QLEC: A Machine-Learning-Based Energy-Efficient Clustering Algorithm to Prolong Network Lifespan for IoT in High-Dimensional Space

Ke Li[†]
Shanghai Jiao Tong University
Shanghai, China
like19971019@sjtu.edu.cn

Haowei Huang[†]
Shanghai Jiao Tong University
Shanghai, China
H1270927224@sjtu.edu.cn

Xiaofeng Gao[*†]
Shanghai Jiao Tong University
Shanghai, China
gao-xf@cs.sjtu.edu.cn

Fan Wu[†]
Shanghai Jiao Tong University
Shanghai, China
fwu@cs.sjtu.edu.cn

Guihai Chen[†]
Shanghai Jiao Tong University
Shanghai, China
gchen@cs.sjtu.edu.cn

ABSTRACT

With the emergence of Internet of Things (IoT), many battery-operated sensors are deployed in different applications to collect, process, and analyze useful information. In these applications, sensors are often grouped into different clusters to support higher scalability and better data aggregation. Clustering based on energy distribution among nodes can reduce energy consumption and prolong the network lifespan. In our paper, we propose a machine-learning-based energy-efficient clustering algorithm named QLEC to select cluster heads in high-dimensional space and help non-cluster-head nodes route packets. QLEC first selects cluster heads based on their residual energy through successive rounds. Besides, we prove the optimal cluster number in a high-dimensional wireless network and adopt it in our QLEC algorithm. Furthermore, Q-learning method is utilized to maximize residual energy of the network while routing packets from sensors to the base station (BS). The energy-efficient clustering problem in high dimensional space can be formed as an NP-Complete problem and QLEC is proved to solve it in the running time $O(kX)$, where k is the cluster number and X is the number of updates Q-learning needs to converge. Extensive simulations and experiments based on a large-scale dataset show that the proposed scheme outperforms a newly proposed FCM-based algorithm and k-means clustering in terms of network lifespan, packet delivery rate, and transmission latency. To the best of our knowledge, this is the first work adopting Q-learning method in clustering problems in high-dimensional space.

[*]Xiaofeng Gao is the corresponding author.
[†]Ke Li, Haowei Huang, Xiaofeng Gao, Fan Wu and Guihai Chen are from Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University

ICPP 2019, August 5–8, 2019, Kyoto, Japan

ACM ISBN 978-1-4503-6295-5/19/08…$15.00
https://doi.org/10.1145/3337821.3337926

CCS CONCEPTS

• **Theory of computation** → **Facility location and clustering**; • **Hardware** → *Power estimation and optimization*; • **Computing methodologies** → Reinforcement learning;

KEYWORDS

IoT, Energy-Efficient Clustering, Q-learning, Lifespan-Extended Network, High-Dimensional Space

ACM Reference Format:
Ke Li, Haowei Huang, Xiaofeng Gao, Fan Wu, and Guihai Chen. 2019. QLEC: A Machine-Learning-Based Energy-Efficient Clustering Algorithm to Prolong Network Lifespan for IoT in High-Dimensional Space. In *48th International Conference on Parallel Processing (ICPP 2019), August 5–8, 2019, Kyoto, Japan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3337821.3337926

1 INTRODUCTION

Internet of Things (IoT) is a new technology paradigm envisioned as a global network of machines and devices capable of interacting with each other. The main objective of the IoT is a common aspect in our everyday life and depends on the behavior of users. There are two technologies traditionally considered as key enablers for the IoT paradigm: the radio-frequency identification (RFID) and the wireless sensor networks (WSN). While RFID is well established for low-cost identification and tracking [18], WSNs bring IoT applications richer capabilities for both sensing and actuation. IoT WSN systems have wide applications in air monitoring, water monitoring, and forest monitoring [16].

However, WSNs suffer from many issues such as coverage, security, energy-efficiency, localization, and etc. Among these issues, energy-efficiency is the critical one, as sensor nodes are battery-operated [9]. Moreover, it is established that most of energy is consumed in the process of data transmission and reception. Therefore, energy-saving routing protocols are required.

Energy-efficient protocols based on clustering can be designed to adapt to various characteristics of wireless sensor networks, in order to prolong the lifetime of the network. In WSN, nodes around the base station (BS) will function as the relay for nodes which

are far away from BS by using clustering algorithms to extend the lifetime of sensors. A clustering technique transforms the global communication into the local communication for saving energy.

Due to various energy conditions of different sensors, a traditional cluster-based routing protocol is not good enough to support large wireless sensor networks. To make more progress, the hierarchical cluster-based protocol is proposed. In this protocol, high energy nodes are selected for processing and sending information, while low energy nodes are used to sense and send data to the cluster head (CH). In this way, energy is used more efficiently and the lifespan of networks can be prolonged. Distributed Energy Efficient Clustering (DEEC) is one of the cluster-based hierarchical protocol used especially for multilevel communication in a heterogeneous routing environment.

In DEEC protocol, the selection of cluster heads is based on the ratio between the residual energy of each node and the average energy of the network [11] so that high energy nodes are more probable to be chosen as cluster heads than nodes with lower initial and residual energy. Thus, DEEC protocol is more stable than other heterogeneous protocols. However, existing works rarely focus on the integration of DEEC clustering problems and the reinforcement learning method which is designed to make choices according to the reward function provided by the environment.

In most cases, people assume that wireless nodes are on a two-dimensional plane. However, in many environment like mountainous areas or underwater regions, node deployment is often not flat, resulting in high dimensional space. Given that communication between nodes in high dimensional space is more complicated and restricted with the environment, clustering algorithms in high dimensional space tend to have higher time complexity. What is more, many optimal algorithms in two-dimensional space can not directly apply to high-dimensional situation. Therefore, it is challenging to design an efficient clustering algorithm in high dimensional space.

In this paper, we focus on energy-efficient clustering in a 3-dimensional wireless sensor network of IoT by adopting the Q-learning method to make clustering choices for sensors. In order to make energy consumption uniformly distributed to extend the lifespan, we combine DEEC algorithm with Q-learning method and design a novel clustering algorithm called QLEC algorithm with two phases: *Cluster Head Selection* and *Data Transmission*. In *Cluster Head Selection Phase*, the improved DEEC algorithm is adopted to select cluster heads for a network through successive rounds. In *Data Transmission Phase*, non-cluster-head nodes dynamically choose cluster heads for packet transmission and data fusion with Q-learning method as to avoid direct communication with BS. This problem is proved to be NP-Complete and QLEC can solve it in the running time $O(kX)$, where k is the cluster number and X is the number of updates Q-learning needs to converge. Our contributions in this paper is summarized as follows.

- Since energy-efficiency is critical in wireless sensor networks, we provide some improvements to DEEC algorithm to restrict the minimum energy that nodes need to be selected as cluster heads. We also prove the optimal cluster number in a 3-dimensional wireless network. To the best of our knowledge, this is the first work demonstrating the result.
- Based on the cluster heads already selected, we adopt Q-learning method to help non-cluster-head nodes dynamically choose the cluster head for packet transmission and data aggregation. Our QLEC is proved to solve the problem with an acceptable running time.
- Performance evaluation results validate the efficiency of our algorithm, which outperforms a newly proposed FCM-based algorithm and classic k-means clustering in terms of network lifespan, packet delivery rate, and transmission latency. We also conduct experiments based on a large-scale dataset to prove the efficiency of QLEC in the real world.

