i = x_i(t)$. Besides, after the sink node received $x_i(t)$, it also updates its stored sensory data to $b_i = x_i(t)$. After the sink node has received all sensory data $\{x_i(t) | x_i \in \mathcal{K}^q, |x_i(t) - b_i| \ge \alpha \times (q_i - p_i)\}$, the sink node estimates the answer of Q by the similar method described in the PLF-AE algorithm.

The detailed operations for the TC-AE algorithm is described in Algorithm 5. Algorithm 5 is proposed to estimate the answers of continuous queries with their (k, β)-relative kernel datasets by piecewise linear fitting method. Algorithm 5 can further reduce the sensory data transmitted and computed in an IoT system by applying the temporal correlations of sensory data.

6 The performance analysis

6.1 Computation complexity of the RKDR algorithm

The computation complexity of computing the Pearson correlation coefficients between query Q and n attributes is $O(mn)$. Then, the computation complexity of calculating the Pearson correlation coefficients between any two attributes is $O(mn^2)$. Therefore, the computation complexity of Step 1 is $O(mn^2)$. The computation complexity of sorting n Pearson correlation coefficients in Step 2 is $O(n \log n)$. In conclusion, the computation complexity for the RKDR algorithm is $O(mn^2)$.

6.2 Computation complexity of the PLF-RKDR algorithm

The computation complexity of Step 1 in the PLF-RKDR algorithm consists of three parts. Firstly, the computation complexity of preprocessing the training set is

$O(nm \log m)$. Then, the computation complexity of calculating optimal segment points for n attributes is $O(nm^3)$. Finally, the computation complexity of calculating the correlation coefficients between query Q and attributes is $O(nm)$. Therefore, the computation complexity of Step 1 is $O(nm^3)$. The computation complexity of calculating candidate relative kernel dataset in Step 2 is $O(mn^2)$. In Step 3, the computation complexity of sorting all correlation coefficients is $O(n \log n)$. In conclusion, the computation complexity for the PLF-RKDR algorithm is $O(nm^3)$.

6.3 Computation complexity of the PLF-AE algorithm and the TC-AE algorithm

The computation complexity of the PLF-AE algorithm and the computation complexity of the TC-AE algorithm are similar. For each attribute $x_i \in \mathcal{K}^q$, the following computations are conducted. Firstly, the computation complexity of sorting the extracted tuples $\{(x_i^1, y_q^1), \cdots, (x_i^m, y_q^m)\}$ is $O(m \log m)$. Then, the computation complexity of calculating optimal segment points is $O(m^3)$. Finally, the computation complexity of calculating the Pearson correlation coefficients and piecewise linear functions for all segments is $O(m)$. Therefore, the computation complexity of the algorithm is $O(km^3)$.

7 Simulation results and discussion

In this section, the performances of the RKDR algorithm and the PLF-RKDR algorithm are evaluated by extensive simulations on both simulation dataset and real dataset. The RKDR algorithm and the PLF-RKDR algorithm retrieve the (k, β)-relative kernel datasets for continuous queries. The application of the RKDR algorithm and the PLF-RKDR algorithm is to estimate the answers of the continuous queries given by users in IoT systems with the retrieved (k, β)-relative kernel datasets. Therefore, to evaluate the performances of the RKDR algorithm and the PLF-RKDR algorithm, we evaluate the accuracy of the answers estimated by the retrieved (k, β)-relative kernel datasets of continuous queries. The PLF-AE algorithm and the TC-AE algorithm are proposed in this paper to estimate the answers of continuous queries by the retrieved (k, β)-relative kernel datasets.

