A graph-based sensor recommendation model in semantic sensor network

Yuanyi Chen¹, Yihao Lin¹,², Peng Yu¹,², Yanyun Tao¹,², and Zengwei Zheng¹

Abstract
In the past few years, introducing ontology to describe the concepts and relationships between different entities in semantic sensor network enhances the interoperability between entities. Existing works mostly based on SPARQL retrieval ignore the user’s specific requirements of sensor attributes. Therefore, the recommendation results cannot satisfy the user’s needs. In this article, we propose a graph-based sensor recommendation model. The model mainly includes two parts: (1) Filtering nodes in data graph. In addition to using the traditional graph matching algorithm, we propose a threshold pruning algorithm to narrow the matching scope and improve the matching efficiency. (2) Recommending top-k sensors. We use the improved fast non-dominated sorting algorithm to obtain the local optimal solutions of sensor data set, and we apply the simple additive weight algorithm to characterize and sort local optional solutions. Finally, we recommend the top-k sensors to the user. By comparison, the graph-based sensor recommendation algorithm meets user’s needs more than other algorithms, and experiments show that our model outperforms several baselines in terms of both response time and precision.

Keywords
Graph matching, threshold pruning algorithm, sensor selection, semantic sensor network, fast non-dominated sorting algorithm

Date received: 10 May 2021; accepted: 1 September 2021
Handling Editor: Yanjiao Chen

Introduction
In the Internet of things (IoT), the Internet is used as medium to control all kinds of physical devices for collecting real-time environmental data and transferring the data to middleware’s specific function modules. And the middleware returns the results to users. At present, the IoT is widely employed in intelligent home, intelligent city, intelligent healthcare, intelligent transportation, and other fields. For example, apps on smart phones enable many sensors to monitor traffic jam and optimize routes in intelligent transportation. Type-B ultrasonic can use ultrasonic sensors to know every move and development status of the fetal, which can avoid some unexpected circumstances. Numerous sensors are applied in our daily life.

Due to the development of communication technology and the emergence of various intelligent objects, a large number of physical objects are connected to the

¹Hangzhou Key Laboratory for IoT Technology and Application, Zhejiang University City College, Hangzhou, China
²Department of Computer Science and Technology, Zhejiang University, Hangzhou, China

Corresponding author:
Yanyun Tao, Department of Computer Science and Technology, Zhejiang University, 866 Yuhangtang Road, Hangzhou, Zhejiang 310058, China. Email: taoyanyun@zju.edu.cn
Zhang et al.4 proposed a schedule that it expressed recommendation under the semantic sensor network. Retrieval has made significant progress in sensor recommendation. Rasyid et al.6 data modeling in semantic sensor networks and laid a foundation for sensor recommendation. The above paper solved the problem of semantic IoT-Lite) with better scalability by following these principles. The specific definitions and models were also given. In the face of the descriptive data in the semantic sensor network, many traditional sensor recommendation algorithms need to process large-scale sensor data, such as collaborative filtering algorithm, convolutional neural network, and other algorithms, which must converse all semantic sensor data into normal form. This processing of data conversion takes a lot of time and energy, so this traditional sensor recommendation algorithm is not applicable in semantic sensor networks. How to recommend sensor based on semantic sensor data is a problem to overcome.

In recent years, many studies3–6 based on SPARQL retrieval have made significant progress in sensor recommendation under the semantic sensor network. Zhang et al.4 proposed a schedule that it expressed sensor data into Resource Description Framework (RDF) form based on semantic sensor network standard, including XML-based mapping language algorithm (SASML) and Sensor Data to RDF Mapping (SDRM) algorithm. Bermudezdo et al.5 reduced the semantic description’s complexity and processing time in the semantic sensor network. They listed 10 semantic model design principles, then design models (such as IoT-Lite) with better scalability by following these principles. The above paper solved the problem of semantic data modeling in semantic sensor networks and laid a foundation for sensor recommendation. Rasyid et al.6 created a project to provide users with sensor data. The PROTEGE editor was used to model the sensor data in the csv file into an entity and store it in the database in the form of tuple. And it used the SPARQL integrated in the Sesame framework to search database and display results to users. Gong et al.9 constructed a CASSF framework, and they used SRSF to analyze user needs’ semantics and SPARQL to retrieve sensors. After semantic analysis, Comparative Priority-based Weighted Index (CPWI) was used to characterize and rank sensor options. Relying heavily on SPARQL for parsing semantic sensor data in sensor recommendation algorithm has a big problem that ignoring sensor attributes and user preferences causes the recommended sensors fail to meet user needs.

Based on the above problems, we propose a graph-based sensor recommendation model. This model parses the sensor semantic data based on the semantic sensor network ontology to model a weighted data graph, and it takes user’s preference for sensor attributes into account to model a weighted search graph. From the perspective of the user, the weighted data graph is characterized and pruned using threshold pruning algorithm. Then, we use the improved fast non-dominated sorting algorithm and Simple Additive Weighting (SAW) algorithm to sort sensor options. The main contributions of our model are as follows:

- We redefine the matching method between the two weighted graphs, where we parse semantic sensor data to model a weighted data graph and get the user’s preference of sensor’s attributes to model a weighted search graph.
- We propose a threshold pruning algorithm to narrow the matching range and improve the matching efficiency.
- We use the improved fast non-dominated sorting algorithm to obtain the local optimal solution of the sensor data set after pruning, which improves the accuracy of the algorithm.

The rest of this article is organized as follows: section “Related work” investigates the related work of sensor recommendation algorithms under the IoT. The proposed graph-based sensor recommendation model is introduced in section “Sensor recommend model.” Section “Experiment evaluation” reports and discusses the experimental results. Finally, we present our conclusions and future work in section “Conclusion.”

Related work

In this section, we first discuss related work of sensor recommendation algorithms in the IoT and summarize the differences between our proposed model and previous work. Zhou et al.7 divided existing sensor recommendation algorithms into two categories, which include context-based algorithm and content-based algorithm.

Context-based sensor recommendation algorithms rely on different accessible context information in the user’s description of sensor. The existing sensor recommendation algorithms conduct the sensor recommendation from two aspects. On one hand, previous sensor recommendation works in semantic sensor network mainly contain two parts. Some simplify the sensor ontology for easy implying, and others use contextual content to sort sensors after using SPARQL to retrieve semantic sensor data. On the other hand, sensor recommendation algorithms based on general data sets
improve themselves by increasing precision or reducing response time.

