International Journal of Digital Earth

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/tjde20

Sponet: solve spatial optimization problem using deep reinforcement learning for urban spatial decision analysis

Haojian Liang, Shaohua Wang, Huilai Li, Liang Zhou, Hechang Chen, Xueyan Zhang & Xu Chen

To cite this article: Haojian Liang, Shaohua Wang, Huilai Li, Liang Zhou, Hechang Chen, Xueyan Zhang & Xu Chen (2024) Sponet: solve spatial optimization problem using deep reinforcement learning for urban spatial decision analysis, International Journal of Digital Earth, 17:1, 2299211, DOI: 10.1080/17538947.2023.2299211

To link to this article: https://doi.org/10.1080/17538947.2023.2299211

INTERNATIONAL JOURNAL OF DIGITAL EARTH
2024, VOL. 17, NO. 1, 2299211
https://doi.org/10.1080/17538947.2023.2299211

Sponet: solve spatial optimization problem using deep reinforcement learning for urban spatial decision analysis

Haojian Liang[a], Shaohua Wang[b,c,d], Huilai Li[a,e], Liang Zhou[f], Hechang Chen[a], Xueyan Zhang[g] and Xu Chen[a]

[a]School of Artificial Intelligence, Jilin University, Changchun, People's Republic of China; [b]Key Laboratory of Remote Sensing and Digital Earth Chinese Academy of Sciences, Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing, People's Republic of China; [c]International Research Center of Big Data for Sustainable Development Goals, Beijing, People's Republic of China; [d]Key Laboratory of Digital Earth Science, Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing, People's Republic of China; [e]School of Mathematics, Jilin University, Changchun, People's Republic of China; [f]Faculty of Geomatics, Lanzhou Jiaotong University, Lanzhou, People's Republic of China; [g]Viterbi School of Engineering, University of Southern California, Los Angeles, CA, USA

ABSTRACT

Urban spatial decision analysis is a critical component of spatial optimization and has profound implications in various fields, such as urban planning, logistics distribution, and emergency management. Existing studies on urban facility location problems are based on heuristic methods. However, few studies have used deep learning to solve this problem. In this study, we introduce a unified framework, SpoNet. It combines the characteristics of location problems with a deep learning model SpoNet can solve spatial optimization problems: p-Median, p-Center, and maximum covering location problem (MCLP). It involves modeling each problem as a Markov Decision Process and using deep reinforcement learning to train the model. To improve the training efficiency and performance, we integrated knowledge SpoNet. The results demonstrated that the proposed method has several advantages. First, it can provide a feasible solution without the need for complex calculations. Second, integrating the knowledge model improved the overall performance of the model. Finally, SpoNet is more accurate than heuristic methods and significantly faster than modern solvers, with a solution time improvement of more than 20 times. Our method has a promising application in urban spatial decision analysis, and further has a positive impact on sustainable cities and communities.

ARTICLE HISTORY
Received 15 June 2023
Accepted 20 December 2023

KEYWORDS
Urban spatial decision analysis; spatial optimization problems; p-Median; MCLP; attention model; deep reinforcement learning

1. Introduction

Urban spatial decision analysis plays a critical role in the development of sustainable cities and communities. With the acceleration of urbanization, cities have encountered lots of challenges, including traffic congestion, outdated infrastructure, and disaster risks. These challenges were caused by insufficient and uneven utilization of urban space, inadequate allocation of public facilities, impractical transportation planning, and immature emergency warning mechanisms. As an important strategy of sustainable urban development, spatial optimization maximizes the utilization and optimization

CONTACT Shaohua Wang wangshaohua@aircas.ac.cn Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100094, People's Republic of China

of available spatial resources through rational planning, design, and management. The urban Facility Location Problem (FLP) is a significant spatial optimization problem that plays an essential role in logistics, communication (Gao et al. 2022), and emergency management (Bleha and Ďurček 2023). The city-oriented facility location problem is a key component of urban spatial analysis, providing a feasible decision basis for urban planning (Zhao, Sailor, and Wentz 2018), urban management and design (Beairsto et al. 2022) by analyzing the layout and spatial distribution of existing facilities in the city (Guo et al. 2022). During this process, the influence of factors on facility construction should be considered, which can greatly satisfy urban residents' requirements and promote sustainable development (Huang et al. 2021), including traffic (Jiang et al. 2022; Xia et al. 2022), environment (Zhang et al. 2020), socioeconomic factors (Rabiei-Dastjerdi et al. 2021), policy and regulation.

In urban spatial decision analysis, p-Median (Hakimi 1965), coverage (Church,1972), and p-Center (Hakimi 1965) are three location problems that offer valuable insights and methodologies to aid in the process of identifying optimal locations. Typical methods for solving location problems can be categorized into three classes: exact, approximate, and heuristic algorithms. Exact algorithms are essentially an exhaustive enumeration of the solution space, which can provide the optimal solution to the problem. The typical algorithms include Branch and Bound, Lagrange Relaxation, and Column Generation. However, for a large-scale problem, it is infeasible to solve in polynomial time as the number of nodes increases and the solution space increases exponentially. Hence, scholars have suggested providing a practical solution to satisfy the need for practical problems without determining the best possible outcome (Williamson and Shmoys 2011). The approximate algorithm theoretically provides the existing gap between the approximate and optimal solutions. Heuristic and metaheuristic algorithms cannot theoretically address the gap between the achieved solution and the ideal optimal solution (Beheshti and Shamsuddin 2013), while they always can provide a feasible solution to large-scale practical problems and have diverse applications across various fields. The representative algorithms are genetic algorithms, simulated annealing, and variable neighborhood search.

The algorithms mentioned above have been widely applied to urban spatial decision analyses. Recently, some deep learning methods have been utilized to address the challenges of urban sustainable development and optimization problems. Although deep learning methods have been extensively employed to solve routing problems (Mazyavkina et al. 2021), such as the Traveling Salesman Problem (TSP) and Vehicle Routing Problem (VRP), few studies apply these techniques to location problems. Previous studies mainly focused on designing different heuristic methods to solve problems (Kaveh and Khayatazad 2012; Song and Chen 2018; Li, Li, and Ma 2022). These well-designed heuristic approaches can undoubtedly enhance performance, while they need to be redesigned for different problems and the performance of algorithms, which largely depend on the intuition and experience of experts. Deep learning can replace the handcrafted design of artificial rules to some extent (Mecheter et al. 2022; Xiao, Xu, and Wan 2016).

The neural-spatial optimization method (NeurSPO) utilizes deep learning to address spatial optimization problems. Deep construction and deep improvement are two main directions for utilizing the deep learning method to address spatial optimization problems. Deep construction refers to the initial generation or creation of solutions using deep learning techniques. Deep improvement pertains to the refinement or enhancement of these solutions through further deep learning processes. This study mainly focuses on the use of deep construction to realize an end-to-end solution to solve spatial optimization problems.

The primary contributions of our work are as follows.

- We focus on three classic FLPs, p-Median, p-Center and maximum covering location problem (MCLP) which are crucial for urban spatial decision analysis. It is challenging to obtain the optimal solution due to the NP hardness. Therefore, we propose a novel algorithm and design a unified structure-SpoNet for solving FLPs.

- We formulate the p-Median, p-Center and MCLP as Markov Decision Process (MDP) and adopt the Deep Reinforcement Learning (DRL) to train the SpoNet. Based on allocation relationship between nodes, we combined the characteristics of the FLPs to incorporate local features, order and distribution relations, and coverage information into the network models.
- Extensive benchmark experiments demonstrate that SpoNet outperforms heuristic methods, linear programming solvers, and a deep learning baseline in terms of solution quality and efficiency. The ablation study further emphasizes the effectiveness of dynamic coverage information and GRU structure in SpoNet.

To provide a clear and organized presentation, we structured the paper as follows. Section 2 provides an overview of related work in this field, while Section 3 presents the preliminary knowledge and problem description. In Section 4, we describe our methodology, which focuses on adopting reinforcement learning to train the model, and introduces SpoNet. We discuss the comparison analysis, ablation study, and transfer study in section 5. Finally, Section 6 concludes the paper.

2. Related work

In this section, we provide an overview of the relevant research in two main aspects: urban FLPs and deep learning for spatial optimization problems.

2.1. Urban facility location problems

Real-world urban spatial decision problems can be effectively tackled using optimization concepts from operations research, often transferred into p-Median, p-Center, and MCLP.

Hakimi (1964) proposed the single-facility p-Median and p-Center problem, and then this work was extended to the multi-facility situation (Hakimi 1965), which is the p-Median and p-Center problem. Subsequently, ReVelle and Swain (1970) proposed a discrete p-Median problem and the first MILP model. Teitz and Bart (1968) first proposed the interchange heuristic (or vertex substitution), which starts with a feasible p-Median candidate set and iteratively checks possible exchanges with other candidates until the optimal is reached. A heuristic method introduced by Alcaraz, Landete, and Monge (2012) for large-scale instances consists of three key components: subgradient column generation, one core heuristic, and one clustering process. Kazakovtsev and Stupina (2014) proposed a multistage method comprising clustering, VNS, and exact methods. MCLP was first introduced by Church and Velle (1974), who demonstrated a distinct definition, Integer Linear Programming (ILP) of MCLP, and greedy addition and greedy addition with substitution. Later, diverse variations in the MCLP and corresponding heuristic methods were proposed. Pirkul and Schilling (1989) used Lagrange Relaxation (LR) to solve a capacitated MCLP, Berman and Krass (2002) proposed a Lagrangian relaxation algorithm, and ReVelle, Scholssberg, and Williams (2008) utilized a metaheuristic method and heuristic concentration to solve the MCLP. In addition, there are many other algorithms for solving the MCLP, such as Benders decomposition, nonlinear programming, Tabu algorithm, and local search algorithm.

