Wireless Networks
https://doi.org/10.1007/s11276-023-03635-8

ORIGINAL PAPER

Density-based anti-clustering for scheduling D2D communications

Ahmed Elsheikh[1] · Ahmed S. Ibrahim[2] · Mahmoud H. Ismail[3]

Accepted: 19 December 2023

Abstract
Wireless link scheduling in device-to-device (D2D) networks is an NP-hard problem. As a solution, multiple supervised deep learning (DL) models have been recently proposed, which depend on the geographical information of D2D pairs. However, such DL models require labeled training data. In this paper, we focus on unsupervised learning of scheduling. More specifically, this paper proposes using a Density-Based anti-Clustering for Scheduling D2D Communications (DBSCHedule). The proposed algorithm is a two-step approach that consists of clustering and anti-clustering. First, clustering aims at identifying the non-interfering groups of D2D pairs. Then, anti-clustering aims at identifying the maximally separated sub-groups to minimize the interference. The clustering step uses a fully-automated unsupervised density-based spectral-clustering of applications with noise (DBSCAN) and the anti-clustering uses the inverse of the objective function of the k-means clustering. Results show comparable performance with the optimal FPLinQ scheduler yet without requiring any channel information nor is there a requirement to solve a complex optimization problem. Moreover, a comparable performance to the previous attempts using DL and modified clustering is achieved while being completely adaptive and easily accommodating to changes in the network layout.

Keywords Device-to-device · Machine learning · Scheduling · Unsupervised learning · Anti-clustering

1 Introduction

Device-to-device (D2D) communication is an essential component in modern cellular systems mainly because of its support in offloading traffic from the base stations [1]. D2D communication avoid the centralized computational and communication bottleneck to serve better quality of service as well as save energy, which makes it an important step towards green communication as well [2]. However, the problem of link scheduling in D2D communication is NP hard and requires solving complex optimization problems, which could be solved using strategies like FPLinQ and FLashLinQ [3]. Some attempts sought sub-optimal solutions to reduce the complexity such as fractional programming algorithms that optimize ratios of functions and then undergo a certain approximate transformation [4], heuristics, which can use any form of minimization or maximization of utility under certain assumptions [5], and adaptive learning optimization, which uses deep learning for the pruning step instead of calculating all possibilities for a certain branching operation [6, 7].

All of the aforementioned algorithms, however, rely on the existence of channel estimates to do the predictions. This can impede the development of fast algorithms that can cope with fast network dynamics and communication service requirements. Several attempts were made in the literature to try and alleviate the need for channel estimation and use only distance information to solve the D2D scheduling problem. Using distance information only to solve D2D scheduling problems is very challenging. This is why several attempts in the literature used other algorithms' outputs to train Deep Learning (DL) models. In this paper, we propose a novel solution for D2D scheduling that can inherently do scheduling without the need of another algorithm to work with. We first summarize the previous related works on solving D2D scheduling problems and

✉ Ahmed Elsheikh
ahmed.elsheikh@eng.cu.edu.eg

[1] Department of Mathematics and Engineering Physics, Cairo University, Giza 12613, Egypt

[2] Department of Electrical and Computer Engineering, Florida International University, Miami, FL 33174, USA

[3] Department of Electrical Engineering, American University of Sharjah, PO Box 26666, Sharjah, UAE

Published online: 27 January 2024

build up the base for the contributions of this work in the next section.

2 Related work

DL has found its way in numerous applications nowadays including wireless communications [8]. Resource allocation is one of the most challenging tasks in communication networks where previous research explored DL in power allocation in the framework of reinforcement learning and fusion [9–11], to learn optimization solutions for spectrum sharing [12], and to solve linear sum assignment problems [13]. The use of DL specifically in D2D link scheduling was explored by [14] and [15]. The work in [14] extracts interference and distance information based on kernel filters that are learned from synthetically generated data. However, in order to learn a sufficient number of filters to solve the scheduling problem, a significant amount of training data is required. Federated learning was also explored to distribute the computational burden to multiple devices and a central node [16]. On the other hand, the graph embedding approach in [15, 17] eases the burden of learning the mappings through kernel filters by preparing neighborhood graphs describing the network through pairwise distances. Recurrent neural networks (RNNs) were also studied using sequence-to-sequence learning [18] and geometrical manifolds [19].

