Coverage hole detection in WSN with force-directed algorithm and transfer learning

Yue-Hui Lai1 · Se-Hang Cheong1 · Hui Zhang2 · Yain-Whar Si1

Accepted: 24 July 2021 / Published online: 12 August 2021
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract
Coverage hole detection is an important research problem in wireless sensor network research community. However, distributed approaches proposed in recent years for coverage hole detection problem have high computational complexity. In this paper, we propose a novel approach for coverage hole detection in wireless sensor networks called FD-TL (Force-directed and Transfer-learning) which is based on layout generation capability of Force-directed Algorithms and image recognition power of Convolutional Neural Network with transfer learning. In contrast to existing approaches, the proposed approach is a pure topology-based approach since FD-TL can detect both triangular and non-triangular coverage holes from a wireless sensor network based on the input network topology without relying on the physical locations of the anchor nodes. In FD-TL, a Force-directed Algorithm is used to generate a series of possible layouts from a given input topology. Next, a Convolutional Neural Network is used to recognize potential coverage holes from the generated layouts. During the training phase, a transfer learning method is used to aid the recognition process. Experimental results show that FD-TL method can achieve 90% sensitivity and 96% specificity for coverage hole detection in wireless sensor networks.

Keywords Wireless sensor networks · Coverage hole detection · Force-directed algorithm · Convolutional neural network · Transfer learning

1 Introduction

In recent years, wireless sensor networks (WSN) are increasingly used in various applications involving military aviation [1], residential properties [2], health care [3], and among others. A WSN is composed of multiple sensors or edge devices with perceptive ability. The aim of each device is to collect and process the perceived objects’ information [4].

Various topologies are used in WSNs. These topologies include star topology, tree topology, and mesh topology [5]. There are two types of mesh topology; partial mesh topology and fully connect topology [6]. In this paper, we focus on the WSNs with partial mesh topology. In WSNs, data is transmitted directly from one edge device to another which is located within the communication range. When the battery power of an edge device is exhausted or the distance between the device and its neighbours exceeds the communication range, a collection of such devices could form a coverage hole within a WSN. Due to these coverage holes, a WSN may not be able to accurately collect information of edge devices in the affected area. Therefore, it is crucial to detect and rectify coverage holes [7].

A simplified example of coverage holes in a WSN is depicted as Fig. 1. For the purpose of simplification, in this paper, we use the term “nodes” to represent the sensors or edge devices from the WSNs. In Fig. 1(a), the nodes of the WSN are depicted in pink dots. Next, by using the sensing range of the node ($R_S$) as the radius, we can draw pink circles around each node. These circles are depicted in Fig. 1(b). From Fig. 1(b), we can observe that some of the areas are not covered in pink colour. These areas
represents coverage holes in the WSNs. Finally, based on this observation, we coloured the detected coverage hole in Fig. 1(c) using a different colour, which consists of edges surrounding the coverage hole and corresponding boundary nodes.

In this paper, we propose a novel approach called FD-TL (Force-directed and Transfer-learning) which only requires information about the unique identification of nodes (node ID) and the list of edges connecting the nodes in a WSN. In contrast to existing approaches [8, 9], FD-TL does not use physical location of nodes generated by GPS during the coverage hole detection process. The proposed FD-TL method is a centralized approach and can be used to detect both triangular and non-triangular coverage holes effectively.

FD-TL is based on Force-directed (FD) algorithm and Convolutional Neural Network (CNN). FD algorithms are flexible, intuitive and easy to implement. FD algorithms can be used to generate high-quality layouts of the input topologies without the need of node location information [10]. CNNs were first developed in the 1980s. CNNs had not attracted broad attention and were not widely used until 2012. In 2012, AlexNet [11] broke the image classification record in ImageNet challenge [12]. Since then, CNNs are increasingly used in various applications.

The coverage hole detection process of proposed FD-TL method is depicted in Fig. 2. In FD-TL, a possible layout of a wireless sensor network is generated by the FD algorithm from a given input network topology. A generated layout can be considered as a potential node distribution (estimation) among hundreds of such possibilities. Next, a trained CNN model is used to detect coverage holes from the layout and output the detection results.

Specifically, the CNN model performs two tasks in the FD-TL: detecting the position of the coverage holes and segmenting the coverage hole’s boundary to identify the identity of the nodes along the border of the coverage hole. In this paper, we use Mask R-CNN [13], Mask R-CNN with FPN [13], TensorMask [14], PointRend [15] and BlendMask [16] respectively for coverage hole detection. These methods fall into the category of semantic segmentation algorithms. The reason behind adopting the semantic segmentation algorithms instead of object detection algorithms is that semantic segmentation algorithm can achieve object classification at pixel level. Specifically, by using a semantic segmentation algorithm, each pixel of the image could be used to identify whether the specific pixel belongs to a coverage hole or not. Therefore, compared to the bounding boxes generated by the object detection algorithms, semantic segmentation algorithms could more precisely identify the boundary of the coverage holes from a wireless sensor network. In FD-TL method, we train CNN models by using the transfer learning method [17]. Transfer learning method is widely used in image recognition and computer vision applications when the training data is insufficient. Transfer learning can also be used to save training time and computing resources. In FD-TL, pre-trained CNN models of Microsoft COCO data set [18] are fine-tuned with our own coverage hole detection training data set. Since there is no coverage hole data set available in the public domain for WSNs, we create a labeled data set for training the CNN models from scratch.

The main contributions of our approach are as follows:

- A novel method called FD-TL for coverage hole detection in WSN is proposed, which can achieve high...
sensitivity and specificity for coverage hole detection in WSNs.

- FD-TL is based on a hybridization of deep learning approach and information visualization.
- An approach for generation of datasets for training deep learning models is proposed in this paper.
- To alleviate the problem of insufficient training data, a transfer learning method was also used in FD-TL.
- FD-TL is efficient in detecting holes from a generated layout. According to experimental results, FD-TL can detect coverage holes from a generated layout within 1 second.

The rest of the paper is organized as follows. In Section 2, we review the related work. In Section 3, we introduce the overview of proposed FD-TL method. In Section 4, coverage hole detection process by FD-TL is detailed. In Section 5, experiments are conducted to evaluate to effectiveness of the proposed approach. In Section 6, we conclude the paper with future work.

2 Related work

In WSNs, all nodes have similar status [19]. These nodes are often equipped with battery power to operate. However, the lifespan of the batteries are limited. Since replacement cannot be immediately carried out in many application scenarios, the affected nodes may no longer function in these networks. Such situations can lead to the formation of coverage holes in WSNs.

Coverage holes can be also caused by external factors such as earthquake or thunderstorm. In these situations, some of the nodes could be entirely destroyed or malfunctioned. Coverage holes formed during such situations may affect the nodes’ role in monitoring and transmitting of collected data. Therefore, detecting coverage holes is one of the important research problems in WSNs.

Existing coverage hole detection methods can be roughly categorized as geometrical, statistical, and topological methods respectively [20]. Geometrical methods utilize the geographic coordinates of the node [21]. Topological methods utilize the connectivity information of the network. However, these methods do not require exact node location information for detection [21]. In the statistical methods, the distribution of nodes follows certain statistical functions [22].

2.1 Geometrical methods

Geometrical methods assume that the location of the nodes and the relative distance between the nodes can be obtained by using Global Positioning System (GPS) or other positioning devices. In these methods, geometric figures such as Voronoi diagram and Delaunay triangulations are used to identify coverage holes with location information of the nodes [21].

