Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.048240

ARTICLE

# Mobile Crowdsourcing Task Allocation Based on Dynamic Self-Attention GANs

**Kai Wei**[1], **Song Yu**[2] **and Qingxian Pan**[1,*]

[1]School of Computer and Control Engineering, Yantai University, Yantai, 264005, China

[2]School of Computer and Information Science, Southwest University, Chongqing, 400715, China

*Corresponding Author: Qingxian Pan. Email: pqx@ytu.edu.cn

Received: 01 December 2023 Accepted: 19 February 2024 Published: 25 April 2024

**ABSTRACT**

Crowdsourcing technology is widely recognized for its effectiveness in task scheduling and resource allocation. While traditional methods for task allocation can help reduce costs and improve efficiency, they may encounter challenges when dealing with abnormal data flow nodes, leading to decreased allocation accuracy and efficiency. To address these issues, this study proposes a novel two-part invalid detection task allocation framework. In the first step, an anomaly detection model is developed using a dynamic self-attentive GAN to identify anomalous data. Compared to the baseline method, the model achieves an approximately 4% increase in the F1 value on the public dataset. In the second step of the framework, task allocation modeling is performed using a two-part graph matching method. This phase introduces a P-queue KM algorithm that implements a more efficient optimization strategy. The allocation efficiency is improved by approximately 23.83% compared to the baseline method. Empirical results confirm the effectiveness of the proposed framework in detecting abnormal data nodes, enhancing allocation precision, and achieving efficient allocation.

**KEYWORDS**

Mobile crowdsourcing; task allocation; anomaly detection; GAN; attention mechanisms

## 1 Introduction

Crowdsourcing is a novel method for problem-solving that leverages decentralized swarm intelligence [1]. Given the challenges inherent in task allocation within a crowdsourcing environment, numerous researchers have proposed viable solutions [2–6] and outlined directions for subsequent research [7–10]. For instance, Cheng et al. [11–14] improved the timeliness of crowdsourcing through prediction-based task assignments, utilizing data on workers' locations and trajectories. Zhao et al. [15–17] developed a preference-aware task assignment method, enhancing assignment volume by considering workers' temporal and spatial preferences. In a distinct approach, Ding et al. [18] introduced a dynamic delayed decision-making method for task assignment, alleviating issues such as poor quality resulting from hasty decisions by incorporating the timing of task assignment. Wang et al. [19] concentrated on robustness, devising an offline initiative to create a resilient task allocation scheme capable of handling unforeseen disruptions. This effort significantly bolstered the robustness of the allocation scheme.

Chen et al. [20] delved into optimal task allocation among workers, introducing an impact-aware approach that maximizes task volume by calculating workers' affinity for tasks. Finally, Zhao et al. [21] employed worker turnover predictions in task allocation to maximize the total reward from task allocation.

On crowdsourcing platforms, the presence of malicious actors among task workers and requesters can pose significant challenges [22]. For instance, a requester may publish an invalid task, which cannot be completed when assigned to a worker and yields no reward. Such malicious requests undermine task quality and accuracy. Crowdsourcing abnormal activities are not only limited to malicious requests issued by task requesters but also include malicious behaviors of task workers. For example, workers may not arrive at the task location on time when performing tasks. To address this concern, Gadiraju et al. [23] investigated everyday malicious activities within crowdsourcing environments. They proposed a method to evaluate these activities by analyzing worker behavior, thereby assessing their trustworthiness. Spurling et al. [24] studied the threat posed by malicious workers to crowdsourcing platforms, employing interval-valued labels and worker reliability metrics to detect anomalous behavior. Zhao et al. [25] introduced a framework for detecting anomalies in malicious task requests or workers. Meanwhile, Kieu et al. [26,27] proposed solutions for handling anomalous data in multivariate time series, with the aim of enhancing robust anomaly detection in time series through unsupervised learning. This body of work underscores the importance of addressing malicious activities to ensure the integrity and effectiveness of crowdsourcing platforms. Li et al. [28–31] presented a variety of deep learning-based solutions for anomaly detection problems, utilizing tools such as generative adversarial networks and innovative autoencoders to tackle various challenges in anomaly detection. Similarly, Sattar et al. [32,33] leveraged deep learning and crowdsourcing technologies to address abnormal activities in different scenarios, including trajectory detection and business fraud. Commonly employed deep learning models for anomaly detection research include self-encoders and generative adversarial networks. The self-encoder, a widely used deep learning model [34–36] for unsupervised learning, comprises encoder and decoder components essential for data dimensionality reduction, feature learning, and generative modeling. The Generative Adversarial Network (GAN) [37,38] is a generative model consisting of generator and discriminator components. The network trains these two components against each other, creating a competitive environment. GAN has proven successful in enhancing images and data. The Self-Attention Mechanism [39] is an attentional mechanism used for processing sequential data. It enables the model to focus on information at different locations during data processing and is commonly employed in Natural Language Processing (NLP) tasks.

Their work has contributed numerous practical methods for future research. Despite considerable progress, the current research field faces two significant challenges. Firstly, a majority of existing methods concentrate on typical crowdsourcing anomalies, neglecting the abnormal activities jointly carried out by workers and task requesters. Hence, it is imperative to incorporate the combination of fraudulent workers and malicious tasks into the analysis. Furthermore, the present methods lack adequate recognition of abnormal activities. Due to the limited occurrence of malicious workers and tasks in real-time data streams, the existing methodologies do not adequately address these anomalies, posing a considerable challenge for effective identification in large-scale data streams. Secondly, the efficiency of allocation algorithms becomes compromised when confronted with datasets containing a substantial number of abnormal nodes.

Given the challenges outlined above, this study introduces a novel mechanism for invalid detection, referred to as Dynamic Self-Attention Generative Adversarial Networks (DSAGD). The purpose of DSAGD is to identify abnormal nodes in real-time data streams, enhancing the efficiency and reliability of crowdsourcing allocation. The paper bifurcates the invalid detection mechanism into two main components, as illustrated in Fig. 1. The first segment comprises an autoencoder and a dynamic self-attention GAN network. This component models the issue of identifying malicious task requests and fraudulent workers as a related time series anomaly detection problem. Its primary goal is to assess whether the modeled multivariate time series data conforms to the expected distribution of the model and to handle the resulting abnormal nodes. Diverging from traditional methods, we employ two GANs, denoted as cross-generative adversarial networks, to train the autoencoder. This strategy enhances the autoencoder's ability to learn the distribution of the original data more effectively. The second part focuses on the matching process between workers and tasks. The paper transforms task allocation into bipartite graph matching. This matching process incorporates the abnormal attribute values of each worker or task calculated in the first part. Subsequently, an efficient allocation algorithm is employed for comprehensive task allocation. The specific block diagram of the framework is illustrated in Fig. 1. The primary contributions of this paper can be summarized as follows:

1) We propose a cross GAN (C-GAN) network anomaly detection model to cross-train the autoencoder.