The simulation dataset consists of two sub-datasets, which are about two functions $y_q = f_1(x_1, \cdots, x_{15})$ and $y_q = f_2(x_1, \cdots, x_{15})$, respectively. There are 15 attributes of the IoT systems, which are $x_1, x_2, \cdots, x_{15}$. There are two continuous queries of the IoT systems, which are query Q_1 and query Q_2. Each training example in the simulation dataset includes the values of 15 attributes and the target values of query Q_1 and query Q_2 following function f_1 and f_2, respectively. We apply the RKDR algorithm and the PLF-RKDR algorithm to retrieve the (k, β)-relative kernel dataset for query Q_1 and query Q_2. Then, we apply the PLF-AE algorithm and the TC-AE algorithm to estimate the target values of query Q_1 and query Q_2 with the retrieved (k, β)-relative kernel dataset.

The real dataset is collected from a mobile device. The collected data from the mobile device contains 6 attributes, i.e. the X-, Y-, and Z-axes of a three-axis accelerometer and the X-, Y-, and Z-axes of a three-axis gyroscope of the mobile phone, and a motion type of a person holding the mobile phone. Each training example is generated by a person holding the mobile device and making 7 types of motions, which are numbered as motion 1 to 7. Each training example contains values of 6 attributes and a target value y_q indicating which type of motion happens. We apply the RKDR algorithm and the PLF-RKDR algorithm to retrieve the (k, β)-relative kernel dataset for the motion query, where the (k, β)-relative kernel dataset is a subset of 6 attributes of the mobile phone. Then, we apply the PLF-AE algorithm and the TC-AE algorithm to estimate the motion type of the person holding the mobile phone with the retrieved (k, β)-relative kernel dataset.

In the following simulations, two types of error are applied to evaluate the performance of the proposed algorithms. The first one is the absolute error $er_1 = |\frac{\widehat{y_q}(t) - y_q(t)}{y_q(t)}|$, where $\widehat{y_q}(t)$ is the target value estimated by sensory data of attributes in the (k, β)-relative kernel dataset $\mathcal{K}^q$ retrieved by the proposed algorithms and $y_q(t)$ is the true target value of query Q at time slot t. However, the true target value of query Q is always unknown or inaccessible in the monitored physical world. As a consequence, another type of error replaces $y_q(t)$ by $y'_q(t)$, where $y'_q(t)$ is the target value estimated by sensory data of all n attributes. The second type of error is called as the relative error, denoted as $er = |\frac{\widehat{y_q}(t) - y'_q(t)}{y'_q(t)}|$.

In the following subsections, the comparison experiments are firstly conducted to compare the performances of the RKDR algorithm and the PLF-RKDR algorithm. Then, both absolute error and relative error are evaluated on simulation dataset to verify the performance of the proposed algorithms. Finally, experiments on real dataset are carried out to verify the performance of the proposed algorithms.

7.1 Comparison experiments of the RKDR algorithm and the PLF-RKDR algorithm

We compare the RKDR algorithm and the PLF-RKDR algorithm by conducting a group of comparison experiments on simulation dataset. We compare the relative error $er = |\frac{\widehat{y_q}(t) - y'_q(t)}{y'_q(t)}|$ of the RKDR algorithm and the PLF-RKDR algorithm when the size of the (k, β)-relative kernel dataset increases. The simulation results are presented in Fig. 1. Each data point presented in Fig. 1 is the average of simulation results on 500 times of query.

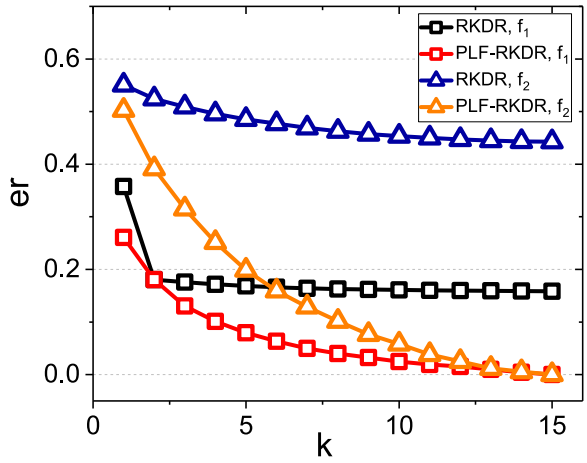

Fig. 1 The relative error of RKDR algorithm and PLF-RKDR algorithm on the simulation dataset

