Proceedings of the 16th Conference on Computer Science and Intelligence Systems pp. 1–7
DOI: 10.15439/2021F001
ISSN 2300-5963 ACSIS, Vol. 25

# An Efficient Connected Swarm Deployment via Deep Learning

Kiril Danilchenko
*School of Electrical and Computer Engineering*
*Ben-Gurion University of the Negev*
Beer-Sheva, Israel
kirild@post.bgu.ac.il

Michael Segal
*School of Electrical and Computer Engineering*
*Ben-Gurion University of the Negev*
Beer-Sheva, Israel
segal@bgu.ac.il

***Abstract*—In this paper, an unmanned aerial vehicles (UAVs) deployment framework based on machine learning is studied. It aims to maximize the sum of the weights of the ground users covered by UAVs while UAVs forming a connected communication graph. We focus on the case where the number of UAVs is not necessarily enough to cover all ground users.**

**We develop an UAV Deployment Deep Neural network (*UDDNNet*) as a UAV's deployment deep network method. Simulation results demonstrate that *UDDNNet* can serve as a computationally inexpensive replacement for traditionally expensive optimization algorithms in real-time tasks and outperform the state-of-the-art traditional algorithms.**

## I. Introduction

IN THIS work we consider the case where Unmanned Aerial Vehicles (UAVs) provide coverage to ground users within the serving area. Thus, each UAV serves as an aerial Base Station (BS). Different researchers introduced different objectives and proposed different aerial BSs deployment algorithms in order to solve them. The objective function of these algorithms varied from maximizing the system capacity to minimizing the required number of aerial BSs.

In the last years, the researchers have focused on deployment optimization algorithms that minimize the required number of aerial BSs providing wireless coverage to all ground users. Opposite to most existing work, where the number of UAVs is assumed to be enough to cover all ground users [13], [14], [17], in this study, we consider a more realistic scenario: UAVs are given, and their number is not necessarily enough to cover all ground users. This assumption is reasonable in any practical scenario. For example, in emergency cases, the number of available UAVs is limited, but the number of first responders (firefighters, medicals, etc.) is larger than the UAVs can cover.

In this work, we assume that each ground user has a *rank* that defines the importance of the user's coverage. The rank and location of ground users can be changed from time to time. Thus, we are facing with the question: "Who should be covered and who should not, at any point in time?" For a given particular time snapshot, this problem is known to be NP-hard [4]. Additionally, we require the local connectivity between the UAVs among themselves (see Figure 1) and not through some global entity that connects between them.

We adopt the model and problems first proposed in [3]. Formally, we consider a set $S$ of $n$ points distributed in the plane, where each point $s_i \in S, i = 1, \ldots, n$, has a positive weight $w(s_i)$. We assume that all UAVs fly at the same fixed altitude. All UAVs fly at the same fixed altitude, and covering disks on the ground have the same radius. Denote this radius as $R_{COV}$, and each UAV has a communication radius ($R_{COM}$). Now, we define the covering problems formally where UAVs provide connectivity between themselves. Consider a set $P$ of $m$ disks (represent the covering disks of the UAV's) of radius $R_{COV}$, where set $C$ contains the centers of these disks.

**Connected max(S,m)**($Cmax(S, m)$): Given a set $S$ and a parameter $R_{COM}$, place the disks from the set $P$ such that:

1) The total weight of points from the set $S$ covered by the disks is maximized.
2) The undirected graph $G = (C, E)$ imposed on $P$ should be connected, where an edge $(u, v) \in E$ if $d(u, v) \leq R_{COM}$, for $d(u, v)$ being the $L_2$ distance between the centers $v, u \in C$.

Similarly to the above, the **Connected-Dynamic max(S,m)** ($CDmax(S, m)$) problem aims to maintain the disks from $P$ under dynamic updates of $S$.

State-of-the-art solutions often involve exhaustive searches or the optimization of various heuristics. We tackle our problem from a different perspective; we leverage recent deep learning (DL) advances to design a novel deep neural network (DNN) architecture to provide better performance with reduced runtime. The proposed DNN approach establishes a connection between the total weight cover maximization problems under connectivity constraints while minimizing a loss function during DNN training. Additionally, it relies on an efficient network training and ensemble mechanism to beat state-of-the-art solutions.

The main contributions of this work can be summarized as follows. First, we propose a UAV's deployment strategy using a DNN. To this end, we offer a novel DNN structure trained on the optimal solutions to a target problem. We use a supervised learning approach to solve our problem compared to another approaches that use the unsupervised or reinforcement methods. The uniqueness of our study lies in the fact that our training set is constructed being based on optimal solutions for this problem. Therefore, the solutions derived by our DNN

have efficiency close to that of an optimal solution. Second, the performance of the proposed DNN approach is verified through extensive evaluation. The simulation results confirm that the proposed DNN achieves outstanding approximation solutions to the target problem with shorter than the state-of-the-art evaluated solutions computation times.