2) We propose a dynamic self-attention model to enhance the representation of important features by dynamically adjusting the attention weight of the channel.

3) We propose a P-queue KM algorithm in the second stage of the framework, which effectively solves the problem of low allocation efficiency in the two-part graph matching process.

4) The effectiveness and allocation efficiency of the proposed framework are verified by comparing it with real datasets.

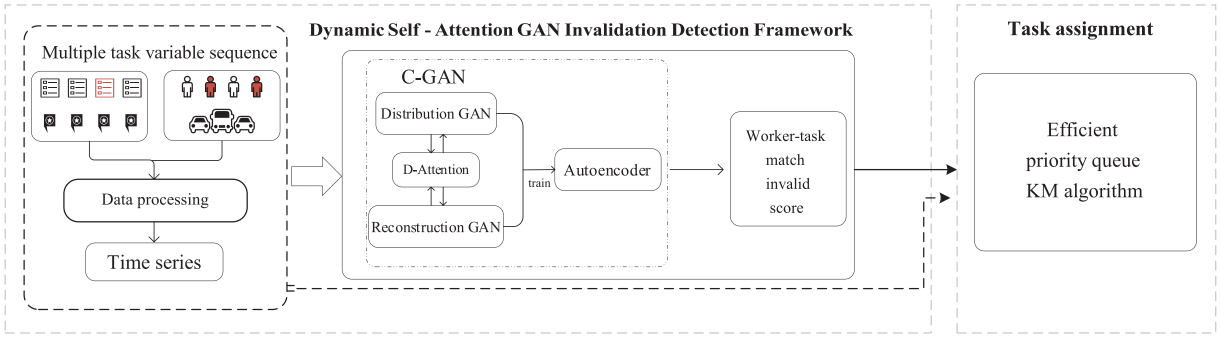
Multiple task variable sequence
Data processing
Time series
Dynamic Self - Attention GAN Invalidation Detection Framework
C-GAN
Distribution GAN
D-Attention
Reconstruction GAN
train
Autoencoder
Worker-task match invalid score
Task assignment
Efficient priority queue KM algorithm

**Figure 1:** Dynamic self-attention GAN invalidation detection framework

## 2 Related Work

### *2.1 Autoencoder*

The autoencoder is a type of unsupervised learning mechanism that consists of a feed-forward fully connected neural network comprising two components: An encoder and a decoder. Its primary purpose is to acquire efficient data representations. In the context of time series anomaly detection, the autoencoder can effectively identify the normal data distribution within the time series. By comparing the reconstructed data with the original data and measuring their discrepancy against a predefined threshold, one can infer whether there are anomalous nodes present in the original data. Consequently, it enables us to detect anomalies or reconstruction errors within data. The encoding-decoding process of an autoencoder can be formally described as follows:

$$f_E(a_i) : \mathbb{R}^m \rightarrow \mathbb{R}^h \tag{1}$$

$$f_D(t) : \mathbb{R}^h \rightarrow \mathbb{R}^m \tag{2}$$

Eqs. (1) and (2) denote that the encoder maps the high-dimensional input to the potential space $h$, and the decoder maps back to the initial dimension to get the reconstructed output $X^R$. Autoencoder can adapt to data learning of different structures and learn data features well.

### *2.2 Generating Adversarial Networks*

The Generative Adversarial Network (GAN), a deep learning model, comprises two primary components: The Generator and the Discriminator. This fundamental concept leverages an adversarial game, where the Generator and Discriminator are trained reciprocally in an antagonistic manner. In anomaly detection scenarios, GANs address the limitations of traditional statistical models that struggle to adapt to complex data distributions and identify anomalous sample data. This is achieved by training the generator to fit the distribution of actual samples and subsequently calculating the discrepancy between real samples and generated ones. The objective functions for training both generator and discriminator are as follows:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(Z)}[\log(1 - D(G(z)))] \tag{3}$$

where $X$ is the real data, $D(x)$ denotes the probability from the real data, $G(z)$ denotes the possibility of generating a pseudo-sample, $z$ is the noisy data, and the a priori value of the noisy data. The generative adversarial network can learn data representation, generate new data samples, and improve the model's sensitivity to abnormal data through adversarial mechanisms.

### *2.3 Self-Attention Mechanism*

The attention mechanism is a deep learning technique primarily employed to augment a model's capacity for processing and comprehending input data. In the context of anomaly detection, the objective is to identify outliers within the dataset. These outliers may exhibit distinct characteristics and statistical properties compared to normal nodes. By capturing dependencies and contextual information among data elements, the self-attention mechanism enables the model to gain a better understanding of the data distribution and effectively detect anomalies. The inputs to the self-attention mechanism consist of queries, keys, and values derived through linear transformations using trainable parameter matrices. The formal representation of the self-attention mechanism can be seen in Eq. (4):

$$A(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

## 3 Method

### *3.1 Definition of the Problem*

Problem 1 Crowdsourcing task: A crowdsourcing task can be represented as a ternary $t = \langle a, e, l \rangle$, represented by m-dimensional vectors, respectively, and the ternary describes the task arrival time and the end time $t.e$, as well as the task location $t.l$.

Problem 2 Crowdsourcing workers: A crowdsourced worker triad can be expressed as a triad that denotes the worker's arrival time $w.a$, the worker's end time of performing the task $w.e$, and the worker's position $w.l$, respectively.

Problem 3 Anomaly detection: In multivariate time series anomaly detection, the aim is to assign the corresponding attribute value to the input sequence for each timestamp $T$, where the attribute value V is computed from the original sequence with the reconstructed fitted sequence.

Problem 4 Task utility: In two-part graph matching, worker-task matching edges all have combined weight values computed from the base utility value and the anomalous attribute value of the form $W = U + V$, with an initialized random value.