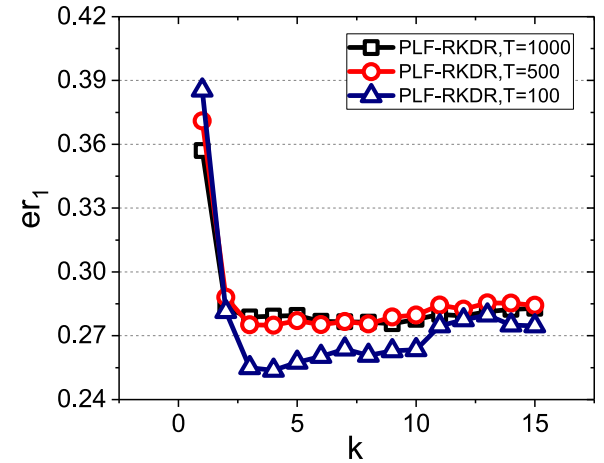

Fig. 2 The impact of k on the absolute error of PLF-RKDR algorithm on the simulation dataset

In Fig. 1, the relative error of both algorithms decreases when k increases. However, the relative error of the PLF-RKDR algorithm is much smaller than that of the RKDR algorithm no matter how much k is. In particular, the relative error of the PLF-RKDR algorithm is almost a half of the RKDR algorithm when k is larger than 5. Therefore, the PLF-RKDR algorithm has better performance than the RKDR algorithm. Besides, Fig. 1 shows that the relative error of both algorithms on simulation dataset with function f_1 is much less than that of function f_2, which reveals that linear correlation cannot describe the relationship between sensory data and query Q_2 perfectly.

7.2 The performance of proposed algorithms on simulation dataset

7.2.1 The absolute error on simulation dataset

The first group of simulations is conducted on the simulation dataset to evaluate the performance of the PLF-RKDR algorithm through the absolute error. Firstly, the PLF-RKDR algorithm is applied to retrieve the (k, β)-relative kernel datasets of continuous queries. Then, the PLF-AE algorithm is applied to estimate answers of continuous queries. Finally, the absolute error of the estimated answers of the continuous queries are evaluated. The simulation results are presented in Figs. 2 and 3.

Firstly, we investigate the impact of k, the size of the (k, β)-relative kernel dataset, on the absolute error of the estimated answers of the continuous queries. Figure 2 presents the absolute error under three test sets, whose size are $T = 1000$, $T = 500$, and $T = 100$. The data points presented in Fig. 2 are the average of simulation results on 1000, 500, and 100 times of query, respectively. Figure 2 presents that the absolute error decreases with the increasing of k. It is shown that the absolute error can reach to 0.258 when k is only 5, which is pretty small for users.

Then, we study the impact of m, the size of training set, on the absolute error of estimated answers of the continuous queries. Figure 3 presents the absolute error when the size of the (k, β)-relative kernel dataset are $k = 5$, $k = 8$, and $k = 11$. Each data point presented in Fig. 3 is the average of simulation results on 500 times of query. Figure 3 reveals that the absolute error of estimated answers of the continuous queries decreases sharply with the increasing of m.

7.2.2 The relative error on simulation dataset

The second group of simulations is carried out on the simulation dataset to evaluate performance of the PLF-RKDR algorithm. Firstly, the PLF-RKDR algorithm is applied on the simulation dataset to retrieve the (k, β)-relative