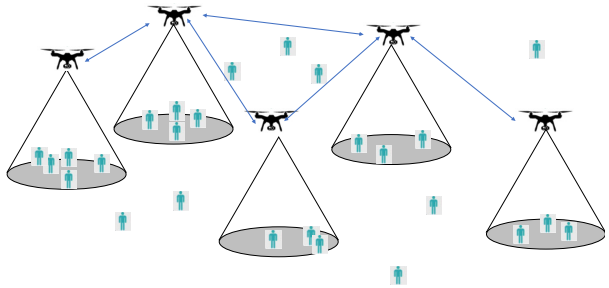

Fig. 1: The UAVs provide connectivity among themselves.

The remainder of this paper is organized as follows. Next Section II discusses recent related studies. The details of the proposed Deep Learning architecture are described in Section III.In section IV we summarized all notation used in this paper. In Section V we describe all aspects of evaluation setups. The simulation results are described in Section VI. Finally, conclusions and suggestions for future research are presented in Section VII.

## II. Related Work

In this section, we first briefly review the typical approaches. There is a growing number of researches on the topic of UAV-based stations (aerial-BS) placement. UAV deployment under different constrains have been widely discussed in recent years and resulted in the development of various heuristics [13], [14], [17], [22], [23], [20], [20]. The problem of placing $m$ disks that cover a maximal weight of given points (without connectivity requirement) has been given some attention in the past. The authors of [4] presented the problem of covering the maximum number of points in the point set $S$ with $m$ unit disks, without the demand for disks connectivity. They gave a $(1-\varepsilon)$-approximation algorithm with time complexity $O(n\varepsilon^{-4m+4})\log^{2m-1}(\frac{1}{\varepsilon})$. The problem to place $m$ rectangles such that the sum of the weights of the points in $S$ covered by these rectangles is maximized is considered in [10]. For any fixed $\varepsilon > 0$, the authors present efficient approximation schemes that can find a $(1-\varepsilon)$-approximation to the optimal solution in $O(\frac{n}{\varepsilon}\log(\frac{1}{\varepsilon}) + m(\frac{1}{\varepsilon})^{O(\min(\sqrt{m},\frac{1}{\varepsilon}))})$ runtime. In [6] the authors presented a PTAS for a more general case different covering shapes (disks, polygons with $O(1)$ edges), running in $O(n\frac{1}{\varepsilon}^{O(1)} + \frac{m}{\varepsilon}\log m + m(\frac{1}{\varepsilon})^{O(\min(m,\frac{1}{\varepsilon}))})$ time. The authors of [19] solve the relevant problem to place two disks in the plane to ensure both maximal covering and full connectivity by providing two algorithms having $O(n^4)$ and $O(n^3\log n)$ time complexity, respectively. Another related problem is presented in [9], [5]. In these works, the authors formulate the following problem: given a set of $n$ discs in the plane, select a subset of $k$ disks that maximize the area of their union, under the constrain that this union is connected. The authors of [2] consider the $Cmax(S,m)$. They gave $O(\frac{1}{\sqrt{m}})$ with time complexity $O(\beta^2 mn\log n)$, where $\beta \cdot R_{COM} = d_{max}$ and $d_{max}$ be the largest $L_\infty$ distance defined by a pair of points in $S$. In [3] the authors gave $O(1)$ approximation for $Cmax(S,m)$, and presented an algorithm for $CDmax(S,m)$ using $O(m\sqrt{m})$ UAVs with the approximation ratio $O(1)$.

We continue with researches that use Machine Learning techniques. The authors of [8] aim how to maximize the number of users covered by the system in an emergency scenario. They proposed the use of RL (Q-learning) to determine the optimal position of the UAVs. The proposed solution was compared to different positioning strategies and outperformed all other methods in all considered metrics. In [16] the authors considered the problem of the optimal deployment of multiple UAVs to maximize throughput for ground users with different requirements. The authors use Reinforcement Learning (RL) to calculate the locations of the UAVs. Qiu et al. [18] considered the problem of maximizing the coverage rate of $N$ ground users by the simultaneous placement of multiple UAVs with a limited coverage range. They applied the Deep Reinforcement Learning method to cope with this problem. Liu et al. [11] proposed a deep RL (DRL), a method for energy-efficient UAV control to provide communication coverage for ground users. The control policy considers the UAV movements in each time slot, and the aim is to optimize the communication coverage, fairness, energy consumption, and connectivity. Liu et al. [12] developed a fast positioning algorithm for the deployment of aerial BSs, where the objective is to maximize the sum of the downlink rates in the multiple UAV communication network. They designed a geographical position information (GPI) learning algorithm. Dai et al. [1] investigated the problem of the efficient deployment of UAVs in order guarantee the quality-of-service requirements. The UAV played the role of a coordinator to provide high-quality communication service for ground users and maximize the benefits of caching. The authors proposed an RL-based approach to solving the multi-objective deployment problem while maintaining an optimal tradeoff between the power consumption and backhaul saving. They adopted the RL approach to determine the 3D placement and minimum transmit power, and cache strategy of each UAV.