The rest of our paper is organized as follows. In Section 2, we describe the related works in detail. Section 3 provides the improved DEEC algorithm, the Q-learning method, and the description of our problem. Motivated by the two algorithms above, we develop our QLEC algorithm and analyze it theoretically in Section 4. The performance evaluation is illustrated in Section 5 and finally we come to an conclusion of the paper in Section 6.

2 RELATED WORK

A number of different clustering methods are studied in previous works, such as partitioning clustering, hierarchical clustering, graph-based clustering, and so on. Recent years, many energy-efficient clustering methods based on routing protocols have been proposed to save energy and prolong the network lifespan.

Low-Energy Adaptive Clustering Hierarchy (LEACH) is an self-organizing, adaptive clustering protocol that uses randomization-based probability to distribute the energy load equally to sensor nodes in the network [5]. It is one of the most common and classic energy-efficient protocols. However, LEACH does not take residual energy of sensors into consideration and may lead to unevenly distributed cluster heads. Hence, many improved algorithms have been studied to solve these problems. Loscri et al. proposed a TL-LEACH protocol in [10] to build a two-level hierarchy of cluster heads in WSN. ED-LEACH protocol in [13] studied the Euclidean distance between sensors to improve location of cluster heads in a region. DEEC [11] is also based on LEACH protocol and designed for heterogeneous wireless sensor networks. It uses residual energy of nodes to choose the cluster heads and has been studied widely.

Fuzzy C-Means (FCM) clustering algorithm employs the concept of maximizing residual energy when choosing cluster heads as well. An FCM-based scheme in [14] divides the WSN into different hierarchies based on the distance to the BS and a dynamic multi-hop routing algorithm is designed. There are also some energy-efficient clustering algorithms based on distributed approaches [15, 17].

Some recent works focus on using Q-learning method to design routing protocols in WSN. QELAR algorithm in [6] took residual energy of a node, as well as the energy distributed among its neighbour nodes as the reward function to maximize residual energy of the network when routing packets, following the Q-learning-based approach. Basagni et al. proposed HyDRO in [2] to select routing relay in a harvesting-aware underwater WSN using Q-learning.

However, with all the works about clustering algorithms and Q-learning utilization in wireless networks introduced above, none of them have adopted reinforcement learning algorithms to solve the clustering problem in IoT WSN.

3 SYSTEM MODEL AND PROBLEM DESCRIPTION

In this section, we first give a brief introduction of Distributed Energy Efficient Clustering (DEEC) algorithm with some improvements. Then we demonstrate the proof of the optimal cluster number in 3-dimensional networks. Afterwards, we analyze the Q-learning method. Finally, we describe the clustering problem in high dimensional space in detail.

3.1 An Improved DEEC Algorithm

DEEC algorithm periodically selects cluster heads among all nodes in a wireless network according to their residual energy to prolong the lifetime of the network.

The algorithm can be divided into two phases. In the clustering phase, the cluster heads are selected based on the ratio between the energy of nodes and the average energy of the network. Each cluster head represents a cluster for the network. Nodes that are not selected as cluster heads dynamically choose the nearest cluster head as its routing node and join in its cluster. Then for the stable phase, nodes in a cluster send their sensing data to their cluster head for data fusion. Finally, cluster heads directly transmit the aggregated data to the base station. In DEEC, the energy consumption of communication between sensor nodes and the sink is minimized and nodes with more energy are more likely to be chosen as cluster heads so that the chance for any sensor in the network to be out of power is minimized as well.

Figure 1 depicts a simple 3-dimensional network structure after implementing DEEC clustering algorithm. We assume N nodes are randomly distributed in a $M \times M \times M$ cube. The green node in the center is the sink node while the black ones are the cluster heads selected with DEEC algorithm. If non-cluster-head nodes like gray ones intend to communicate with the sink node, they are supposed to transmit data to their own cluster head for data fusion in the first place. After that, the cluster head is responsible for the transmission of all the data collected from its own cluster.

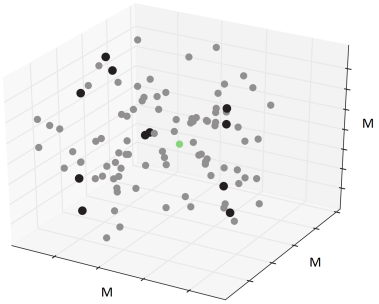


Figure 1: Network structure of clustering

As a result of the mobility of wireless sensor networks, DEEC algorithm is conducted through successive rounds to dynamically select nodes with the most residual energy to serve as cluster heads. For a given network with N nodes b_i $(i = 1, 2, ...N)$ randomly distributed in a $M \times M \times M$ cube, let n_i denote the number of rounds between b_i being selected as a cluster head twice, and it is referred to as the rotating epoch. Thus, $p_i = 1 / n_i$ stands for the average probability for b_i to be a cluster head during n_i rounds.

Energy is usually not distributed evenly among nodes in a network. Thus, nodes with more energy should be given more probability to be chosen as cluster heads so that the energy consumption of the network is reasonable. DEEC determines whether a node b_i can be selected as the cluster head according to its residual energy at round r, which is denoted as $E_i(r)$. Then the probability p_i is given as

$$p_i = p_{opt}[1 - \frac{\overline{E}(r) - E_i(r)}{\overline{E}(r)}] = p_{opt}\frac{E_i(r)}{\overline{E}(r)} \tag{1}$$

where p_{opt} is the optimal probability and $\overline{E}(r)$ is the average energy of the network at round r. In order to reduce the time complexity of the algorithm, we can give an estimate of $\overline{E}(r)$ at round r with the following equation:

$$\overline{E}(r) = \frac{1}{N}E_{initial}(1 - \frac{r}{R}) \tag{2}$$

where $E_{initial}$ is the initial energy of the whole network while R denotes total rounds of the lifespan of the network. [7] gives a detailed description of how to estimate R from energy consumption model.

Equation (1) guarantees that the average number of cluster heads in the network per round is equal to

$$\sum_{i=1}^{N} p_i = \sum_{i=1}^{N} p_{opt}\frac{E_i(r)}{\overline{E}(r)} = Np_{opt} = k_{opt}$$

It is the optimal number of cluster heads we want to achieve.

With the probability of each node to be selected as a cluster head at round r, a threshold $T(b_i)$ is adopted to determine whether a node can become a cluster head:

$$T(b_i) = \begin{cases} \dfrac{p_i}{1 - p_i(r \mod \frac{1}{p_i})} & \text{if } b_i \in C \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where C is the *candidate nodes set* consisting of nodes which are qualified to be cluster heads at round r. If a node b_i has not been chosen as a cluster head in the recent n_i rounds, then it can be added into the *candidate nodes set* C. At each round, if a node b_i finds that it is eligible to be a cluster head, it will compute its threshold value $T(b_i)$ and generate a random number between 0 and 1. If the number is smaller than $T(b_i)$, the node will be selected as the cluster head.