## 3.2 *Dynamic Self-Attention GAN Invalidation Detection*

The Dynamic Self-Attention Generative Adversarial Network (DSAGD) is introduced in this section as a novel solution for invalid detection. In this study, we model the real-time task-worker request as a multivariate time series, where abnormal nodes are represented by invalid tasks issued by the task requester and malicious behaviors of the task worker. Consequently, we reframe the problem of identifying abnormal nodes in worker-task relationships as an outlier detection challenge within time series data. To address this challenge, our proposed framework incorporates a Cross-Generative Adversarial Network (C-GAN), which utilizes both perceptual node distribution GAN and reconstruction error GAN to cross-train the autoencoder and learn the distribution pattern of task time series. This section outlines the workflow of our anomaly detection mechanism, with Fig. 2 illustrating the corresponding flow chart.

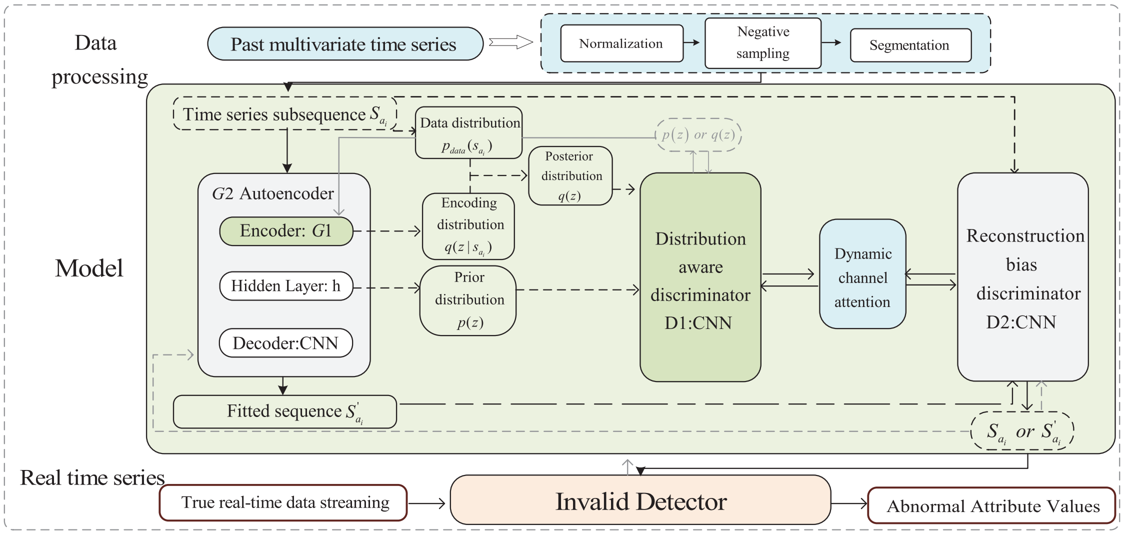
Data processing
Past multivariate time series
Normalization
Negative sampling
Segmentation
Model
Time series subsequence $S_{a_i}$
Data distribution $p_{data}(s_{a_i})$
$p(z)$ or $q(z)$
Posterior distribution $q(z)$
G2 Autoencoder
Encoder: G1
Hidden Layer: h
Decoder:CNN
Encoding distribution $q(z|s_{a_i})$
Prior distribution $p(z)$
Distribution aware discriminator D1:CNN
Dynamic channel attention
Reconstruction bias discriminator D2:CNN
Fitted sequence $S'_{a_i}$
$S_{a_i}$ or $S'_{a_i}$
Real time series
True real-time data streaming
Invalid Detector
Abnormal Attribute Values

**Figure 2:** Invalid detector flowchart

### 3.2.1 *Process Overview*

The overall process is depicted in Fig. 2. Initially, the framework preprocesses the historical multivariate sequence data. Subsequently, the system transmits the preprocessed data to the autoencoder. The autoencoder then forwards various probability distributions to the distribution-aware GAN discriminator and receives feedback. This procedure enables the autoencoder to capture data features comprehensively and learn distribution patterns effectively. Afterward, the autoencoder generates fitted data and sends it to the reconstruction error GAN for comparison with the original data. Upon the arrival of real-time data, the trained model performs anomaly detection.

#### *3.2.2 Cross-Generative Adversarial Network*

In the Cross-Generative Adversarial Network (C-GAN), both the distribution-aware GAN and the reconstruction error-based GAN are integrated, enabling them to be trained adversarially. Regarding the network structure composition, the discriminator of the distribution-aware GAN consists of five convolutional layers. Each layer employs a $4 \times 4$ kernel size, a stride of 2, and a padding of 1. Conversely, the discriminator for the reconstruction error GAN comprises only one feature extraction layer and one classifier.

(1) Distribution-Aware GAN

The distribution-aware GAN (depicted in dark green in Fig. 2) employs the encoder (CNN) as the generator G1, and the decoder functions as the discriminator D1. The encoder consists of five convolutional layers, with each layer utilizing a $4 \times 4$ kernel, a stride of 2, and a padding of 1. Following each convolutional layer, there is a batch normalization layer and a subsequent ReLU activation function. The decoder is composed of six convolutional layers. The initial deconvolution layer employs a $4 \times 4$ convolution kernel, a stride of 1, and zero padding. The sequence is employed as the network input, and the hidden layer yields output $z$. The distribution-aware GAN, in an effort to learn the data distribution pattern within the sequence and fit the data distribution, bootstraps the computation between the posterior distribution $q(z)$ of the original data distribution $p_{data}\left(s_{a_i}\right)$ and the encoded data distribution $q\left(z|s_{a_i}\right)$, and the prior distribution $p(z)$ of the hidden variable $z$. Given that the encoded value is a continuous distribution of sequential data rather than a discrete point, the computation formula for $q(z)$ can formally be expressed as follows:

$$q(z) = \int p_{data}\left(s_{a_i}\right) q\left(z|s_{a_i}\right) ds_{a_i} \tag{5}$$

Assuming that the coding distributions are Gaussian, $q\left(z|s_{a_i}\right)$ is each coding distribution's probability density function [40,41]. Finally, backpropagation is performed using an encoder. Coding distribution is a probability distribution that represents the frequency or possibility of data elements. The combination of data distribution and coding distribution is designed to understand the uncertainty of model parameters better and enhance the model's robustness. The hidden layer outputs a prior distribution to better control the generated samples.