In summary, recent studies have used DL to solve aerial BS (UAVs) deployment under different objectives. We propose a novel method of using DL to solve the UAVs deployment such that the UAVs cover a maximal total weight of ground users under the connectivity requirement between UAVs. We use a supervised approach to solve our problem compared to other researches using the unsupervised or reinforcement methods. The uniqueness of our study lies in the fact that our training set is constructed basing on optimal solutions for this problem. Therefore, the solutions derived by our DNN have efficiency

close to that of an optimal solution.

## III. *UDDNNet* STRUCTURE

In this section, we describe the proposed *UDDNNet*, including the details of the DNN design and a training process based on supervised learning.

In the following, we detail the proposed DNN architecture and discuss how training and testing are performed.

*1) Network Structure:* Our proposed approach uses a fully connected neural network with two input layers, $L = 8$ fully connected hidden layers, and one output layer.

The first input for the proposed network is a matrix with dimensions of $3 \times n$, where entry $i$ of the matrix represents a location and a weight of ground user $i$.

The second input is a binary matrix with dimensions of $2 \times \mathcal{K}$, where a number of rows without zero elements equals the maximal number of UAVs that is possible to use in this scenario. Note that the $\mathcal{K}$ represents the maximum value of $m$ in our experiments. Denote this matrix as *FilMat*. We use this input as a binary filter matrix, meaning a non-zero entry in the matrix signifies that we can put an UAV in this location. Therefore, we use this matrix to avoid a scenario in which the DNN located more UAVs than given in advance.

The first hidden layer reshapes the input matrix into a one-dimensional vector with a length of $5000$. The second hidden layer reshapes its input into a one-dimensional vector with a length of $4000$. In this manner, the following five hidden layers perform reshaping until the output has dimensions of $100$. The $8^{-th}$ hidden layer reshapes the one-dimensional vector with a length of $100$ into a matrix with dimensions of $2 \times \mathcal{K}$. This matrix multiplied by *FilMat* and the result of this multiplication is an input of the output layer, where a activation function Eq. 1 was applied. The output of the network is the location of $m$ UAV's.

The $ReLU(x)$ function is used as the activation function for the hidden layer, where $ReLU(x)$ is the rectified linear unit function $\max(x, 0)$ [15]. Additionally, to enforce the location constraint we adopt a special activation function from [21] for the output layer of the DNN, as shown below.

$$y(x) = \min(ReLU(x), \sqrt{A}), \tag{1}$$

We apply this activation function in the output layer to limit the output location to the range of $[0, \sqrt{A}]$, where $A$ is the square zone of interest area.

We let $l_k$ to denote the number of neurons in the $k^{-th}$ layer. The $k^{-th}$ layer is a hidden layer and its output is calculated as follows:

$$c_k = ReLU(W_k \cdot c_{k-1}), \tag{2}$$

where $c_{k-1}$ and $c_k$ are the output vectors of the previous and current layers with dimensions of $l_{k-1} \times 1$ and $l_k \times 1$, respectively. $W_k$ is the $l_k \times l_{k-1}$ weight matrix.

A detailed explanation of the DNN architecture is provided in Fig. 2. The motivation of the proposed DNN architecture is to "shrink" the inputs (location and weight of the ground users) into $m$ two dimensional coordinates, the locations of the UAVs.

## IV. NOTATIONS

The notations used in this paper are summarized in Table I

| Symbol | Meaning |
|---|---|
| $S$ | The set of $n$ ground users |
| $C$ | The set of the centers of the disks |
| $R_{COV}$ | The covering disk radius in the case of |
| $R_{COM}$ | The communication radius |
| $m$ | The number of available UAVs |
| $A$ | The area of zone of interest |
| $\mathcal{K}$ | The maximum value of $m$ |

TABLE I: Summary of notations used in this study.

## V. EXPERIMENTAL SETUP

To evaluate the proposed *UDDNNet* performance, we conducted experiments with a different number of ground users and the different number of available UAVs. This section describes the data generation process, splitting of whole data set to training, validation and testing sets, training details, and testing process.

*1) Data Generation:* The *UDDNNet* was trained using optimal solutions implemented in the Wolfram Language. Data was generated in the following manner.

First, we randomly distribute $m$ disks in an area of interest with dimensions of $5000 \times 5000$ m$^2$, such that the disk graph imposed on their centers is connected. We set $m$ to be $m \sim U[2, \ldots, \mathcal{K}]$, where $\mathcal{K} = 10$. Next, we distributed on these disks between $10\% - 30\%$ of ground users. Finally, we randomly distributed the ground users in an area of interest. Also, for each ground user we randomly assign a weight $w(s_i) \sim U[0, 1]$.