Parsing semantic sensor data is the key in the semantic sensor network, which is why many studies are based on SPARQL. Rasyid et al.\textsuperscript{6} created a project to provide users with semantic sensor data. In terms of data processing, the information collected by the sensor was stored in the csv file, and the PROTEGE editor was used to transform sensor information into entity which was stored as the tuples in the database. The Sesame framework is used to retrieve the database after the user’s needs converted into the SPARQL search form, and the search information was fed back to the user. The CASSF\textsuperscript{8} framework enriched the sensor attributes of the semantic sensor network ontology and used RDF to store entity data for maximizing the preservation of semantic information, where the CASSF framework also used SPARQL for retrieval sensor data. Perera et al.\textsuperscript{9} proposed a context-based sensor recommendation model CASSARAM, which considered user preferences of sensor attributes, such as reliability, accuracy, and battery life. This model employed CPWI to characterize and sort sensor options. Chirila et al.\textsuperscript{10} proposed a proxy-based architecture, which can perform sensors’ discovery and recommendation at the same time. And it used a web service clustering method during recommending to reduce the search space of candidate services. Meclibah et al.\textsuperscript{11} proposed a mechanism based on concept catalogs to speed up the search of semantic resources. When users searched for semantic resources, they will first search the directory. It used SPARQL to search the database if queries were in directory, which avoided an invalid search. Gomes et al.\textsuperscript{12} proposed the semantic-based service discovery architecture QoDisco, which was composed of a set of independent repositories. QoDisco also designed synchronous and asynchronous retrieval mechanisms. The above research is based on semantic sensor network for sensor recommendation, which utilized SPARQL to retrieve sensor data.

In terms of improving the accuracy of traditional sensor recommendation algorithms, many excellent models had been proposed. Neha et al.\textsuperscript{13} proposed the Vector Based Sensor Ranking (VBSR) model, which calculated a preference-based weighted index (PBWI) for each sensor option, which relies on the sensors’ attributes value and context attributes value input by the user. Nunes et al.\textsuperscript{14} took sensor recommendation as a multi-criteria decision analysis problem. In order to combine the high accuracy of the fast non-dominated sorting algorithm and the low response time of TOPSIS algorithm,\textsuperscript{15} they proposed ES algorithm, which reduced the time complexity of the fast non-dominated sorting algorithm by limiting the size of sensor data set. Kertiou et al.\textsuperscript{16} used the dynamic skyline algorithm to filter sensor data, which improved the accuracy of multi-criteria decision analysis algorithm. After filtering the data, the sensor data were characterized, sorted, and recommended to the user. In order to select the most desired sensor according to user needs, Nithya et al.\textsuperscript{17} proposed a clustering method to optimize the sensors’ selection in the IoT. Bharti et al.\textsuperscript{18} proposed a value of information-based sensor ranking mechanism (VoISRAM), which considered sensor context information and sensor service level as information attribute values for modeling. And this model attempted to balance between services’ QoS requirements and energy consumption. Above research is about non-semantic sensor recommendation, and some algorithms are used to improve the accuracy of recommendation, such as the fast non-dominated sorting algorithm, the dynamic skyline algorithm, and clustering algorithm.

Content-based sensor recommendation algorithms retrieve the sensor history output data based on the user’s demand for the sensor.\textsuperscript{19} They are required to process a large amount of sensor historical data. If algorithms process semantic sensor data, it mainly consumes a lot of time and energy of the sensor. Therefore, the content-based sensor recommendation algorithm cannot refer to the semantic sensor data. The existing content-based sensor recommendation algorithms make improvements by increasing the accuracy and decreasing the response time of sensor recommendation algorithms. In recent years, machine learning models have also been involved. Truong et al.\textsuperscript{20} borrowed the idea of Google retrieving pictures and adopted the “case-by-case search,” which avoided inaccurate input of keywords by users. It took the sensor’ historical output as a comparison object and used fuzzy set to efficiently calculate the similarity score for obtaining a ranked list of matching sensor options. Ostermaier et al.\textsuperscript{21} proposed a real-time prediction model Dyser for the IoT, which supported web infrastructure to publish sensor entity data and retrieve data based on the specified sensor’s type. When Dyser returned the search results, it outputted part of the relevant sensor data in descending order according to the predicted consequences, which reduced a lot of communication overhead. Truong and Romer\textsuperscript{22} proposed a lightweight prediction model CSS based on fuzzy logic to estimate the probability of a sensor option matching a search query. This model implemented content-based sensor search in the IoT, which had low communication overhead and computational efficiency performances. Zhang et al.\textsuperscript{23} proposed a low-cost and high-precision prediction model based on quantitative values. This model used a multi-step prediction method, and they apply approximate values to evaluate the next state of the sensor at any time. Zhang et al.\textsuperscript{24} proposed a sensor state prediction method to estimate the short-term sensor’s state. This prediction method could make the best use of the time correlation among sensor data, and then
In recent years, machine learning has also been applied to recommend sensors in IoT. Mietz and Romer used the highly correlated characteristics of many sensors’ output to learn the relevant structure from sensors’ historical data and then modeled it as a Bayesian network (BN). This model could estimate the sensor option’s recommended probability without knowing the current sensor output, and it recommended sensor options to the user with a superior acquisition probability. Zhang et al. established a prediction model based on historical temperature and humidity information, after using the collected environmental data to learn and train the back propagation (BP) network. When the change trend of environmental parameters exceeds the threshold, an early warning is performed to discover hidden danger of the equipment in advance. Li et al. first introduced deep learning to the edge computing environment, and they had also designed a new offload strategy to optimize the performance of deep learning applications based on edge computing due to the limited processing power of existing edge nodes. Although there are machine learning algorithms involved in the sensor recommendation field, they are not mainstream because a lot of time cannot be expended in training models in the IoT.

At present, there are relatively few sensor recommendation algorithms for semantic data, but this aspect is of great significance for the IoT to solve the heterogeneity of devices. Most of recommendation algorithms for the semantic sensor network rely excessively on the use of SPARQL to retrieve semantic sensor data, which results in the recommended sensor options being locally optimal. The SPARQL query semantic database operation is equivalent to query a relational database using SQL statements, which enable the query results fail to meet the user’s needs without involving user’s preference information. Therefore, we propose a graph-based sensor recommendation model. This model can make better use of RDF semantic data stored in graphs, and it considers sensors’ attributes and user’s preference for sensor recommendation.

### Sensor recommend model

In this part, we will first give the problem definition of our model. Then, we introduce background knowledge of recommending semantic sensor network. Finally, we state our graph-based sensor recommendation model.