The distribution-aware GAN is designed to learn the original data distribution and bootstrap matching with the fitted data, with a loss function expressed as:

$$D_1 = \mathbb{E}_{z\sim p(z)}[\log D(z)] \tag{6}$$

$$+ \mathbb{E}_{S_{a_i}\sim P_{data}\left(S_{a_i}\right)}\left[\log\left(1 - D\left(G\left(S_{a_i}\right)\right)\right)\right] \tag{6}$$

The distribution-aware GAN tries to maximize the loss function as a whole, and the generator hopes to minimize the corresponding loss function, so the joint description of the two objectives is $\mathbb{C}_1$:

$$\begin{aligned} \mathbb{C}_1 &= \max(D_1) + \min(G_1) \\ &= \max(D_1) - \max(G_1) \\ &= \max\{\mathbb{E}_z\sim p(z)[\log D(z)]\} \end{aligned} \tag{7}$$

(2) Reconstruction Error GAN

Within the Reconstruction Error GAN (depicted in the gray area of Fig. 2), the autoencoder serves as a generator in this study, taking in an input sequence and reconstructing the output $S'_{a_i}$. The Reconstruction Error GAN is designed to guide the comparison between the original and reconstructed sequences, thereby preventing overfitting during the training process. The loss function for the Reconstruction Error GAN can be formally expressed as $\mathbb{C}_2$:

$$\begin{aligned}&\mathbb{C}_2 = \max(D_2) + \min(G_2)\\&= \max(D_2) - \max(G_2)\\&= \max\left\{\mathbb{E}_{S_{a_i}} \sim P_{data}\left[\log D_2\left(S_{a_i}\right)\right]\right\}\end{aligned} \tag{8}$$

### 3.2.3 *Dynamic Channel Attention*

To effectively learn the feature weights between channels, this paper introduces a module specifically designed for learning channel weight attention, the Dynamic Channel Attention Module (DCAM). We guide the feature computation between two discriminators in the distribution-aware GAN and the reconstruction error GAN to assign higher feature weights to the data focused on inter-channel [42], thereby enhancing the network's capability to detect abnormal data nodes.

In the context of dynamic channel attention, the total operations executed within the network layer can be collectively represented as follows:

$$Z_c = F(c) = \frac{1}{B \times I}\left(\sum_{i=1}^{H}\sum_{j=1}^{L} c(i,j)\right) \tag{9}$$

B represents the batch size, which is the number of samples processed in each iteration. Meanwhile, I signifies the input dimension size. To enhance learning capabilities, a reduction operation [43] is executed on the channel inputs to train crucial channels. Subsequently, another operation is performed to restore the features back to their original dimensions. Following these operations, the model assigns higher weights to essential features.

### 3.2.4 *Abnormal Detection*

The anomaly detection phase aims to compute the value of the anomalous attribute between the network input sequence $S_{a_i}$ and the fitted output sequence $S'_{a_i}$ using the formula:

$$Value = \left\| S_{a_i} - S'_{a_i} \right\| \tag{10}$$

In this study, the aforementioned attribute values are incorporated into the worker-task assignment process, and workers or tasks with higher values are more likely to be identified as abnormal nodes.

### 3.2.5 *Assignment of Tasks*

In a two-stage task assignment, task assignment is transformed into two-part graph matching. Let us assume there is an undirected graph with a vertex set in the left set V and an edge set in the right set E, where the vertex set V consists of a left-hand side worker set W, and a right-hand side task set T. When tasks and workers arrive in real-time at a specific timestamp, i.e., there are matchable nodes in both sets, the matching edges are added to the set. Initially, each edge is assigned an initial weight $U(w,t)$. This paper introduces anomalous attribute values for each edge $Value\left(w_i, t_j\right)$, where unequal anomalies may be present between workers or tasks. Consequently, the total weight of the corresponding edges connecting workers and tasks can be represented as:

$$W\left(w_i, t_j\right) = U\left(w, t\right) * \{1 - Value_{\max}\left[(w, w'), (t, t')\right]\} \tag{11}$$

Here, those with high anomaly attribute values are likelier to have lower anomaly weight utility. To achieve efficient two-part graph matching, this paper proposes an efficient allocation algorithm, namely the P-queue KM algorithm, as described in Algorithm 1.

(1) Overview of algorithm flow

This section presents an overview of the P-queue KM algorithm, providing a detailed introduction to its process and critical components. Initially, the algorithm initializes global variables and the left worker nodes, followed by a call to the breadth-first search algorithm to find the augmented path. During the path search, the algorithm adds the starting node to the priority queue and sorts it based on the relaxation value. If the queue is not empty, the algorithm removes the first element, marks it as accessed, traverses the right set, and calculates the slack value. If the augmented path is found, the algorithm updates the matching relationship; otherwise, it updates the label array based on the descending order of the relaxation value. Finally, the algorithm calculates the sum of the maximum matching weights and returns the matching results. The optimization method utilizes a priority queue to minimize the number of traversals by actively selecting the next vertex for extension. The time complexity of the improved KM algorithm in extending the next vertex is reduced from *O(n)* to *O(log n)*.

**Algorithm 1:** P-queue KM

```
Input: weighting matrix
Output:   sum of weights and matches
1    for $i = 1$ to $N + 1$ do
2        initialization;
3    end
4    while que do
5        u = return minimum (queue);
6        for $v = 1$ to $N + 1$ do
7            relaxation action;
8            If v is not matched then
9                expansion path;
10           else
11               add queue;
12           end
13       end
14   end
```

## 4 Experimental Section

In order to evaluate the performance of the anomaly detection mechanism proposed in this paper, the experimental design uses an actual multivariate time series data set, soil moisture active and passive remote sensing satellite (SMAP) [44], to simulate the crowdsourcing worker-task data stream. Table 1 shows the statistics of the data set. The statistical table includes the number of subsets, the size of the training set, and the size of the test set. We use PyTorch (v1.11 with Python 3.7.1) on Windows machines equipped with NVIDIA GTX1050Ti for experimentation.