We repeated the process described above multiple times to generate a dataset. The final dataset contained approximately $100000$ instances. We randomly split the dataset into three sets for training, validation and testing, where the sizes of each set were $70\%$, $10\%$ and $20\%$ of the entire dataset, respectively.

*2) Training Process:* We used the entire *training dataset* to optimize the weights of the neural network. The loss function we adopted was the mean absolute error (MAE) as a loss between the optimal UAV's location and the network's output. We used the ADAM optimizer [7] for optimization. We analyzed the impact of the batch size and learning rate of *UDDNNet*. Based on the results presented in Fig.3 and Fig.4, we selected a batch size of $256$ and the learning rate was set to $0.0001$.

In Fig. 5 we can see the the training error and the validation error as a function of the training epoch (rounds) with the parameters chosen in Fig.3 and Fig.4. We can see that a validation error decreases when the number of rounds is increased.

*3) Testing Process:* In the testing stage, we used the testing dataset, passed each instance through the trained *UDDNNet*, and collected the results-location of the UAVs. We then compared the resulting total covering weight by UAVs achieved

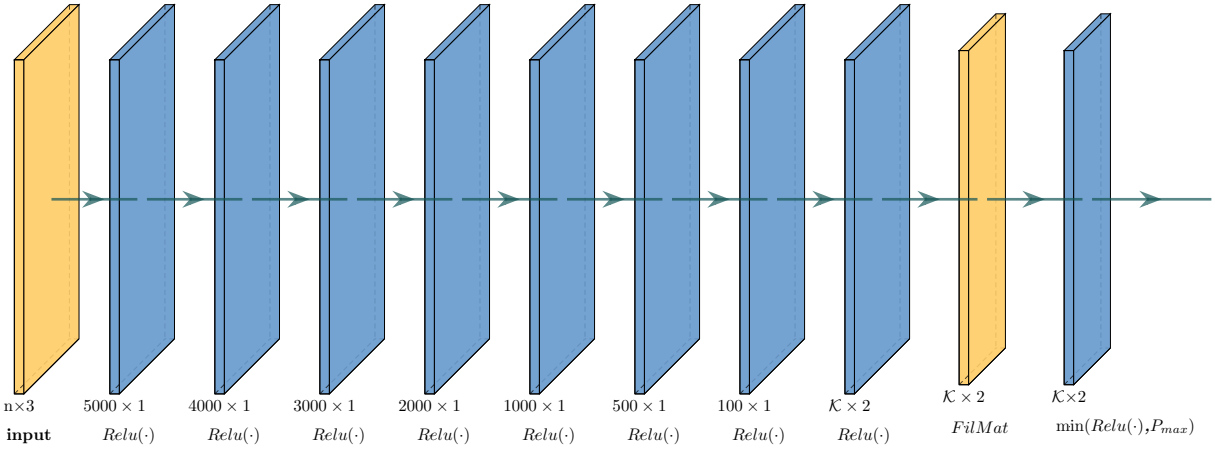


Fig. 2: DNN architecture for UAVs deployment.

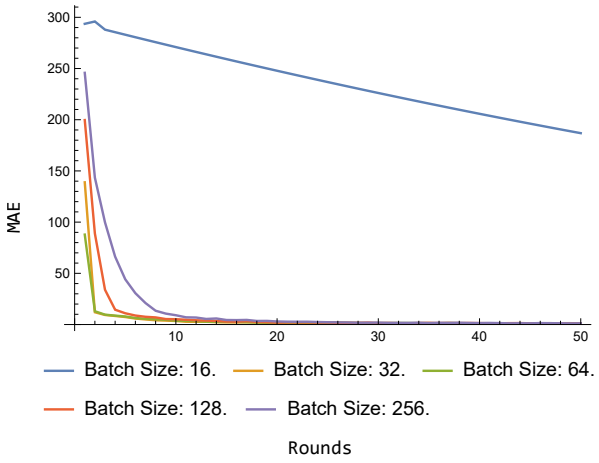


Fig. 3: Batch size selection

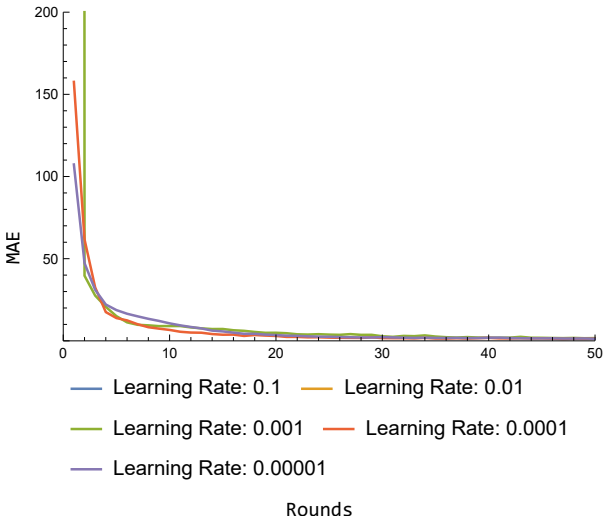


Fig. 4: Learning rate selection

by the compression scheme and the solution based on the locations generated by *UDDNNet*.