In [23], Meguerdichian et al. proposed a centralized computation model. By combining the computational geometry and graph theoretic techniques such as Voronoi diagram, an optimal polynomial time algorithm was established for coverage calculation. In [24], So and Ye used Voronoi diagrams in a centralized manner to solve the coverage hole detection problem in WSNs. Each node was used to detect the vertices of the Voronoi cells. Next the Voronoi diagram was used for coverage hole detection. Zhang et al. [8] proposed a kind of Voronoi diagram-based screening strategy to detect coverage holes. In their work, Zhang et al. introduced a new virtual edge method to compute the boundary nodes. Information about the coverage hole’s position and shape was also determined by the proposed method. In [9], Soundarya et al. proposed a Delaunay-based method that incorporated the virtual edge, which could detect the coverage holes more accurately than existing tree-based methods [25]. In [26], Ma et al. proposed a computational geometry approach based on distributed coverage hole detection protocol to detect the coverage holes in a post deployment scenario, where the sensing range of each node was considered to be uniform. However, all of these geometrical methods require node location information.

In summary, in geometrical methods, it is necessary to obtain the geographic information of the node to detect the coverage holes. When exact node location information is unavailable, geometrical methods cannot detect the coverage hole effectively. Therefore, geometrical methods are not suitable for coverage hole detection when the geographic information is unavailable.

2.2 Statistical methods

Statistical methods for coverage hole detection assume that node distribution follows certain statistical functions [22]. In [27], the optimal node mobility strategy and corresponding target mobility strategy were used to improve the coverage of sensor networks. In this approach, Liu et al. assumed that the WSN is composed of a random distribution of nodes. In [28], Fekete et al. proposed an approach to identify the boundary nodes by applying methods from stochastic, topology and geometry. The statistical arguments were also used to to determine the thresholds to differentiate boundary nodes. The approach proposed in [28] aimed to solve the coverage hole detection problem of large-scale dense sensor network. It was developed for determining the region’s topology, Voronoi diagram of boundaries, and the outside boundary. However, their approach can only be
applied to wireless sensor network with high-density node distribution.

2.3 Topological methods

Topological methods do not require the node’s exact geographic information for coverage hole detection. Instead, the distance and connectivity information between different nodes are used for coverage hole detection [21].

In [29], Kanno et al. applied a coverage hole-equivalent planar graph to preserve the coverage hole position. Next a maximal simplicial complex was constructed, which contained the coverage hole information. The approach proposed by Kanno et al. can be used for coverage hole detection in WSNs with or without node position information. In [30], Ramazani et al. proposed a centralized coverage hole detection algorithm. In their approach, a coverage planar graph of the WSN was constructed, and the locations of the nodes were estimated using received signal strength. Next, simplicial complex was adopted to identify coverage hole boundaries from the coverage planar graph. In [31], De Silva et al. proposed a centralized coverage hole detection algorithm based on the homology of a simplicial pair called Vietoris–Rips complexes. The Rips complex was used to check the coverage by verifying the first homology group of the Rips complex. However, the algorithm proposed in [31] could not detect triangular coverage holes effectively.

In [32], a distributed approach was proposed. In this approach, homology computation of the Rips complex and the combinatorial Laplacian flows were used to verify the existence of coverage holes and obtain the coverage hole position. In [33], Yan et al. proposed a homology-based distributed coverage hole detection method. The proposed method could obtain the location of coverage holes and effectively detect the nontriangular coverage holes. However, the proposed method could not detect triangular coverage holes effectively, and the complexity was high. In [34], Yan et al. proposed a simplicial complex reduction algorithm for coverage hole detection. Although the detection accuracy was high for non-triangular coverage holes, it could not detect triangular coverage holes accurately.

Therefore, to alleviate the shortcomings of the existing approaches, we propose a novel coverage hole detection method called FD-TL in this paper. The proposed approach is designed to detect triangular and nontriangular coverage holes effectively. In our approach, we use FD algorithms to generate the potential (estimated) layouts of wireless sensor network. Next, we exploit the image recognition capabilities of CNNs to detect coverage holes. Since FD-TL is based on estimated layouts, our method does not depend on the location information generated by GPS. Specifically, the proposed approach is entirely based on the information of nodes’ identity and edges connecting these nodes.

3 Force-directed algorithms and convolutional neural networks

In this section, we briefly review the FD algorithms and CNNs adopted in our approach.

3.1 Force-directed algorithms

FD algorithms were applied in many application areas such as information visualization, sensor networks [35–37], graph drawing, routing algorithms, scheduling, etc [38]. In addition, FD algorithms are flexible, intuitive, and can be easily implemented. They can produce relatively good quality graphs from the network topology alone without the information of nodes’ physical location [10]. Due to these reasons, we have adopted FD algorithms for solving coverage hole detection problem in WSNs.

There are several models of FD algorithms, such as accumulated force models, energy function minimization models, and combinatorial optimization models [38]. Accumulated force models follow the simulation of a spring system. Among the accumulated forces models, there are Eades algorithm [39], Fruchterman-Reingold (FR) algorithm [40], and ForceAtlas2 (FA2) algorithm [41]. Energy function minimisation models use the spring system to minimise the difference between the visual distance and theoretical graphed distance. Kamada-Kawai (KK) algorithm [42] is a well-known FD algorithm which was designed based on the Energy function minimisation model. Combinatorial optimisation models are probabilistic algorithms and often inspired by evolutionary mechanisms. Combinatorial optimization models include Davidson-Harel (DH) algorithm [43] and Kudelka algorithm [44]. In our previous work, we adopted several FD algorithms for evaluation, such as ForceAtlas2 (FA2) algorithm, Kamada Kawai with multiple node selection and decaying stiffness (KK-MS-DS) algorithm [10] and Davidson Harel (DH) algorithm. Because the quality of WSN layout generated by KK-MS-DS algorithm is better than the layout generated by the FA2 and DH algorithms in WSNs, we adopt KK-MS-DS algorithm in the FD-TL method.

The objective of KK-MS-DS algorithm is to push the nodes in the outer boundary away from the inner nodes based on a decaying stiffness \( m \) assigned to the nodes. The higher the decaying stiffness value \( m \), the further the distance the node can be moved.

The simplified control flow of KK-MS-DS algorithm is depicted in Fig. 3. First, a node with the highest average
degree is selected as the starting node $s$. Second, 2-hop neighbors of starting point $s$ are then selected to construct the starting area ($WT$). Third, the stiffness $m$ of the nodes from the starting area $WT$ are updated as time progresses (see (1)).

$$m' = m - z p^t$$

(1)

where $t$ is the number of times the node has been selected for updating. $p$ is the decay rate and $z$ is the remaining energy of the node.

Fourth, nodes from the outer area are inserted into $WT$ if the stable status $r$ of the starting area is less than the threshold $\varepsilon$. The radius $r$ can be calculated as follows (see (2)).

$$r = \frac{1}{l} \sum_{i=1}^{l} |\hat{L}_i - L_i|$$

(2)

where $l$ is the total number of edges. $\hat{L}_i$ is the edge length on the intermediate iteration and $L_i$ is the edge length on the input network topology. $\sigma$ is the estimated difference (in terms of the distance) between the input network topology and an intermediate iteration. KK-MS-DS algorithm terminates when $WT$ is large enough to cover the entire graph. Finally, $WT$ is fine turned and KK-MS-DS algorithm returns the layout of $WT$. 