**Table 1:** Statistics of datasets

| Dataset name | Subset number | Training sets | Testing sets |
|---|---|---|---|
| SMAP | 55 | 337680 | 225120 |

### *4.1 Anomaly Detection Evaluation*

Model Comparison for Evaluation. This paper compares the performance of four models, namely, VAE [45], SA-GAN-Learn [25], SA-GAN-Random [25], and DSAGD. For parameter settings, the proposed anomaly detector in this study uses Adam as the optimizer, a learning rate of 0.001, a momentum β1 of 0.9, a momentum β2 of 0.999, a batch size of 256, a sequence length of 128, and 20 training iterations. The F1-score serves as the metric to evaluate the performance of the four aforementioned models in this paper, with a higher value indicating better model performance. The anomaly threshold selection range is set to [0,1], with an initial value of 0 and a step size of 0.01. Afterwards, the anomaly attribute values of different models are normalized to a fixed interval using min-max normalization. Then, the F1-score is calculated for each threshold, and the optimal F1-score is determined. We compare the following methods when evaluating the detection performance of malicious workers and invalid task requests:

1) VAE: Variational autoencoder, an autoencoder extension. The network structure of VAE in this paper is the same as the other two SA-GAN networks.

2) SA-GAN-Random: This model uses GAN for anomaly detection, especially proposes social awareness, and uses a random strategy for update.

3) SA-GAN-Learn: This model uses changing learning rates to update awareness.

Table 2 shows the performance of different models on the data set. In the SMAP dataset, DSAGD is about 4.07% superior to the best-performing SA-GAN-Random model. The overall value of DSAGD is approximately 3.9% better than the top SA-GAN-Random model. The above data comparison shows that the model in this paper has a significant performance improvement compared with other models. The model's performance also varies because the data distribution varies across different datasets. DSAGD enhances weight learning between channels to assign a higher weight to abnormal nodes, enabling the model to identify abnormal nodes more effectively.

**Table 2:** F1-score

| | SMAP | Totally |
|---|---|---|
| VAE | 0.332 | 0.212 |
| SAGAN-Random | <u>0.541</u> | <u>0.538</u> |
| SAGAN-Learn | 0.496 | 0.494 |
| DSAGD | **0.563** | **0.559** |

Tables 3 and 4 compare the precision and recall rates of different models. The model in this paper is ahead of other models in terms of accuracy comparison. Regarding SMAP and overall value, the model in this paper is significantly ahead of other models. The above two comparisons can show the model's effectiveness in this paper.

**Table 3:** Comparison of accuracy of different models

| | SMAP | Totally |
|---|---|---|
| VAE | 0.312 | 0.312 |
| SAGAN-Random | <u>0.39</u> | <u>0.379</u> |
| SAGAN-Learn | 0.34 | 0.35 |
| DSAGD | **0.41** | **0.392** |

**Table 4:** Recall rate comparison of different models

| | SMAP | Totally |
|---|---|---|
| VAE | 0.532 | 0.69 |
| SAGAN-Random | <u>0.77</u> | 0.642 |
| SAGAN-Learn | 0.06 | <u>0.86</u> |
| DSAGD | **0.97** | **0.979** |

### *4.2 Task Allocation Evaluation*

Evaluation Comparison. For the two-part graph matching problem, the following five algorithms are compared in this paper:

1) KM algorithm

2) Greedy algorithm

3) Greedy algorithm based on anomalous attribute values

4) KM algorithm based on anomalous attribute values

5) P-queue KM algorithm based on anomalous attribute values

Evaluation Metrics. This study employs four critical metrics for evaluation: The total utility value of task assignment, the accuracy rate of worker assignment, the accuracy rate of crowdsourcing task assignment, and the time consumed in task assignment. Considering the scale of workers and tasks, we evaluate each of these indicators. The growth rate for the number of workers and tasks is set at 500, i.e., 500, 1000, 1500, and 2000, to emulate the large-scale data flow prevalent in real-world task allocation scenarios.

#### *4.2.1 Comparison of Total Weight Values*

In Table 5, the total task utility value for all algorithms demonstrates an upward trend with an increase in the number of workers or tasks. Specifically, for the SMAP dataset, the average enhancements are about 5.245% and 5.952% under the respective dimensions. It is apparent from the table that the total utility value increases with the increase in the number of workers and tasks. The increase in the total utility value is due to adding worker set and task set nodes in the bidirectional graph. In the SMAP datasets, the total utility value achieved by the proposed algorithm surpasses that of the greedy algorithm, is comparable to the KM algorithm, and slightly lags behind the KM algorithm in specific dimensions. Including abnormal attribute values during edge matching impacts the total utility value, contributing to this slight underperformance. The above four tables illustrate the proposed algorithm's superiority in different dimensions, indicating its competitive edge over other algorithms.

**Table 5:** SMAP the total weight of task allocation

| Number of workers | Greedy | KM | A-Greedy | A-KM | P-KM |
|---|---|---|---|---|---|
| 500 | 4975 | 6013 | 4973 | 5991 | **6010** |
| 1000 | 10210 | 11910 | 10190 | 11902 | **11980** |
| 1500 | 16002 | 17745 | 15919 | 17627 | **17777** |
| 2000 | 21448 | 22060 | 21359 | 21964 | **22100** |
| Number of tasks | Greedy | KM | A-Greedy | A-KM | P-KM |
| 500 | 4961 | 5633 | 4955 | 5617 | **5640** |
| 1000 | 10225 | 11501 | 10216 | 11452 | **11550** |
| 1500 | 15968 | 17165 | 15908 | 17123 | **17200** |
| 2000 | 21587 | **24312** | 21438 | 24199 | 24198 |

### *4.2.2 Comparison of Worker Distribution Accuracy*

Table 6 compares the worker assignment accuracy of different algorithms, expressed as the ratio of correctly assigned workers to the total number of workers. In the SMAP dataset, the improvements are 0.10% and 0.16% for the two respective scenarios. Given that the majority of abnormal activities in the crowdsourcing data stream originate from malicious task requester requests, various algorithms encounter a reduction in worker assignment accuracy. The proposed algorithm outperforms other algorithms in most scenarios in the worker count dimension. However, under the task count dimension, the algorithm's performance is slightly inferior to the Greedy algorithm in specific systems, possibly due to its insufficient robustness in handling abnormal situations, indicating certain algorithm limitations. Overall, in both dimensions of the SMAP dataset, all algorithms demonstrate relative stability. The accuracy of the worker assignment algorithm in most scenarios surpasses that of other algorithms, signifying its competitive advantage over other algorithms.