*4) Schemes for Comparison:* Besides *UDDNNet* we also implemented the algorithms presented in [3]. The authors divide the area of interest into a grid with cell size $r$. They represent each cell as a node in the graph with a weight equal to the total sum of ground users' weights belong to this cell. They gave $O(1)$ approximation for $Cmax(S,m)$and presented an algorithm for $CDmax(S,m)$ using $O(m\sqrt{m})$ UAVs to keep the approximation ratio $O(1)$, where each update takes $O(\log n)$ runtime. We compare the performance of *UDDNNet* with the algorithms that solve $Cmax(S,m)$ and $CDmax(S,m)$ from [3].

## VI. Evaluation Results

The proposed DNN approach was implemented in Wolfram 12.3 on a single desktop computer with the hardware specifications listed below.

1) Intel CPU Core i7-8700K @ 3.70 GHz
2) Nvidia GPU GeForce GTX 1080Ti

The GPU was used in the training stage to reduce training time but was not used in the testing stage.

### A. Numerical Results

We conducted numerical simulations to verify the effectiveness of the *UDDNNet* and compare it to the heuristic presented in [3]. Detailed simulations allowed us to study the performance of the proposed *UDDNNet*, which is defined as the total weight of covered users and the runtime required by *UDDNNet*. Specifically, we examined the performance obtained by *UDDNNet* for different numbers of ground users and available UAVs to solve $CDmax(S,m)$ and $CDmax(S,m)$. Table II gathers the parameters that, unless otherwise speci-

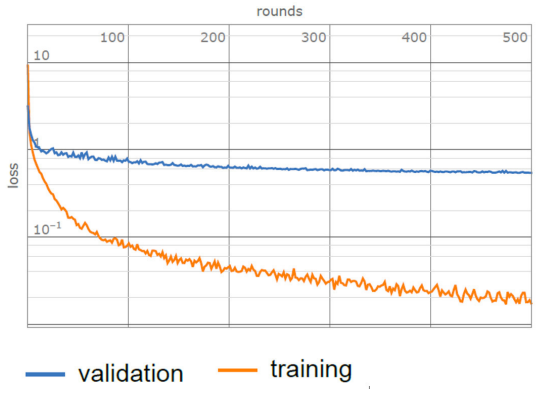


Fig. 5: Traning Procces

fied, we have used for the network model, regardless of the simulation environment.

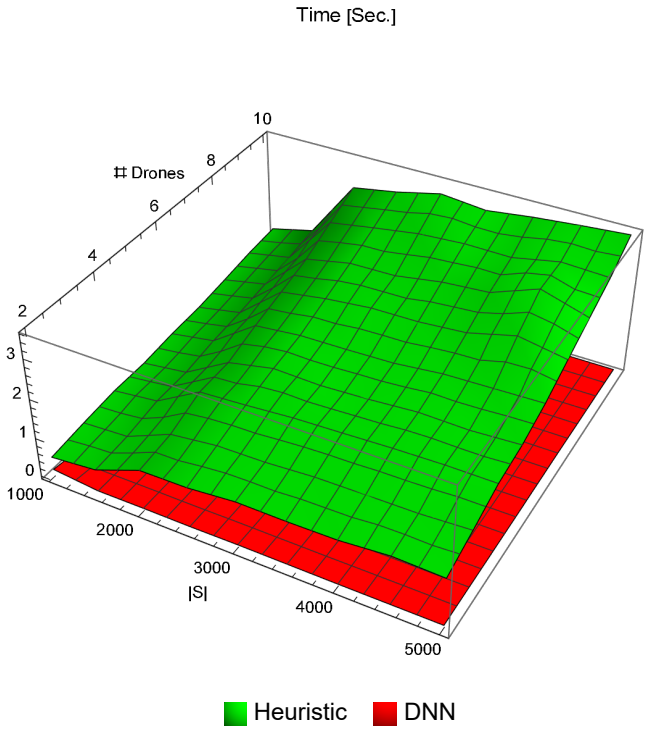


Fig. 6: Time complexity of *UDDNNet* v.s. $Cmax(S,m)$ heuristics from [3].