The DEEC algorithm above is a randomized algorithm because it produces a random number to decide whether a node can be a cluster head so that it requires less time and space complexity to solve the problem. However, it still has some problems so we would like to propose some improvements.

Firstly, even though nodes with more energy are given a bigger chance to become a cluster head in DEEC algorithm, nodes that are about to be out of power are still possible to be chosen. It may accelerate the process of running out of battery for these nodes, shortening the lifespan of the network. Hence, we define a energy threshold $E_{i,th}$ to impose a restriction on the minimum energy of

a node to be selected as the cluster head. The threshold should be related to the initial energy of a node $E_{i,initial}$ and decrease with the round r growing:

$$E_{i,th}(r) = [1 - (\frac{r}{R})^2] \times E_{i,initial} \tag{4}$$

With the energy threshold defined, nodes which are selected to be cluster heads at round r with DEEC algorithm must have more energy than the threshold, i.e. $E_i(r) > E_{i,th}(r)$ to truly become the cluster heads for this round. If a node possesses less energy than needed, the improved DEEC algorithm will choose another node up to the demand to replace it.

Another drawback of traditional DEEC algorithm is that the number of cluster heads per round is uncertain. In reality, too many clusters of a network is a waste of resources while too few will aggravate the burden of the base station. It is very important to set a certain cluster number for each round with specific cluster coverage area to better allocate resources for the network. If we choose k cluster heads per round, then we can define the cluster coverage radius of each cluster d_c as

$$d_c = \sqrt[3]{\frac{3}{4\pi k}} M \tag{5}$$

With the radius d_c defined, each node which is selected as a cluster head will immediately broadcast a message to the nodes in its cluster coverage range indicating that it has been chosen as a cluster head and how much energy it has. Every node receiving this kind of message will compare the energy indicated in the message with its own energy. If it has less energy than the cluster head, it will exit the competition of being a cluster head in this round. Otherwise, it will remain as the candidates to become a new cluster head. This mechanism guarantees that no redundant nodes are selected as cluster heads to save the overall energy resources of the network.

3.2 Optimal Cluster Number in 3-D Wireless Network

In the last section, we selected k cluster heads for each round to determine the cluster coverage radius of each cluster. Many works have been done to determine the optimal cluster number in 2-dimensional wireless networks [4, 11]. However, there has not been a paper explaining how to determine it in a 3-dimensional wireless network, so we are going to prove it in this subsection. We assume that each non-cluster-head node sends L bits data to the cluster each round so that the total energy dissipated in the network during a round is given by *first-order radio model* [4]:

$$E_r = L(2NE_{elec} + NE_{DA} + k\epsilon_{mp} d_{toBS}^4 + N\epsilon_{fs} d_{toCH}^2) \tag{6}$$

where E_{elec} is the energy dissipated per bit to run the transmitter or the receiver circuit, E_{DA} is the data aggregation cost expended in the cluster-heads, and ϵ_{fs} and ϵ_{mp} are two constants derived from free space and multi-path fading channel models. ϵ_{fs} is the free space constant and usually set as $10pJ/bit/m^2$ while ϵ_{mp} is the multi-path constant and usually set as $0.0013pJ/bit/m^4$. d_{toBS} is the average distance between the cluster heads and the base station (BS). According to [1], d_{toBS} can be approximated by the average distance between the nodes and BS. d_{toCH} is the average distance between the cluster nodes and the cluster head.

Lemma 1. The average distance d_{toCH} of nodes is determined by the cluster number with $d_{toCH}^2 = \frac{4\pi}{5}(\frac{3}{4\pi})^{\frac{5}{3}} \frac{M^2}{k^{\frac{2}{3}}}$.

Proof. The volume occupied by each cluster is approximately M^3 / k. It is assumed that cluster nodes are uniformly distributed in the area of a ball centered on the cluster head so the density of nodes in this area is $\rho = (1 / (M^3 / k))$. Thus, the expected squared distance from cluster nodes to the cluster head is given by

$$\begin{aligned} E\{d_{toCH}^2\} &= \rho \iiint (x^2 + y^2 + z^2) dxdydz \\ &= \rho \iiint r^4 \sin\phi dr d\phi d\theta \\ &= \rho \int_0^{d_c} r^4 dr \int_0^{2\pi} d\theta \int_0^{\pi} \sin\phi d\phi \end{aligned} \tag{7}$$

where $E\{\cdot\}$ denotes the expected value. Substitute d_c and ρ into Equation (7), we can get the average d_{toCH}^2. □

Theorem 1. The optimal cluster number in a 3-dimensional wireless network is $k_{opt} = \frac{3}{4\pi}(\frac{8\pi N\epsilon_{fs}}{15\epsilon_{mp}})^{\frac{3}{5}} \frac{M^{\frac{6}{5}}}{d_{toBS}^{\frac{12}{5}}}$.

Proof. Substitute Lemma 1 into Equation (6) and set the derivative of E_r with respect to k to 0, and then we can obtain the optimal cluster number k_{opt}. □

3.3 Q-Learning Algorithm

After selecting cluster heads with improved DEEC algorithm in Subsection 3.1, then the rest nodes need to choose a cluster head and start to transmit sensing data to the cluster head for data fusion. Nodes choosing the same cluster head will form one cluster together with the cluster head. In our QLEC clustering algorithm, we adopt the Q-learning method to determine which cluster a node belongs to. Thus, we will briefly introduce the Q-learning algorithm below.

Q-learning algorithm is an off-policy temporal difference approach that can yield near-optimal policies for reinforcement learning without much computations and the underlying model [6]. The reinforcement learning problem is meant to be a straightforward framing of the problem of learning from interaction to achieve a goal [12]. Figure 2 depicts agent-environment interaction in reinforcement learning. The learner or the decision-maker is called the *agent*. Anything outside the agent is called the *environment*. It continually interacts with the agent and responds to its actions. Besides, the environment presents new states to the agent for the next step. Meanwhile, the environment provides rewards, special numerical values that the agent tries to maximize over time, to the agent policy decision mechanism to determine the next action of the agent.

In detail, the agent and environment interacts at a sequence of discrete time $t = 0, 1, 2, \ldots$. At each time step t, the environment provides the agent with its current *state* $S_t \in S$. S is the set of all the possible states that the environment may belong to. The agent then takes *action* $A_t \in A(S_t)$ under some agent *policy* π_t based on

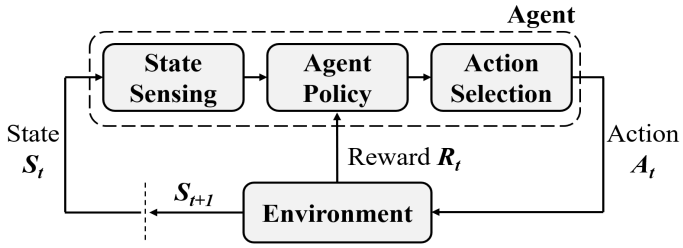


Figure 2: The agent-environment interaction in reinforcement learning

the state S_t. $A(S_t)$ represents the set of actions the agent can take in state S_t. Finally, the action A_t stimulates the environment to generate a reward R_t back to the agent and transform itself into a new state S_{t+1}. The agent adapts the policy on the basis of R_t and receives the new state S_{t+1}, entering the next step.