---

Fig. 3 A simplified control flow of KK-MS-DS algorithm
The proposed FD-TL relies on KK-MS-DS for generating the layouts of the wireless sensor network from a given input topology. The worst-case complexity of the KK-MS-DS algorithm is $O(n^3)$, where $n$ is the number of nodes in the network.

### 3.2 Convolutional neural networks

CNN is a kind of deep neural network from the area of Artificial Intelligence. CNN was first developed in the 1980s and gained attention in the field of image processing and computer vision. CNNs have been successfully used in various applications such as object detection, semantic segmentation, and instance segmentation [45–47]. CNNs can detect and recognize general patterns of the objects from an input image [48]. Particularly, CNNs can be used for object localization of remote sensing images, fault diagnosis, and classification in WSNs.

In [49], Long et al. adopted CNNs to detect objects and solve the problem of localization in remote sensing images. In [50], Hou et al. made fault diagnosis of rolling bearing based on CNNs and WSN, which improved the reliability of the machinery and operating efficiency. In [51], Tong et al. proposed a CNN-based approach to improve the accuracy of event classification in homogeneous sensor networks. In the proposed FD-TL, five CNN models namely Mask R-CNN [13], Mask R-CNN with FPN [13], TensorMask [14], PointRend [15] and BlendMask [16] are adopted for coverage hole detection in WSNs. In the coverage hole detection process, CNNs are used to detect both coverage holes’ positions and boundaries from the WSNs. In the following paragraphs, we briefly review each CNN models used in the FD-TL.

**Mask R-CNN**

Mask R-CNN [13] is a kind of two-stage framework. First, it scans the image and extracts the feature maps by a backbone network. Region Proposal Network (RPN), which is a lightweight neural network looks for the areas where there is a target and generates region proposals [52]. After getting the final region proposals, it passes the final region proposals to a Region of Interest Align (RoIAlign) layer. Second, it makes classification and boundary box regression to obtain the precise position of the bounding box and generates the mask [13]. Compared with Faster R-CNN [52], Mask R-CNN replaces RoIPooling with RoIAlign layer. It also adds a parallel branch network, which outputs a binary mask. RoI Pooling is not pixel-to-pixel alignment, and it leads to inaccurate result in mask generation. To solve the problem, RoIAlign uses bilinear interpolation to find more precise region [13], which could get a better match between the feature map and the original image. The binary mask is a matrix. The pixels that belong to the target object are represented by 1 and the others are represented by 0. The binary mask indicates whether a given pixel is part of the target object, which could make more accurate object segmentation.

**FPN**

In the object detection tasks, CNN cannot detect the small object effectively. This is because most of the algorithms only use neural network’s high-level feature maps to detect the object. Although the high-level feature map contains rich global information, it contains limited local information which is often important for small object detection. Feature Pyramid Network (FPN) [53] solves the problem of multiple-size object detection. FPN consists of two parts. One is a bottom-up process and another one is the fusion of top-down and lateral connections [53]. The bottom-up process is the same as a regular CNN. A CNN can be divided into different layers. Each layer corresponds to one level of the feature pyramid. The top-down process amplifies the high-level feature map to the same size as the previous layer by up-sampling. With up-sampling of the high-level feature map, it makes use of the abstract semantic features of the high layer for classification as well as the high-resolution information of the lower layer for location. A lateral connection structure is proposed in [53] to fuse low-level local features and high-level semantic features. Lateral connection fuses the high-level features with the same-resolution features of the previous layer after the high-level features have been up-sampled. By taking full advantage of high-level and low-level features, rich semantic information could be kept while maintaining high resolution in the same time, which greatly improves the effect of small object detection. Aiming to identify small coverage holes effectively, in this paper, we combine Mask R-CNN [13] with FPN. Specifically, FPN is used as the backbone of Mask R-CNN for feature extraction in FD-TL.

**TensorMask**

TensorMask [14] is a kind of dense sliding-window object segmentation algorithm. TensorMask uses structured high-dimensional tensors to represent the image in a set of dense sliding windows [14]. One head of TensorMask is to predict mask and it is responsible for generating mask in sliding window. The other head is used to make the classification and it is responsible for predicting the target category. In TensorMask, each spatial position’s output has its own spatial dimension, which is essentially different from Mask R-CNN. When TensorMask is tested with the Microsoft COCO data set [18], the results reveal that the performance of TensorMask and Mask R-CNN are similar.

**PointRend**

The aim of PointRend [15] is to optimize the sampling method and predict the object’s edge effectively. It is similar to the idea of computer graphics rendering. PointRend treats the segmentation problem as a similar
rendering problem. Non-uniform sampling is conducted in PointRend, and the segmentation result of each sampling point is calculated. The main idea is to render the image efficiently by computing only those points that are most likely to differ from the surrounding pixels. Then the result is mapped to a regular grid. An adaptive point selection strategy is then used to predict the segmentation labels [15].

BlendMask The overall architecture of BlendMask[16] contains a detector module and a BlendMask module. The detector module is FCOS [54]. The BlendMask module is composed of three parts. The first part is bottom module, which is used to process the feature maps extracted by the backbone network to predict the bases. Next the convolution layer which is on the detection towers predicts the top-level attention masks and bounding boxes [16]. Finally, blender module fuses the bases and the corresponding attentions to get the final prediction [16]. The performance of BlendMask is better than Mask R-CNN since BlendMask uses features with higher resolution and better information fusion. BlendMask can encode two kinds of information. One is semantic masks which can be used to judge whether a pixel belongs to an object. The other one is position-sensitive features which can judge whether a pixel is on object’s certain part [16]. BlendMask can learn richer feature representation in this way. The base of BlendMask module extracts position-sensitive features which could help to separate instances effectively. Semantic masks could also help to make more precise predictions. As a result, BlendMask performs better in the segmentation of instances. BlendMask fuses the advantage of different models. When testing with Microsoft COCO data set [18], the precision and speed of BlendMask exceed other single-stage instance segmentation algorithms.

4 FD-TL Method

In this section, we describe the details of the proposed FD-TL method.

4.1 Preliminaries

We outline the characteristics of the nodes, connections, and the type of coverage holes considered in our approach as follows:

- We assume that each node in the WSN has a unique identification number (node ID).
- The position of each node is random and their exact \( x \) and \( y \) coordinates are unknown.
- The sensing range (\( R_S \)) of nodes considered in our approach is uniform and can be defined by the users.
- A coverage hole is a polygonal shape consisting of nodes and edges. The interior of the coverage hole cannot contain intersecting edges.
- A coverage hole in a WSN must be a closed area and it must be surrounded by nodes.
- In FD-TL, the external coverage holes are not considered since they are actually the boundaries of the WSN.

FD-TL is based on the FD algorithm and CNN. The overall architecture of FD-TL is depicted as Fig. 4. First, CNN models are trained with the labelled input images. Second, FD algorithm (KK-MS-DS) is used to generate the layouts of the wireless sensor network from the given input topologies. Finally, the trained CNN models are used to detect the coverage holes from the generated layouts.

The activities performed in FD-TL can be described in 2 sessions, namely model training and testing processes. The overview of the activities performed in FD-TL is depicted

![Fig. 4 The overall architecture of FD-TL](image)
in Fig. 5. First, in the training session, we generate the training images by a prototype WSN layout generator called CNCAH [55] which was developed based on several state-of-the-art FD algorithms. Next, the training images are used to train CNN models. In the testing session, we generate the testing datasets using the same FD algorithms from the CNCAH generator. Then, coverage holes are detected from the testing datasets using trained CNN models. Finally, we compare the output image of CNN models with ground-truth image and evaluate the detection results.