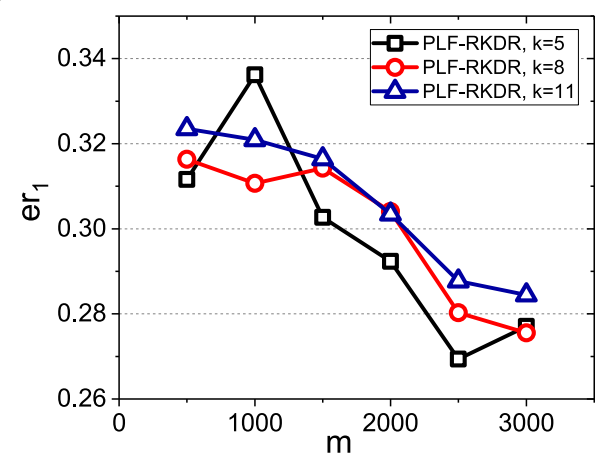

Fig. 3 The impact of m on the absolute error of PLF-RKDR algorithm on the simulation dataset

kernel datasets of query Q_1 and query Q_2. Then, the PLF-AE algorithm is applied to estimate answers of query Q_1 and query Q_2. Finally, the relative error of the estimated answers of query Q_1 and query Q_2 is evaluated. The simulation results are presented in Figs. 4 and 5. Each data point presented in Figs. 4 and 5 is the average of simulation results on 500 times of query.

Firstly, we investigate the impact of k, the size of the (k, β)-relative kernel dataset, on the relative error of the estimated answers of query Q_1 and query Q_2. The size of training set is set as $m = 3000$. When the size of the (k, β)-relative kernel dataset is fixed, the relative error of the estimated answers of query Q_1 and query Q_2 are different since different queries have different relative kernel datasets. That explains why we need to retrieve the relative kernel dataset for each given query. Figure 4 shows that the relative error of the estimated answers of both queries decreases with the increasing of k. Specially, Fig. 4 presents that the relative error is reduced to 0.05 when k is only a half of n. That means the IoT system can save a half of energy consumption when sacrificing only a few accuracy.

Then, we study the impact of m, the size of training set, on the relative error of the estimated answers of the continuous queries. In the simulations, different scales of training set are evaluated and the size of the (k, β)-relative kernel dataset is set as $k = 8$. We evaluate the relative error with the size of training set m varying from 500 to 3000 with increment of 500. Figure 5 presents that the relative error decreases slowly with m increases. Similarly, the relative error of the estimated answers of query Q_1 is less than that of query Q_2. It is worth noting that even m is only 500, the relative error is less than 0.15 when k is no less than a half of n.

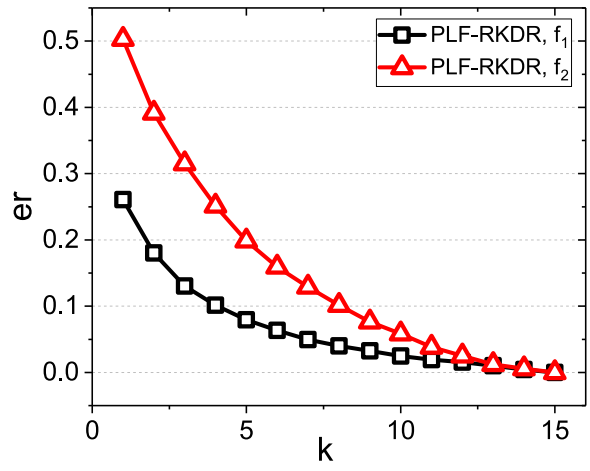

Fig. 4 The impact of k on the relative error of PLF-RKDR algorithm on the simulation dataset

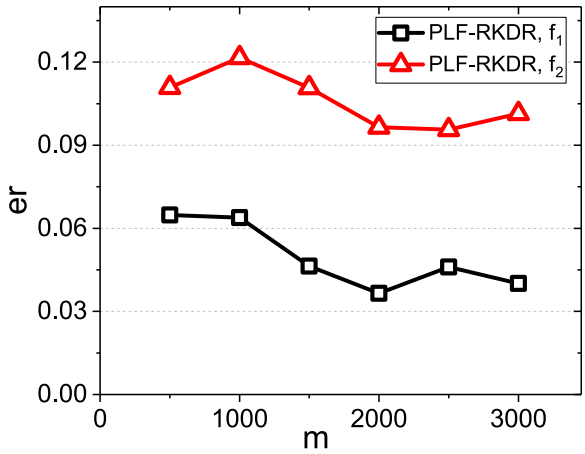

Fig. 5 The impact of m on the relative error of PLF-RKDR algorithm on the simulation dataset