A reinforcement learning task that satisfies the Markov property is called a *Markov Decision Process* (MDP). If the state and action spaces are finite, then it is called a *finite Markov Decision Process* (*finite* MDP) [12].

A typical *finite* MDP is defined by its *state-action pair*. Given the state s and the action a, the *state-transition probability* of going to state s' from state s after taking action a is denoted as

$$P^a_{ss'} = Pr\{S_{t+1} = s' | S_t = s, A_t = a\} \tag{8}$$

and the expected rewards for *state–action–next-state* triples,

$$R^a_{ss'} = E\{R_t | S_t = s, A_t = a, S_{t+1} = s'\} \tag{9}$$

With state-transition probability and expected rewards defined in Equation (8) and Equation (9), we can then give the expression of R_t after taking an action A_t from the state S_t at time t:

$$R_t = \sum_{S_{t+1} \in S} P^{A_t}_{S_t S_{t+1}} R^{A_t}_{S_t S_{t+1}} \tag{10}$$

We have presented above that the goal of the agent in reinforcement learning is to maximize the accumulated rewards in the long term by adjusting its agent policy. Now we have the sequence of rewards $R_t, R_{t+1}, R_{t+2}, \ldots$ after taking action A_t. Thus we want to maximize the *expected return* G_t, which is a defined as a specific function of the reward sequence. Typically, the agent tries to select action A_t so that the *discounted reward* it receives in the future is maximized. In particular, the reward is defined as:

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k}$$

where γ is a parameter, $0 \leq \gamma \leq 1$, called the *discount rate*. γ is used to balance the influence current actions and future ones have on current values. Usually, recent actions have a greater impact on current values than future ones do. The typical value of γ is within $[0.5, 0.99]$ [6].

There are *value functions* estimation in the framework of almost all reinforcement learning algorithms. The function of states $V^\pi(s)$ is often used to estimate *how good* it is for the agent to be in a given state s under the policy π. It is defined as the expected total return in the future when starting in the state s and following the policy π thereafter. For MDPs, we can formally define $V^\pi(s)$ as:

$$V^\pi(s) = E_\pi\{G_t | S_t = s\} = E_\pi\{\sum_{k=0}^{\infty} \gamma^k R_{t+k} | S_t = s\} \tag{11}$$

where $E_\pi\{\cdot\}$ denotes the expected value under the policy π.

The same as $V^\pi(s)$ in Equation (11), the function of state-action pairs can also be defined to estimate *how good* it is to take a given action in a given state. The function is denoted $Q^\pi(s, a)$ and represents the value of taking action a in state s under the policy π:

$$\begin{aligned} Q^\pi(s,a) &= E_\pi\{G_t | S_t = s, A_t = a\} \\ &= E_\pi\{\sum_{k=0}^{\infty} \gamma^k R_{t+k} | S_t = s, A_t = a\} \end{aligned} \tag{12}$$

Equation (11) is called the *Bellman Equation* for V^π [12]. With definitions above, we can safely obtain the *optimal value* of a state under the *optimal policy* by solving the equation:

$$\begin{aligned} V^*(s) &= \max_\pi V^\pi(s) \\ &= \max_\pi E_\pi\{R_t + \sum_{k=1}^{\infty} \gamma^k R_{t+k} | S_t = s\} \\ &= \max_a [R_t + \gamma \sum_{S_{t+1} \in S} P^a_{S_t S_{t+1}} V^*(S_{t+1}) | S_t = s] \end{aligned} \tag{13}$$

Combining Equation (13) with Equation (12), we can get

$$V^*(s) = \max_a Q^*(s, a) \tag{14}$$

where $Q^*(s, a)$ is the special case of $Q^\pi(s, a)$ where the policy π is the *optimal policy*. Q-learning algorithm adopts Equation (14) to obtain near-optimal policies without too much computations and the underlying model [6].

Comparing Equation (13) and Equation (14), we can derive

$$\begin{aligned} Q^*(S_t, A_t) &= R_t + \gamma \sum_{S_{t+1} \in S} P^{A_t}_{S_t S_{t+1}} V^*(S_{t+1}) \\ &= R_t + \gamma \sum_{S_{t+1} \in S} P^{A_t}_{S_t S_{t+1}} \max_a Q^*(S_{t+1}, a) \end{aligned} \tag{15}$$

Equation (15) is used in our QLEC algorithm to update V values for each node, which will be illustrated in detail in Section 4, so that V can converge very fast. By adopting Q-learning algorithm to solve the reinforcement learning problem, nodes are capable of computing the Q values of all the actions based on their own knowledge to update V values rather than take real actions to all cluster heads to get the *optimal value* of a state.

3.4 Problem Description

Generally, given a 3-dimensional wireless network with N nodes distributed in a $M \times M \times M$ cube, the objective of energy-efficient clustering problem is to divide the whole space into k subspaces and maximize the average lifespan of nodes in each subspace. Since the lifespan of each node depends extremely on its residual energy $E_i(r)$ and its distance to the cluster head d_{toCH}, we design lifespan determined function

$$LS(E_i(r), d_{toCH}) = 1 / f(E_i(r), d_{toCH})$$

where $f(E_i(r), d_{toCH})$ is a weighted integration where $E_i(r)$ represents its chance to become a cluster head and d_{toCH} stands for its energy consumption to transmit data at round r. $f(E_i(r), d_{toCH})$ is the *lifespan decrease function* which we want to minimize in order to prolong network lifespan.

Definition 1 (Energy-Efficient Clustering Problem (EECP))**.** Given a 3-dimensional wireless network with N nodes distributed in a $M\times M\times M$ cube, nodes are divided into several clusters and have lifespan decrease function $f(E_i(r), d_{toCH})$. Our objective is to minimize the average $f(E_i(r), d_{toCH})$ in each subspace to prolong overall network lifespan with a specific clustering algorithm.

In order to analyze EECP, we present the definition of k-means clustering problem [8] in Definition 2, which is a classic NP-hard problem.

Definition 2 (k-means Clustering Problem)**.** Given a network, divide it into k subspaces and minimize the average distance to the nearest center d_{toCH} in each subspace.

Theorem 2. Energy-Efficient Clustering Problem (EECP) is NP-Complete.

Proof. Firstly, we prove that EECP is NP. The decision version of EECP can be easily defined when given some constants to constrain the problem. It is obvious that given any instance of the decision problem, in polynomial time we are able to decide whether it is desirable. Thus, EECP is NP.