As much as these works are successful, a significant part of learning is labelled data. In contrast, unsupervised learning techniques have the advantage of requiring only data without being labelled, which makes them more versatile especially with dynamic deployment in roadside units (RSUs) in vehicular networks, relays, and base stations. There has been an attempt to use clustering algorithms, which fall under unsupervised learning, to achieve the required task in [20]. The authors modified several well-known clustering algorithms in order to achieve link scheduling with equal-size clusters. The number of clusters was determined using the final number of network active links as a percentage of the result obtained from the optimal FPLinQ scheduler.

Motivated by the above, this paper proposes using a Density-Based anti-Clustering for Scheduling D2D Communications (DBSCHedule). The proposed algorithm is a two-step approach where the first step is to cluster D2D pairs that are in spatial proximity defined by a certain radius within which interference is significant. The second step is to identify, within each cluster, a set of pairs that are as far away from each other as possible. These objectives can be achieved using density-based spectral-clustering of applications with noise (DBSCAN) [21] and anti-clustering [22], respectively, for D2D link scheduling. DBSCAN clusters D2D pairs based on their spatial location and density where more dense regions are considered clusters with high risk of interference. It is a highly adaptive model and is computationally far less expensive than other supervised models such as neural networks. This makes it a prominent model for dynamic networks because first, the model is based on the density of D2D pairs, which is a relative measure to the network layout and not an absolute one. Second, the reduction in computational requirements, which is due to the straightforward decision making process requiring very few iterations and hence, faster decisions. Third, it only requires distance information, which does not cause a lot of overhead to acquire or estimate. Anti-clustering then identifies D2D pairs that are furthest apart within each cluster to minimize the interference within that cluster. The number of anti-clusters sought is simply 2, one will be scheduled and the other will not. Anti-clustering will be based on the resulting maximum sum of separating distances. The algorithm will require as input only the locations of the D2D pairs that can be acquired using global positioning systems, and a preset radius of interference. No knowledge about channel state information (CSI) is needed.

The main contributions of this paper can thus be summarized as follows:

- Adopting a completely unsupervised technique for D2D link scheduling with automated hyperparameter selection.
- Using density-based clustering and anti-clustering to tackle the D2D link scheduling problem.
- Achieving comparable performance to optimal as well as solutions obtained using supervised learning with no need for running a trainer optimizer to acquire guidance, and minimal computational requirements.

The paper is laid out as follows: Sect. 3 describes the system model, then the proposed link scheduling algorithm is described in Sect. 4. The simulation setup and results are described in Sect. 5. Finally, Sect. 6 concludes the paper findings.

3 System model

We consider a network with N_d D2D pairs that are assumed to be located randomly in a two-dimensional square area with length L and the separation distances between the D2D pairs are uniformly distributed between $l_{\min}$ and $l_{\max}$. The D2D pairs are allowed to move at a typical pedestrian speed in random directions while keeping the separation distance constant. Furthermore, we assume the transmission power is constant. A graphical representation of the network layout is shown in Fig. 1 with several D2D pairs