7.3 The performance of the proposed algorithm on real dataset

In this subsection, simulations are conducted on the real dataset to evaluate the performance of the PLF-RKDR algorithm. Each training example of the real dataset consists of the values of 6 attributes and a target value indicating which type of motion happens. For each type of motion i, the target value is set as 1 if motion i happens; otherwise, the target value is set as 0. Firstly, the (k, β)-relative kernel datasets for motion 1 to 7 are retrieved by the PLF-RKDR algorithm. Then, the PLF-AE algorithm is applied to estimate the target value of each motion, respectively. Finally, if the target value of a motion i calculated by the PLF-AE algorithm with its (k, β)-relative kernel dataset retrieved by the PLF-RKDR algorithm is larger than 0.5, it is thought that motion i happens. To evaluate the performance of the PLF-RKDR algorithm, we evaluate the precision of the judgement of each motion. The simulation results are presented in Figs. 6 and 7.

The first group of simulations are conducted to study the impact of k on the motion judgement precision. The size of training set is fixed to $m = 4796$. The simulation results are presented in Fig. 6, where each data point is the average of the simulation results on 1511 times of query. Figure 6 presents that the precision for each motion increases with k increases. The reason is similar to the analysis of simulations on the simulation dataset. For each type of motion, more than 80% of the queries are correctly answered by the PLF-RKDR algorithm and the PLF-AE algorithm.

The second group of simulations are carried out to study the impact of m on the motion judgement precision. The simulation results are presented in Fig. 7, where the size of the (k, β)-relative kernel dataset is set as $k = 4$. Each

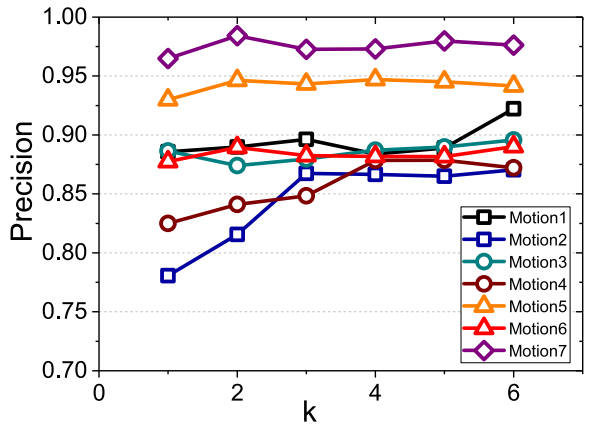

Fig. 6 The impact of k on the precision of PLF-RKDR algorithm on the real dataset

data point presented in Fig. 7 is the average of simulation results on 585 times of query. Figure 7 presents that more than 80% of the motions are correctly judged.

8 Related works

Reducing the energy consumption of an IoT system through reducing sensory data transmitted and computed in the IoT system has been studied for a long time. The existing data reduction algorithms can be divided into two categories. The first category of algorithms are based on sampling. The second category of algorithms are based on compressed sensing.