Secondly, we notice that k-means clustering problem, a classic NP-hard problem [8], in Definition 2 can be reduced to EECP. Given any instance p_1 of k-means clustering problem, p_1 can be reduced to an instance p_2 of EECP by assigning the parameters of function f, such that $f(E_i(r), d_{toCH}) = d_{toCH}$. Thus, if a solution p in k-means clustering problem gets a reasonable result, its converted version is also available in EECP and vice versa. Hence, EECP is also NP-hard. It means that EECP is NP-Complete. □

Since EECP is NP-Complete, we can hardly find an optimal solution in polynomial time unless NP = P. In this case, we propose a machine-learning-based energy-efficient clustering algorithm called QLEC to prolong network lifespan in high dimensional space.

The important notations adopted in this section and the following sections are summarized in Table 1.

Table 1: Symbol Description

Symbol	Description
N	Total number of nodes in a 3-dimensional network
M	Side length of the cube where nodes are distributed
p_i	Average probability for node b_i to be a cluster head
n_i	Rotating epoch for b_i, which is the reciprocal of p_i
r, R	Current round and total rounds of QLEC algorithm
$E_i(r)$	Residual energy of b_i at round r
$\overline{E}(r)$	Average energy of the whole network at round r
$T(b_i)$	Threshold to determine whether b_i can be the cluster head
$E_{i,th}(r)$	Energy threshold for b_i to be the cluster head at round r
k, d_c	Number of cluster head and cluster coverage radius
H	Set of cluster heads h_j ($j = 1, 2, \ldots k$)
$S(b_i)$	State space of Q-learning method for node b_i
$A(b_i)$	Action space of Q-learning method for node b_i
R_t	Reward function produced by environment to the agent
$P^{a_j}_{b_i h_j}, R^{a_j}_{b_i h_j}$	Packet successful transmission rate and reward function from node b_i to cluster head h_j after taking action a_j
x_{b_i}, x_{h_j}	Equal to the residual energy of b_i and h_j
$y(b_i, h_j)$	Energy needed to transmit a packet from b_i to h_j based on the distance d between them
γ	Discount parameter to balance the influence of current and future actions on current values

4 QLEC ALGORITHM

This section presents our QLEC clustering algorithm in detail. Given a 3-dimensional wireless network with N modes b_i ($i = 1, 2, \ldots N$), our QLEC algorithm is divided into *Cluster Head Selection Phase* and *Data Transmission Phase*.

Cluster Head Selection Phase. Given a 3-dimensional wireless network with N modes b_i ($i = 1, 2, \ldots N$), the improved DEEC algorithm introduced in Subsection 3.1 selects cluster heads through successive rounds. At each round r, only when the node b_i has not been selected as the cluster head in the nearest n_i rounds and possesses more energy than the energy threshold $E_{i,th}(r)$, can it be eligible to be the candidates for cluster heads. Besides, the cluster coverage radius d_c guarantees that the number of selected cluster heads k is very close to the optimal cluster number k_{opt} proved in Subsection 3.2.

Data Transmission Phase. With k cluster heads h_j ($j = 1, 2, \ldots k$) already been selected at each round, non-cluster-head nodes choose a cluster head to transmit sensing data based on Q-learning algorithm illustrated in Subsection 3.3 when they want to forward a packet. At the end of each round, cluster heads perform data fusion and send the processed data to the base station (BS).

In this section, we will give the description of our two-phase QLEC algorithm in detail.

4.1 Cluster Head Selection Phase

Our QLEC clustering algorithm is demonstrated in Algorithm 1. In our algorithm, we firstly compute the optimal cluster number k_{opt} of a given network we try to achieve from Theorem 1. Substituting k_{opt} into Equation (5), we can derive the cluster coverage radius d_c which will be used to reduce redundant cluster heads later in the algorithm. In the *Cluster Head Selection Phase*, the improved DEEC algorithm starts selecting cluster heads through successive rounds until it reaches a setting number of total rounds of the lifespan of the network R in Line 4-16. In each round r, the average energy of the whole network is estimated from Equation (2) and the set of all cluster heads is initialized to $\varnothing$. p_{opt} is the expected probability of a node to be selected as the cluster head and can be given by k_{opt}. With parameters calculated above, Algorithm 2 and Algorithm 3

work together to form the final set of cluster heads H for round r in Line 8-9.

Afterwards, the algorithm goes to the *Data Transmission Phase*, where non-cluster-head nodes dynamically choose cluster heads to send data to for data fusion if they possess a packet taking the base station (BS) h_{BS} as the destination in Line 10-12. Finally, the cluster heads perform data fusion with all the data they have received in this round to compress the data size and lower the transmission energy consumption. After the cluster heads send processed data directly to the base station, they update their own V values in Line 15 and the system goes to the next round.

Algorithm 1: QLEC Algorithm

```
1  Compute the optimal cluster number k_opt of the network
     according to Theorem 1;
2  Substitute k_opt into Equation (5) and calculate the cluster
     coverage radius d_c of each cluster;
3  r = 0;
4  while r < R do
5      Compute Ē(r) of the whole network according to
         Equation (2);
6      H = ∅;                          /* Initialize the set */
7      p_opt = k_opt / N;
8      Call Function Cluster-Head-Selection(H);
9      Call Function Reduce-Redundancy(H);
10     while it is still in round r do
11         foreach b_i ∈ B ∩ b_i ∉ H do
12             if b_i has a packet to transmit to BS then Call
                 Function Send-Data(b_i);
13     foreach h_j ∈ H do
14         Transmit processed data directly to BS;
15         V*(h_j) = Q*(h_j, a_BS) =
             R_t + γ(P^{a_BS}_{h_j h_BS} V*(h_BS) + P^{a_BS}_{h_j h_j} V*(h_j))
16     r = r + 1;                      /* Go to next round */
17 return;
```

The operations for selecting k cluster heads h_j ($j = 1, 2, ...k$) are implemented in Algorithm 2. For each node in the wireless network, we compute the probability and energy threshold of it to be selected as the cluster head in Line 2-3. At each round r, only when the node b_i has not been selected as the cluster head in the nearest n_i rounds and possesses more energy than the energy threshold $E_{i,th}(r)$, can it be eligible to be the candidates for cluster heads.

If a node is eligible to be the cluster head, it will calculate a threshold $T(b_i)$ and compare it with a randomly generated number $z \in [0, 1]$. $z < T(b_i)$ makes b_i become one cluster head of the network in H and send a *HELLO* message to all nodes in the range of its cluster coverage radius d_c, indicating that it has been selected as a cluster head and how much energy it possesses. This mechanism guarantees that the number of selected cluster heads k is very close to the optimal cluster number k_{opt} proved in Subsection 3.2.