4.2 Training phase

One of the most important requirements of training with CNN models is to prepare appropriate datasets for feature extraction. In the following sections, we detail the process of data generation for CNN models.

4.2.1 Training data set and validation data set preparation

To the best of authors’ knowledge, there is no WSN coverage hole dataset available in the public domain for training CNN models. Therefore, we prepare a labeled dataset for training the CNN model based on the following steps. First, we generate synthetic network topologies which contain randomly distributed nodes and corresponding edges by using a prototype system called CNCAH network generator [55].

FD is a kind of graph drawing algorithm. FD algorithms can generate visualizations based on the information of the nodes (i.e. node ID) and the edges connecting these nodes. In FD-Tl, FD algorithm’s output is the approximate layout of the wireless sensor network. Due to the nondeterministic nature of the FD algorithms, for a given input (topology), different layouts can be generated by the FD algorithm at a specific point in time. In other words, a layout could be considered as one of many possible snapshots of the WSN at a specific time point. Moreover, in majority of the FD algorithms, the layouts generated from the input topology can be exported into images or plain text files for post-processing. In our work, we adopt KK-MS-DS algorithm for evaluation. The training and validation data set preparation process of FD-TL is depicted in Fig. 6.

During the process, we transform each generated network topology into a bitmap image (1024 x 1024). Next, we draw a circle on each node using the sensing range of the node as radius in the bitmap image. In this paper, the sensing range of a node is defined as $R_S$. The sensing range of the node is a circle whose center is the node where the node is located [56]. In this paper, we regard the pixel distance of the WSN layout image as the measurement unit of sensing range of the node. We set the number of nodes $n$ per image as 500, 1000 and 2000 respectively. Ideally, the sensing range of the nodes should cover the entire WSN layout image ($1024 \times 1024$) without forming coverage holes. However, in most cases, the layout image of a WSN cannot be covered by the sensing range of the nodes. In our approach, $R_S$ is defined as follows:

$$\pi \times R_S^2 = \frac{1024 \times 1024}{n} \quad (3)$$

In (3), when $n$ is set to 500, the value of $R_S$ is equal to 26. By adopting this formulation, the layout image of a WSN can be covered by the whole sensing range of nodes when $n$ is set to 500, 1000 and 2000 respectively.

Based on the above definition, the example of a network topology, corresponding bitmap image, and the bitmap image with sensing range of the nodes are depicted in Fig. 7. The white areas which are not covered by pink circle represent the coverage holes. Based on this process, we could identify the position of coverage holes in the bitmap image.

---

1https://www.cis.um.edu.mo/~fstasp/tool_eric_CNCAHNetGenerator.html
By using the images generated from the above process, we label the coverage holes in the bitmap images manually with LabelMe tool [57] according to the definitions stated in Section 4.1. The image labeling process is depicted in Fig. 8, which contains 3 steps. First, we input the bitmap images into the LabelMe tool. Second, we label the coverage holes manually with the LabelMe tool. Third, the labeled image and annotation file are recorded with the nodes’ coordinates which are located on the boundaries of the coverage holes. Finally, we split the labeled data randomly into training and validation data set, which are used for training the CNN models. The number of labeled images used for training the CNN models is 800. The pixel size of the labeled images is $(1024 \times 1024)$. In the labeled images, the average number of coverage holes per image is 50. The average pixel size of a coverage holes is $(70 \times 70)$.

4.2.2 Training CNN models with transfer learning method

There are many successful applications of CNN in computer vision such as image classification, localization, and detection [58]. CNN is mainly composed of convolutional layers, pooling layers, and fully connected layers [59]. Convolution layer and pooling layer connect to each other alternately in the network. Convolution layer extracts the image’s features and outputs the feature map as input into the next layer. Pooling layer performs down-sampling to reduce the training parameters. Pooling layer is also used to avoid over-fitting and to make sure that the object’s global information is extracted in the final convolutional layer. The function of fully connected layer is to fuse the extracted feature map and to make the prediction.

In the coverage hole detection process, CNN detects both the positions of coverage holes and the boundary of the coverage holes. It means that CNN needs to effectively recognize the coverage holes through the bounding boxes and locate the precise pixels of each coverage hole to achieve pixel-level segmentation. Therefore, this paper uses Mask R-CNN [13], Mask R-CNN with FPN [13], TensorMask [14], PointRend [15] and BlendMask [16] respectively in FD-TL to detect the coverage holes.
Sufficient training datasets and effective feature extraction method are essential for designing a CNN model which contains a large number of parameters. Without sufficient datasets for training, the generalization ability of CNN could be significantly affected. In this situation, the model can be affected by irrelevant information such as noise from the training data. During the training period, it is also difficult for the loss function to converge. Besides, CNN model cannot learn effective features and information if there is insufficient training data. To alleviate the problem of insufficient training data, researchers have proposed an approach to make use of pretrained CNN models that have been trained on large amount of public data sets, and apply the structures and weights to solve the specific problem. This approach is commonly referred to as transfer learning [17]. In transfer learning, the universal feature representation and low-level features such as contours and textures are learnt by a pretrained model. After the low-level features have been extracted by the pretrained model, we fine tune the weight of higher layers by training with our own datasets. With transfer learning, a better training result can be obtained even though sufficient training datasets are unavailable. In addition, there are a large number of parameters needed to be trained and updated in CNN models. With transfer learning, we can also reduce the number of parameters needed to be trained and updated.

In our proposed method, Mask R-CNN [13] model, Mask R-CNN with FPN [13] model, TensorMask [14] model, PointRend [15] model and BlendMask [16] model are first pretrained with the Microsoft COCO data set [18]. Next, these models are fine-tuned with our own coverage hole detection training datasets which are obtained in the previous step. During the training phase, the validation data set is used to verify the performance of the trained CNN models. According to the validation results, we tune the hyperparameters of the CNN models and re-train the CNN models with the training data set if necessary. This process is repeated for many times until the validation result is better than a predefined threshold. Finally, we output the trained CNN models. The overall training process is depicted in Fig. 9.

### 4.3 Testing phase

The testing phase of FD-TL is depicted in Fig. 10. First, we generate the topology of WSNs by using the CNCAH network generator [55]. Next we input the topology of WSN into KK-MS-DS [42] algorithm. The topology of a WSN contains the node IDs and the list of edges. Then KK-MS-DS algorithm generates the layout of the given WSN from the input topology.

For any input WSN topology, KK-MS-DS algorithm is able to generate a layout at different time points, which
approximates the actual physical layout of the WSN. The generated layout is gradually improved as the execution time grows. KK-MS-DS algorithm adjusts the position of different nodes according to the force calculation, and the iteration process terminates when the nodes’ position remain unchanged. Each iteration produces a layout, and we use the layouts as the test dataset for coverage hole detection. The process of iteratively improving the generated layouts by KK-MS-DS algorithm is depicted in Fig. 11(a) and (b). The layouts of a WSN depicted in Fig. 11(a) and (b) are generated by the KK-MS-DS algorithm at the 20th and 100th second respectively. In this example, the topology contains 500 nodes and 1500 edges.

Next, we input the layouts of a wireless sensor network generated by KK-MS-DS algorithm into the trained CNN model for coverage hole detection. The sample output images which contain the coverage holes detected by a CNN model are depicted in Fig. 12(a) and (b).