### Problem definition

For ease of the following presentation, we define the key data structures and notations used in the proposed model. Table 1 lists the relevant notations used in this article.

| Symbol | Description |
|--------|-------------|
| T      | The sensor type |
| N      | The maximum number of sensors |
| K      | The threshold |
| G_d    | The data graph |
| G_s    | The search graph |
| D_p    | The sensor data set after pruning |
| F      | The first non-dominated front |
| S_l    | The score of latitude-longitude |
| S_p    | The score of preference |
| S_t    | The total score |
| U_r    | The attributes of user required sensor |
| U_p    | The preferences of sensor’s attributes |

### Definition 1 (data graph).
In this article, we parse the semantic sensor data and store them in different files according to the sensor type, which is convenient to access. Then, we read the specific file based on the user required sensor type to model a weighted data graph $G_d = \{Node[1] : \{n[1], n[2], \ldots, n[i], \ldots\}, \ldots, Node[j] : \{n[1], w[2], \ldots, n[i], \ldots\}\}$. And each $Node[j]$ contains different number of $n[i]$, and $n[i]$ may include sensor attribute values and nodes related to the node $Node[k]$. The edges in the data graph are the attributes or relationships defined by the sensor ontology in the semantic sensor network. The weight in the data graph is the evaluation of the sensor attributes based on the current sensor attribute values. An example of data graph is shown in Figure 1.

### Definition 2 (search graph).
We model a weighted search graph $G_s = \{p[1], p[2], \ldots, p[i], \ldots\}$ according to the user’s input requirements in the user interface, where the weight $p[i]$ is the user’s preferences of sensor’s attributes. An example of search graph is shown in Figure 2.
Definition 3 (the first non-dominated front). In multi-criteria decision analysis problems, there may be conflicts and incomparability between multiple criteria. One solution may be the best in one criterion, and the worse in another. We select two sensor options ($S_1$ and $S_2$) randomly. When every sensor attribute value in $S_1$ is no less than the corresponding sensor attribute value of $S_2$, we call $S_1$ dominant $S_2$. In other words, if the sensor option $S_1$ is not dominated by other sensor data, $S_1$ is called the non-dominated solution. The set of all non-dominated solutions is the first non-dominated front $F$.

With the aforementioned definitions, the problem of graph-based sensor recommendation model can be formally stated as follows:

Given semantic sensor data set and user input sensor preferences of sensor’s attributes, the problem of graph-based sensor recommendation model aims to select top-$k$ sensors for user by graph matching algorithm.

Background

RDF. RDF is advocated by W3C in order to describe resources on the World Wide Web and their relationships. The core of the RDF data model includes resources, properties, RDF statements, and so on, where each resource has a Uniform Resource Identifier (URI). Using RDF data model to describe data in the form of triples eliminates the heterogeneity between devices.

Sensor ontology description. According to the Single Sign-On (SSO) design pattern, the semantic sensor network ontology can be described as four main aspects: sensors, observations, systems, features, and attributes. This article mainly describes the sensor information from the perspective of the sensor. The main concepts and relationships are shown in Figure 3.

Graph matching algorithm. The graph matching algorithm is mainly to find the subgraph isomorphism of the search graph $G_s$ in the data graph $G_d$. The Ullmann Algorithm is the most widely used among traditional graph matching algorithms, and many graph matching algorithms are based on this algorithm. However, using the Ullmann Algorithm to get subgraph isomorphism
is a non-deterministic polynomial-time (NP) hard problem. In this article, after we use the threshold pruning algorithm to initially filter the nodes, we use the Ullmann Algorithm to get subgraph isomorphism of the search graph in data graph. The Ullmann Algorithm is mainly based on the following theorem to determine subgraph isomorphism.

Given search graph $G_s = (V_s, E_s)$ and data graph $G_d = (V_d, E_d)$, our aim is to find all the isomorphic subgraphs of the data graph $G_d$ and the search graph $G_s$. We denote the number of nodes and edges in the data graph as $p_a, q_a$ respectively, and we denote the number of nodes and edges in the search graph $G_s$ as $p_b, q_b$, respectively.

The adjacency matrices of $G_s$ and $G_d$ are $A = [a_{ij}]$ and $B = [b_{ij}]$, respectively. And then, we define a mapping matrix $M'$, and there is the mapping matrix $M'$ when $G_d$ contains subgraph isomorphism of $G_s$. It is composed of $p_a \times p_b$ elements, where each row can contain only one “1,” and each column can contain at most one “1.” We use this matrix $M'$ to perform a series of row and column transformations on matrix $B = [b_{ij}]$ to obtain our matrix $C$, which is defined as follows:

$$C = [C_{ij}] = M'(M'B)^T \tag{1}$$

where $T$ is the transpose of the matrix.

If there is an isomorphic matrix of figure $A$ in figure $B$, then it must satisfy follow equation

$$\forall i \forall j : (A[i][j] = 1) \implies (C[i][j] = 1) \tag{2}$$

Therefore, $M'$ indicates an isomorphic mapping of the retrieval graph and the data graph.

Finally, whether this is right of mapping matrix $M'$ or not can be determined by the following equation

$$C = M'(M'B)^T$$

then, $\forall i \forall j : (A[i][j] = 1) \implies (C[i][j] = 1) \tag{3}$

As shown in Figure 4, there is a mapping relationship between the data graph $G_d$ and the search graph $G_s$ to construct the isomorphic subgraph.

The sensor recommendation model

The current semantic sensor recommendation model relies too much on SPARQL retrieval. When retrieving semantic data, the user’s preference information of sensor attributes and the semantic information of the sensor itself cannot be considered, so that the sensor options recommended cannot meet user’s needs. In order to solve the problem of using SPARQL retrieval and better use of RDF graph semantic information, we use graph matching algorithm to filter sensors and provide users with sensor option. When matching the weighted search graph and the weighted data graph, we not only use the classic graph matching method to obtain the isomorphic graph but also propose a threshold-based pruning method to filter the sensors. The hierarchical structure of sensor recommendation model based on graph matching is shown in Figure 5.

**Step 1: filtering nodes in data graph**

1. Data processing. We parse RDF data, and then we write sensor options to different files according to sensor type. In order to obtain the range of values of different sensor attributes, we created a table to record the attributes of different sensor types and their range of values during parsing RDF data. The content of the table is shown in Table 2, where “qualitative” is used to measure sensor attribute characteristic. When the qualitative value of the sensor attribute is 1, it means that the larger the sensor attribute value, the better. On the contrary, if the qualitative value of the sensor attribute is −1, the smaller the sensor attribute value, the better. When the data are updated, only the corresponding sensor file needs to be changed.