When a node which has been selected as the cluster head hears such a *HELLO* message, it realizes immediately that there is another

Algorithm 2: Cluster-Head-Selection(H)

```
1  foreach b_i ∈ B do
2      p_i = p_opt E_i(r) / Ē(r);
3      Compute E_{i,th}(r) according to Equation (4);
4      if b_i has not been selected as the cluster head for the last
         n_i rounds ∩ E_i(r) > E_{i,th}(r) then
5          Compute T(b_i) according to Equation (3);
6          Generate a random number z ∈ [0, 1];
7          if z < T(b_i) then
8              b_i becomes one cluster head in H;
9              Send a HELLO message to all nodes in the range
                 of d_c from b_i with energy information
                 E_HELLO = E_i(r);
10 return;
```

cluster head in the range of d_c from it. As a result, it compares its own eneygy with the energy of its neighbour cluster head, indicated in the *HELLO* message. If it possess less energy that its neighbour cluster head, it will quit the competition of being a cluster head. Otherwise, it remains in H to be a cluster head. This method guarantees that there are no redundant cluster heads in the network and nodes with more energy will be selected as cluster heads, prolonging the lifespan of the network.

Algorithm 3: Reduce-Redundancy(H)

```
1  foreach b_i ∈ H do
2      if b_i receives a HELLO message then
3          Get the energy information E_HELLO from the
             message;
4          if E_i(r) < E_HELLO ∩ b_i ∈ H then Remove b_i from
             H;
5  return;
```

4.2 Data Transmission Phase

We now have a wireless network with k cluster heads selected. In the *Data Transmission Phase*, non-cluster-head nodes dynamically choose cluster heads as a relay for forwarding a packet to BS. The concept of how to choose a *better* cluster head is basically based on the Q-learning method introduced in Subsection 3.3. At the beginning, all the V values and Q values are initialized to 0. For each non-cluster-head node b_i, the *state space* $S(b_i)$ includes itself, the base station, and all the cluster heads, i.e. $S(b_i) = \{b_i, h_{BS}\} \cup H$. H represents all the possible cluster heads it may choose while b_i stands for the unsuccessful packet transmission and state transition. Besides, b_i can also directly communicate with the base station h_{BS}.

Node b_i makes clustering choices based on the set of possible *actions* it can take from the state b_i, denoted as $A(b_i)$. Each action a_j in set $A(b_i)$ means forwarding the packet to cluster head h_j. $P^{a_j}_{b_i h_j}$ is the packet successful transmission rate from node b_i to cluster head h_j after taking action a_j. Poor communication environment

or limited storage caches of cluster heads may lead to packet loss so $P^{a_j}_{b_i h_j} = 1$ does not always hold. Similar to the mechanism adopted by TCP/IP protocol, an *ACK* message will be delivered from the cluster head h_j to the node b_i indicating that the packet sent by b_i is successfully received and processed by h_j. Hence, the link probability $P^{a_j}_{b_i h_j}$ can be estimated by the ratio between the successfully transmitted packets and all the packets sent by b_j recently [2]. It is obvious that $P^{a_j}_{b_i b_i} = 1 - P^{a_j}_{b_i h_j}$. Besides, γ is the *discount rate* set between $[0, 1]$ to balance the impact current actions and future ones have on current V and Q values.

Next, we analyze the *reward* function R_t produced by the *environment* when the *agent* takes an action a_j in state b_i, which is critical to our QLEC algorithm. Derived from Equation (10), R_t can be calculated as follows when node b_i tries to take action a_j sending the packet to cluster head h_j:

$$R_t = P^{a_j}_{b_i h_j} R^{a_j}_{b_i h_j} + P^{a_j}_{b_i b_i} R^{a_j}_{b_i b_i} \tag{16}$$

where $R^{a_j}_{b_i h_j}$ and $R^{a_j}_{b_i b_i}$ stand for the reward function for taking action a_j in state b_i and ending in state h_j and b_i respectively. Specifically, $R^{a_j}_{b_i h_j}$ is the reward of transmitting the packet successfully while $R^{a_j}_{b_i b_i}$ is the reward of unsuccessful packet transmission.

The reward function $R^{a_j}_{b_i h_j}$ for state-action pair (b_i, a_j) in our QLEC algorithm is defined as:

$$R^{a_j}_{b_i h_j} = -g + \alpha_1[x(b_i) + x(h_j)] - \alpha_2 y(b_i, h_j) \tag{17}$$

where $-g$ is a constant *punishment* when a node tries to send a packet because any transmission consumes energy of the network. $x(b_i)$ and $x(h_j)$ are equal to the residual energy of node b_i and cluster head h_j at round r. In this way, non-cluster-head nodes tend to send packets to the cluster head with more residual energy because it can give more positive rewards to the agent. $y(b_i, h_j)$ represents energy needed to transmit the packet from b_i to h_j based on the distance between them. We adopt the *communication energy consumption model* in [4] to estimate the energy consumed to send a packet consisting of L bits data at a distance d from b_i to h_j:

$$y(b_i, h_j) = \begin{cases} L\epsilon_{fs} d^2 & d < d_0 \\ L\epsilon_{mp} d^4 & d \geq d_0 \end{cases} \tag{18}$$

where $d_0 = \sqrt{\frac{\epsilon_{fs}}{\epsilon_{mp}}}$ is the distance threshold.

It takes more energy to transmit packets at a longer distance so the larger $y(b_i, h_j)$ is, the smaller $R^{a_j}_{b_i h_j}$ will be. $x(b_i)$, $x(h_j)$, and $y(b_i, h_j)$ are introduced to take the lifespan of the network into consideration. α_1 and α_2 are the weights of them respectively.

As we mentioned before, non-cluster-head nodes may join in no cluster and communicate directly with the base station. It will aggravate the burden of BS and is what we want to avoid. Thus, we add a penalty l to Equation (17) if a node b_i tries to send data directly with the base station h_{BS}. The revised reward function is as follows and l is set to be an arbitrarily large number:

$$\begin{aligned} R^{a_{BS}}_{b_i h_{BS}} = & -g + \alpha_1[x(b_i) + x(h_{BS})] \\ & - \alpha_2 y(b_i, h_{BS}) - l \end{aligned} \tag{19}$$

If node b_i attempts to transmit data to h_j and fails, the reward function is defined as:

$$R^{a_j}_{b_i b_i} = -g + \beta_1 x(b_i) - \beta_2 y(b_i, h_j) \tag{20}$$

where β_1 and β_2 are the weights that we can assign to show different importance of different parts of energy.

Substituting Equation (17)-(20) into Equation (16), we can calculate Q values for each action a_j in state b_i in Line 1 of Algorithm 4. Then we update the V values of state b_i with the maximum Q values in Line 2 and choose the corresponding cluster head $h_{j_{opt}}$ for data transmission in Line 3-4.

Algorithm 4: Send-Data(b_i)

1 **foreach** $h_j \in H$ *and* h_{BS} **do**
  $Q^*(b_i, a_j) = R_t + \gamma(P^{a_j}_{b_i h_j} V^*(h_j) + P^{a_j}_{b_i b_i} V^*(b_i))$;
2 $V^*(b_i) = \max_{a_j \in A(b_i)} Q^*(b_i, a_j)$; /* Update */
3 $j_{opt} = arg\max_{a_j \in A(b_i)} Q^*(b_i, a_j)$;
4 Forward the packet to cluster head $h_{j_{opt}}$
5 **return**;

4.3 Analysis of QLEC

We respectively analyze the performance and complexity of QLEC in two phases under our model.