4.4 Comparing the holes detected by CNN models with the ground truth

To evaluate the coverage hole detection results, we compare the coverage holes detected by CNN models with the ground truth. Recall that FD algorithms are designed to approximate the physical layouts of the input topology using calculated force (energy) value. As the execution progresses, KK-MS-DS algorithm iteratively adjust the locations of the nodes. Therefore, for any input topology, the layout of wireless sensor network generated by KK-MS-DS algorithm can vary at different execution time points. Although such adjustment is being carried out at each time point, the node ID and the connection information between different nodes remain the same for a specific input topology. We identify the node IDs from the boundary of the detected coverage holes. Next, the IDs of these coverage holes are compared against the coverage holes from the ground truth dataset. Pseudo code for identifying the ID
of the nodes along the coverage hole’s boundary from the ground truth and from the output image of the CNN model is listed in Algorithm 1. The overall procedure is depicted in Fig. 13. If the IDs of the nodes along the boundary in a detected coverage hole are the same as that of ground truth, the coverage hole is counted as correctly detected by the trained CNN models.

**Algorithm 1** Algorithm for identification of nodes’ ID along the coverage hole’s boundary.

Input: Ground-truth image, the output image of CNN model, network topology of wireless sensor network (T).

Output: The ID of nodes along the coverage hole’s boundary \((B - \text{nodes} - \text{ID})\) in the ground-truth image and the output image of CNN model.

// Variables \((V)\) initialization.
let \(V\) be the set of nodes in \(T\);
let coverage hole color be red in the ground-truth image;
let coverage hole color be blue in the output image of CNN model;
\(B - \text{nodes} = \{\}\);
\(B - \text{nodes} - \text{ID} = \{\}\);

// 1) Find the coverage hole contours from the ground-truth image and the output image of CNN model with color.
let \(C\) be the list of contours found from the ground-truth image and the output image of CNN model;
for each contour \(c\) in \(C\) do
  for each node \(v\) in \(V\) do
    if PointTest \((v, \text{contour}) = \text{TRUE}\) then
      // 2) Save the specific nodes along the coverage hole’s boundary.
      \(B - \text{nodes} \leftarrow \{v\}\);
    end
  end
// 3) Return the IDs of the nodes along the coverage hole’s boundary.
return \(B - \text{nodes} - \text{ID}\).

4.4.1 Evaluation criteria

The notations used in this paper are listed in Table 1. TP, TN, FP, and FN in Table 1 represent true positive, true negative, false positive, and false negative. The performance evaluation criteria are listed in Table 2. Note that we did not measure the precision criteria in the experiments because of the imbalanced data problem. In WSNs, the number of nodes belonging to coverage holes and the number of nodes not belonging to coverage holes are imbalanced. In other words, most of the nodes from the test dataset are not on the boundary of the coverage holes. Therefore, the number of TP and TN in datasets can vary significantly. Because of the imbalanced data, the precision of coverage hole detection cannot be used to reflect the CNN models’ real performance. Aiming to solve this problem, we use sensitivity and specificity as the benchmark criteria for evaluating the CNN’s performance. Sensitivity measures the proportion of true positives which are correctly classified, and specificity measures the proportion of true negatives that are correctly classified.

5 Experiments

In the experiments, we evaluate the performance of FD-TL based on the evaluation criteria outlined in the previous section. In the experiments, KK-MS-DS is combined with Mask R-CNN, Mask R-CNN with FPN, TensorMask, PointRend and BlendMask for evaluation.

5.1 Experiment settings

The experiments were conducted on computer with Intel Core i7-10875H processor, graph card NVidia Geforce RTX2070S with 16GB’s memory. The parameters used for configuration of CNN models are listed in Table 3.

In the experiments, the number of nodes per image is set to 500, 1000 and 2000 respectively. The average degree \(d\) is set to 6, 8 and 10 to generate the topologies. In total, 9 possible combination of topologies are generated by using CNCAH network generator [55].

In addition, KK-MS-DS algorithm is used to generate the layouts of these topologies which are stored as testing data set. Specifically, during the execution [42], KK-MS-DS algorithm improves and updates the layout iteratively, and the updating process terminates when the nodes’ position remain unchanged. Each iteration produces a layout, and we use the generated layouts as the testing data set for coverage hole detection with the trained CNN models. An example of a layout with \(n = 500\) and \(d = 6\), \(d = 8\) and \(d = 10\) are depicted in Fig. 14(a), (b) and (c) respectively, where the pink dots represent the location of each node.

5.2 Experiment results of WSNs with 500 nodes

In this experiment, the number of node \(n\) is set to 500, and the average degree \(d\) is set to 6, 8, and 10 respectively. The
experimental results from Table 4 and Fig. 15 show that FD-TL can obtain 85% sensitivity in all cases. Specifically, the sensitivity can reach to 90% when the average degree is 6 (see Fig. 14(a)). We can also observe that all CNN models can achieve relatively stable sensitivity during the coverage hole detection.

Experiment results in Table 4 and Fig. 16 show that FD-TL achieves at least 89% specificity for 5 different CNN models when \( n = 500 \). Specifically, when the average degree is 6, the specificity of FD-TL is similar for 5 different CNN models, which is above 96% (see Fig. 16(a)). As depicted in Fig. 16(c), when the average degree is 10, BlendMask outperforms TensorMask and PointRend. The specificity of Mask R-CNN with FPN structure is the same as that of Mask R-CNN without FPN structure, which has the worst performance. According to the experimental results depicted in Fig. 16(a) and (c), the specificity of FD-TL with a low average degree (\( d = 6 \)) is better than specificity of a high average degree (\( d = 10 \)).

In summary, all CNN models show a stable performance for coverage hole detection when \( n = 500 \). Sensitivity and specificity for different average degrees are above 85%. When average degree is 6, sensitivity and specificity of FD-TL are above 90% and 96% respectively.

5.3 Experiment results of WSNs with 1000 nodes

In this experiment, the number of node \( n \) is set to 1000, and the average degree \( d \) is set to 6, 8, and 10 respectively. The experimental results in Table 5 and Fig. 17 show that FD-TL can achieve approximately 79% sensitivity for all CNN models. Specifically, the sensitivity can reach to 90% when the average degree is 8 (see Fig. 17(b)), which is higher than the sensitivity when average degree is 6 and 10. According to the experimental results depicted in Figs. 17(a) and (c), the sensitivity of FD-TL with a low average degree (\( d = 6 \)) is better than the sensitivity of a high average degree (\( d = 10 \)).

The experimental results in Table 5 and Fig. 18 show that FD-TL can achieve approximately 84% specificity in majority of the cases when \( n = 1000 \). The specificity of FD-TL with a high average degree (\( d = 10 \)) is better than that of a low average degree (\( d = 6 \) and 8). When the average degree is 10, FD-TL can obtain above 92% specificity for all CNN models (see Fig. 18(c)). As depicted in Fig. 18(a), when the average degree is 6, TensorMask, PointRend and Mask R-CNN achieve similar performance and the specificity of BlendMask is approximately equal to that of Mask R-CNN with FPN structure.

When \( n = 1000 \), the sensitivity and specificity for different average degrees are above 79% and 84% respectively. When the average degree is 8, the sensitivity and specificity of FD-TL increase up to 89%.

5.4 Experiment results of WSNs with 2000 nodes

In this experiment, number of nodes \( n \) is set to 2000, and the average degree \( d \) is set to 6, 8, and 10 respectively.