2. Construct data graph and search graph. We create a user visual interface, and let the user input required sensor information including sensor type, location, attributes, and corresponding preferences. Then, we extract the user input information as a weighted search graph, where the weight is the user’s preferences of the sensor’s attributes. When constructing the data graph, we first match user input sensor’s type with all the sensors’ type in sensor attribute table. If the matching is unsuccessful, the result will be returned. If the matching is successful, the corresponding sensor file will be parsed according to the sensor type, and the sensor...
attribute table will be read. Then, the data graph’s weight is calculated according to the sensor semantic information and the value range of the sensor attribute. Finally, the data graph is constructed

\[ q^0_{ij} = \begin{cases} \frac{q_i - q^\text{min}_j}{q^\text{max}_j - q^\text{min}_j} & \text{qualitative} = 1 \\ q^\text{max}_j - q_i & \text{qualitative} = -1 \end{cases} \]  

(4)

where \( q_i \) is the \( j \)th attribute value of the \( i \)th sensor, and \( q^0_{ij} \) represents the weight of the \( j \)th attribute value of the \( i \)th sensor in the data graph. The two parameters \( q^\text{max}_j \) and \( q^\text{min}_j \) are obtained by reading the sensor attribute table, where \( q^\text{max}_j \) represents the maximum value of the \( j \)th sensor attribute in the data graph, and \( q^\text{min}_j \) represents the minimum value of \( j \)th sensor attribute in the data graph. When the corresponding qualitative value of the attribute in the sensor attribute table is “1,” we use first equation (4) to calculate \( q^0_{ij} \), and when the corresponding qualitative value of the attribute in the sensor attribute table is “-1,” we use second equation (4) to calculate \( q^0_{ij} \).

3. Threshold-based pruning. We first use the traditional graph matching algorithm for pre-prune. We first compare the degree of the search graph with the degree of the node in the data graph. When the degree of the node in the data graph is larger than the degree of the search graph, the node is retained. When the degree of the node in the data graph is lower than the degree of the search graph, the node is deleted. In addition, in order to prune the data graph according to the user’s preferences of sensor’s attributes, we propose a threshold pruning algorithm, which mainly considers two aspects: \( S_L \) is the matching degree of longitude and latitude, which compute the nodes of data graph and the search graph. And \( S_P \) is the degree of the nodes’ attribute value in the data graph meets the user’s requirements for the sensor attributes. According to the above two aspects, we characterize each sensor option(node) in the data graph after pre-prune. And we determine a threshold \( K \) by experiments, and the node whose \( S \) does not exceed this threshold \( K \) has to be deleted. In order to reduce the subsequent processing time.

Figure 5. The hierarchical structure of sensor recommendation model.

Table 2. Sensor property table.

| Sensor type         | Attribute | Attribute minimum value | Attribute maximum value | Qualitative |
|---------------------|-----------|-------------------------|-------------------------|-------------|
| Temperature sensor  | Sensitivity | 20                      | 100                     | 1           |
|                     | The response time | 15                      | 30                      | -1          |
Algorithm 1. A pruning algorithm based on user preference.

Require: Sensor type $T$; Maximum number of sensors: $N$; Threshold value $K$; Data graph $G_d$; Search graph $G_s$;
Ensure $D_p$: Sensor data set after pruning;

1: function PRUNINGFUNCTION $T$, $G_s$, $G_d$, $N$, $K$
2:     Get the attributes’ value range and qualitative according to $T$ in sensor property table, table
3:     Gets the number of child nodes in the search graph $G_s$, sum
4:     for each in $G_s$.keys(): do
5:         if Matchfunction($G_s$.keys(), $G_d$.keys[()].keys()) then
6:             Use equation (6) to compute score of latitude and longitude $S_i$
7:                 for each $j$ in $G_d$.keys(): do
8:                     Use equation (7) to compute score of preference $S_p$
9:                     end for
10:         end if
11:     end for
12:     Use equation (5) to compute total score $S$
13:     if $S$ $>$ $K$ then
14:         if len($D_p$.keys()) $<$ $N$ then
15:             $D_p$ += $S$
16:         else
17:             end if
18:     end if
19: end for
20: return $D_p$
21: end function

of sensor data, a parameter $N$ is set in this algorithm to limit the number of sensors after pruning. The main contents of the threshold pruning algorithm are as follows, and Algorithm 2 is presented below

\[
S = S_l + S_p \quad (5)
\]

\[
S_l = \frac{1}{\sqrt{\Delta \text{lat} + \Delta \text{lon} + 1}} \quad (6)
\]

where $\Delta \text{lat}$ is the absolute values of the latitude between the search graph and the nodes of the data graph, and $\Delta \text{lon}$ is the absolute values of the longitude between the search graph and the nodes of the data graph

\[
S_p = \begin{cases} 
\sum_{i=1}^{N} \left( w_i \times q_i^s \right) & \text{qualitative} = 1 \\
\sum_{i=1}^{N} \left( -w_i \times q_i^s \right) & \text{qualitative} = -1 
\end{cases} \quad (7)
\]

where $w_i$ is the user input preference of the $i$th sensor’s attribute.

The pruning algorithm above requires the sensor type $T$, the maximum number of sensors $N$, the threshold $K$, the data graph $G_d$, and the search graph $G_s$. The return value of the pruning algorithm is filtered sensor data set $D_p$, and the size of sensor data set is less than $N$. Reading the content of the sensor attribute table obtains the range of sensors’ attributes value and qualitative of different sensor attributes in line 2. Comparing search graph with the nodes in the data graph according to the traditional graph matching algorithm in line 5. If the graph composed of nodes and neighbor nodes in the data graph is an isomorphic graph of the search graph, we calculate the difference between latitude and longitude in line 6 and attribute satisfaction in line 8. Next, the values calculated above are normalized to get total characterization score $S$ in line 10. Finally, we compare the characterizing value $S$ with the set threshold $K$. If the $S$ is less than $K$, no processing has to be carried out. If the $S$ is greater than $K$, the size of the sensor data set and $N$ are judged, and subsequent operations are performed. If the size of the sensor data set is lower than $N$, adding the node of the data graph to the sensor data set in line 13. If the size of the sensor set is not less than $N$, the sensor nodes in the sensor data set are scored according to $S$ in reverse order, and then the node with the lowest $S$ is deleted and the nodes that passed the filter is added.

Step 2: recommending top-$k$ sensors. Sensor recommendation can be considered as a multi-criteria decision analysis problem. How to balance user requirements on multiple sensor’s attributes has become the primary task of sensor recommendation algorithm. In previous studies, many scholars used local optimal solutions of sensor data set to improve traditional multi-criteria decision analysis algorithms, such as the fast non-dominated sorting algorithm and dynamic Skyline algorithm, but the high time complexity of these algorithms greatly affects the response time of the sensor recommendation. In order to conquer this problem, we proposed the improved fast non-dominated sorting algorithm last year. This algorithm combines the idea of quick sort algorithm to improve the fast non-dominated sorting algorithm, and it reduces the time complexity of the fast non-dominated sorting algorithm from $O(mn^2)$ down to $O(n \log mn)$. Experiments show that the algorithm is significantly better than the previous model. In this article, we use an improved fast non-dominated sorting algorithm for improving the accuracy of sensor recommendation model. Instead of classifying all solution sets, we pursue the local optimal solution. The time complexity of the improved fast non-dominated sorting algorithm is $O(n \log mn)$, the time complexity analysis is as follows.