Lemma 2. The total time complexity of *cluster head selection phase* is $O(RN)$, where N is the number of sensors and R is the total rounds of QLEC algorithm.

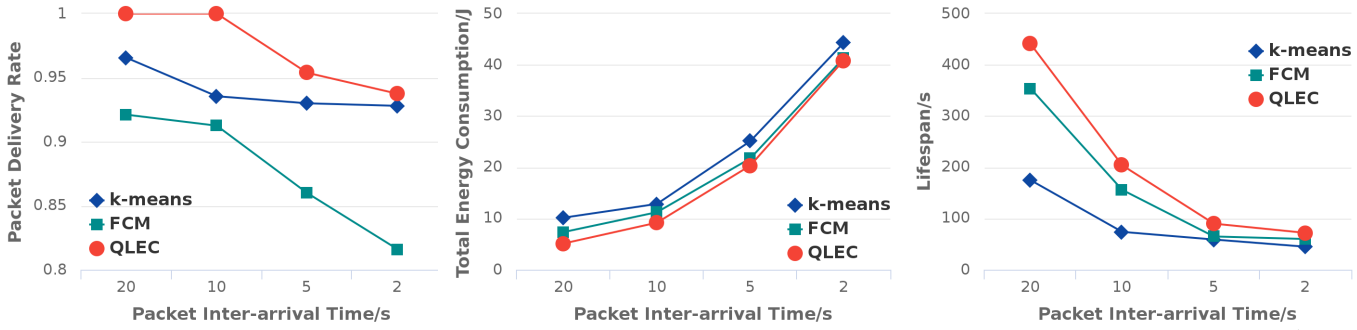
Proof. Since *cluster head selection phase* of QLEC extremely relies on Algorithm 2 and Algorithm 3, we mainly analyze their running time. In Algorithm 2, we only compute $T(b_i)$ for each node and compare it with z to determine whether it can be a cluster head so it runs in $O(N)$ time. Similarly, Algorithm 3 runs in $Q(k)$ time. Considering k is the number of clusters and $k < N$, the total time complexity of *cluster head selection phase* is $O(RN)$ for R rounds. □

Lemma 3. Q-learning algorithm runs in $O(kX)$ time until it converges, where k is the cluster number and X is the times of calculations to make V values converge.

Proof. We establish a matrix to store the V values of each node in the network. As we can see in Algorithm 4, we update $k + 1$ elements of V matrix each time until it converges. If we assume that it takes X updates for the matrix to converge, the total time complexity of Q-learning algorithm to help non-cluster-head nodes choose clusters runs in $O(kX)$ time. □

Theorem 3. QLEC is a machine-learning-based energy-efficient clustering algorithm with the running time $O(kX)$.

Proof. Combining Lemma 2 and Lemma 3, we can get that QLEC runs in $O(RN + kX)$ time. However, it usually takes many times to update all V values in a large-scale wireless sensor networks. Hence, X tends to be much larger than N or R. As a result, QLEC runs in $O(kX)$ time in general. □



(a). Packet Delivery Rate (b). Total Energy Consumption (c). Lifespan

Figure 3: Simulation results of QLEC algorithm compared with FCM-based algorithm and *k*-means clustering

5 PERFORMANCE EVALUATION

In this section, we first present our experiment settings. Then we show our evaluation results with corresponding analysis. Finally, a large-scale dataset in the real world is adopted to test the efficiency of QLEC clustering algorithm.

5.1 Experiment Settings

To evaluate the performance of our QLEC clustering algorithm, we randomly generate $N = 100$ nodes with the same initial energy $5J$ on a space of $200 \times 200 \times 200$ units. According to Theorem 1, the optimal cluster number k_{opt} is approximately 5. Thus, we adopt our QLEC, a newly proposed FCM-based algorithm in [14], and classic k-means algorithms respectively to cluster the nodes and conduct data transmission in $R = 20$ successive rounds. Other various system parameters are demonstrated in Table 2.

Table 2: Simulation Parameters

System Parameters	Settings
discount rate γ	0.95
free space constant ϵ_{fs}	$10pJ/bit/m^2$
multi-path constant ϵ_{mp}	$0.0013pJ/bit/m^4$
weights $\alpha_1, \alpha_2, \beta_1, \beta_2$	0.05, 1.05, 0.05, 1.05
compression ratio at cluster heads	50%

We focus on the lifespan, total energy consumption, and packet delivery rate of the network to evaluate the performance of our algorithm. It is assumed that the network dies when there exists one sensor possessing less energy than a given energy death line. To achieve a stable network condition, we lower the energy death line while measuring the total energy consumption and packet delivery rate.

5.2 Experiment Results

The packet generation time in the network follows the poisson distribution. λ is the average packet inter-arrival time for the network. The smaller λ is, the more congested the network is. We simulate four network conditions with different λ and compare the network indexes under different conditions after clustering with QLEC, FCM-based algorithm, and k-means.

First, we evaluate the packet delivery rate of the network. Poor communication environment or limited storage caches of cluster heads may lead to packet loss. Figure 3(a) exhibits the packet delivery rate of three networks clustered with QLEC, FCM-based algorithm, and k-means under different network conditions. As we can see, QLEC can maintain a packet delivery rate equal to 1 while other two algorithms will lose some packets when the network is relatively idle. As networks get congested, all three packet delivery rate decreases because the long queue at cluster heads leads to discarding more packets. However, QLEC is capable of retaining a higher rate than other two algorithms under all circumstances. What is more, FCM-based algorithm tends to discard more than 10% packets when the network is congested because it takes multi-hops to transmit a packet to the BS under this model.

Next, Figure 3(b) compares the total energy consumption in the three networks above. QLEC clustering consumes less energy than other two algorithms to transmit packets in 20 rounds. k-means clusters nodes based on the distance between them while QLEC attaches some importance to the energy distribution as well as distance. FCM-based algorithm also takes residual energy into consideration, but hierarchical network in it consumes more energy to deliver packets. As a result, QLEC can save more energy when transmitting a packet in a given network.

Finally, Figure 3(c) demonstrates the lifespan of networks. Since QLEC adopts the DEEC algorithm to select nodes with relative more residual energy as the cluster heads and uses Q-learning method to lower the energy dissipation in data transmission, it can definitely prolong the lifespan of the network as we see in Figure 3(c). A longer lifespan is critical to a given network because it may be difficult to charge the sensor nodes under some environmentally harsh conditions like mountainous area or underwater monitoring. Hence, QLEC is quite practical for a 3-dimensional wireless sensor network.

5.3 Experiment based on a Large-scale Dataset

In this subsection, we conduct experiments based on a large-scale dataset of nodes with given energy in China from Global Power Plant Database [3] to prove the efficiency of our QLEC clustering algorithm in reality. Note that this dataset is not strictly a dataset of IoT, but we can utilize the data of energy in it to simulate a WSN. Besides, we randomly assign a height value to each node to convert

the 2-dimensional network of the dataset into a 3-dimensional one. In the dataset, we have 2896 nodes in China in total, not counting the base station (BS). According to Theorem 1, $k_{opt} = 272$ cluster heads are selected to perform data fusion and transmission for the network.