### *4.2.3 Comparison of Task Allocation Accuracy*

Table 7 illustrates a comparison of task allocation accuracy across different algorithms. In the SMAP dataset, the average enhancements under the worker count and task count dimensions are about 0.06% and 0.14%. In the SMAP dataset, under the worker count dimension, task allocation accuracy peaks when the worker count is 1000 and subsequently diminishes with an increase in the number of workers. Reduced accuracy suggests that increasing the number of workers may disrupt optimal task matching. Additionally, when the worker count is 1500, the performance of the proposed algorithm lags behind that of the outlier-based Greedy algorithm. However, when the worker count escalates to 2000, the outlier-based Greedy algorithm significantly decreases performance. Conversely, the proposed algorithm takes the lead when the worker count is 2000 and maintains this lead as the worker count continues to increase. This trend suggests that the proposed algorithm is more suitable for application scenarios with large groups of workers. In the task count dimension, all algorithms demonstrate relatively stable performance. The proposed algorithm leads under all four task counts, suggesting its suitability for large-scale task allocation. This suggests that the algorithm has some advantages.

**Table 6:** SMAP the accuracy of worker allocation

| Number of workers | Greedy | KM | A-Greedy | A-KM | P-KM |
|---|---|---|---|---|---|
| 500 | **0.663** | 0.661 | 0.662 | 0.661 | 0.662 |
| 1000 | 0.661 | 0.661 | 0.661 | 0.661 | **0.661** |
| 1500 | 0.662 | 0.663 | 0.662 | 0.662 | **0.663** |
| 2000 | 0.662 | 0.662 | 0.662 | 0.662 | **0.663** |
| Number of tasks | Greedy | KM | A-Greedy | A-KM | P-KM |
| 500 | 0.66 | 0.659 | 0.659 | 0.659 | **0.661** |
| 1000 | 0.663 | 0.663 | 0.662 | 0.663 | **0.6631** |
| 1500 | **0.663** | 0.662 | 0.662 | 0.662 | 0.6622 |
| 2000 | 0.662 | 0.661 | 0.6622 | 0.662 | **0.6626** |

**Table 7:** SMAP the accuracy of task allocation

| Number of workers | Greedy | KM | A-Greedy | A-KM | P-KM |
|---|---|---|---|---|---|
| 500 | 0.991 | 0.9899 | 0.991 | 0.99 | **0.991** |
| 1000 | 0.99 | 0.991 | 0.99 | 0.991 | **0.9919** |
| 1500 | 0.991 | 0.99 | **0.9919** | 0.99 | 0.99 |
| 2000 | 0.9916 | 0.9915 | 0.9918 | 0.9917 | **0.9918** |
| Number of tasks | Greedy | KM | A-Greedy | A-KM | P-KM |
| 500 | 0.991 | 0.9915 | 0.99 | 0.992 | **0.9925** |
| 1000 | 0.9918 | 0.9926 | 0.9928 | 0.992 | **0.993** |
| 1500 | 0.992 | 0.991 | 0.992 | 0.991 | **0.993** |
| 2000 | 0.991 | 0.9915 | 0.9918 | 0.9917 | **0.993** |

### *4.2.4 Comparison of Execution Time*

Fig. 3 delineates the task execution time for each algorithm. The objective of task allocation in this study is to minimize time expenditure. As the number of workers escalates, the time cost for all algorithms increases. In the SMAP dataset, the proposed algorithm holds the lowest execution time cost. At a worker count of 1000, the average efficiency reaches its zenith at 20.70%, and at a task count of 500, the highest average efficiency is 21.84%. The efficiency improvement arises from the proposed algorithm utilizing a more efficient search strategy than the traditional algorithm. This reduction in time cost for bidirectional graph search contributes to the enhanced execution efficiency of the algorithm.

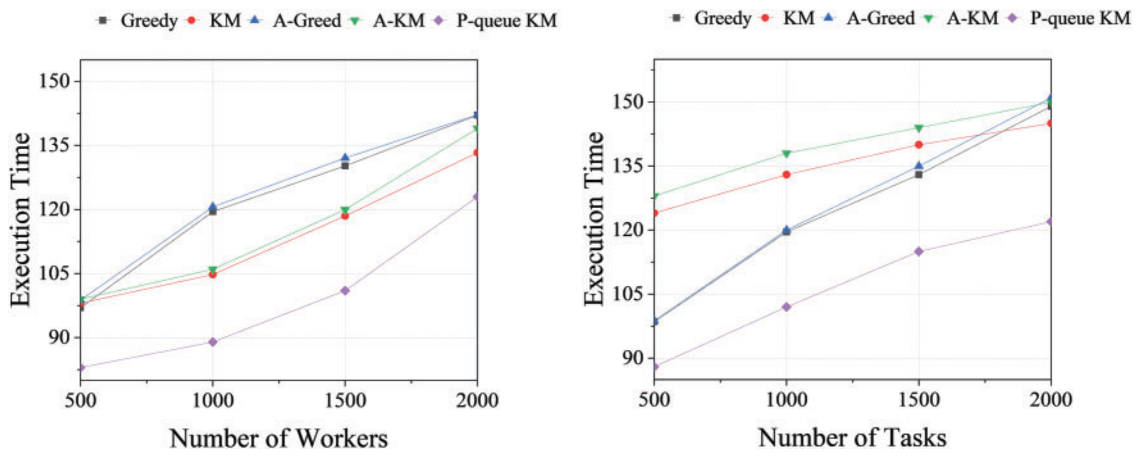

**Figure 3:** The total task execution time of SMAP dataset in different environments

After conducting a comparative analysis of the four groups of experiments mentioned above, it can be concluded that the P-queue KM algorithm proposed in this paper effectively optimizes the search operation and reduces time complexity. When dealing with large-scale data, this algorithm exhibits higher allocation efficiency compared to both the KM and greedy algorithms. Notably, comprehensive experiments are conducted in this study considering different dimensions such as the number of workers and tasks, providing an effective demonstration of the algorithm's excellent performance and validating its effectiveness.

## 5 Conclusion

This paper proposes a novel and efficient anomaly detection framework, namely dynamic self-attention invalid detection framework, which can effectively identify abnormal data nodes in the multivariate time series of crowdsourcing workflows, thus promoting efficient task allocation. The framework learns historical distribution patterns of multivariate time series data to train the detector model, thus capturing anomalous data in actual crowdsourcing data streams. Moreover, this paper proposes an efficient P-queue KM matching algorithm, significantly reducing the time complexity of matching search for large-scale data. Numerous experiments demonstrate that the framework can effectively recognize and handle abnormal data while enhancing the allocation accuracy of workers and tasks in task allocation. In future work, we will delve into developing a more robust model for detecting a minimal number of abnormal nodes in crowdsourcing data and seek to optimize the limitations of the allocation algorithm in specific scenarios.