We start by examining our method's performance in the static version $Cmax(S,m)$, for the case where the node $i$ weight is uniformly distributed $w_i = U \sim [0,1]$. The results of this examination we can see in Fig. 7 and Fig.6 showing the superiority of *UDDNNet* approach. In particular, Figure 7 presents the total weight of covered users achieved by *UDDNNet* versus the solution from [3]. In this Figure, one can see that *UDDNNet* achieves better performance than the solution from [3] for a problem with different numbers of nodes and available UAVs. In Figure 6, we present the running

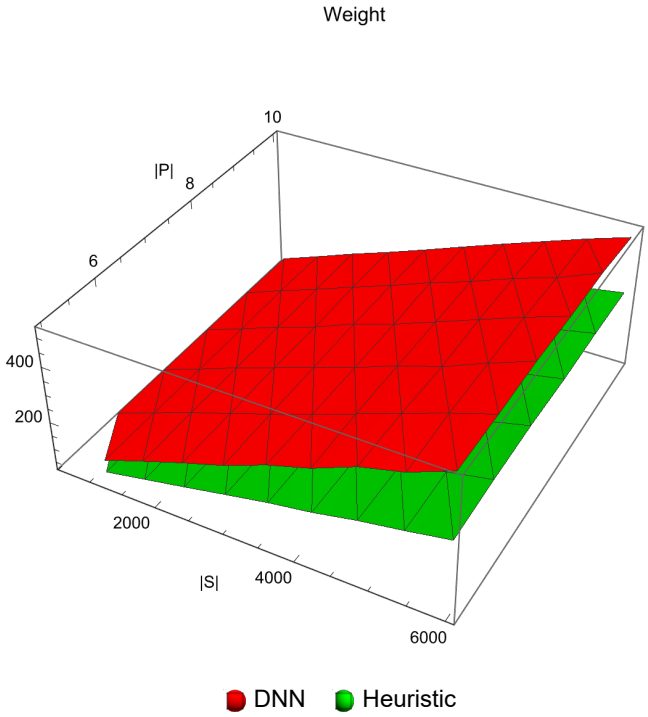


Fig. 7: Weight covered by *UDDNNet* and $Cmax(S,m)$ heuristics from [3].

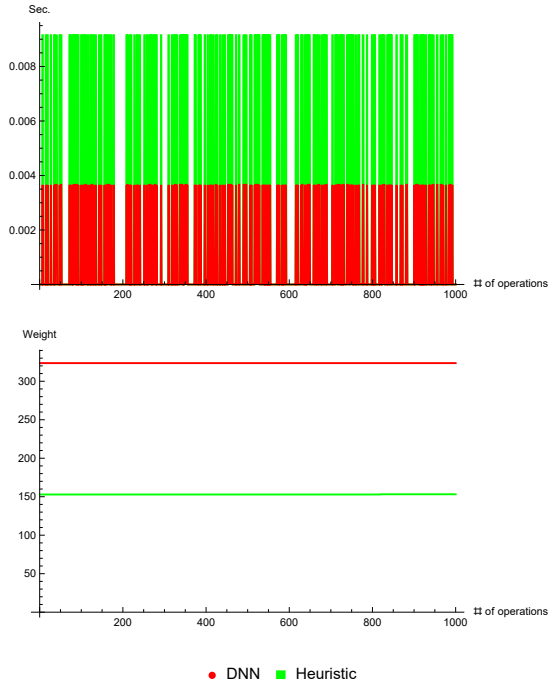


Fig. 8: Maintenance of Dynamic Covering Set. The number of ground users is 3000 and the number of UAVs is 3.

time of *UDDNNet* versus that of heuristic solution from

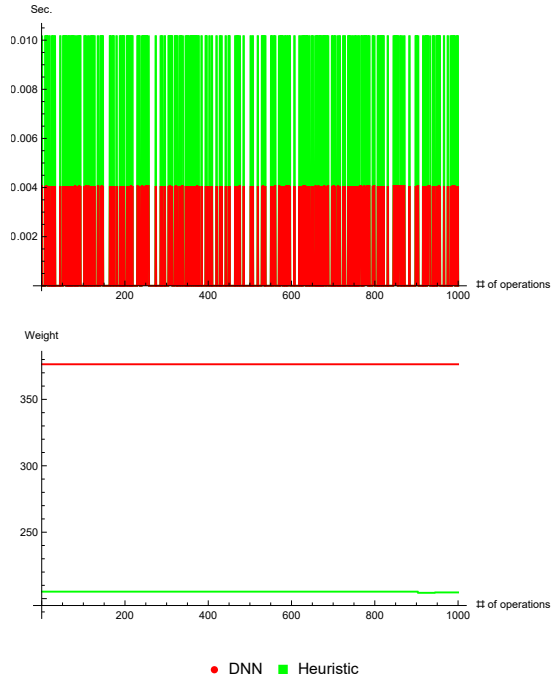


Fig. 9: Maintenance of Dynamic Covering Set. The number of ground users is 3500 and the number of UAVs is 4.