2.2. Deep learning for spatial optimization problems

The classical methods mentioned above for solving FLPs have been widely applied in practice. However, due to the NP-hard nature of these problems, these methods often provide only a feasible solution, particularly in large-scale scenarios. Hence, there is an urgent need to explore novel approaches to solving FLPs. Deep reinforcement learning is effective in handling spatial decision-making and providing a robust foundation for the development of new methods.

In recent years, for the routing problem, there has been a significant amount of research exploring spatial optimization problems with deep learning. The pioneering work on utilizing the deep learning technique to solve COPs was introduced by Vinyals, Fortunato, and Jaitly (2015). Inspired by the seq2seq structure of machine translation, Pointer Network (Ptr-Net) was proposed to solve the Convex Hull problem and Traveling Salesman Problem by adding a pointer to seq2seq. Afterward, as the transformer model (Vaswani et al. 2017) became popular in NLP, Deudon et al. (2018) improved the Ptr-Net based on the Multi-head Attention (MHA) mechanism in the transformer. They used the REINFORCE method to train the model, and utilized the 2-opt method for further search solution space after the solution was generated, which effectively enhanced the quality of the solution. Kool, van Hoof, and Welling (2018) proposed the Attention Model (AM) to tackle different types of routing problems, such as TSP, Capacitated VRP (CVRP), Orienteering Problem (OP), and Prize Collecting TSP (PCTSP). Inspired by AM, Peng, Wang, and Zhang (2020) proposed an AM with dynamic characteristics that can significantly enhance the performance of the model. Xin et al. (2021) proposed the use of Multi-Decoder AM (MDAM) to solve VRP. MDAM comprises a single encoder and multiple decoders, containing the same structure but different parameters, and it obtains promising results on six routing problems. Kwon et al. (2020) revised the RL algorithm in the AM, raised Policy Optimization with Multiple Optima (POMP), and obtained SOTA results.

The above-mentioned research mainly focuses on TSP, VRP, PCTSP, OP, and other routing problems, but little research has been conducted on urban facility location problems. Using neural networks to address p-Median problems can be traced back to the Hopfield network (Domínguez and Muñoz 2008). However, this neural network can only learn and solve a single small-scale p-Median problem each time. For the new problem, it still needs to be retrained and compared with traditional methods. Liang et al. (2022) proposed a new method for solving the p-Center problem by combining Graph Convolutional Networks (GCN) with a clustering algorithm. Wang et al. (2023) proposed the DeepMCLP to solve MCLP with the DRL. Our study is situated at the intersection of big data-driven spatial decision support systems and connectivity-based graph neural networks. Recent studies (Andronie et al. 2021; Nica et al. 2023) emphasized data-driven approaches in sustainable urban governance and smart urban economies. Lăzăroiu et al. (2020) explored sustainability in urban corporate economies. These sources inform our research, highlighting the significance of data-driven solutions in complex urban environments.

3. Preliminaries

3.1. Problem definition

For all the urban facility location problems in this study, let $N = \{1, 2, \ldots, n\}$ be the set of demand points and $M = \{1, 2, \ldots, m\}$ be the set of facilities. Subsequently, we introduced three spatial optimization problems. The decision variables are defined as follows.

$$x_{ij} = \begin{cases} 1, & \text{if demand point } i \text{ is assigned to the facility } j, \\ 0, & \text{otherwise.} \end{cases}$$

$$y_j = \begin{cases} 1, & \text{if and only if facility } j \text{ is open,} \\ 0, & \text{otherwise.} \end{cases}$$

3.1.1. p-Median problem

The p-Median problem can be modeled using integer linear programming (ILP). The formulation is as follows (ReVelle and Swain 1970).

$$\text{Minimize} \quad \sum_{i=1}^{n} \sum_{j=1}^{m} d_{ij} x_{ij} \tag{1}$$

$$\text{Subject to} \quad \sum_{j=1}^{m} x_{ij} = 1, i = 1, \ldots, n, \tag{2}$$

$$\sum_{j=1}^{m} y_j = p, \tag{3}$$

$$x_{ij} \le y_j, \ i = 1, \ldots, n, j = 1, \ldots, m. \tag{4}$$

$$x_{ij}, y_j \in \{0, 1\} \ i = 1, \ldots, n, \ j = 1, \ldots, m. \tag{5}$$

Constraint (2) guarantees that each point is assigned a unique median value. Condition (3) ensures that only p median points exist. Inequality (4) ensures the coherence of the solutions; that is, demand point i cannot be assigned to facility j if facility j is not determined as a median.

3.1.2. p-Center problem

Hakimi (1964) first proposed the p-Center problem. The essence of this problem is to find p facilities to open and allocate n customers to these p facilities. In the p-Center problem, the objective function minimizes the maximum distance from each demand point to the nearest facility. The Integer Linear Programming format of the p-Center problem is as follows (Daskin 1997):

$$\text{Minimize} \quad z \tag{6}$$

$$\text{Subject to} \quad \sum_{j=1}^{m} y_j = p, \tag{7}$$

$$\sum_{j=1}^{m} x_{ij} = 1, \ i = 1, \ldots, n \tag{8}$$

$$x_{ij} \le y_j, \ i = 1, \ldots, n, j = 1, \ldots, m \tag{9}$$

$$\sum_{j=1}^{m} d_{ij} x_{ij} \le z, \ i = 1, \ldots, n \tag{10}$$

$$x_{ij}, \ y_j \in \{0, 1\} \ i = 1, \ldots, n, \ j = 1, \ldots, m. \tag{11}$$

Equation (7) ensures that the number of open facilities is p. Constraint (8) assigns each demand point to only one facility: Inequality (9) guarantees that each demand point is assigned to an open facility: Constraint (10) requires that z to be the upper bound of the distance between any demand point to the assigned facility. It is evident that $z = max_{i \in N} \sum_{j=1}^{m} d_{ij} x_{ij}$ is the optimal cover radius.

3.1.3. MCLP

The essential goal for the MCLP is to cover the most demand points (or serve more customers) under the condition that the number and radius of facilities are known. The ILP formulation of the MCLP is as follows (Church and Velle 1974):

$$\text{Maximize} \quad z = \sum_{i=1}^{n} a_i y_i \tag{12}$$

$$\text{Subject to} \quad \sum_{j \in N_i} x_j \ge y_i, \ i = 1, \ldots, n \tag{13}$$

$$\sum_{j=1}^{m} x_j = p, \tag{14}$$

$$x_j,\ y_i \in \{0, 1\}\ i = 1, \ldots, n,\ j = 1, \ldots, m. \tag{15}$$

where S represents the distance beyond which a customer is considered 'uncovered'. d_{ij} is the shortest distance from point i to point j. a_i is the population to be served at demand point i. $N_i = \{j \in M | d_{ij} \leq S\}$. x_j and y_i are two decision variables.

$$x_j = \begin{cases} 1, & \text{if a facility } i \text{ is allocated to site } j, \\ 0, & \text{otherwise.} \end{cases}$$

$$y_i = \begin{cases} 1, & \text{if and only if facility } i \text{ is open,} \\ 0, & \text{otherwise.} \end{cases}$$

Constraint (13) ensures that y_i equals 1 only when one or more facilities are established at sites in the set N_i. Condition (14) restricts the number of facilities allocated to be p.

3.2. Deep learning model

3.2.1. Attention model

The Attention Model (AM), a deep learning model, was proposed by Kool, van Hoof, and Welling (2018) to solve various routing problems and consists of an Encoder and Decoder. For the initial input node X, the original vector is first embedded into the d_x high-dimensional space through a linear mapping layer and then through N-th attention layers, which contain both multi-headed attention and feed-forward neural networks. The formula used is as follows:

$$\hat{h}_i = BN^l(h_i^{l-1} + MHA_i^l(h_1^{l-1}, \ldots, h_n^{l-1})) \tag{16}$$

$$h_i^l = BN^l(\hat{h}_i + FF^l(\hat{h}_i)) \tag{17}$$

where h_i is the node embedding, MHA represents the multi-head attention layer, BN is the batch normalization layer, and FF represents the feed-forward layer. During the decoding process, one node is selected per step. In step t, the decoder is context embedding based on the embedding characteristic of the hidden layer from the encoder and decoder output at the first and $t-1$ steps. The context embedding is calculated as follows:

$$h_c^N = \begin{cases} [\bar{h}^N, h_{\pi_{t-1}}^N, h_{\pi_1}^N] & t \geq 1 \\ [\bar{h}^N, v_1, v_f] & t = 1. \end{cases} \tag{18}$$

3.2.2. Gate recurrent unit

GRU, a neural network, is a variant of LSTM (Hochreiter and Schmidhuber 1997), which can efficiently handle sequence data and can be used to solve the gradient disappearance problem in long-term memory and backpropagation problems during training. The GRU contains a reset gate and an update gate. The reset gate acts on the previous hidden state to determine how much of the previous information should be forgotten, and the update gate acts on the current and previous hidden units to determine which valid information needs to be retained from the previous and current moments. The updated formula is as follows (Chung et al. 2014):

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$

$$\bar{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \bar{h}_t$$

where r_t represents the reset gate, and h_{t-1} contains the previous information. Multiplying the two by elements is used to measure whether the previous information is useful for future predictions. z_t denotes the update gate and determines how to combine the new input information with previous memory. The problem of gradient decay can be effectively addressed by GRU, whereas GRU can more efficiently capture dependencies with larger intervals in sequential data.