Assuming all possible inputs are equally likely, and then all partitioning cases are equally likely. We will take each value in the interval, with equal probability, so the average time complexity of the quick sort algorithm is as follows
The improved fast non-dominated sorting algorithm requires data set Fastlist' and ensure the first non-dominated front DFront. The first to fifth lines define the comparison function for the domination between different solutions. The sixth to thirtieth lines define the main part of improved fast non-dominated sorting algorithm. It first judges the value of start and end. Then, it starts looping from start to end and end to start to judge the dominance in lines 11–20 and lines 21–29, respectively. In line 31, it is a recursive process that calls itself.

After using the improved fast non-dominated sorting algorithm to obtain the local optimal solutions, we use the SAW algorithm to characterize and sort the local optimal solution. The SAW algorithm is one of the most commonly used in multi-criteria decision analysis algorithms, which is the basis of other multi-criteria decision analysis algorithms. The SAW algorithm mainly includes the following three steps:

1. Normalizing the sensor data set

\[ s_j' = \frac{s_j - s_j^{\min}}{s_j^{\max} - s_j^{\min}} \]  
\[ s_j' = \frac{s_j^{\max} - s_j}{s_j^{\max} - s_j^{\min}} \]  

where equation (10) is for the maximization criterion, and equation (11) is for the minimization criterion.

2. Calculating the score for each sensor option

\[ S_s = \sum_{j=1}^{N} (w_j \times s_j') \]  

where \( w_j \) is the user preferences of sensor's attributes, and \( N \) is the number of sensor attributes.

3. Sorting sensor data set is in descending order according to the sensor options' score \( S_s \).

**Example of model applications**

In order to simplify computing, we assume the sensor data set contains five sensor nodes in different regions. We need to use the graph-based sensor recommendation model to get the optimal recommendation among sensor data set.

**Step 1: filtering nodes in data graph.** According to the sensor's location, we divide the sensor data set and record data in different files. After analyzing the input information in the user interface, we get sensor's location and user's preferences of different sensor's attributes. We construct search graph with user input information.
After filtering sensor data set according to the sensor location of the user input information, we have determined three sensor nodes that meet the requirements and the specific information of the nodes is shown in Table 3. According to equation (4), the response time weights of sensor nodes S1, S2, and S3 are calculated as 0.5, 1, and 0, respectively, and the sensitivity weights of sensor nodes S1, S2, and S3 are 0, 1, and 1, respectively. We construct the data graph with weight information, and the data graph and the search graph are shown in Figures 6 and 7, respectively.

Next, using the threshold pruning algorithm to sort the size of the data graph is mainly used with large data sets. When the size of data set is small, we can directly omit this step. In the current, there are only three sensor nodes in the data graph. In order to clearly show the specific implementation of the threshold pruning algorithm, we use the threshold pruning algorithm as set in the model. According to equations (10)–(12), the scores of nodes S1, S2, and S3 in the data graph are $-4.72$, $-13.84$, and $1.13$, respectively. According to the threshold setting in the original model, these three sensor nodes do not meet the requirements, but the original model mainly solves the sensor recommendation problem under big data, so we only delete the worst S2 node.

**Step 2: recommending top-k sensors.** According to the improved fast non-dominated sorting algorithm, the response time of sensor node S3 is better than S1 and the sensitivity of S3 is also better than S1, so S3 dominates S1. Therefore, we need to recommend sensor node S3 to users. The process of recommending options nodes is shown in Figure 8. When the number of local optimal solution sets obtained using the fast non-dominated sorting algorithm is greater than 1, we need to characterize and sort the sensor nodes in the set according to the TOPSIS algorithm.

**Experiment evaluation**

In this part, we describe the specific preparation content of the experiment from three aspects including data set, comparison methods, and evaluation indicators, and assess the performance of the sensor recommendation model based on graph matching and the price comparison method through the test results.
Experimental settings

Data sets. There is currently no large-scale public data set to provide sensor and context information. In order to obtain large-scale data, we simulate sensor data according to the 22 different sensor property rules described by the public sensor website “Array of Things,” where the sensor property rules are shown in Table 4. We integrated real data and simulation data to construct four scale sensor data sets, including 50,000, 100,000, 150,000, and 200,000. This combination of data can provide a large amount of sensor data, which contribute to better understand the behavior of the sensor recommendation algorithm.

According to the definition of the sensor ontology in the semantic sensor network, we use Protege to construct the sensor semantic model after constructing the sensor data set. And then, the four sensor data sets constructed are poured into the Protege, and they are converted into semantic data. In Protege, ontology is formed on various annotations, including RDF/XML, N3, N-Triples, and so on. We use RDF/XML to annotate the ontology, which is also the simplest form.

Table 4. The value range of sensor attributes.

| Sensor attribute         | Unit   | Value range   |
|--------------------------|--------|---------------|
| Life                     | y      | 5 or 10       |
| Sensitivity              | nA/ppm | [2,100]       |
| Measurement accuracy     | ppm    | [0.1]         |
| Response time            | s      | 8, 15, or 30  |
| Start time               | ms     | 70 or 300     |
| Energy consumption       | uW     | 2.7 or 50     |
**Comparative methods**

- **TOPSIS**: TOPSIS algorithm\(^\text{15}\) normalizes the sensor data set matrix. Next, it obtains the best point and the worst point according to the objective function, and it calculates the distance between each sensor option in the sensor data set matrix to the best point and the worst point. Finally, it characterizes, ranks, and recommends sensor options.

- **ES**: after using TOPSIS to sort sensor options, ES algorithm\(^\text{14}\) sets SR parameters to limit the number of sensor options input to the fast non-dominated sorting algorithm. Finally, top-\(k\) sensor options of non-dominant frontiers are recommended.

- **Dynamic skyline algorithm**: this method\(^\text{16}\) obtains the user request, and it calculates the local dynamic skyline for reducing the size of sensor data set. Next, it calculates the global dynamic skyline to filter sensor options again. Finally, it uses the SAW algorithm to sort the sensor options.