The first category of data reduction algorithms is sampling-based data reduction algorithms. The authors in [12] investigate Bernoulli sampling-based aggregation algorithms, which can satisfy arbitrary precision requirement. They firstly adapt the sampling probability to satisfy the arbitrary precision requirement given by users. Then, the aggregation is approximately computed with the sampled sensory data. The work in [13] proposes a sampling-based algorithm to compute approximate quantiles in sensor networks. The authors apply random sampling in the algorithm and provide a guarantee that the computed ϕ-quantile are within error ϵ with a constant probability which can arbitrarily close to 1. The algorithms proposed in [14] adapt the sampling frequencies for energy-hungry sensors in wireless sensor networks according to the real needs of the monitored physical environment. As a consequence, the proposed algorithms can reduce the sensory data by sampling and then reduce the energy consumption of a wireless sensor network. Another work in [15] adaptively adjusts the sampling frequency to retrieve the critical data points of each sensor, which can significantly reduce the energy consumption. The authors in [16] apply adaptive sampling in snow monitoring applications of wireless sensor networks. They proposed algorithms to dynamically estimate the sampling frequencies of sensors to minimize sensory data sensed and transmitted in sensor networks while maintaining acceptable accuracy of the monitoring application.

The second category of data reduction algorithms is compressed sensing-based data reduction algorithms. The authors in [17] propose an algorithm combining the compressed sensing with the principal component analysis. The algorithm can recover the whole dataset through a small subset of data, which can significantly reduce the sensory data transmitted in the network. Other two works [18, 19] are proposed to compress the sensory data in a wireless sensor network by the principal component analysis technique. The work in [20] applies the compressed sensing in both single-sink and multi-sink wireless sensor networks to efficiently gather sensory data. The authors in [21] propose

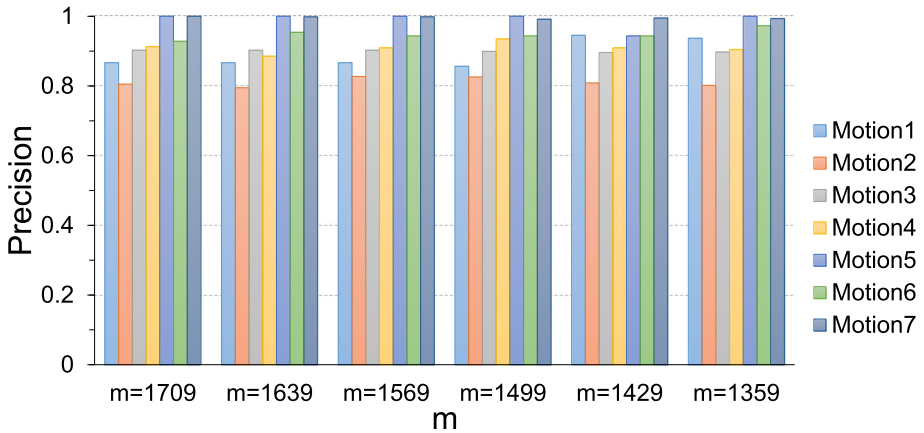

Fig. 7 The impact of m on the precision of PLF-RKDR algorithm on the real dataset

a compressed sensing-based approach to monitor with vehicular networks. The tradeoff between the communication cost and the estimation accuracy is studied in this work. The authors in [22] proposed an algorithm to minimize the energy consumption of sensor networks through joint routing and compressed aggregation. Compressed sensing-based data reduction algorithms compress raw sensory data and transmit compressed sensory data in the network.

However, the sampling based data reduction algorithms in the first category are only effective for some simple statistic queries, such as maximum, quantile, and average. These algorithms cannot reduce sensory data for arbitrary queries. Besides, the compressed sensing-based data reduction algorithms in the second category only consider the temporal and spatial correlations between sensory data but ignoring the correlation between the sensory data and the queries. The sensory data can be further reduced if the correlations between sensory data and queries of IoT system are taken into consideration. And the data reduction for a given query is not enough in these algorithms. Another work is proposed in [23, 24] to retrieve the dominant dataset from big sensory data for a wireless sensor network. However, the dominant dataset defined in [23, 24] is a general one for all queries instead of a specific one for a given query. As a consequence, the existing algorithms cannot retrieve the relative kernel dataset for continuous queries in IoT systems.