5.4.1 Sensitivity

The experimental results in Fig. 19 reveal that five CNN models used in the experiment have varying performance.
Table 1  Notations

| Notation | Description |
|----------|-------------|
| $n$      | The number of nodes of the wireless sensor network. |
| $d$      | Average degree of the wireless sensor network, $d = 2(\text{num}_\text{edge}/\text{num}_\text{node})$. |
| $TP$     | The node which is on the detected coverage hole’s boundary as well as on the actual coverage hole boundary. |
| $TN$     | The node which is not on any of the detected coverage holes’ boundary. This node is also not on any of the actual coverage hole boundaries. |
| $FP$     | The node which is on the detected coverage hole’s boundary but not on the actual coverage hole boundary. |
| $FN$     | The node which is not on any of the detected coverage holes’ boundary. But, this node is on the actual coverage hole boundary. |

Table 2  Performance evaluation criteria

| Criteria            | Description                      |
|---------------------|----------------------------------|
| Sensitivity         | $Sensitivity = TP/(TP + FN)$     |
| Specificity         | $Specificity = TN/(TN + FP)$     |

Table 3  The parameters used for configuration of CNN models

| Configuration       | BlendMask | Mask R-CNN + FPN | Mask R-CNN | TensorMask | PointRend |
|---------------------|-----------|------------------|------------|------------|-----------|
| Activation Function | RELU      | RELU             | RELU       | RELU       | RELU      |
| Normalization       | Batch Normalization, Group Normalization | Batch Normalization, Group Normalization | Batch Normalization, Group Normalization | Batch Normalization, Group Normalization | Batch Normalization, Group Normalization |
| Base Learning Rate  | 0.01      | 0.01             | 0.01       | 0.01       | 0.01      |
| Warmup Iterations   | 1000      | 1000             | 1000       | 1000       | 1000      |
| Weight Decay        | 0.0001    | 0.0001           | 0.0001     | 0.0001     | 0.0001    |
| Backbone            | ResNet-50-FPN | ResNet-50-FPN | ResNet-50-C4 | ResNet-50-FPN | ResNet-50-FPN |
| Images_Per_Batch    | 2         | 2                | 2          | 1          | 2         |
| Input_Image_Size    | 1024*1024 | 1024*1024        | 1024*1024  | 1024*1024  | 1024*1024 |

Fig. 14  An example of a layout with $n = 500$ and (a) $d = 6$, (b) $d = 8$ and (c) $d = 10$
Table 4  Sensitivity and specificity of coverage hole detection when $n = 500$ and $d = 6$, $d = 8$ and $d = 10$

|                | Sensitivity |                  | Specificity |                  |
|----------------|-------------|------------------|-------------|------------------|
|                | $d=6$ | $d=8$ | $d=10$ | $d=6$ | $d=8$ | $d=10$ |
| BlendMask      | 0.90    | 0.85   | 0.86   | 0.96 | 0.94   | 0.93   |
| Mask R-CNN+FPN | 0.91    | 0.88   | 0.89   | 0.97 | 0.93   | 0.89   |
| Mask R-CNN     | 0.89    | 0.86   | 0.89   | 0.97 | 0.94   | 0.89   |
| TensorMask     | 0.88    | 0.86   | 0.89   | 0.97 | 0.96   | 0.91   |
| PointRend      | 0.89    | 0.86   | 0.89   | 0.97 | 0.95   | 0.90   |

for different average degree. From Table 6 and Fig. 19, we can observe that BlendMask algorithm achieves the best performance among all CNN algorithms. Specifically, the sensitivity obtained by BlendMask algorithm is above 90% for all settings of average degrees when $n = 2000$. It is because BlendMask can learn a richer feature representation by extracting position-sensitive features. Besides, BlendMask is able to perform better low-layer and high-layer information fusion. Since position-sensitive features are sensitive to different positions, BlendMask could judge whether a specific pixel is in a certain part of the coverage hole or not. This capability can help to separate the coverage holes better and as a result, the segmentation and performance in detection is significantly improved.

When Mask R-CNN uses FPN as its backbone for feature extraction network, its performance is better than Mask R-CNN without FPN structure. On average, the sensitivity of Mask R-CNN with FPN structure is 10% higher than the sensitivity of Mask R-CNN without FPN structure when average degree is set to 6, 8 and 10. Specifically,
sensitivity of Mask R-CNN with FPN structure is 89% when degree is 8, while the sensitivity of Mask R-CNN without FPN structure is 66% (see Fig. 19(b)). The reason behind the difference in performance is caused by the fact that FPN structure uses a multi-level feature map to detect the coverage holes of different sizes. The experiment results show that the FPN structure is indeed effective in detecting coverage holes from test datasets.

From the experiment results, we can also observe that the sensitivities of TensorMask algorithm when $d = 6,8$ and 10 are lower than 75% (see Fig. 19(a), (b), and (c)). According to the results depicted in Fig. 19(a) and (b), the performance of PointRend algorithm is the worst among all CNN algorithms tested. Compared with the previous experiments on $n = 500$ and 1000, the sensitivity of the PointRend algorithm drops sharply when $n = 2000$. It shows that the performance of PointRend is unstable. The overall performance of PointRend is poor when when $n$ is set to 2000. All in all, we can observe that the sensitivity of coverage hole detection for WSNs with $n = 2000$ is

![Fig. 16 Specificity of coverage hole detection when $n = 500$ and (a) $d = 6$, (b) $d = 8$ and (c) $d = 10$](image)

| Table 5 | Sensitivity and specificity of coverage hole detection when $n = 1000$ and $d = 6, d = 8$ and $d = 10$ |
|-------------|-----------------|-----------------|-----------------|
| $n=1000$ | Sensitivity | | Specificity |
| | d=6 | d=8 | d=10 | d=6 | d=8 | d=10 |
| BlendMask | 0.89 | 0.90 | 0.83 | 0.84 | 0.89 | 0.92 |
| Mask R-CNN+FPN | 0.89 | 0.89 | 0.84 | 0.84 | 0.92 | 0.96 |
| Mask R-CNN | 0.85 | 0.87 | 0.79 | 0.89 | 0.93 | 0.98 |
| TensorMask | 0.85 | 0.88 | 0.79 | 0.88 | 0.93 | 0.97 |
| PointRend | 0.85 | 0.88 | 0.79 | 0.89 | 0.93 | 0.97 |
Fig. 17  Sensitivity of coverage hole detection when \( n = 1000 \) and (a) \( d = 6 \), (b) \( d = 8 \) and (c) \( d = 10 \)

Fig. 18  Specificity of coverage hole detection when \( n = 1000 \) and (a) \( d = 6 \), (b) \( d = 8 \) and (c) \( d = 10 \)
less promising than WSNs with $n = 500$ and 1000. It is because node distribution is denser when $n = 2000$. As a result, CNN models such as Mask R-CNN, TensorMask and PointRend algorithm cannot maintain a stable performance. Among the five CNN models, BlendMask performs the best, whose sensitivity is approximately 90% when $n$ is 500, 1000 and 2000. PointRend performs the worst, whose sensitivity drops to 54% when $n$ is 2000.