3.3. Markov decision process and deep reinforcement learning

3.3.1. Markov decision process

The MDP is a mathematical model that imitates the stochastic strategy and reward of an agent in an environment, and the state of the environment has a Markov Property. It contains a finite state set S, action set A, state transition probability P, discount factor γ, and reward function R, which is denoted as quintuple (S, A, P, R, γ). A deep learning model can be viewed as an agent, and the problem can be regarded as the environment. By sensing the state of the external environment, the agent makes decisions, acts on the environment, and adjusts its strategy through rewards. The state of the environment varies with changes in agent action, and the variation in the environment can be sensed by the agent. The environment can provide feedback and relative reward for the decision made by the agent.

Taking p-Median as an example, we considered AM as the agent, the p-Median problem as the environment, and the state of the environment as relevant variables of the problem, such as the node coordinates of users and facilities, the distance between each user and facilities, whether facilities are selected, whether users are served, etc. A facility is chosen by AM as the median (center) from the facility set, according to the present state information s_t. Afterward, AM calculates the current total weighted distance (i.e. p-Median problem) based on the selected median and differentiates the total weighted distance between steps t and step $t-1$, This difference is used as the reward of that step or is the feedback made by the environment to the action. We then iterate until p actions are selected.

3.3.2. Deep reinforcement learning

The spatial optimization problems are all *NP*-hardness problems. For small-scale problems, a large set of labeled data can be generated by a solver, and supervised learning can be utilized to train the model. However, creating enough labeled data is time-consuming, labor-consuming, and even challenging. We use DRL to train the model and enable it to learn how to optimize through continuous interaction between the agent and environment.

The task of DRL is always described by the MDP, and this study mainly applies the REINFORCE algorithm (Williams 1992) which is a renewed policy gradient method based on the Monte Carlo method. The policy gradient applies a random gradient ascent measure to update the parameters under the given objective function, and the random gradient ascent formula is as follows:

$$\theta_{t+1} = \theta_t + \alpha \nabla L(\pi) \tag{19}$$

In our problem, because the goal is to minimize the objective function, the negative value of the optimization objective for each problem is used to replace the value of q in the above equation to denote the sum of the rewards after t. When $L(\pi)$ increases, it suggests that the reward of the current state achieves the maximum and the parameter transfers in the direction of increasing the probability of it being in the current state. The larger the reward, the greater the gradient update.

4. Methodology

We adopt reinforcement learning to train the model to solve unseen instances. First, we convert the location problem into an MDP and introduce a quintuple $M = (S,\ A,\ P,\ r,\ \gamma)$ to express the

problem. We then propose a unified deep model, SpoNet, for such spatially optimal problems. The dynamic covering information and gated recurrent unit (GRU) affect the Encoder and Decoder, respectively. Finally, we provided the REINFORCE algorithm with a rollout baseline to train our models. The SpoNet framework is shown in Figure 1.

4.1. Reformulate as MDP form

The key issue of p-Median, p-Center, and MCLP is how to select p facilities from N candidate facilities to meet the objective function. We used a constructive method to model the three spatial optimization problems with a unified framework. In the decision-making process, we select a facility point at each step until we choose p facility points. We essentially modeled this selection process as an MDP.

The MDP is usually represented by a five-tuple, including state, action, state transition matrix, reward, and discount factor. The following terms are defined.

State: State $s_T = (M_t)$ represents the current solution sets in the construction process in step t, where M_T includes all selected facility points up to time step t, M_0 represents an empty solution set.

Action: In the process of constructing the solution, each step must select a node as the facility point. Let a_T be denoted as z_j. In other words, node z_j at time step t as the facility point.

Transition: State transition indicates that in state s_t, the model adopts action a_t to enter the next state $s_{t+1} = M_{t+1} = (M_t;\ z_j)$, that is, the solution of the next state s_{t+1} is spliced together by the current state s_t and point z_j corresponding to the action.

Discount factor: The discount factor was set to 1. When calculating the reward function, we take the objective function as the final reward, and the cumulative reward is not calculated.

Reward: The strategic network's optimization direction is determined by the reward function. The three spatial optimization problems have different optimization goals, so different reward functions need to be set.

Denoting our policy by p_θ, the policy network selects a node at each time step until p points are acquired as the facilities. The other demand points are then assigned to the nearest facilities, and the final objective function is calculated. It provides a feasible solution to the problem, denoted by

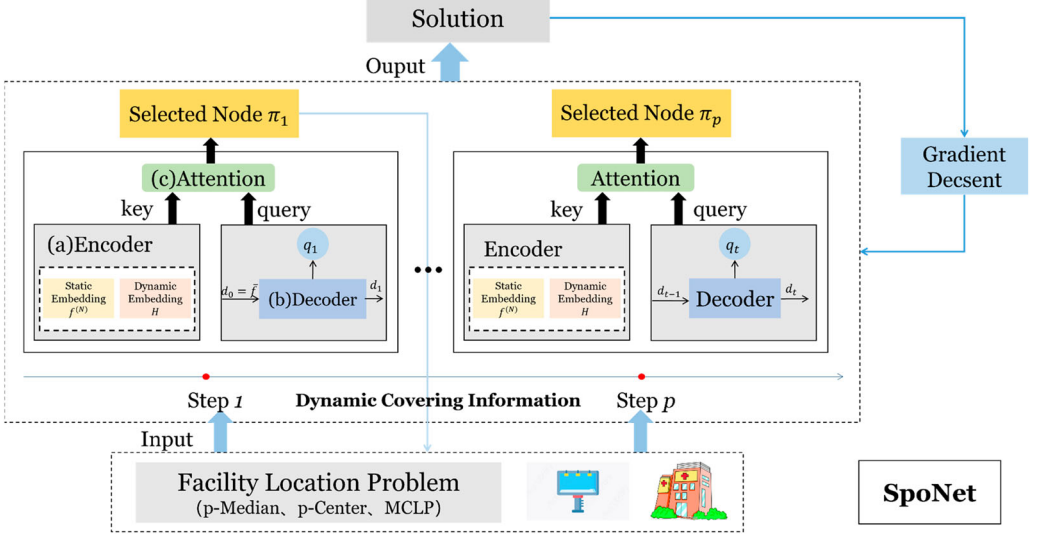

Figure 1. Total Workflow of Our SpoNet. (a) Encoder. Fusing static location information and dynamic coverage information of nodes to generate key. (b) Decoder. The Encoder is based on a GRU structure to generate the query. (C) Attention. At each step, the probability of selecting each node is computed.

$\pi = \{\pi_1, \ldots, \pi_p, p \leq n\}$, and only one node is selected at each step, so our SpoNet strategy $p(\pi|s)$ can be expressed as

$$p_\theta(\pi|s) = \prod_{t=1}^{p} p_\theta(\pi_t|s, \pi_{1\sim t-1}), \ p \leq n \tag{20}$$

To train our model, we defined the loss $L(\theta|s) = E_{p\theta(\pi|s)}[L(\pi)]$: the expectation of cost $L(\pi)$. For p-Median, the cost function $L(\pi)$ is

$$L(\pi) = \sum_{i=1}^{N}\sum_{j=1}^{M} d_{ij}x_{ij} \tag{21}$$

For p-Center, the cost function $L(\pi)$ is:

$$L(\pi) = \max_{i\in I}\sum_{j=1}^{M} d_{ij}x_{ij} \tag{22}$$

For MCLP, the cost function $L(\pi)$ is:

$$L(\pi) = -\sum_{i=1}^{N} a_i y_i \tag{23}$$

Unlike the routing problem, there is no sequence relationship between the points selected in location problems. However, a point must be selected at each step during the modeling process, and the previous point selection influences the current point selection. Therefore, the DRL algorithm solves the coverage relationship of points with changing environmental conditions. The Encoder uses the original features x and generates the final embedding for each point. The Decoder takes the encoder's final embedding as the input, combines the previously selected point information $\pi_{\{1\sim t-1\}}$, and finally outputs π_t. Apparently, only one point is output at each step, and the decoder observes the mask to indicate coverage and the point to choose to deploy the facility.

4.2. The SpoNet

We first described the encoder–decoder structure model to introduce dynamic coverage information. The encoder encodes the original features to generate the hidden embedding for each node, while the decoder takes hidden embedding as input. Finally, the selection probability for each node was the output. In the location problem, a covering relationship exists between nodes. If a node has been covered, it will no longer be selected, or the chances of selection will be reduced. Therefore, we propose a dynamic coverage model that encodes the static coordinates of cities and the coverage states between cities to improve model performance. The SpoNet model is illustrated in Figure 2.

4.2.1. Encoder

The encoder can transform realistic problems into mathematical problems, and the original features are partitioned into two components: static and dynamic embedding. Static embedding represents the two-dimensional coordinate information of each point, which does not affect the features during the model training. In contrast, dynamic embedding indicates that the node state constantly changes as it is covered. Therefore, the dynamic feature represents coverage information.