9 Conclusion

This paper studies the problem of retrieving the relative kernel dataset from big sensory data for continuous queries in IoT systems. The RKDR algorithm and PLF-RKDR algorithm are proposed to retrieve the (k, β)-relative kernel dataset for continuous queries. Besides, the PLF-AE algorithm and TC-AE algorithm are proposed to estimate the answers of continuous queries based on their (k, β)-relative kernel datasets. Extensive simulations are carried out to evaluate the performances of the proposed algorithms.

Abbreviations

IoT: Internet of Things; LS-PLF: Least-squares-based piecewise linear fitting; PLF-AE: Piecewise linear fitting-based answer estimating; PLF-RKDR: Piecewise linear fitting-based relative kernel dataset retrieving; RKDR: Relative kernel dataset retrieving; TC-AE: Temporal correlation-based answer estimating

Acknowledgements

This work is partly supported by the National Natural Science Foundation of China under Grant NO. 61632010, 61502116, U1509216, 61370217, and 61602129 and the National Science Foundation (NSF) under grant NO.1741277.

Funding

This work is partly supported by the National Natural Science Foundation of China under Grant NO. 61632010, 61502116, U1509216, 61370217 and 61602129 and the National Science Foundation (NSF) under grant NO.1741277.

Availability of data and materials

No applicable.

Authors' contributions

TX designed the idea, performed the experiments, and wrote the manuscript. JB and SY gave valuable suggestions on the design of the study. YS assisted in processing the data and helped to finish the simulation results. JZ supervised the work and helped to check and revise the manuscript. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Author details

[1]School of Computer Science and Tech, Harbin Institute of Technology, Harbin, China. [2]Department of Computer Science, Georgia State University, Atlanta, USA.

Received: 2 October 2018 Accepted: 3 May 2019
Published online: 27 May 2019

References

1. G. G. Says, 6.4 billion connected things will be in use in 2016, up 30 percent from 2015 (2015). http://www.gartner.com/newsroom/id/3165317
2. J. T. Overpeck, G. A. Meehl, S. Bony, D. R. Easterling, Climate data challenges in the 21st century. Science. **331**(6018), 700–702 (2011). American Association for the Advancement of Science
3. J. Yu, Y. Qi, G. Wang, X. Gu, A cluster-based routing protocol for wireless sensor networks with nonuniform node distribution. AEU-Int. J. Electr. Commun. **66**(1), 54–61 (2012). Elsevier
4. J. Yu, N. Wang, G. Wang, Constructing minimum extended weakly-connected dominating sets for clustering in ad hoc networks. J. Parallel Distrib. Comput. **72**(1), 35–47 (2012). Elsevier
5. J. Yu, X. Ning, Y. Sun, S. Wang, Y. Wang, in *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*. Constructing a self-stabilizing cds with bounded diameter in wireless networks under sinr (IEEE, 2017), pp. 1–9
6. T. Shi, S. Cheng, Z. Cai, J. Li, in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. Adaptive connected dominating set discovering algorithm in energy-harvest sensor networks (IEEE, 2016), pp. 1–9
7. J. Li, S. Cheng, Z. Cai, J. Yu, C. Wang, Y. Li, Approximate holistic aggregation in wireless sensor networks. ACM Trans. Sens. Netw. (TOSN). **13**(2), 11 (2017). ACM
8. Q. Chen, H. Gao, Z. Cai, L. Cheng, J. Li, in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. Energy-collision aware data aggregation scheduling for energy harvesting sensor networks (IEEE, 2018), pp. 117–125
9. Z. He, Z. Cai, S. Cheng, X. Wang, Approximate aggregation for tracking quantiles and range countings in wireless sensor networks. Theor. Comput. Sci. **607**, 381–390 (2015). Elsevier
10. S. Cheng, Z. Cai, J. Li, Curve query processing in wireless sensor networks. IEEE Trans. Veh. Technol. **64**(11), 5198–5209 (2015)
11. X. Zheng, Z. Cai, J. Li, H. Gao, A study on application-aware scheduling in wireless networks. IEEE Trans. Mob. Comput. **16**(7), 1787–1801 (2017). IEEE
12. S. Cheng, J. Li, Q. Ren, L. Yu, in *Proceedings of the 29th Conference on Information Communications*. Bernoulli sampling based (ε, δ)-approximate aggregation in large-scale sensor networks (IEEE Press, 2010), pp. 1181–1189
13. Z. Huang, L. Wang, K. Yi, Y. Liu, in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*. Sampling based algorithms for quantile computation in sensor networks (ACM, 2011), pp. 745–756
14. C. Alippi, G. Anastasi, M. D. Francesco, M. Roveri, An adaptive sampling algorithm for effective energy management in wireless sensor networks with energy-hungry sensors. IEEE Trans. Instrum. Meas. **59**(2), 335–344 (2010)