- **An efficient preference-based sensor selection algorithm**: this algorithm\(^\text{1}\) narrows the sensor data set based on the user’s preferences of sensor’s attributes, and it sets the number of sensors input to the improved fast non-dominated sorting algorithm. Decreasing the number of processing sensors makes the algorithm’s response time lower than the original algorithm. Then, the obtained results are sorted and recommended to users through TOPSIS algorithm.

**Evaluation methods.** In the IoT, the computing and storage capacity of sensors is not as good as those devices in the Internet, which requires our algorithm to provide optimal sensor recommendations under low time complexity. From the analysis of the characteristics of the IoT and the satisfaction of user, we use two evaluation indicators including precision and response time to estimate our model. Response time refers to the time interval from a user submitting a request on the application to feedback. The lower the response time, the better the algorithm. The precision mainly refers to the ratio between the number of sensors satisfied by the user and the number of sensors recommended by the algorithm. A higher ratio indicates a higher precision of the algorithm, as following

\[
\text{Precision} = \frac{S}{K} \tag{13}
\]

where \(S\) is the number of sensors satisfied by user, and \(K\) is the number of sensors recommended by the algorithm.

**Experimental results**

In this part, we first introduce the effect of parameter settings on the graph-based sensor recommendation model. After determining the parameters of the model, we will compare the performance of our proposed model with the above four sensor recommendation algorithms in terms of the response time and precision.

**Impact of parameter setting.** In the graph-based sensor recommendation model, we need to determine two parameters including the maximum number of sensors \(N\) and the threshold \(K\). Among them, the maximum number of sensors limits the scope of the threshold pruning algorithm, and the number of sensors obtained after using the threshold pruning algorithm is lower than maximum number of sensors. Under the condition of high precision of threshold pruning algorithm, limiting the maximum number of sensors helps to reduce the response time of the algorithm. The threshold is used to limit the strength of the threshold pruning algorithm. This node can be temporarily retained only when the score of the node in the data graph is greater than the threshold, otherwise the node and its neighbors need to be deleted.

As described above for the threshold pruning algorithm, the threshold pruning algorithm characterizes the nodes in the data graph. The total score \(S\) of the node includes the latitude–longitude score \(S_L\) and the preference score \(S_p\). In order to clearly understand the score range and distribution of the two parts score of the node in the data graph, we perform the threshold pruning algorithm without setting the maximum number of sensors. Latitude–longitude score range and preference score range are shown in Figure 9(a), and the distribution of total scores in the data graph is shown in Figure 9(b). As we can see from Figure 9(a), the node’s latitude–longitude score range is from 0.235 to 0.412, and the node’s preference score range is from 0.013 to 0.814. The node’s preference score range is

|                | 5     | 10    | 15    | 20    |
|----------------|-------|-------|-------|-------|
| TOPSIS         | 0.204 | 0.457 | 0.787 | 1.577 |
| ES             | 9.962 | 10.010| 10.999| 12.190|
| Dynamic skyline| 4.997 | 9.375 | 14.094| 19.543|
| An efficient preference-based sensor selection algorithm | 0.196 | 0.402 | 0.568 | 0.814 |
| Our(now)       | 0.058 | 0.059 | 0.122 | 0.124 |

Table 5. The detail information of the response time of different data sets.
much larger than the node’s latitude–longitude score range. Moreover, the latitude–longitude score of the node does not change much with the total score increasing while the node’s preference score shows an increasing state with the increase in the total score. Figure 9(a) illustrates that preference score largely determines the total score. It can be seen from Figure 9(b) that the total scores of nodes in the data graph generally conform to the normal distribution. When the total score is greater than 0.9, the number of sensors after pruning is less than 3000. Combining the above information, the higher the preference score of nodes in the data graph is, the higher the total score of nodes will be and the fewer sensors will be. Therefore, we assume that the threshold $K$ is 0.9 for the follow experiments to obtain the maximum number of sensors, and measured indicators are also precision and response time.

We assume the threshold is 0.9, and we conduct the maximum number of sensors $N$ selection experiment on the data set whose size is 100,000. In the experiment, we use 50 as the base unit, and the maximum number of sensors $N$ ranges from 50 to 300. The precision (shown in Figure 10(a)) and the response time (shown in Figure 10(b)) are calculated when our model recommends the top three sensors for user. As shown in Figure 10(a), when $N$ is less than 250, the precision of our model increases with $N$ increasing. When $N$ is greater than 250, the precision of the algorithm is stable at 94.87%. As shown in Figure 10(b), the response time of our model is stable and the response time fluctuates in a small range, which is around 0.112 s. Based on the above analysis, when we set the threshold to 0.9 and set the maximum number of sensors to 250, the precision and the response time of our model remain stable. Therefore, the maximum number of sensors we selected is 250.
In the above experiment, we have determined that the maximum number of sensors is 250, and we make a threshold selection experiment based on this result. In order to eliminate the influence of the size of the data set on the threshold selection experiments, we use four data sets in this experiment. In addition, total score of nodes in the data graph computed by our threshold pruning algorithm is less than 1.2, so we use 0.01 as the basic unit and threshold varies from 0 to 1.2. The precision of our model (as shown in Figure 11(a)) and the response time (as shown in Figure 11(b)) show as follows. As shown in Figure 11(a), the precision of our model is more stable as the size of the data set becomes larger, and the precision curve almost overlaps when the size of the data set is 150,000 and 200,000. When the threshold of the four data sets is less than 0.92, the preference score is less than the latitude and longitude score, which shows in Figure 9(a). And this keeps precision unchanged with threshold less than 0.92. From Figure 11(a), we can know that the precision is 79.8% and 92.3% separately when the size of data set is 50,000 and 100,000. And the precision is 92.3% when the size of data set is 150,000 or 200,000. The precision of the four data sets changes when the threshold is greater than 0.92. When the data set size is 100,000, 150,000, and 200,000, stability occurs when the threshold is greater than 0.95. At this time, the preference score is greater than the latitude–longitude score, which makes the precision higher with an accuracy of 94.87%. When the threshold is greater than 1, the precision of the four data sets greatly reduce, because the number of sensors selected at this time is less than 500 (as shown in Figure 9(a)) and the coverage area of sensor is too small. As shown in Figure 11(b), the number of sensors that are filtered out is smaller and smaller with the increase in the threshold on the four data sets; therefore, the response time of the algorithm also decreases. Based on the above analysis, we choose a threshold of 0.95, at which time the precision remains stable and the response time is very low.