We adopt QLEC in such a network based on the large-scale dataset. Figure 4 shows how the ratio between energy consumption and initial energy is for all nodes in the network. As shown in the figure, nodes with high energy consumption rate that are marked in red are evenly distributed in the network, which means QLEC tends to make energy equally dissipated among nodes to prevent a few nodes from running out of energy too early. In this way, the lifespan of the network is guaranteed and sensors can function as long as possible.

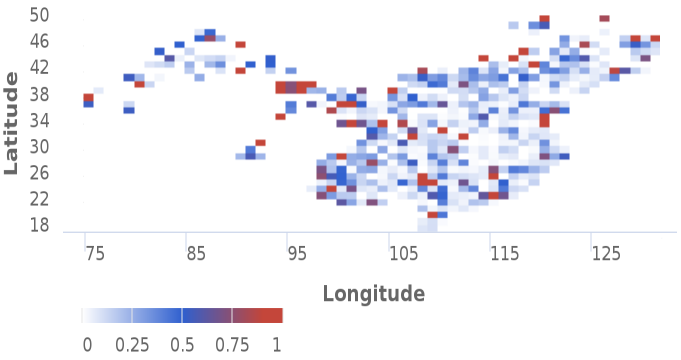


Figure 4: Energy consumption rate in the network based on a large-scale dataset after clustering with QLEC

Generally speaking, QLEC performs efficiently not only in a simulation network, but also in a real-world network based on a large-scale dataset.

6 CONCLUSION

In this paper, we propose a machine-learning-based two-phase algorithm called QLEC for clustering problems in high dimensional heterogeneous wireless sensor networks.

First, we improve traditional Distributed Energy-Efficient Clustering (DEEC) algorithm with some constraints of energy and prove the optimal cluster number in a 3-dimensional wireless network to select nodes with more residual energy as the cluster heads for data fusion. Then we give a detailed analysis to Q-learning method and adopt it in the network to help non-cluster-head nodes choose cluster heads for data transmission. In this way, energy consumption is distributed evenly in the network and the lifespan of it is prolonged. We prove that our QLEC algorithm has the running time $O(kX)$. Finally, numerical simulations and experiments based on a large-scale dataset validate the efficiency of our design. To the best of our knowledge, we are the first work to design an energy-efficient clustering algorithm combined with Q-learning method in high dimensional space of IoT.

ACKNOWLEDGMENT

This work was supported by the National Key R&D Program of China [2018YFB1004703]; the National Natural Science Foundation of China [61872238, 61672353, 61672348]; the Shanghai Science and Technology Fund [17510740200]; the Huawei Innovation Research Program [HO2018085286]; the State Key Laboratory of Air Traffic Management System and Technology [SKLATM20180X]; the Tencent Social Ads Rhino-Bird Focused Research Program; the Open Project Program of the State Key Laboratory of Mathematical Engineering and Advanced Computing [2018A09]; and Alibaba Group through Alibaba Innovation Research Program.

REFERENCES

[1] S. Bandyopadhyay and E.J. Coyle. 2003. An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks. In *Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. 1713–1723.
[2] Stefano Basagni, Valerio Di Valerio, Petrika Gjanci, and Chiara Petrioli. 2018. Harnessing HyDRO: Harvesting-aware Data Routing for Underwater Wireless Sensor Networks. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*. 271–279.
[3] Logan Byers, Johannes Friedrich, Roman Hennig, Aaron Kressig, Xinyue Li, Laura Malaguzzi Valeri, and Colin McCormick. 2018. Global Power Plant Database. http://datasets.wri.org/dataset/globalpowerplantdatabase. (Aug. 2018).
[4] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan. 2002. An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications (TWC)* 1, 4 (2002), 660–670.
[5] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. 2000. Energy-efficient communication protocol for wireless microsensor networks. In *Hawaii International Conference on System Sciences (HICSS)*. 1–10.
[6] T. Hu and Y. Fei. 2010. QELAR: A Machine-Learning-Based Adaptive Routing Protocol for Energy-Efficient and Lifetime-Extended Underwater Sensor Networks. *IEEE Transactions on Mobile Computing (TMC)* 9, 6 (2010), 796–809.
[7] Nadeem Javaid, Muhammad Rasheed, Muhammad Imran, Mohsen Guizani, Zahoor Khan, Turki Alghamdi, and Manzoor Ilahi. 2015. An Energy Efficient Distributed Clustering Algorithm for Heterogeneous WSNs. *EURASIP Journal on Wireless Communications and Networking (J WIREL COMM)* 2015, 1 (2015), 151.
[8] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. 2002. An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 24, 7 (2002), 881–892.
[9] C. Lin, Y. Zhou, H. Dai, J. Deng, and G. Wu. 2018. MPF: Prolonging Network Lifetime of Wireless Rechargeable Sensor Networks by Mixing Partial Charge and Full Charge. In *IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 1–9.
[10] V. Loscri, G. Morabito, and S. Marano. 2006. A two-levels hierarchy for low-energy adaptive clustering hierarchy (TL-LEACH). In *IEEE Vehicular Technology Conference (VTC)*.
[11] L. Qing, Q. Zhu, and M. Wang. 2006. Design of a distributed energy-efficient clustering algorithm for heterogeneous wireless sensor networks. *Computer Communications (CC)* 29, 12 (2006), 2230–2237.
[12] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement learning: An introduction*. MIT Press. 1–322 pages.
[13] Meenakshi Tripathi, R. B. Battula, M. S. Gaur, and V. Laxmi. 2013. Energy Efficient Clustered Routing for Wireless Sensor Network. In *IEEE International Conference on Mobile Ad-hoc & Sensor Networks (MSN)*. 330–335.
[14] Zijing Wang, Xiaoqi Qin, and Baoling Liu. 2018. An energy-efficient clustering routing algorithm for WSN-assisted IoT. In *IEEE Wireless Communications and Networking Conference (WCNC)*. 1–6.
[15] Dali Wei, Yichao Jin, Serdar Vural, Klaus Moessner, and Rahim Tafazolli. 2011. An Energy-Efficient Clustering Solution for Wireless Sensor Networks. *IEEE Transactions on Wireless Communications (TWC)* 10, 11 (2011), 3973–3983.
[16] F. Wu, C. Rudiger, and M. R. Yuce. 2017. Design and field test of an autonomous IoT WSN platform for environmental monitoring. In *IEEE International Telecommunication Networks and Applications Conference (ITNAC)*. 1–6.
[17] O. Younis and S. Fahmy. 2004. HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on Mobile Computing (TMC)* 3, 4 (2004), 366–379.
[18] Jia Zhao, Wei Gong, and Jiangchuan Liu. 2018. X-Tandem: Towards Multi-hop Backscatter Communication with Commodity WiFi. In *ACM International Conference on Mobile Computing and Networking (MobiCom)*. 497–511.