There are two parts for processing static embedding: node embedding and multi-headed attention (MHA) layer. The original features are fed into the network and obtain the node embedding after a layer of linear mapping, then run through N layers of the attention layer, and finally generate the static embedding of each point. Each Attention layer is composed of one MHA layer and a feed-

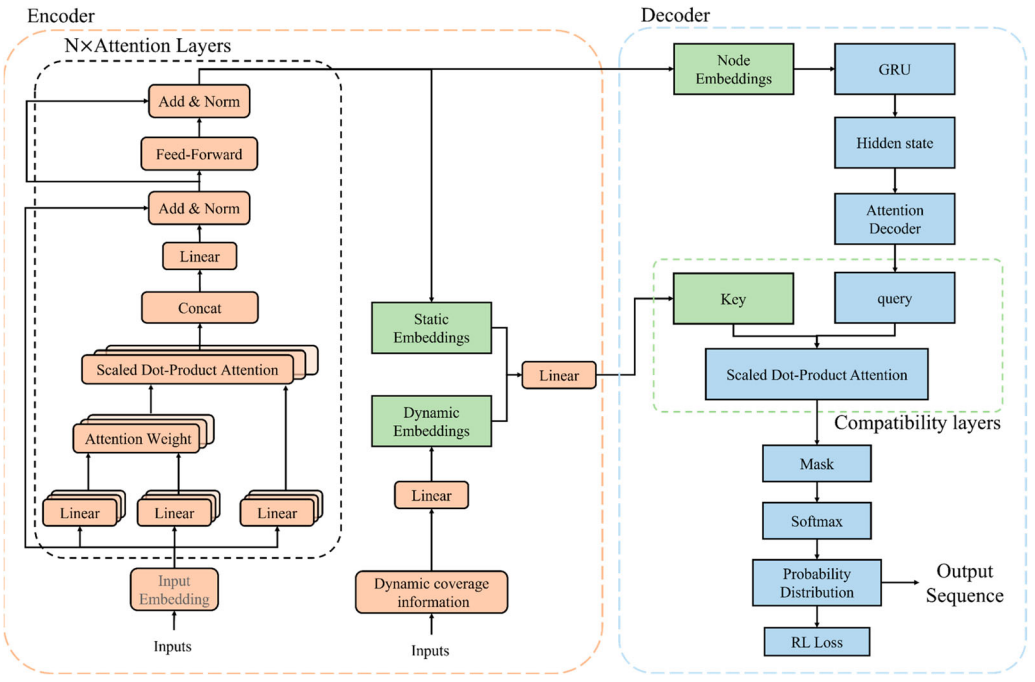

Figure 2. The Spatial Optimal Neural Network (SpoNet).

forward neural network. Each sublayer was connected to a skip-connection structure and normalized layer *BN*. In other words, for each sublayer neural network, given an input x, there is $BN(x + sublayer(x))$. The details of the encoder are as follows:

First, we mapped the $d_x(d_x = 2)$ dimensional location information of each point to the $d_h(d_h = 128)$ dimensional node embedding.

$$f_i^{(0)} = W^{(0)}x_i + b^{(0)} \tag{24}$$

where $W^{(0)}$ and $b^{(0)}$ are learnable parameters. The final static embedding is generated by passing $f_i^{(0)}$ through N layers of the attention layers. The output of each layer depends on the input of the previous layer, and the calculation for the k_{th} layer is as follows:

$$\hat{f} = BN(f^{(k-1)} + MHA(f^{(k-1)})) \tag{25}$$

$$f^{(k)} = BN(\hat{f} + FF(\hat{f})) \tag{26}$$

After N attention layers, the final static embedding is $f^{(N)}$.

We then introduce dynamic coverage information as dynamic features. We propose two types of handling of coverage information. The first one is that if a node is selected or covered, it will not be selected in subsequent operations, and this idea is easy to understand and implement. If a node is covered but not selected, it still has the chance to be selected. For the second approach, we introduced a dynamic coverage vector I_i for each point i to represent its coverage status, and dynamically updated I_i in each step.

We used π_t to denote the node selected in step t, $C_r(\pi_t)$ to denote the set of nodes covered by the node π_t under coverage radius r, $N_{C_{r(\pi_t)}}$ to represent the number of nodes in the set $C_r(\pi_t)$, sorted each point in $C_r(\pi_t)$ according to its distance to π_t from near to far, and c_i to express the sort index of the i_{th} point. The nearest sort to π_t is 1 and the farthest point is sorted as $N_{C_{r(\pi_t)}}$.

First, for each point, I_i initialized as 1:

$$I_i = 1, i = 1, \ldots, n \tag{27}$$

In each step of the decoding process, I_i is dynamically updated according to Equation (28) or Equation (29):

$$I_i = I_i \times \frac{c_i}{N_{C_{r(\pi_t)}}}, i = 1, \ldots, n. \tag{28}$$

$$I_i = \begin{cases} 0, & \text{if } i \in C_r(\pi_t) \\ I_i, & \text{otherwise} \end{cases} \tag{29}$$

According to the above equation, if node i is covered several times, I_i is reduced, which indicates that the probability of this selected node is reduced. Furthermore, the designed dynamic coverage vector contains coverage information for each node. A node's coverage and distance from the central node can serve as a useful guide for selecting the next node. Then we let the model learn how to use I_i to adjust probabilities. I_i is updated according to the following equation:

$$H_i = I_i W^I \tag{30}$$

where W^I is the learnable parameter, and H_i is the final dynamic embedding.

Hence, according to static and dynamic embedding, we constructed the key of each node as the input to the decoder, and updated the formula as follows:

$$k_i = W^k f_i^{(N)} H_i, \ i = 1, \ldots, n. \tag{31}$$

where k_i denotes the key of node i, and W^k is the learnable parameter. The model can capture environmental information and learn better strategies by adding dynamic coverage information.

4.2.2. Decoder

We designed a learnable decoder to select the facilities and construct feasible solutions based on the attention mechanism. The Decoder consists of two parts: the RNN for outputting the query, and the attention structure for calculating the probability of each city being selected.

The decoding process operates sequentially, and the decoder outputs a facility at each step t. This decision was based on the static and dynamic embeddings of the nodes, which were acquired from the encoder, as well as the current state of the partial solution. For each node $i = 1, \ldots, n$, as in the study by Li et al. (2021), the query q_t is generated by using the Decoder through the RNN and self-attention, and the vector u_i of each node is computed by combining the k_i obtained from the encoder:

$$u_i^t = \frac{q_t^T k_i}{\sqrt{d_k}} \tag{32}$$

where $\sqrt{d_k}$ is called the scaling factor, $d_k = \frac{d_h}{M}$, and M denotes the number of self-attentions in the multi-headed attention. Then, u_i was modified according to the coverage information to obtain $\bar{u}_i$:

$$\bar{u}_i^t = \begin{cases} -\infty, & \text{if } i \in \pi_{1\sim t-1} \\ \frac{q_t^T k_i}{\sqrt{d_k}}, & \text{otherwise} \end{cases} \tag{33}$$

Finally, we used softmax to calculate the probability of each node being selected.

$$p_i = \frac{e^{\bar{u}_i^t}}{\sum_{j=1}^{n} e^{\bar{u}_i^t}}, \ i = 1, \ldots, n. \tag{34}$$

Here, we applied a greedy strategy to select nodes, that is, we chose the node with the highest probability as the location of the currently deployed facility.

4.3. Training of the model

Given instance s, and the model defined in Equation (20), the solution π for the problem was obtained from the samples. Our goal is to choose a point at each step to deploy the facility and try to maximize or minimize the objective function. Thus, a reinforcement learning approach is utilized to train the model. Commonly, given the triple (s, a, r), the parameters of the model can be updated by gradient descent using a reinforcement learning algorithm with baseline $b(s)$ (Williams 1992):

$$\nabla L(\theta|s) = E_{p\theta(\pi|s)}[(L(\pi) - b(s))\nabla \log p\theta(\pi|s)] \tag{35}$$

The algorithm for solving problems using reinforcement learning is as follows.

Algorithm 1: Training with Reinforcement Learning by Rollout Baseline
Input: Training set S, Number of epochs N_{epoch}, Number of training steps T, Batch size B
1: Initialize the policy network parameters θ and baseline policy parameter θ^*, $\theta^* \leftarrow \theta$
2: **for** *epoch* $\leftarrow$ 1: N_{epoch} **do**
3: **for** *step* $\leftarrow$ 1: T **do**
4: $s_i \leftarrow SampleInput(S)$ for $i \in \{1, \ldots, B\}$
5: $\pi_i \leftarrow p_\theta(s_i)$. The policy by sampling
6: $\pi_i^* \leftarrow p_{\theta^*}(s_i)$. The policy by rollout baseline
7: $\nabla\mathcal{L} \leftarrow \sum (L(\pi_i) - L(\pi_i^*))\nabla_\theta \log p_\theta(\pi_i)$
8: $\theta \leftarrow \theta + \nabla_\theta\mathcal{L}(\theta)$
9: **end for**
10: **if** p_θ performs better than p_{θ^*} **then**
11: $\theta^* \leftarrow \theta$
12: **end if**
13: **end for**

5. Experiments and results

We present the benchmark experimental results of SpoNet for solving the p-Median, p-Center, and MCLP. First, instances with 20, 50, and 100 points were generated to train the models. The two-dimensional coordinates (x, y) of each point are independently sampled in a uniform distribution of $[0, 1]$, and the weight of every pair of points is calculated using the Euclidean distance. All the instances are complete graphs, that is, each point in the graph is directly connected to other points by edges. Therefore, the covering radius was added to mask the covered points to realize dynamic updates and improve the performance of the model. The choice of the covering radius is based on experience. In the MCLP, the covering radius is set to 0.3, 0.2, and 0.15, for problems of 20, 50, and 100 points, respectively. The covering radius of facilities in the original p-Median and p-Center problems is infinite by default, and the optimization objective of the p-Center problem is to find the minimum covering radius. Therefore, we chose the approximate upper bound of the p-Center target value as the reference. For the problems with 20, 50, and 100 points, the covering radius was set to 0.32, 0.24, and 0.16, respectively. All experiments were implemented using Python and run on two RTX3090 GPUs and an Intel(R) Xeon(R) Gold 6230 CPU @2.10 GHz. The code has been released https://github.com/HIGISX/SpoNet.