After the above two experiments, we finally set the maximum number of sensors to 250 and set the threshold to 0.95. Based on the above settings, we compared the proposed graph-based sensor recommendation model with the other four algorithms in terms of the precision and the response time. In order to reduce the experimental error of equipment or accident, we will conduct experiments to obtain the average value of precision for five times and response time as the final value. During experiments, in order to reduce the uncontrollable error of human operation, we use quantitative indicators when evaluating user satisfaction with the recommended sensor, which not only avoids human operational error but also improves the efficiency of experiments. Below we evaluate the efficiency of the proposed graph-based sensor recommendation model and the other four sensor recommendation algorithms in terms of precision and response time.

The precision comparison of sensor selection algorithm. We evaluate the precision of our proposed graph-based sensor recommendation model and the other four recommendation algorithms on four data sets, as shown in Figure 12. The precision of five algorithms has different change tendency from recommend one sensor to recommending ten sensors for user under different data sets, but the precision of the algorithm between them is relatively stable. Among the five algorithms, the graph-based sensor recommendation model has the highest precision while the TOPSIS algorithm has the worst precision. The remaining three sensor recommendation algorithms are ranked as an efficient preference-based sensor selection algorithm, ES algorithm, and dynamic...
skyline algorithm. Taking the data set of 150,000 as an example, the precision difference between of the five sensor recommendation algorithms is specifically analyzed from recommend one sensor to recommending three sensors for user. The precision of our proposed graph-based sensor recommendation model, an efficient preference-based sensor selection algorithm, ES algorithm, dynamic skyline algorithm, and TOPSIS algorithm is, respectively, 98.29%, 81.26%, 76.99%, 63.88%, and 46.44%.

The TOPSIS algorithm is not appropriate for recommendation under large-scale data set. As the size of data set increasing, the precision of the TOPSIS algorithm decreases, and the TOPSIS algorithm is unstable. As shown in Figure 12, the precision of TOPSIS algorithm is 50.00%, 46.15%, 42.30%, and 38.46% under the four data sets when recommending a sensor to user. And the precision of the TOPSIS algorithm first increases and then decreases when the size of data sets is 150,000 and 200,000, which indicates the TOPSIS algorithm has poor ability to characterize sensors under large-scale data sets. The precision of other four algorithms including graph-based sensor recommendation model, an efficient preference-based sensor selection algorithm, ES algorithm, and dynamic skyline algorithm increases with the size of the data set increasing. Under the large-scale data sets, these four algorithms are under a strong ability to represent sensor options. Among the four algorithms, the precision change trend of the graph-based sensor recommendation model, an efficient preference-based sensor selection algorithm, and the ES algorithm is reduced from recommending one sensor to recommend ten sensors for user. In this way, when the above algorithms recommend sensors

![Figure 12. The precision in different data sets: (a) Dataset1, (b) Dataset2, (c) Dataset3, and (d) Dataset4.](image-url)
for users, users can find satisfactory sensors among as few sensors as possible. In contrast, although the dynamic skyline algorithm is suitable for recommendation in large-scale data set, it cannot recommend satisfactory sensor options for users when there are few recommended sensors.

Response time comparison of sensor selection algorithm. The response time is the time interval from the user submitting the request on the sensor recommendation model to return recommendation results. In the experiment to obtain the response time of the sensor recommendation algorithm, we calculated the average of the response time of experiments for each algorithm on four different data sets, and the experimental results are shown in Figure 13. Because the response time of the five sensor recommendation algorithms is relatively large, the table is listed for more detail. The performance of these five algorithms on different data sets is relatively stable and the response time of these five algorithms is ranked from low to high as the graph-based sensor recommendation model, an efficient preference-based sensor selection algorithm, TOPSIS algorithm, ES algorithm, and dynamic skyline algorithm. We can analyze the above results from the perspective of algorithm time complexity. The time complexity of the graph-based sensor recommendation model is $O(n \log mn)$ ($n \leq 250$). The time complexity of an efficient preference-based sensor selection algorithm is $O(n \log mn)$ ($n \leq 900$). The time complexity of TOPSIS algorithm is $O(mn)$. The time complexity of ES algorithm is $O(mn^3)$ ($n \leq 2095$). The time complexity of dynamic skyline algorithm is $O(mn^2)$.

Stability of different sensor recommendation algorithms in response time is measured by the difference of response time between the data set of 50,000 and the data set of 200,000, are shown in Table 5. The proposed graph-based sensor recommendation model is the best while dynamic skyline algorithm is worst. Specifically, the difference in response time of the graph-based sensor recommendation model is 0.0605 s. The difference in response time of an efficient preference-based sensor selection algorithm is 0.618 s. The difference in response time of the TOPSIS algorithm is 1.373 s. The difference in response time of the ES algorithm is 2.223 s, and the difference in response time of the dynamic skyline algorithm is 14.546 s.

Conclusion

In order to deal with the heterogeneity of devices in the IoT, scholars proposed a semantic sensor network, which has made the development of the IoT a giant step forward. However, sensor recommendation based on the semantic sensor network is still a problem. Some existing research mainly relies on SPARQL to retrieve the semantic database to recommend sensors for users, where SPARQL matches data in the database without considering the user’s preferences of sensor’s attributes, which leads to this type of sensor recommendation model fail to meet the needs of users. Therefore, we propose a graph-based sensor recommendation model. For the first time, the weighted search graph and the weighted data graph are employed in sensor recommendation of semantic sensor network, where this maximizes the use of RDF semantic graph. In order to narrow the scope of sensor search, we propose a threshold pruning algorithm. Experiments show that threshold pruning algorithm can efficiently characterize the nodes in the data graph. Next, we adopt the improved fast non-dominated sorting algorithm to obtain the local optimal solutions in the sensor data set after pruning, and we use the SAW algorithm to sort the local optimal solutions. Experiments show that our proposed graph-based sensor recommendation model is superior to other algorithms. In the following research, we intend to further parse the semantic data dynamically and expand the existing model.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported, in part, by the National Natural Science Foundation of China under grants 61802343 and 62072402; in part, by the Zhejiang Provincial Natural
Reference

1. Zheng Z, Tao Y, Chen Y, et al. An efficient preference-based sensor selection method in Internet of Things. *IEEE Access* 2019; 7: 168536–168547.

2. Evans D. The Internet of Things: how the next evolution of the Internet is changing everything. CISCO White paper. CISCO, San Jose, CA, 2011.

3. Huang YH and Li GY. The semantic Web of Things: strategy for Internet of Things intrinsic contradiction. *Appl Res Comput* 2010; 27: 4087–4090.

4. Zhang X, Zhao Y and Liu W. Transforming sensor data to RDF based on SSN ontology. *Adv Sci Technol Lett* 2015; 81: 95–98.