### 5.4.2 Specificity

The experimental results listed in Table 6 and depicted in Fig. 20 show that the specificity for different average degree values varies greatly when $n = 2000$. Among all CNN models tested, PointRend has the highest specificity, but its sensitivity is very low. PointRend algorithm cannot correctly detect the coverage hole when $n = 2000$, and it

![Graph showing sensitivity of coverage hole detection](image)

#### Table 6  Sensitivity and specificity of coverage hole detection when $n = 2000$ and $d = 6$, $d = 8$ and $d = 10$

|       | Sensitivity |                      | Specificity |                      |
|-------|-------------|-----------------------|-------------|-----------------------|
|       |             | d=6  | d=8  | d=10  |             | d=6  | d=8  | d=10  |
| BlendMask | 0.91  | 0.91  | 0.90  | 0.74  | 0.82  | 0.91  |
| Mask R-CNN+FPN | 0.86  | 0.89  | 0.84  | 0.81  | 0.89  | 0.95  |
| Mask R-CNN | 0.81  | 0.66  | 0.78  | 0.83  | 0.93  | 0.97  |
| TensorMask | 0.72  | 0.75  | 0.74  | 0.86  | 0.93  | 0.97  |
| PointRend | 0.54  | 0.56  | 0.76  | 0.92  | 0.95  | 0.97  |

Fig. 19  Sensitivity of coverage hole detection when $n = 2000$ and (a) $d = 6$, (b) $d = 8$ and (c) $d = 10$
tends to predict the true coverage hole as non-coverage hole, thereby resulting in low sensitivity and high specificity. The performance of TensorMask algorithm and Mask R-CNN algorithm are similar when $n = 2000$ and the specificity is above 93% when the average degree is 8 and 10 (see Fig. 20(b) and (c)). Specificity of Mask R-CNN with FPN algorithm is high when average degree is 8 and 10, but its specificity is relatively low when the average degree is 6. From Fig. 20(c), we can observe that when the average degree is 10, the specificity of BlendMask can reach up to 91%. However, when the average degree is 6 and 8, the specificity are low, and it indicates that BlendMask tends to predict true non-coverage holes as coverage holes when $d = 6$ and 8, resulting in high sensitivity and low specificity.

All in all, when $n = 2000$, there is a significant difference in the sensitivity and specificity of five CNN models when $d = 6, 8$ and 10. When $n = 2000$, PointRend, TensorMask and Mask R-CNN cannot detect coverage holes precisely. In this experiment, BlendMask achieves the best performance.

In summary, we analyze the results of coverage hole detection using Mask R-CNN, Mask R-CNN with FPN, TensorMask, PointRend and BlendMask models in the experiments. Our analysis also reveals the reasons behind the different performance results of CNN models. According to the results, when $n = 500, 1000$ and 2000, BlendMask achieves the best result because it can learn a richer feature representation by extracting position-sensitive features. BlendMask also makes superior low-layer and high-layer information fusion, which allows the BlendMask to accurately detect and segment the coverage holes of different positions. The performance of Mask R-CNN with FPN structure is superior than that of Mask R-CNN without FPN structure, which proves the effectiveness of FPN structure in coverage hole detection. Mask R-CNN without FPN structure and TensorMask have relatively low sensitivity when $n = 2000$. Their sensitivity decreases when $n$ increases from 500 to 2000, which means that their performance is not stable enough for detecting coverage holes. We found that the performance of PointRend is unstable. PointRend cannot detect coverage holes effectively when the number of node is set to 2000. The experimental results show that the non-uniform
sampling and adaptive point selection method of PointRend did not perform well for the coverage hole detection task when \( n \) is set to 2000. From the experiment results, we found that the performance of CNN models for coverage hole detection deteriorates when the number of nodes \( n \) is increased. It is because when \( n \) is increased, node distribution becomes denser.

6 Conclusion

In this paper, we propose a novel coverage hole detection approach for wireless sensor network called FD-TL which is based on FD algorithm and CNN with transfer learning. To the best of authors’ knowledge, the proposed FD-TL is the first coverage hole detection method which is based on information visualization and deep learning approach. In addition, an approach for generating training datasets for coverage hole detection was proposed in this study. Transfer learning was also used to alleviate the problem of insufficient training data.

Experimental results show that the FD-TL method can effectively detect the coverage holes from WSNs. FD-TL method can achieve 90% sensitivity and 96% specificity for coverage hole detection in wireless sensor networks. In terms of efficiency, the proposed FD-TL method can detect coverage holes from an image within 1 second. By integrating information visualization technique and deep learning-based image classification method in FD-TL, we hope that the proposed approach opens the door for new research directions in the WSN community.

It should be noted that depending on the energy level of the node, the layout of a WSN could change during the monitoring. That is, some of the nodes could be undetectable or drop out completely if their battery power is too low. Note that the input data used for the proposed model is a simple topology information containing the ID of the nodes and their connection only. It does not contain time stamps or additional attributes from latest monitoring task. In other words, the input data could be considered as a current snapshot of a WSN and the proposed model is designed to detect holes from a given snapshot. Since our model is able to detect holes from a snapshot from a given time point, it can be easily extended to detect holes from continuous stream of snapshots (dynamic data) generated from the monitoring scenario. In addition, the FD algorithm (KK-MS-DS) used in the experiments can be easily replaced by a dynamic version which is designed for streaming data. The detailed designs of several dynamic FD algorithms have been reported [60]. As for the future work, we are planning to extend the FD-TL to deal with the dynamic data generated from a continuous monitoring activity. In addition, we are also planning to test methods from computer graphics area such as contour tracing algorithm for recognizing the holes from the generated layouts.

Acknowledgements This research was funded by University of Macau (File no. MYRG2019-00136-FST and MYRG2017-00029-FST).

References

1. Azzabi T, Farhat H, Sahli N (2017) A survey on wireless sensor networks security issues and military specificities. In: 2017 International conference on advanced systems and electric technologies (IC_ASET), pp 66–72. IEEE
2. Sun W, Yuan X, Wang J, Li Q, Chen L, Mu D (2017) End-to-end data delivery reliability model for estimating and optimizing the link quality of industrial wsns. IEEE Trans Autom Sci Eng 15(3):1127–1137
3. Mohanty P, Kabat MR (2016) Energy efficient reliable multi-path data transmission in wsn for healthcare application. Int J Wireless Inf Netw 23(2):162–172
4. Xueli C, Wenyuan C, Suhua C et al (2010) Notice of retraction: Role of wireless sensor networks in forest fire prevention. In: 2010 2nd international conference on computer engineering and technology, vol 4, pp V4–12. IEEE
5. Reina DG, Toral SL, Barrero F, Bessis N, Asimakopoulou E (2013) The role of Ad Hoc networks in the internet of things: a case scenario for smart environments. Springer, Berlin, pp 89–113
6. McGrath MJ, Scannell CN (2013) Sensor network topologies and design considerations. Apress, Berkeley, pp 79–95
7. Zhu C, Zheng C, Shu L, Han G (2012) A survey on coverage and connectivity issues in wireless sensor networks. J Netw Comput Appl 35(2):619–632
8. Zhang Y, Zhang X, Wang Z, Liu H (2013) Virtual edge based coverage hole detection algorithm in wireless sensor networks. In: 2013 IEEE wireless communications and networking conference (WCNC), pp 1488–1492. IEEE
9. Soundarya A, Santhi V (2017) An efficient algorithm for coverage hole detection and healing in wireless sensor networks. In: 2017 1st International conference on electronics, materials engineering and nano-technology (IEMENTech), pp 1–5. IEEE
10. Cheong S-H, Si Y-W (2016) Accelerating the kamada-kawai algorithm for boundary detection in a mobile ad hoc network. ACM Trans Sens Netw (TOSN) 13(1):1–29
11. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
12. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M et al (2015) Imagenet large scale visual recognition challenge. Int J Comput Vis 115(3):211–252
13. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 2961–2969
14. Chen X, Girshick R, He K, Dollár P (2019) Tensormask: A foundation for dense object segmentation. In: Proceedings of the IEEE international conference on computer vision, pp 2061–2069
15. Kirillov A, Wu Y, He K, Girshick R (2020) Pointrend: Image segmentation as rendering. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9799–9808
16. Chen H, Sun K, Zhi T, Chunhua S, Yongming H, Youliang Y (2020) Blendmask: Top-down meets bottom-up for instance segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 8573–8581
Coverage hole detection in WSN with force-directed algorithm and transfer learning