**Acknowledgement:** In the research process of this paper, we thank Mr. Pan Qingxian for his careful guidance, and thank Mr. Yu Song and his laboratory classmates Fan Zequn and Li Jinru for their suggestions on the experimental content and article writing. It is their help that makes the research of this topic go smoothly. At the same time, I also thank the editors and reviewers of CMC journals, thank you for your review of the content of the article, and thank you for your valuable suggestions. Finally, thanks to the National Natural Science Foundation of China for funding this project.

**Funding Statement:** National Natural Science Foundation of China (62072392).

**Author Contributions:** The authors confirm contribution to the paper as follows: Study conception and design: Wei Kai, Pan Qingxian; data collection: Wei Kai; analysis and interpretation of results: Wei Kai, Pan Qingxian, Yu Song; draft manuscript preparation: Wei Kai, Pan Qingxian. All authors reviewed the results and approved the final version of the manuscript.

**Availability of Data and Materials:** Data not available due to researches' restrictions. Due to the nature of this research, participants of this study did not agree for their data to be shared publicly, so supporting data is not available.

**Conflicts of Interest:** The authors declare that they have no conflicts of interest to report regarding the present study.

## References

[1] Y. Tong, Z. Zhou, Y. Zeng, L. Chen, and C. Shahabi, "Spatial crowdsourcing: A survey," *VLDB J.*, vol. 29, no. 1, pp. 217–250, Jan. 2020. doi: 10.1007/s00778-019-00568-7.
[2] X. Yu, G. Li, Y. Zheng, Y. Huang, S. Zhang and F. Chen, "CrowdOTA: An online task assignment system in crowdsourcing," in *IEEE Int. Conf. Data Eng.*, Paris, France, Oct. 2018, pp. 1629–1632.
[3] F. Basik, B. Gedik, H. Ferhatosmanoglu, and K L. Wu, "Fair task allocation in crowdsourced delivery," *IEEE Trans. Serv. Comput.*, vol. 14, no. 4, pp. 1040–1053, Jul. 2018. doi: 10.1109/TSC.2018.2854866.
[4] L. Qiao, F. Tang, and J. Liu, "Feedback based high-quality task assignment in collaborative crowdsourcing," in *Proc. Int. Conf. Adv. Inf. Netw. Appl. AINA*, Aug. 2018, pp. 1139–1146.
[5] L. Wang, Z. Yu, Q. Han, B. Guo, and H. Xiong, "Multi-objective optimization based allocation of heterogeneous spatial crowdsourcing tasks," *IEEE Trans. Mob. Comput.*, vol. 17, no. 7, pp. 1637–1650, Nov. 2017. doi: 10.1109/TMC.2017.2771259.
[6] H. Sun, B. Dong, B. Zhang, W. H. Wang, and M. Kantarcioglu, "Sensitive task assignments in crowdsourcing markets with colluding workers," in *IEEE Int. Conf. Data Eng.*, ICDE, Oct. 2018, pp. 377–388.
[7] Y. Tong, L. Chen, Z. Zhou, H. V. Jagadish, L. Shou and W. Lv, "SLADE: A smart large-scale task decomposer in crowdsourcing," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 8, pp. 1588–1601, Jan. 2018. doi: 10.1109/TKDE.2018.2797962.
[8] A. Moayedikia, K. L. Ong, Y. L. Boo, and W. G. Yeoh, "Task assignment in microtask crowdsourcing platforms using learning automata," *Eng. Appl. Artif. Intell.*, vol. 74, no. 1, pp. 212–225, Jul. 2018. doi: 10.1016/j.engappai.2018.06.008.
[9] T. Zhao, Y. Yang, E. Wang, S. Mumtaz, and X. Cheng, "Task bundling in worker-centric mobile crowdsensing," *Int. J. Intell. Syst.*, vol. 36, no. 9, pp. 4936–4961, Jun. 2021. doi: 10.1002/int.22497.
[10] K. Tan and Q. Tao, "Market-driven optimal task assignment in spatial crowdsouring," *in Web-Age Information Management*, pp. 224–235, Jun. 2016. doi: 10.1007/978-3-319-47121-1.
[11] P. Cheng, X. Lian, L. Chen, and C. Shahabi, "Prediction-based task assignment in spatial crowdsourcing," in *Proc. Int. Conf. Data Eng.*, May 2017, pp. 997–1008.
[12] D. Li, J. Zhu, and Y. Cui, "Prediction-based task allocation in mobile crowdsensing," in *Int. Conf. Mob. Ad-Hoc Sens. Networks*, Shenzhen, China, Dec. 2019, pp. 89–94.
[13] Y. Zhao, K. Zheng, Y. Cui, H. Su, F. Zhu and X. Zhou, "Predictive task assignment in spatial crowdsourcing: A data-driven approach," in *Proc. Int. Conf. Data Eng.*, Apr. 2020, pp. 13–24.
[14] A. Ben Said, A. Erradi, A. G. Neiat, and A. Bouguettaya, "A deep learning spatiotemporal prediction framework for mobile crowdsourced services," *Mob. Netw. Appl.*, vol. 24, no. 3, pp. 1120–1133, Jun. 2019. doi: 10.1007/s11036-018-1105-0.
[15] Y. Zhao *et al.*, "Preference-aware task assignment in spatial crowdsourcing," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 2629–2636.

[16] X. Wei, B. Sun, J. Cui, and M. Qiu, "Location-and-preference joint prediction for task assignment in spatial crowdsourcing," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 42, no. 3, pp. 928–941, Jul. 2022. doi: 10.1109/TCAD.2022.3188960.

[17] X. Lian, W. He, L. Cui, H. Li, and L. Liu, "Task assignment with consensus decision for spatial crowdsouring," in *IEEE Int. Symp. Parallel Distrib. Process. Appl., IEEE Int. Conf. Big Data Cloud Comput., IEEE Int. Symp. Soc. Comput. Netw. IEEE Int. Conf. Sustain. Comput. Commun.*, Exeter, UK, Dec. 2020, pp. 713–720.

[18] Y. Ding, L. Zhang, and L. Guo, "Dynamic delayed-decision task assignment under spatial-temporal constraints in mobile crowdsensing," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 4, pp. 2418–2431, Mar. 2022. doi: 10.1109/TNSE.2022.3163925.