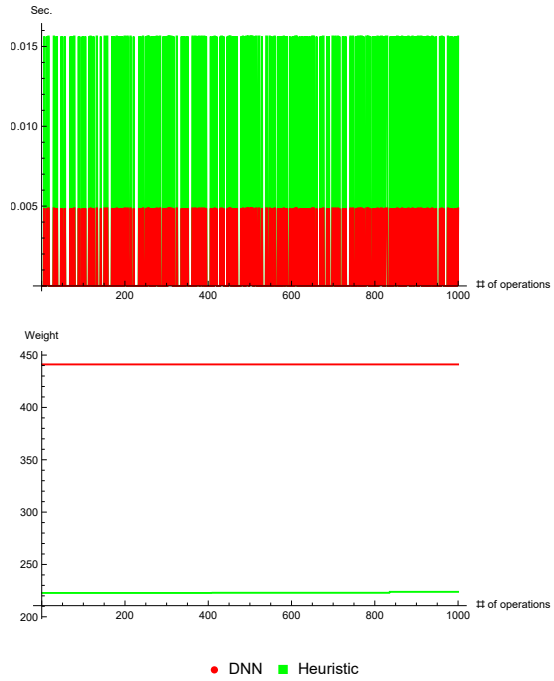


Fig. 10: Maintenance of Dynamic Covering Set. The number of ground users is 400 and the number of UAVs is 6.

[3]. One can see that *UDDNNet*'s run-time is approximately constant and is a magnitude lower than that of the solution

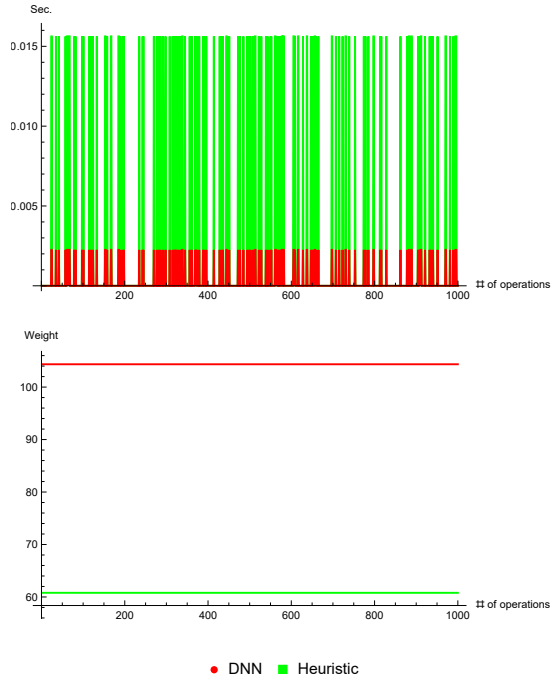


Fig. 11: Maintenance of Dynamic Covering Set. The number of ground users is 1000 and the number of UAVs is 2.

| **Parameter** | **Value** |
|---|---|
| Number of ground users | $1000 - 5000$ |
| Ground user's weights | $w(s_i) \sim U[0,1]$ |
| Number of UAVs (drones) | $2 - 10$ |
| Simulation Playground Size | $5000 \times 5000$ m$^2$ |
| $R_{COV}$ | 100 m |
| $R_{COM}$ | 200 m |
| $m$ | $2 - 10$ |

TABLE II: Simulation Configuration

from [3] that we use as a baseline for a problem with different numbers of nodes and available UAVs.

Now we deal with the dynamic version $CDmax(S, m)$. We examined the performance obtained by [3] and *UDDNNet* for different configurations, where we allow the use of a different number of ground users and UAVs. Note that, at each change of $S$, we solve the problem from scratch by *UDDNNet*. Therefore, *UDDNNet* solves at each change the static version of the given set $S$.

The Figures 8-11 represent the maintenance of set $P$ under insertions or deletions of a point from set $S$. Each subfigure of these figures includes two graphs. The top graph represents the time complexity of dynamic maintenance of solution from [3] versus the time complexity of *UDDNNet*. The bottom graph represents the total weight covered by *UDDNNet* and out scheme for comparison. In both graphics, axis $x$ represents the trace of 1000 operations on set $S$, where each operation may

be insertion or deletion. Additionally, the underneath graph's axis $y$ represents the total weight covered by *UDDNNet* and solution from [3], and the $y$ axis in the left graph represents the time required to execute the operation. We again can witness the better performance of *UDDNNet* in terms of runtime and obtained weight of covered ground users.

## VII. Conclusions and Future Work

In this study, we considered the connected version of the covering problem motivated by the coverage of ad-hoc UAVs swarm. Inspired by recent advances in artificial intelligence, we proposed the use of Deep Learning to address this problem. We developed a fully connected multi-layer neural network that takes a ground user's location and a number of available UAVs as inputs and outputs the location of the UAVs. A supervised learning strategy was adopted to train *UDDNNet* by using optimal solutions as a training dataset.

Our results are encouraging in many respects. The time complexity of the proposed DNN solutions is the most important factor among our results. Therefore, the key takeaway from our research is that a DNN can serve as a computationally inexpensive component to replace expensive optimization algorithms for real-time tasks while providing very good performance compared to state-of-the-art methods.