Hyperparameters: For all weighted parameters in a neural network, a uniform initialization $U \sim \left(-\frac{1}{d}, \frac{1}{d}\right)$ is used, where d denotes the dimension of the input. During the training of each model, 250 epochs were trained, and each epoch processed 250 batches with a batch size of 512.

For each instance, the two-dimensional position coordinates of the nodes are linearly projected onto 128-dimensional vectors and then sent to the encoder. After three multi-head attention layers, a 128-dimensional hidden embedding was obtained. The models were trained using the ADAM optimizer (Kingma and Ba 2014) and the learning rate was fixed at 10^{-4}. Table 1 lists the hyperparameter configurations used during the training process.

5.1. Main results

5.1.1. Comparison analysis

We compared our SpoNet with three types of methods:(1) three state-of-the-art solvers for spatial optimization problems, including Mathematical Programming Solver, Gurobi (2021), Cplex (Manual 1987) and COPT; (2) the deep model baseline, Attention Model (AM), a constructive method based on DRL for solving TSP and CVRP, which we transferred to solve p-Median, p-Center, and MCLP; (3) heuristic and meta-heuristic algorithms, including GA, VNS, SA, TS, and Teitz-Bart. We implemented SA, VNS, and Teitz-Bart for the p-Median problem; SA, TS, and GA for the p-Center problem; and SA, VNS, and GA for the MCLP. To perform the test, 10,000 instances with the same distribution as that of the training data were generated for each group of experiments. We used the objective value, optimality gap, and runtime for the evaluation.

We compare the performance of our SpoNet and all the methods mentioned above for the three problems with different problem sizes in Table 2. Each evaluation indicator was recorded, with an average value of 10,000 instances. Table 2 indicates that SpoNet with sample 1280 is superior to heuristic algorithms in terms of the performance of the objective value and running time. Although the AM with sample 1280 consumes less time than our model, it is acceptable to sacrifice a little time to improve the accuracy. Because heuristic algorithms need to adjust the parameters according to expert experience, the solution accuracy and running time are affected by the parameters. Within a certain range, the longer the solution time, the smaller the gap between the target value and the optimal solution. Here, we present the value of the objective function for the problem within an acceptable solution time.

5.1.2. Efficiency comparison for MCLP

Among all the methods, mathematical programming solvers can provide the optimal solution for the p-Median, p-Center, and MCLP. However, these methods require longer resolution times. With the increase in the scale of the problem size, it is difficult to solve the problem in a short time. We conducted a simple experiment to explain the advantages of the deep model in terms of running time. For MCLP, we expanded the number of demand points to 1000, 2000, and 5000 and displayed the results of the trained SpoNet (greedy) and Gurobi. From Table 3, as the problem scale increases, our model can obtain a flexible solution quickly, but Gurobi cannot report a solution with 5000 demand points in an hour.

5.1.3. Analysis of decoding strategy

In the evaluation experiment, two decoding strategies were adopted. The first is a greedy decoding strategy; that is, the facility point is chosen based on which node has the highest probability. The

Table 1. Hyperparameter configurations.

Hyperparameters	Value
No. of instances per epoch	128,000
No. of epoch	250
Batch size	512
Optimizer	Adam
Learning rate	1e-4
Hidden dimension	128
No. of Encoder layers	3

Table 2. The solution result of p-Median, p-Center, and MCLP with different scale.

		$n = 20, p = 4$			$n = 50, n = 8$			$n = 100, p = 15$		
Problem	Model	Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time
p-Median	Gurobi	2.97	0.00%	0.0204	5.32	0.00%	0.1292	7.68	0.00%	0.5397
	Cplex	2.97	0.00%	0.0288	5.32	0.00%	0.1420	7.68	0.00%	0.5784
	COPT	2.97	0.08%	0.0345	5.32	0.00%	0.1690	7.68	0.02%	1.2992
	AM (greedy)	3.00	1.04%	0.0059	5.52	3.70%	0.0108	8.24	7.29%	0.0191
	AM (sample128)	2.98	0.21%	0.0099	5.38	1.11%	0.0271	7.98	3.92%	0.1115
	SpoNet (greedy)	3.00	0.91%	0.0058	5.40	1.48%	0.0177	7.87	2.50%	0.0303
	SpoNet (sample128)	**2.97**	**0.14%**	**0.0123**	**5.34**	**0.31%**	**0.0345**	**7.72**	**0.57%**	**0.0999**
	SA	3.00	0.88%	0.0076	5.35	0.55%	0.1028	7.72	0.59%	0.9038
	VNS	2.98	0.37%	1.0179	5.42	1.91%	1.0656	8.44	9.89%	1.3051
	Teize-bart	2.98	0.27%	0.0027	5.34	0.45%	0.0311	7.72	0.58%	0.2570
p-Center	Gurobi	0.32	0.00%	0.0503	0.22	0.00%	0.4465	0.16	0.00%	4.5225
	Cplex	0.32	0.00%	0.3493	0.22	0.00%	0.4079	0.16	0.00%	11.5431
	COPT	0.32	0.00%	0.1198	0.22	0.00%	0.4973	0.16	0.00%	3.4479
	AM (greedy)	0.33	3.36%	0.00028	0.25	10.55%	0.0003	0.19	20.85%	0.0005
	AM (sample1280)	0.32	0.95%	0.00875	0.23	4.58%	0.0353	0.18	13.88%	0.0939
	SpoNet (greedy)	0.33	3.27%	0.00005	0.24	8.98%	0.0001	0.19	16.77%	0.0003
	SpoNet (sample128)	**0.32**	**1.02%**	**0.0105**	**0.23**	**2.33%**	**0.0409**	**0.17**	**7.85%**	**0.1571**
	GA	0.32	2.73%	0.2416	0.26	16.52%	0.423	0.21	32.73%	0.7176
	SA	0.34	7.01%	0.2389	0.28	27.57%	0.3094	0.24	48.81%	0.3545
	TS	0.33	3.78%	0.1784	0.25	14.10%	0.299	0.20	26.51%	0.6504
MCLP	Gurobi	18.97	0.00%	0.0113	47.38	0.00%	0.06299	97.51	0.00%	0.246
	Cplex	18.97	0.00%	0.0086	47.38	0.00%	0.01952	97.51	0.00%	0.0491
	COPT	18.97	0.00%	0.0208	47.38	0.00%	0.02946	97.51	0.00%	0.0658
	AM (greedy)	18.64	1.74%	0.00016	44.97	5.08%	0.00022	91.01	6.66%	0.0004
	AM (sample128)	18.88	0.49%	0.00785	46.38	2.09%	0.0291	94.02	3.58%	0.0956
	SpoNet (greedy)	18.65	1.72%	0.00005	45.80	3.32%	0.0001	92.97	4.65%	0.0003
	SpoNet (sample128)	**18.90**	**0.41%**	**0.01036**	**46.88**	**1.04%**	**0.0449**	**95.74**	**1.81%**	**0.164**
	GA	18.54	2.28%	0.09876	43.26	8.68%	0.3967	85.23	12.59%	1.4738
	SA	18.63	1.80%	0.2089	43.70	7.75%	0.7491	85.92	11.90%	1.2422
	TS	18.21	4.00%	0.01489	42.16	11.02%	0.1191	85.89	11.91%	0.9205

second is the sampling decoding strategy. We sampled 1280 solutions and selected the best solution reports. Our model can use GPU operation, and sampling 1280 solutions can be completed in parallel. Therefore, it only takes a short time to obtain more accurate solutions. The experiment showed that the sampling decoding strategy usually obtains a more feasible solution than the greedy decoding strategy.

5.2. Ablation study

In this section, we illustrate the ablation study to prove the effectiveness of the dynamic covering information and the GRU structure.

Table 3. The results on $N = 1000$, 2000 and 5000 of MCLP.

	$N = 1000, p = 15, r = 0.15$			$N = 2000, p = 15, r = 0.15$			$N = 5000, p = 15, r = 0.15$		
Algorithm	Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time
SpoNet	889	6.81%	0.2971	1778	5.83%	1.511	4447	/[a]	8.248
Gurobi	954	0.00%	50.9617	1888	0.00%	711.7398	/	/	/

[a]'/' represents the algorithm that cannot give the solution over one hour.

Table 4. Effect of dynamic covering information.