15. T. Zhu, S. Cheng, Z. Cai, J. Li, Critical data points retrieving method for big sensory data in wireless sensor networks. EURASIP J. Wirel. Commun. Netw. **2016**(1), 18 (2016)
16. C. Alippi, G. Anastasi, C. Galperti, F. Mancini, M. Roveri, in *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE International Conference On*. Adaptive sampling for energy conservation in wireless sensor networks for snow monitoring applications (IEEE, 2007), pp. 1–6
17. R. Masiero, G. Quer, D. Munaretto, M. Rossi, J. Widmer, M. Zorzi, in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*. Data acquisition through joint compressive sensing and principal component analysis (IEEE, 2009), pp. 1–6
18. S. V. Macua, P. Belanovic, S. Zazo, in *Signal Processing Advances in Wireless Communications (SPAWC), 2010 IEEE Eleventh International Workshop On*. Consensus-based distributed principal component analysis in wireless sensor networks (IEEE, 2010), pp. 1–5
19. A. Rooshenas, H. R. Rabiee, A. Movaghar, M. Y. Naderi, in *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2010 Sixth International Conference On*. Reducing the data transmission in wireless sensor networks using the principal component analysis (IEEE, 2010), pp. 133–138
20. H. Wang, Y. Zhu, Q. Zhang, in *INFOCOM, 2013 Proceedings IEEE*. Compressive sensing based monitoring with vehicular networks (IEEE, 2013), pp. 2823–2831
21. H. Zheng, S. Xiao, X. Wang, X. Tian, M. Guizani, Capacity and delay analysis for data gathering with compressive sensing in wireless sensor networks. IEEE Trans. Wirel. Commun. **12**(2), 917–927 (2013). IEEE
22. L. Xiang, J. Luo, A. Vasilakos, in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2011 8th Annual IEEE Communications Society Conference On*. Compressed data aggregation for energy efficient wireless sensor networks (IEEE, 2011), pp. 46–54
23. S. Cheng, Z. Cai, J. Li, X. Fang, in *2015 IEEE Conference on Computer Communications (INFOCOM)*. Drawing dominant dataset from big sensory data in wireless sensor networks (IEEE, 2015), pp. 531–539
24. S. Cheng, Z. Cai, J. Li, H. Gao, Extracting kernel dataset from big sensory data in wireless sensor networks. IEEE Trans. Knowl. Data Eng. **29**(4), 813–827 (2017)
25. T. Kalra, R. Mathew, S. P. Pal, V. Pandey, in *Conference on Algorithms and Discrete Applied Mathematics*. Maximum weighted independent sets with a budget (Springer, 2017), pp. 254–266
26. T. L. L. Zong-tian, Leasts-quares method piecewise linear fitting. Comput. Sci. **39**(6A), 482–484 (2012)
27. S. Cheng, J. Li, Z. Cai, in *INFOCOM, 2013 Proceedings IEEE*. O (ε)-approximation to physical world by sensor networks (IEEE, 2013), pp. 3084–3092
28. J. Li, S. Cheng, H. Gao, Z. Cai, Approximate physical world reconstruction algorithms in sensor networks. IEEE Trans. Parallel Distrib. Syst. **25**(12), 3099–3110 (2014). IEEE