[19] L. Wang *et al.*, "Towards robust task assignment in mobile crowdsensing systems," *IEEE Trans. Mob. Comput.*, vol. 22, no. 7, pp. 4297–4313, Feb. 2022. doi: 10.1109/TMC.2022.3151190.

[20] X. Chen, Y. Zhao, K. Zheng, B. Yang, and C. S. Jensen, "Influence-aware task assignment in spatial crowdsourcing," in *Proc. Int. Conf. Data Eng.*, May 2022, pp. 2141–2153.

[21] Y. Zhao, T. Lai, Z. Wang, K. Chen, H. Li and K. Zheng, "Worker-churn-based task assignment with context-lstm in spatial crowdsourcing," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 9, pp. 9783–9796, Sep. 2023. doi: 10.1109/TKDE.2023.3249828.

[22] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," *in Advances in Cryptology–CRYPTO 2010*, pp. 465–482, Aug. 2010. doi: 10.1007/978-3-642-14623-7.

[23] U. Gadiraju, R. Kawase, S. Dietze, and G. Demartini, "Understanding malicious behavior in crowdsourcing platforms: The case of online surveys," in *Conf. Hum. Fact. Comput. Syst. Proc.*, Apr. 2015, pp. 1631–1640. doi: 10.1145/2702123.

[24] M. Spurling, C. Hu, H. Zhan, and V. S. Sheng, "Anomaly detection in crowdsourced work with interval-valued labels," *Commun Comput. Info. Sci.*, vol. 1601, pp. 504–516, Jul. 2022.

[25] Y. Zhao *et al.*, "Outlier detection for streaming task assignment in crowdsourcing," in *Proc. ACM Web Conf.*, Apr. 2022, pp. 1933–1943.

[26] T. Kieu *et al.*, "Anomaly detection in time series with robust variational quasi-recurrent autoencoders," in *Proc. Int. Conf. Data Eng.*, May. 2022, pp. 1342–1354.

[27] X. Chen, L. Deng, Y. Zhao, and K. Zheng, "Adversarial autoencoder for unsupervised time series anomaly detection and interpretation," in *Proc. ACM Int. Conf. Web Search Data Min.*, Feb. 2023, pp. 267–275.

[28] G. Li and J. J. Jung, "Deep learning for anomaly detection in multivariate time series: Approaches, applications, and challenges," *Inf. Fusion*, vol. 91, no. 2, pp. 93–102, Oct. 2023. doi: 10.1016/j.inffus.2022.10.008.

[29] S. Tuli, G. Casale, and N. R. Jennings, "TranAD: Deep transformer networks for anomaly detection in multivariate time series data," in *Proc. VLDB Endow*. vol. 15, no. 6, pp. 1201–1214, Feb. 2022. doi: 10.14778/3514061.3514067.

[30] A. Chevrot, A. Vernotte, and B. Legeard, "CAE: Contextual autoencoder for multivariate time-series anomaly detection in air transportation," *Comput. Secur.*, vol. 116, no. 5, pp. 102652, Feb. 2022. doi: 10.1016/j.cose.2022.102652.

[31] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "USAD: Unsupervised anomaly detection on multivariate time series," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, Aug. 2020, pp. 3395–3404.

[32] S. Sattar, S. Li, and M. Chapman, "Probabilistic-based crowdsourcing technique for road surface anomaly detection," *Int. Arch. Photogramm., Remote Sens. Spat. Inf. Sci.*, vol. XLIII-B4-2022, no. B4–2022, pp. 281–286, May 2022. doi: 10.5194/isprs-archives-XLIII-B4-2022-281-2022.

[33] X. Zheng, D. Yu, C. Xie, and Z. Wang, "Outlier detection of crowdsourcing trajectory data based on spatial and temporal characterization," *Math.*, vol. 11, no. 3, pp. 620, Jan. 2023. doi: 10.3390/math11030620.

[34] V. Q. Nguyen, V. H. Nguyen, N. A. Le-Khac, and V. L. Cao, "Clustering-based deep autoencoders for network anomaly detection," *in Future Data and Security Engineering* , pp. 290–303, 2020. doi: 10.1007/978-3-030-63924-2.

[35] S. Chen and W. Guo, "Auto-encoders in deep learning—a review with new perspectives," *Math.*, vol. 11, no. 8, pp. 1777, Apr. 2023. doi: 10.3390/math11081777.

[36] A. Takiddin, M. Ismail, U. Zafar, and E. Serpedin, "Deep autoencoder-based anomaly detection of electricity theft cyberattacks in smart grids," *IEEE Syst. J.*, vol. 16, no. 3, pp. 4106–4117, Jan. 2022. doi: 10.1109/JSYST.2021.3136683.

[37] I. Goodfellow *et al.*, "Generative adversarial nets," *Adv. Neural Inf. Process. Syst.*, vol. 27, pp. 2672–2680, Jun. 2014.

[38] E. Brophy, Z. Wang, Q. She, and T. Ward, "Generative adversarial networks in time series: A systematic literature review," *ACM Comput. Surv.*, vol. 55, no. 10, pp. 1–31, Feb. 2023. doi: 10.1145/3559540.

[39] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," *Neurocomputing*, vol. 452, pp. 48–62, Apr. 2021. doi: 10.1016/j.neucom.2021.03.091.

[40] G. Xu, Q. Zhang, Z. Song, and B. Ai, "Relay-assisted deep space optical communication system over coronal fading channels," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 59, no. 6, pp. 8297–8312, Aug. 2023. doi: 10.1109/TAES.2023.3301463.

[41] L. Qu, G. Xu, Z. Zeng, N. Zhang, and Q. Zhang, "UAV-assisted RF/FSO relay system for space-air-ground integrated network: A performance analysis," *IEEE Trans. Wirel. Commun.*, vol. 21, no. 8, pp. 6211–6225, Feb. 2022. doi: 10.1109/TWC.2022.3147823.

[42] N. C. Ristea *et al.*, "Self-supervised predictive convolutional attentive block for anomaly detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2022, pp. 13576–13586.

[43] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Dec. 2018, pp. 7132–7141.

[44] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, Aug. 2018, pp. 387–395.

[45] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Int. Conf. Learn. Represent.,* , Apr. 2014, pp. 1. doi: 10.48550/arXiv.1312.6114.