		$n = 20$, $p = 4$, $r = 0.3$			$n = 50$, $n = 8$, $r = 0.2$			$n = 100$, $p = 15$, $r = 0.15$		
Problem	Model	Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time
p-Median	AM	**3.00**	**1.04%**	0.0059	5.52	3.70%	0.0108	8.24	7.29%	0.0191
	AM + DC1	3.00	1.08%	0.0073	5.51	3.52%	0.0125	8.11	5.71%	0.0239
	AM + DC2	3.00	1.13%	0.0114	**5.46**	**2.64%**	0.0197	**7.96**	**3.65%**	0.0446
p-Center	AM	0.33	3.36%	0.0003	0.25	10.55%	0.0003	0.19	20.85%	0.0005
	AM + DC1	**0.33**	**3.24%**	0.0003	**0.24**	**9.47%**	0.0004	**0.19**	**20.16%**	0.0007
	AM + DC2	0.33	3.97%	0.0003	0.25	10.06%	0.0004	0.20	22.68%	0.0005
MCLP	AM	18.64	1.74%	0.0002	44.97	5.08%	0.0002	91.01	6.66%	0.0004
	AM + DC1	18.64	1.73%	0.0002	44.87	5.29%	0.0002	90.86	6.81%	0.0004
	AM + DC2	**18.70**	**1.44%**	0.0002	**45.16**	**4.68%**	0.0002	**91.17**	**6.50%**	0.0004

5.2.1. Effects of dynamic covering information

In our study, we consider two updated formulations based on coverage information. The first is to remove all covered nodes from the set of candidate facility points according to the coverage radius after selecting facility points. The update formula is as follows Equation (28). Even after a node has been marked as covered, the node is still eligible to be chosen as a facility point. However, the likelihood of it being selected is modified based on the distance between the covered node and the potential facility point. The updated formula is shown in Equation (29): Table 4 shows the Obj, Gaps, and Time of the AM before and after adding dynamic coverage information to solve the three types of problems. As shown, a structure with dynamic coverage information has a smaller gap than the original attention model.

5.2.2. Effect of GRU structure

The process of constructing solutions is a long time series of the decision-making process, and the GRU structure can effectively deal with this type of problem with a long dependence. Table 5 shows the results of the three problems on different scales before and after the GRU structure was added.

5.3. Transfer study

The approach that utilizes a deep learning model to construct solutions has the advantage of being able to handle large-scale problems adaptively. We used the trained model with 100 nodes to evaluate the instances of 200, 300, 400, 500, 800, and 1000 nodes, and15 nodes were finally chosen to deploy facilities. We utilized the Gurobi solver (the solution time was limited to 600 s for each instance) to report the optimal solution, used SpoNet, and simulated annealing algorithm (better than GA and VNS) to provide a feasible solution for the problem. Table 6 presents the results: (1) SpoNet performs better than SA on all issues; and (2) for large-scale problems, Gurobi and SA are time-consuming, but SpoNet can still provide a feasible solution to the problem quickly, and the output is closer to the best possible solution.

Table 5. Effect of GRU structure.

		$n = 20$, $p = 4$, $r = 0.3$			$n = 50$, $n = 8$, $r = 0.2$			$n = 100$, $p = 15$, $r = 0.15$		
	Model	Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time
p-Median	AM	3.00	1.04%	0.0059	5.52	3.70%	0.0108	8.24	7.29%	0.0191
	AM + GRU	**2.99**	**0.81%**	0.0080	**5.42**	**2.64%**	0.0128	**7.93**	**3.29%**	0.0248
p-Center	AM	0.33	3.36%	0.0003	0.25	10.55%	0.0003	0.19	20.85%	0.0005
	AM + GRU	**0.33**	**3.33%**	0.0001	**0.24**	**8.80%**	0.0001	**0.19**	**16.71%**	0.0003
MCLP	AM	18.64	1.74%	0.0002	44.97	5.08%	0.0002	91.01	6.66%	0.0004
	AM + GRU	**18.69**	**1.51%**	0.0000	**45.40**	**4.16%**	0.0001	**91.70**	**5.95%**	0.0003

Table 6. Transfer study.

		$n = 200, p = 15$			$n = 300, p = 15$			$n = 400, p = 15$		
Problem	Algorithm	Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time
p-Median	Gurobi	17.29	0.00%	2.2952	26.77	0.00%	6.4767	36.46	0.00%	14.4155
	SA	17.36	0.42%	4.0136	26.94	0.63%	8.9189	36.78	0.86%	21.1821
	SpoNet	17.35	0.39%	0.2473	26.91	0.54%	0.3856	36.74	0.77%	0.5691
p-Center	Gurobi	0.17	0.00%	131.1513	0.17	0.00%	469.5900	0.17	0.00%	608.0616
	SA	0.22	24.85%	76.7693	0.24	28.98%	107.6598	0.23	23.68%	124.2936
	SpoNet	**0.17**	**3.78%**	**0.2295**	**0.17**	**3.72%**	**0.3792**	**0.18**	**0.28%**	**0.5494**
		$n = 500, p = 15$			$n = 800, p = 15$			$n = 1000, p = 15$		
Problem	Algorithm	Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time
p-Median	Gurobi	46.47	0.00%	47.6388	75.71	0.00%	316.6517	195.35	/[a]	600.0000
	SA	46.96	1.04%	29.4457	76.28	0.75%	125.0781	96.55	/	141.5112
	SpoNet	**46.71**	**0.50%**	**0.6866**	**76.27**	**0.73%**	**1.2508**	**96.19**	**/**	**1.7327**
p-Center	Gurobi	1.19	/	600.0000	1.20	/	600.0000	1.21	/	600.0000
	SA	0.24	/	175.6336	0.24	/	339.5176	0.24	/	404.3000
	SpoNet	**0.18**	**/**	**0.6726**	**0.18**	**/**	**1.2732**	**0.18**	**/**	**1.7350**

[a]Gurobi cannot solve the optimal solution in 10 min. '/' indicates that the gap between the feasible solution and the optimal solution cannot be measured.

5.4. Visualization analysis

To intuitively display the solution results, we randomly generated an example for each of the three problems and used the Gurobi solver, simulated annealing algorithm, and SpoNet (sample1280) to provide a feasible solution to the problem and visualize the final solution results. Figures 3–5 illustrate the differences in performance among the three algorithms for p-Median, p-Center, and MCLP, respectively. SpoNet showed better performance than SA. Although there is a certain gap between it and the optimal solution provided by the Gurobi solver, we achieved significant time savings, which is more obvious in more complex problems.

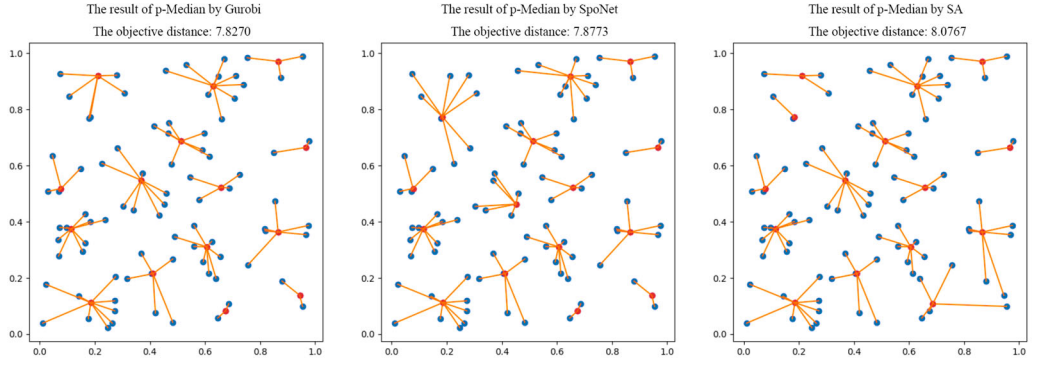

Figure 3. Visualizing solution produced by Gurobi, SpoNet, and SA algorithms for p-Median on $n = 100$, $p = 15$.

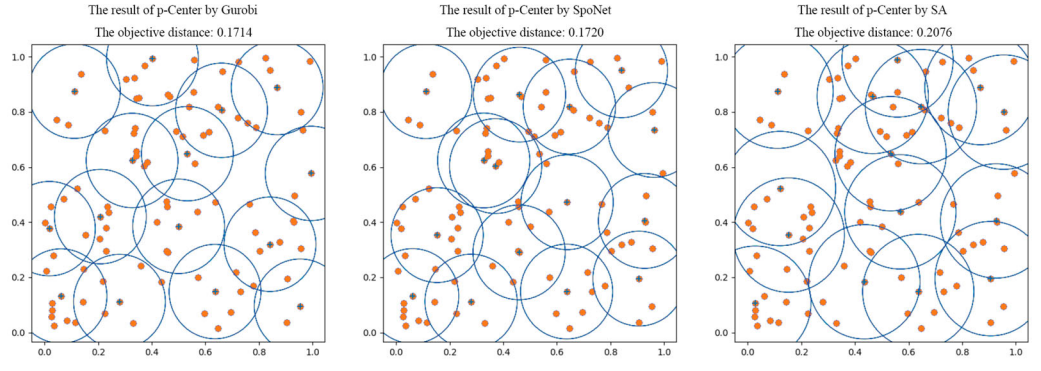

Figure 4. Visualizing solution produced by Gurobi, SpoNet, and SA algorithms for p-Center on $n = 100$, $p = 15$.

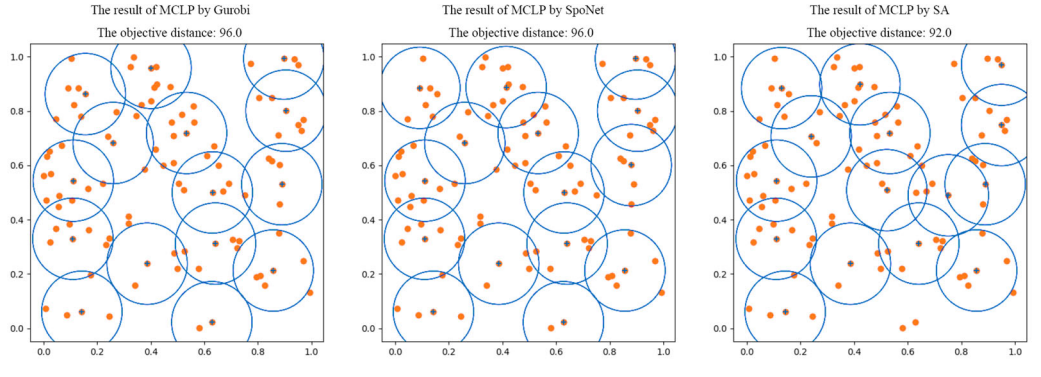

Figure 5. Visualizing solution produced by Gurobi, SpoNet, and SA algorithms for MCLP on $n = 100$, $p = 15$, $r = 0.15$.