17. Pan SJ, Yang Q (2009) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10):1345–1359
18. Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick LC (2014) Microsoft coco: Common objects in context. In: European conference on computer vision, pp 740–755. Springer
19. EAA Abdulla A, Nishiyama H, Ansari N, Kato N (2014) Energy-aware routing for wireless sensor networks. In: The art of wireless sensor networks, pp 201–234. Springer
20. Wang Y, Gao J, Mitchell JSB (2006) Boundary recognition in sensor networks by topological methods. In: Proceedings of the 12th annual international conference on Mobile computing and networking, pp 122–133
21. Zhang J, Chu H, Feng X (2020) Efficient coverage hole detection algorithm based on the simplified rips complex in wireless sensor networks. J Sens 2020
22. Baghdad R, Lamraoui A (2016) Boundary and holes recognition in wireless sensor networks. J Innov Digit Ecosyst 3(1):1–14
23. Meguerdichian S, Koushanfar F, Potkonjak M, Srivastava MB (2001) Coverage problems in wireless ad-hoc sensor networks. In: Proceedings IEEE infocom conference on computer communications. twentieth annual joint conference of the, IEEE computer and communications society (Cat. No. 01CH37213), vol 3, pp 1380–1387. IEEE, 2001
24. Man-Cho So A, Ye Y (2005) On solving coverage problems in a wireless sensor network using voronoi diagrams. In: International workshop on internet and network economics, pp 584–593. Springer
25. Li W, Yuwei W (2016) Tree-based coverage hole detection and healing method in wireless sensor networks. Comput Netw 103:33–43
26. Ma H-C, Sahoo PK, Chen Y-W (2011) Computational geometry based distributed coverage hole detection protocol for the wireless sensor networks. J Netw Comput Appl 34(5):1743–1756
27. Liu B, Brass P, Dousse O, Nain P, Towsley D (2005) Mobility improves coverage of sensor networks. In: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing, pp 300–308
28. Fekete PS, Kröller A., Pfisterer D, Fischer S, Carsten B (2004) Neighborhood-based recognition in sensor networks. In: International symposium on algorithms and experiments for sensor systems, topology, wireless networks and distributed robotics, pp 123–136. Springer
29. Kanno J, Buchart JG, Selmic RR, Phoha V (2009) Detecting coverage holes in wireless sensor networks. In: 2009 17th Mediterranean conference on control and automation, pp 452–457. IEEE
30. Ramazani S, Kanno J, Rastko RS, Brust MR (2016) Topological and combinatorial coverage hole detection in coordinate-free wireless sensor networks. Int J Sens Netw 21(1):40–52
31. Silva VD, Ghrist R (2007) Coverage in sensor networks via persistent homology. Algebr Geom Topol 7(1):339–358
32. Tahbaz-Salehi A, Jadbabaie A (2010) Distributed coverage verification in sensor networks without location information. IEEE Trans Autom Control 55(8):1837–1849
33. Yan F, Vergne A, Martins P, Decreusefond L (2014) Homology-based distributed coverage hole detection in wireless sensor networks. IEEE/ACM Trans Netw 23(6):1705–1718
34. Yan F, Ma W, Shen F, Xia W, Shen L (2020) Connectivity based k-coverage hole detection in wireless sensor networks. Mob Netw Appl 25(2):783–793
35. Zou Y, Chakrabarty K (2003) Sensor deployment and target localization based on virtual forces. In: IEEE INFOCOM 2003. Twenty-second annual joint conference of the ieee computer and communications societies (IEEE Cat. No. 03CH37428), vol 2, pp 1293–1303. IEEE, 2003
36. Wang X, Wang S, Bi D (2007) Virtual force-directed particle swarm optimization for dynamic deployment in wireless sensor networks. In: International conference on intelligent computing, pp 292–303. Springer
37. Efrat A, Forrester D, Iyer A, G Kobourov S, Erten C, Kilic O (2010) Force-directed approaches to sensor localization. ACM Trans Sens Netw (TOSN) 7(3):1–25
38. Cheong S-H, Si Y-W (2020) Force-directed algorithms for schematic drawings and placement: A survey. Inf Vis 19(1):65–91
39. Eades P (1984) A heuristic for graph drawing. Congressus numerantium 42:149–160
40. Fruchterman TMJ, Reingold EM (1991) Graph drawing by force-directed placement. Softw Pract Exp 21(11):1129–1164
41. Jacomy M, Venturini T, Heymann S, Bastian M (2014) Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. PloS one 9(6):e98679
42. Kamada T, Kawai S et al (1989) An algorithm for drawing general undirected graphs. Inf Process Lett 31(1):7–15
43. Kamada T, Kawai S et al (October 1996) Drawing graphs nicely using simulated annealing. ACM Trans Graph 15(4):301–331
44. Kudelka M, Kromer P, Radvansky M, Horak Z, Snasel V (2015) Efficient visualization of social networks based on modified sahmsons mapping. Swarm Evol Comput 25:63–71
45. Lee D-H (2020) Cnn-based single object detection and tracking in videos and its application to drone detection. Multimed Tools Appl 1–12
46. Liu L, Wu F-X, Wang Y-P, Wang J (2020) Multi-receptive-field cnn for semantic segmentation of medical images. IEEE J Biomed Health Inf 24(11):3215–3225
47. Wang W, Lin H, Wang J (2020) Cnn based lane detection with instance segmentation in edge-cloud computing. J Cloud Comput 9:1–10
48. Patterson J, Gibson A (2017) Deep learning: A practitioner’s approach. " O’Reilly Media Inc."
49. Long Y, Gong Y, Xiao Z, Liu Q (2017) Accurate object localization in remote sensing images based on convolutional neural networks. IEEE Trans Geosci Remote Sens 55(5):2486–2498
50. Hou L, Li Z, Qu H (2020) Fault diagnosis of rolling bearing using wireless sensor networks and convolutional neural network. Int J Online Biomed Eng 16(11):32–44
51. Tong C, Li J, Zhu F (2017) A convolutional neural network based method for event classification in event-driven multi-sensor network. Comput Electr Eng 60:90–99
52. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems, pp 91–99
53. Lin T-Y, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2117–2125
54. Tian Z, Shen C, Chen H, Tong H (2019) Fcos: Fully convolutional one-stage object detection. In: Proceedings of the IEEE international conference on computer vision, pp 9627–9636
55. Cheong S-H, Si Y-W (2017) Boundary node detection and unfolding of complex non-convex ad hoc networks. ACM Trans Sensor Netw (TOSN) 14(1):1–34
56. Huang C-F, Tseng Y-C (2005) The coverage problem in a wireless sensor network. Mobile Netw Appl 10(4):519–528
57. Russell BC, Torralba A, Murphy KP, Freeman WT (2008) Labelme: a database and web-based tool for image annotation. Int J Comput Vis 77(1-3):157–173
58. Khan S, Rahmani H, Syed AAS, Bennamoun M (2018) A guide to convolutional neural networks for computer vision. Synth Lect Comput Vis 8(1):1–207
59. Ketkar N, Santana E (2017) Deep learning with python, vol 1. Springer, Berlin
60. Cheong S-H, Si Y-W, Wong RK (2021) Online force-directed algorithms for visualization of dynamic graphs. Inform Sci 556:223–255

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.