## Acknowledgments

This research has been supported by a grant from Pazy Foundation.

## References

[1] Haibo Dai, Haiyang Zhang, Baoyun Wang, and Luxi Yang. The multi-objective deployment optimization of uav-mounted cache-enabled base stations. *Physical Communication*, 34:114–120, 2019.

[2] Kiril Danilchenko and Michael Segal. Connected ad-hoc swarm of drones. In *Proceedings of the 6th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications*, DroNet '20, New York, NY, USA, 2020. Association for Computing Machinery.

[3] Kiril Danilchenko, Michael Segal, and Zeev Nutov. Covering users by a connected swarm efficiently. In *International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics*, pages 32–44. Springer, 2020.

[4] Mark De Berg, Sergio Cabello, and Sariel Har-Peled. Covering many or few points with unit disks. *Theory of Computing Systems*, 45(3):446–469, 2009.

[5] Chien-Chung Huang, Mathieu Mari, Claire Mathieu, Joseph SB Mitchell, and Nabil H Mustafa. Maximizing covered area in the euclidean plane with connectivity constraint. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[6] Kai Jin, Jian Li, Haitao Wang, Bowei Zhang, and Ningye Zhang. Near-linear time approximation schemes for geometric maximum coverage. *Theoretical Computer Science*, 725:64–78, 2018.

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[8] Paulo V Klaine, João PB Nadas, Richard D Souza, and Muhammad A Imran. Distributed drone base station positioning for emergency cellular networks using reinforcement learning. *Cognitive computation*, 10(5):790–804, 2018.

[9] T. Kuo, K. C. Lin, and M. Tsai. Maximizing submodular set function with connectivity constraint: Theory and application to networks. *IEEE/ACM Transactions on Networking*, 23(2):533–546, 2015.

[10] Jian Li, Haitao Wang, Bowei Zhang, and Ningye Zhang. Linear time approximation schemes for geometric maximum coverage. In *International Computing and Combinatorics Conference*, pages 559–571, 2015.

[11] Chi Harold Liu, Zheyu Chen, Jian Tang, Jie Xu, and Chengzhe Piao. Energy-efficient uav control for effective and fair communication coverage: A deep reinforcement learning approach. *IEEE Journal on Selected Areas in Communications*, 36(9):2059–2070, 2018.

[12] Jie Liu, Qiang Wang, Xuan Li, and Wenqi Zhang. A fast deployment strategy for uav enabled network based on deep learning. In *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–6, 2020.

[13] Jiangbin Lyu, Yong Zeng, Rui Zhang, and Teng Joon Lim. Placement optimization of uav-mounted mobile base stations. *IEEE Communications Letters*, 21(3):604–607, 2016.

[14] Mohammad Mozaffari, Walid Saad, Mehdi Bennis, and Mérouane Debbah. Efficient deployment of multiple unmanned aerial vehicles for optimal wireless coverage. *IEEE Comm. Lett.*, 20(8):1647–1650, 2016.

[15] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[16] Yu Min Park, Minkyung Lee, and Choong Seon Hong. Multi-uavs collaboration system based on machine learning for throughput maximization. In *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 1–4. IEEE, 2019.

[17] Luigi Di Puglia Pugliese, Francesca Guerriero, Dimitrios Zorbas, and Tahiry Razafindralambo. Modelling the mobile target covering problem using flying drones. *Optimization Letters*, 10(5):1021–1052, 2016.

[18] Jin Qiu, Jiangbin Lyu, and Liqun Fu. Placement optimization of aerial base stations with deep reinforcement learning. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2020.

[19] Sanaz Soltani, Mohammadreza Razzazi, and Hossein Ghasemalizadeh. The most points connected-covering problem with two disks. *Theory of Computing Systems*, 62(8):2035–2047, 2018.

[20] Anand Srinivas, Gil Zussman, and Eytan Modiano. Construction and maintenance of wireless mobile backbone networks. *IEEE/ACM Transactions on Networking*, 17(1):239–252, 2009.

[21] Haoran Sun, Xiangyi Chen, Qingjiang Shi, Mingyi Hong, Xiao Fu, and Nicholas D Sidiropoulos. Learning to optimize: Training deep neural networks for interference management. *IEEE Transactions on Signal Processing*, 66(20):5438–5453, 2018.

[22] Haijun Wang, Haitao Zhao, Weiyu Wu, Jun Xiong, Dongtang Ma, and Jibo Wei. Deployment algorithms of flying base stations: 5g and beyond with uavs. *IEEE Internet of Things Journal*, 6(6):10009–10027, 2019.

[23] Xiao Zhang and Lingjie Duan. Fast deployment of uav networks for optimal wireless coverage. *IEEE Transactions on Mobile Computing*, 18(3):588–601, 2018.