5.5. Experiments on real scenarios

In this part, we applied SpoNet to a real-world case study to address a critical urban planning challenge. It optimizes the location of emergency facilities in Beijing's Chaoyang District. The aim is to ensure efficient emergency response and coverage for the district's residents while selecting a limited number of central hubs.

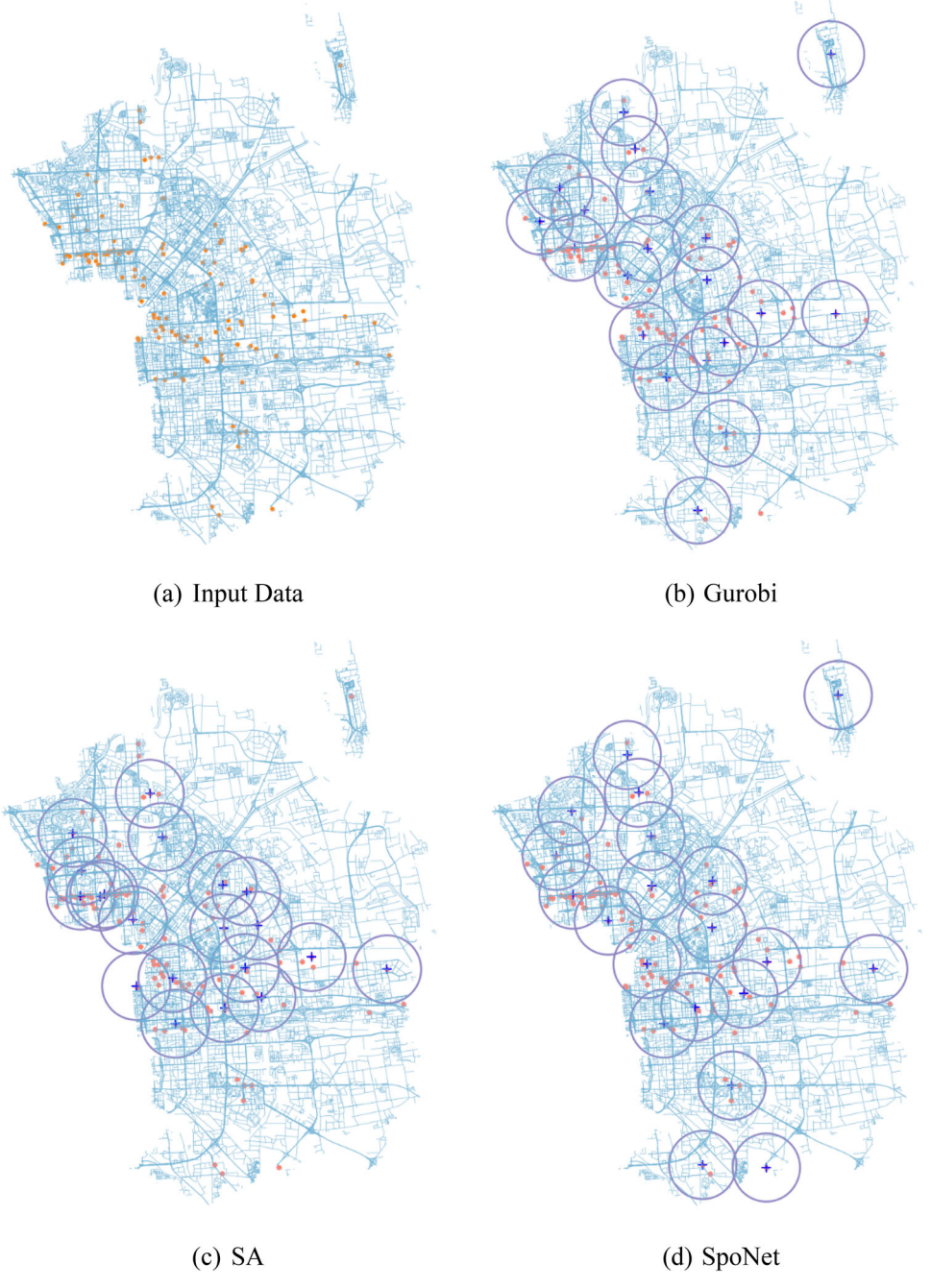

Figure 6. Visualizing solution produced by Gurobi, SA, and SpoNet for the location of emergency facilities.

The dataset comprises a total of 132 emergency facilities. We aim to determine the locations for 20 central hub points out of the 132 available options. We define the maximum service distance for each selected central hub facility as 2000 m. Before conducting any analysis, the dataset underwent essential preprocessing, including coordinate projection and normalization. These steps are crucial to ensure the compatibility of the data with the SpoNet framework and its application to urban planning and emergency response scenarios. Figure 6 presents a visual representation of the input data and a comparative analysis.

This simple yet illustrative example highlights the adaptability of our SpoNet model to real-world problem-solving scenarios, underscoring its vast potential for future research and practical applications. In contrast to mature solvers like Gurobi, which require intricate mathematical formulations, our approach is data-driven and learns how to make decisions through interactions with the environment. We present a novel approach to spatial decision analysis, offering an insightful perspective on problem-solving in complex spatial domains.

6. Conclusion

In conclusion, our work introduced a novel approach to tackle FLPs. We proposed a unified structure, SpoNet. It contains the dynamic covering information to reduce the solution space. It also adopts the GRU structure to manage the dependencies in the long-term sequence by modeling the three facility location problems as MDP and using the RL to train models. Ablation studies and experiments indicated that dynamic coverage information and the GRU structure can effectively enhance the performance of the model. Compared with other modern solvers, there is still a certain gap in optimality. However, SpoNet is suitable for large-scale problems and situations that require fast decision-making.

In future work, we will incorporate new algorithms based on a deep reinforcement learning paradigm to solve additional classical problems. We believe that this approach has great potential and can lead to significant progress in urban spatial decision analysis. We will continue exploring the application of deep learning and reinforcement learning in practical decision support systems, to provide innovative solutions for urban planning and decision-making. Many more challenges and opportunities await as we continue to push the boundaries in advancing this field. Through our ongoing research, we aim to enhance urban sustainability, improve decision accuracy, and create more intelligent urban lifestyles.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

The research were financially supported by the innovation group project of the Key Laboratory of Remote Sensing and Digital Earth Chinese Academy of Sciences (E33D0201-5), CBAS project 2023, and the Beijing Chaoyang District Collaborative Innovation Project (E2DZ050100).

Data and codes availability statement

The data and codes supporting the findings of this study are available at https://github.com/HIGISX/SpoNet.

References

Alcaraz, J., M. Landete, and J. F. Monge. 2012. "Design and Analysis of Hybrid Metaheuristics for the Reliability p-Median Problem." *European Journal of Operational Research* 222 (1): 54–64. https://doi.org/10.1016/j.ejor.2012.04.016.

Andronie, M., G. Lăzăroiu, M. Iatagan, I. Hurloiu, and I. Dijmărescu. 2021. "Sustainable Cyber-physical Production Systems in Big Data-driven Smart Urban Economy: A Systematic Literature Review." *Sustainability* 13 (2): 751. https://doi.org/10.3390/su13020751.

Beairsto, J., Y. Tian, L. Zheng, Q. Zhao, and J. Hong. 2022. "Identifying Locations for New Bike-sharing Stations in Glasgow: An Analysis of Spatial Equity and Demand Factors." *Annals of GIS* 28 (2): 111–126. https://doi.org/10.1080/19475683.2021.1936172.

Beheshti, Z., and S. M. H. Shamsuddin. 2013. "A Review of Population-based Meta-heuristic Algorithms." *International Journal of Advances in Soft Computing and its Applications* 5 (1): 1–35.

Berman, O., and D. Krass. 2002. "The Generalized Maximal Covering Location Problem." *Computers & Operations Research* 29 (6): 563–581. https://doi.org/10.1016/S0305-0548(01)00079-X.

Bleha, B., and P. Ďurček. 2023. "Unambiguous Linkage Between the Vaccination Coverage and the Spread of COVID-19: Geostatistical Evidence from the Slovak LAU 1 Regions." *Journal of Geovisualization and Spatial Analysis* 7 (1): 14. https://doi.org/10.1007/s41651-023-00144-2.

Chung, J., C. Gulcehre, K. Cho, and Y. Bengio. 2014. "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling." arXiv preprint arXiv:1412.3555.

Church, R., and C. R. Velle. 1974. "The Maximal Covering Location Problem." *Papers in Regional Science* 32 (1): 101–118. https://doi.org/10.1111/j.1435-5597.1974.tb00902.x.

Daskin, M. 1997. "Network and Discrete Location: Models, Algorithms and Applications." *Journal of the Operational Research Society* 48 (7): 763–764. https://doi.org/10.1057/palgrave.jors.2600828.

Deudon, M., P. Cournut, A. Lacoste, Y. Adulyasak, and L. M. Rousseau. 2018. "Learning Heuristics for the tsp by Policy Gradient." In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 15th International Conference, CPAIOR 2018, Delft, The Netherlands, June 26–29, 2018, Proceedings 15*, 170–181. Springer International Publishing.