5. Bermudezedo M, Elsaleh T, Barnaghi P, et al. IoT-Lite: a lightweight semantic model for the Internet of Things. In: 2016 international IEEE conferences on ubiquitous intelligence & computing, advanced and trusted computing, scalable computing and communications, cloud and big data computing, Internet of people, and smart world congress, 2016, pp.90–97. https://ieeexplore.ieee.org/document/7816831

6. Rasyid MU, Sayfudin A, Basofi A, et al. Development of semantic sensor web for monitoring environment conditions. In: 2016 international seminar on intelligent technology and its applications (ISITITA), 2016, pp.607–612, https://ieeexplore.ieee.org/document/7828729

7. Zhou Y, De S, Wang W, et al. Search techniques for the Web of Things: a taxonomy and survey. *Sensors* 2016; 16(5): 600.

8. Gong H, Shi Y, Li G, et al. Context-aware sensor search framework in semantic Web of Things. In: 2016 8th international conference on intelligent human-machine systems and cybernetics (IHMSC), 2016, pp.94–98, https://ieeexplore.ieee.org/document/7783568

9. Perera C, Zaslavsky A, Christen P, et al. Context-aware sensor search, selection and ranking model for Internet of Things middleware. In: Proceedings of the IEEE 14th international conference on mobile data management (MDM), Milan, 3–6 June 2013, pp.314–322. Piscataway, NJ: IEEE.

10. Chirila S, Lemmaru C, Dinsoreanu M, et al. Semantic-based IoT device discovery and recommendation mechanism. In: 2016 IEEE 12th international conference on intelligent computer communication and processing (ICCPP), 2016, pp.111–116, https://ieeexplore.ieee.org/document/7737131

11. Mecibah R, Djamaa B, Yachir A, et al. A scalable semantic resource discovery architecture for the Internet of Things. In: 2018 international conference on computer science and its applications (CSA 2018), 2018, pp.37–47, https://link.springer.com/chapter/10.1007/978-3-319-98352-3_5

12. Gomes P, Cavalcante E, Batista T, et al. A semantic-based discovery service for the Internet of Things. *J Internet Serv Appl* 2019; 10(1): 10.

13. Neha SS. Vector method for ranking of sensors in IoT. In: 2016 international conference on inventive computation technologies (ICICT), 2016, pp.1–5, https://ieeexplore.ieee.org/document/7830231

14. Nunes LH, Estrella JC, Perera C, et al. The elimination-selection based algorithm for efficient resource discovery in Internet of Things environments. In: 2018 15th IEEE annual consumer communications & networking conference (CCNC), 2018, pp.1–7, https://ieeexplore.ieee.org/document/8319280

15. Tzeng GH and Huang JJ. Multiple attribute decision making. Boca Raton, FL: Chapman and Hall/CRC, 2011.

16. Kertiou I, Benharzallah S, Kahloul L, et al. A dynamic skyline technique for a context-aware selection of the best sensors in an IoT architecture. *Ad Hoc Netw* 2018; 81: 183–196.

17. Nithya KK, Prathap MV and Babu KRR. Cluster oriented sensor selection for context-aware Internet of Things applications. In: International conference on intelligent data communication technologies and Internet of Things (ICICI) 2018, 2018, pp.981–988, https://link.springer.com/chapter/10.1007/978-3-030-03146-6_113

18. Bharti S, Pattanaik KK and Bellavista P. Value of information based sensor ranking for efficient sensor service allocation in service oriented wireless sensor networks. *IEEE Trans Emerg Topics Comput* 2019; 9: 823–838.

19. Pattar S, Buyya R, Venugopal KR, et al. Searching for the IoT resources: fundamentals, requirements, comprehensive review, and future directions. *IEEE Commun Surv TUTOR* 2018; 20(3): 2101–2132.

20. Truong C, Romer K, Chen K, et al. Fuzzy-based sensor search in the Web of Things. In: 2012 3rd IEEE international conference on the Internet of Things, 2012, pp.127–134, https://ieeexplore.ieee.org/document/6402314

21. Ostermaier B, Römer K, Mattern F, et al. A real-time search engine for the Web of Things. In: 2010 Internet of Things (IOT), 2010, pp.1–8, https://ieeexplore.ieee.org/document/5678450

22. Truong C and Romer KU. Content-based sensor search for the Web of Things. In: 2013 IEEE global communications conference (GLOBECOM), 2013, pp.2654–2660, https://ieeexplore.ieee.org/abstract/document/6831475

23. Zhang P, Liu Y, Wu F, et al. Low-overhead and high-precision prediction model for content-based sensor search in the Internet of Things. *IEEE Commun Lett* 2016; 20(4): 720–723.

24. Zhang P, Liu Y, Wu F, et al. Matching state estimation scheme for content-based sensor search in the Web of Things. *Int J Distrib Sens Netw* 2015; 11(11): 1–12.

25. Vasilev V, Iliev G, Poulkov V, et al. Optimization of wireless node discovery in an IoT network. In: 2015 IEEE Globecom workshops (GC Wkshps), 2015, pp.1–5, https://ieeexplore.ieee.org/document/7414153
26. Tang J and Zhou Z. Searching the Internet of Things using coding enabled index technology. In: *GPC 2018: green, pervasive, and cloud computing*, 2018, pp.79–91, https://link.springer.com/chapter/10.1007/978-3-030-15093-8_6

27. Chen Y, Zhou M, Zheng Z, et al. Time-aware smart object recommendation in social Internet of Things. *IEEE Internet Things J* 2020; 7(3): 2014–2027.

28. Chen Y, Zhang J, Guo M, et al. Learning user preference from heterogeneous information for store-type recommendation. *IEEE Trans Serv Comput* 2020; 13: 1100–1114.

29. Mietz R and Romer KU. Exploiting correlations for efficient content-based sensor search. In: *2011 IEEE sensors*, 2011, pp.187–190, https://ieeexplore.ieee.org/document/6127082

30. Zhang W, Kumar M, Liu J, et al. Multi-parameter online measurement IoT system based on BP neural network algorithm. *Neural Comput Appl* 2019; 31(12): 8147–8155.

31. Li H, Ota K, Dong M, et al. Learning IoT in edge: deep learning for the Internet of Things with edge computing. *IEEE Netw* 2018; 32(1): 96–101.

32. Ullmann JR. An Algorithm for subgraph isomorphism. *J ACM* 1976; 23(1): 31–42.

33. Nunes LH, Estrella JC, Perera C, et al. Multi-criteria IoT resource discovery: a comparative analysis. *Softw Pract Exp* 2016; 47(10): 1325–1341.

34. Memariani A, Amini A, Alnezhad A, et al. Sensitivity analysis of simple additive weighting method (SAW): the results of change in the weight of one attribute on the final ranking of alternatives. *J Optim Ind Eng* 2009; 4(4): 13–18.