Domínguez, E., and J. Muñoz. 2008. "A Neural Model for the p-Median Problem." *Computers & Operations Research* 35 (2): 404–416. https://doi.org/10.1016/j.cor.2006.03.005.

Gao, F., S. Li, Z. Tan, and S. Liao. 2022. "Visualizing the Spatiotemporal Characteristics of Dockless Bike Sharing Usage in Shenzhen, China." *Journal of Geovisualization and Spatial Analysis* 6 (1): 12. https://doi.org/10.1007/s41651-022-00107-z.

Guo, H., D. Liang, Z. Sun, F. Chen, X. Wang, J. Li, J. Bian, et al. 2022. "Measuring and Evaluating SDG Indicators with Big Earth Data." *Science Bulletin* 67 (17): 1792–1801. https://doi.org/10.1016/j.scib.2022.07.015.

Gurobi Optimization, LLC. 2021. "Gurobi Optimizer Reference Manual."

Hakimi, S. L. 1964. "Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph." *Operations Research* 12 (3): 450–459. https://doi.org/10.1287/opre.12.3.450.

Hakimi, S. L. 1965. "Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems." *Operations Research* 13 (3): 462–475. https://doi.org/10.1287/opre.13.3.462.

Hochreiter, S., and J. Schmidhuber. 1997. "Long Short-term Memory." *Neural Computation* 9 (8): 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Huang, X., Z. Li, Y. Jiang, X. Ye, C. Deng, J. Zhang, and X. Li. 2021. "The Characteristics of Multi-source Mobility Datasets and How they Reveal the Luxury Nature of Social Distancing in the US during the COVID-19 Pandemic." *International Journal of Digital Earth* 14 (4): 424–442. https://doi.org/10.1080/17538947.2021.1886358.

Jiang, H., H. Guo, Z. Sun, Q. Xing, H. Zhang, Y. Ma, and S. Li. 2022. "Projections of Urban Built-up Area Expansion and Urbanization Sustainability in China's Cities through 2030." *Journal of Cleaner Production* 367:133086. https://doi.org/10.1016/j.jclepro.2022.133086.

Kaveh, A., and M. Khayatazad. 2012. "A New Meta-heuristic Method: Ray Optimization." *Computers & Structures* 112–113:283–294. https://doi.org/10.1016/j.compstruc.2012.09.003.

Kazakovtsev, L. A., and A. A. Stupina. 2014. "Fast Genetic Algorithm with Greedy Heuristic for p-Median and *k*-Means Problems." In *2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 602–606. IEEE.

Kingma, D. P., and J. Ba. 2014. "Adam: A Method for Stochastic Optimization." *arXiv preprint arXiv:1412.*6980.

Kool, W., H. van Hoof, and M. Welling. 2018. "Attention, Learn to Solve Routing Problems!" International Conference on Learning Representations.

Kwon, Y. D., J. Choo, B. Kim, I. Yoon, Y. Gwon, and S. Min. 2020. "Pomo: Policy Optimization with Multiple Optima for Reinforcement Learning." *Advances in Neural Information Processing Systems* 33:21188–21198.

Lăzăroiu, G., L. Ionescu, M. Andronie, and I. Dijmărescu. 2020. "Sustainability Management and Performance in the Urban Corporate Economy: A Systematic Literature Review." *Sustainability* 12 (18): 7705. https://doi.org/10.3390/su12187705.

Li, X., X. Li, and X. Ma. 2022. "Spatial Optimization for Urban Green Space (UGS) Planning Support Using a Heuristic Approach." *Applied Geography* 138:102622. https://doi.org/10.1016/j.apgeog.2021.102622.

Li, K., T. Zhang, R. Wang, Y. Wang, Y. Han, and L. Wang. 2021. "Deep Reinforcement Learning for Combinatorial Optimization: Covering Salesman Problems." *IEEE Transactions on Cybernetics* 52 (12): 13142–13155. https://doi.org/10.1109/TCYB.2021.3103811.

Liang, H., S. Wang, H. Li, H. Ye, and Y. Zhong. 2022. "A Trade-Off Algorithm for Solving p-Center Problems with a Graph Convolutional Network." *ISPRS International Journal of Geo-Information* 11 (5): 270. https://doi.org/10.3390/ijgi11050270.

Manual, C. U. S. 1987. "Ibm Ilog Cplex Optimization Studio." *Version* 12 (1987–2018): 1.

Mazyavkina, N., S. Sviridov, S. Ivanov, and E. Burnaev. 2021. "Reinforcement Learning for Combinatorial Optimization: A Survey." *Computers & Operations Research* 134:105400. https://doi.org/10.1016/j.cor.2021.105400.

Mecheter, I., M. Abbod, A. Amira, and H. Zaidi. 2022. "Deep Learning with Multiresolution Handcrafted Features for Brain MRI Segmentation." *Artificial Intelligence in Medicine* 131:102365. https://doi.org/10.1016/j.artmed.2022.102365.

Nica, E., G. H. Popescu, M. Poliak, T. Kliestik, and O.-M. Sabie. 2023. "Digital Twin Simulation Tools, Spatial Cognition Algorithms, and Multi-sensor Fusion Technology in Sustainable Urban Governance Networks." *Mathematics* 11:1981. https://doi.org/10.3390/math11091981.

Peng, B., J. Wang, and Z. Zhang. 2020. "A Deep Reinforcement Learning Algorithm Using Dynamic Attention Model for Vehicle Routing Problems." In *Artificial Intelligence Algorithms and Applications: 11th International Symposium, ISICA 2019, Guangzhou, People's Republic of China, November 16–17, 2019, Revised Selected Papers 11*, 636–650. Springer Singapore.

Pirkul, H., and D. Schilling. 1989. "The Capacitated Maximal Covering Location Problem with Backup Service." *Annals of Operations Research* 18 (1): 141–154. https://doi.org/10.1007/BF02097800.

Rabiei-Dastjerdi, H., G. McArdle, S. A. Matthews, and P. Keenan. 2021. "Gap Analysis in Decision Support Systems for Real-estate in the era of the Digital Earth." *International Journal of Digital Earth* 14 (1): 121–138. https://doi.org/10.1080/17538947.2020.1808719.

ReVelle, C., M. Scholssberg, and J. Williams. 2008. "Solving the Maximal Covering Location Problem with Heuristic Concentration." *Computers & Operations Research* 35 (2): 427–435. https://doi.org/10.1016/j.cor.2006.03.007.

ReVelle, C. S., and R. W. Swain. 1970. "Central Facilities Location." *Geographical Analysis* 2 (1): 30–42. https://doi.org/10.1111/j.1538-4632.1970.tb00142.x.

Song, M., and D. Chen. 2018. "A Comparison of Three Heuristic Optimization Algorithms for Solving the Multi-objective Land Allocation (MOLA) Problem." *Annals of GIS* 24 (1): 19–31. https://doi.org/10.1080/19475683.2018.1424736.

Teitz, M. B., and P. Bart. 1968. "Heuristic Methods for Estimating the Generalized Vertex Median of a Weighted Graph." *Operations Research* 16 (5): 955–961. https://doi.org/10.1287/opre.16.5.955.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Łukasz Kaiser, and I. Polosukhin. 2017. "Attention is All You Need." *Advances in Neural Information Processing Systems*, 30.

Vinyals, O., M. Fortunato, and N. Jaitly. 2015. "Pointer Networks." In *Proceedings of the 28th International Conference on Neural Information Processing Systems,* Volume 2 (pp. 2692–2700).

Wang, S., H. Liang, Y. Zhong, X. Zhang, and C. Su. 2023. "DeepMCLP: Solving the MCLP with Deep Reinforcement Learning for Urban Facility Location Analytics." https://doi.org/10.25436/E2KK5V.

Williams, R. J. 1992. "Simple Statistical Gradient-following Algorithms for Connectionist Reinforcement Learning." *Reinforcement Learning*, 5–32. https://doi.org/10.1007/978-1-4615-3618-5_2.

Williamson, D. P., and D. B. Shmoys. 2011. *The Design of Approximation Algorithms*. Cambridge university press.

Xia, Y., H. Chen, C. Zuo, and N. Zhang. 2022. "The Impact of Traffic on Equality of Urban Healthcare Service Accessibility: A Case Study in Wuhan, China." *Sustainable Cities and Society* 86:104130. https://doi.org/10.1016/j.scs.2022.104130.

Xiao, X., D. Xu, and W. Wan. 2016. "Overview: Video Recognition from Handcrafted Method to Deep Learning Method." In *2016 International Conference on Audio, Language and Image Processing (ICALIP)*, 646–651. IEEE.

Xin, L., W. Song, Z. Cao, and J. Zhang. 2021. "Multi-decoder Attention Model with Embedding Glimpse for Solving Vehicle Routing Problems." *Proceedings of the AAAI Conference on Artificial Intelligence* 35 (13): 12042–12049. https://doi.org/10.1609/aaai.v35i13.17430.

Zhang, Z., H. Hu, D. Yin, S. Kashem, R. Li, H. Cai, D. Perkins, and S. Wang. 2020. "A cyberGIS-Enabled Multi-Criteria Spatial Decision Support System: A Case Study on Flood Emergency Management." *International Journal of Digital Earth,* 167–184. https://doi.org/10.1080/17538947.2018.1543363.

Zhao, Q., D. J. Sailor, and E. A. Wentz. 2018. "Impact of Tree Locations and Arrangements on Outdoor Microclimates and Human Thermal Comfort in an Urban Residential Environment." *Urban Forestry & Urban Greening* 32: 81–91. https://doi.org/10.1016/j.ufug.2018.03.022.