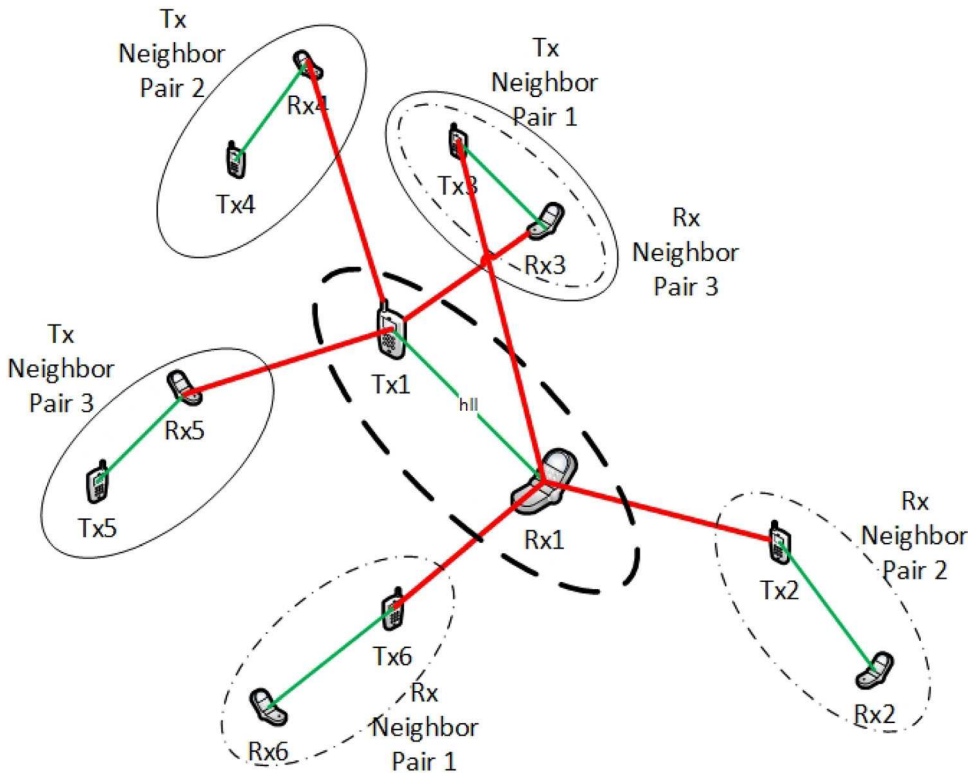

Fig. 1 Channel links between transmitters and receivers in a given D2D network. Each ellipse shows a pair of devices involved in a D2D communication session

and the pair under study is in the middle with the neighboring D2D pairs surrounding it. The channel gain for the communication link between the transmitter (Tx) device ℓ and the receiver (Rx) device k is denoted by $h_{\ell k}$. The goal is to identify which of the links need be active to maximize a certain performance metric. A typical metric for performance would be the sum-rate, where the rate for the link between the Tx device ℓ and the Rx device k where $\ell, k \in \{1, 2, \ldots, N_d\}$ is defined using information theoretic capacity as follows:

$$R_\ell = BW \times \log_2\left(1 + \frac{|h_{\ell\ell}|^2 P_\ell \delta_\ell}{\sum_{k\neq\ell} |h_{\ell k}|^2 P_k \delta_k + \sigma_N^2}\right), \tag{1}$$

where BW is the bandwidth, P_ℓ is the power of the Tx of the ℓth link, σ_N^2 denotes the additive white Gaussian noise power and $\delta_k \in \{0, 1\}$ indicates whether the kth link is active or not. It is worth mentioning here that other forms of communications like base station to user can be considered as part of the interfering term in the denominator of (1). This is done here as a simplification since the main focus of this work is the scheduling problem itself. In addition, this is assumed for the sake of comparison with other works that consider only D2D communication without considering overlaying devices as in [15] and [20].

D2D link scheduling is a typical NP-hard optimization problem whose formulation requires CSI collection at a central node from each of the links and demands a lot of computations to reach an optimal solution. Clearly, the maximization of the sum-rate would entail identifying $\delta_k \in \{0, 1\}$ such that the sum-rate for the N_d devices in the network is maximized as follows:

$$\max_{\delta_k} \sum_{k=1}^{N_d} \delta_k R_k \ . \tag{2}$$

4 DBSCHedule for D2D link scheduling

Clearly, the channel gains between the D2D pair members as well as the interference channels are distance-dependent as shown in (1). Hence, to decrease the interference, the clustering step in DBSCHedule thus identifies the pairs that

are in close proximity to each other. These clusters have inter-separating distances more than the expected harmful interference distance. Hence, these clusters are considered non-interfering clusters. Moreover, the pairs within each of these clusters are at a high risk of interfering on each other if activated all at once. By doing so, the set of active pairs in (1) will no longer be assumed interfering and the denominator can now be considered as follows:

$$I_C = \sum_{k,l \in C, k \neq \ell} |h_{\ell k}|^2 P_k \delta_k + \sigma_N^2, \tag{3}$$

where C is the non-interfering cluster set which the pair ℓ belongs to after clustering.

The clustering step in DBSCHedule uses DBSCAN to cluster the D2D pairs based on their coordinates into N_C clusters where the coordinate of each pair is calculated as the midpoint between the Tx and Rx members of the pair and the clusters are constructed based on a certain preset interference radius r. Assuming that the set of midpoint coordinates between the different D2D pairs can be given by D, then for any two pairs with midpoint coordinates d_m and d_n in $\mathbb{R}^2$ where $m, n \in \{1, 2, ..., N_d\}$ to belong to the same cluster, the following condition must be met:

$$\|d_m - d_n\| \leq r. \tag{4}$$

From the D2D scheduling perspective, this radius defines the region where an active link can cause significant interference on other neighboring active links. The resulting clusters are either singleton, which means they contain only one pair, or contain multiple pairs. In case of a singleton cluster $|C| = 1$, the decision is then to schedule this pair for communication. Otherwise, if there is more than one pair in a given cluster then a fraction of the clustered links is chosen to be active. The factor α is the within-cluster active percentage and it is preset for the scheduler. The scheduler then identifies the pairs within each cluster that are maximally separated as potential pairs for scheduling. This can be achieved by utilizing the anti-clustering technique [23], which is essentially the opposite notion of clustering where members of the same anti-cluster are very well separated from each other, while members of different anti-clusters are very similar to each other. For scheduling purposes, we need only two anti-clusters within each cluster: scheduled and unscheduled. By doing so, R_ℓ will be further maximized by reducing the interference set given in (3) to $C_\alpha \subset C$ such that $|C_\alpha| = \lfloor \alpha |C| \rfloor$ where C_α includes the pairs m and n such that the following is maximized:

$$\sum_{m \in C_\alpha} \sum_{n \in C_\alpha} \|d_m - d_n\| + \sum_{m \in C \setminus C_\alpha} \sum_{n \in C \setminus C_\alpha} \|d_m - d_n\|, \tag{5}$$

So, scheduled anti-clusters will be at the furthest distance from each other, while the unscheduled pairs will be as close as possible to scheduled ones. The proposed approach is summarized in Algorithm 1. Figure 2 shows a typical scenario while scheduling using DBSCHedule. The solid ellipses identify the clusters of D2D pairs that are at risk of interfering on each other because they are in close proximity to each other. The singleton pair (at the bottom right) will be scheduled as well as those potential pairs that are encompassed by the dashed ellipses, which are identified by the anti-clustering step because they are at a maximum distance from each other within the cluster. The within-cluster active percentage α is what decides how many pairs will be active from the potential maximally separated pairs. The final set of pairs that will be active are those with the smallest Tx-Rx distances.

DBSCHedule solution is hence a multi-step heuristic approach for speeding up and simplifying the maximization of the sum rate for D2D communication requiring only location-based information. The first step aims at identifying the D2D pairs that do not contribute to interference and hence, can be scheduled and at the same time reducing the search space for the proper D2D pairs to be scheduled. Figure 3 shows each cluster in the network of $N_d = 30$ D2D pairs, where each cluster has a different color and symbol. Next, two subsets of the D2D clusters that are at risk of interfering with each other will be identified such that these subsets are maximally separated by anti-clustering. Finally, a fraction of those subsets will be chosen based on their separating distance, which in turn maximizes the sum rate. Figure 4 shows the scheduled D2D pairs encompassed by green circles, and the unscheduled D2D pairs marked by a red cross.

Algorithm 1 DBSCHedule

Initialization: Interference radius = r, Within-cluster active ratio = α, A number of scheduling layouts M.
For each D2D pair, calculate the location midpoint between the Tx and Rx.
for $m = 1$ to M **do**
  for $i = 1$ to N_d **do**
    Identify the pairs linking to each other within a radius r.
    Give each cluster of linked pairs a certain label.
    The total number of resulting clusters will be denoted N.
    Each cluster will have a number of pairs N_c.
    for $c = 1$ to N **do**
      if $N_c = 1$ **then**
        Schedule as active.
      else
        Sort pairs in cluster c based on Tx-Rx distance of each pair.
        Anti-cluster cluster c into two anti-clusters.
        Decide which anti-cluster to schedule based on the sum of separating distances.
        Choose the most separated α pairs from the chosen anti-cluster.
      end if
    end for
  end for
end for

5 Simulation results

5.1 Simulation setup

The simulated network layouts are generated using the code from [14] and the parameters described in [15] for proper comparison. The default number of D2D pairs is chosen to be 50 in the square coverage area described in Section refsec:D2D link Scheduling. The number of generated testing samples is 1000. DBSCHedule is fully unsupervised and hence, requires no training data. All the other parameters are summarized in Table 1. The performance is quantified as a percentage of the sum-rate obtained from the optimal scheduler FPLinQ [24]. The results will be compared to those reported in [20] as well as the unsupervised graph embedding model results reported in [15].

5.2 Modelling hyperparameters and their estimation

The performance of DBSCHedule, like any other machine learning technique, depends on a set of hyperparameters that are not learnt from the data and need to be properly chosen. Setting hyperparameters for unsupervised learning techniques is even more difficult due to the lack of any sort of general guidelines [25]. Nevertheless, specific applications can have some guidelines to find adequate estimates of such hyperparameters. To study the impact of changing the two main hyperparameters used in DBSCHedule, which are the radius of interference and the within-cluster activity factor, the following subsections illustrate different experiments to identify the sensitivity of the performance to each hyperparameter, and the best practices for the proposed DBSCHedule algorithm. Some suggested approaches are then provided for estimating these parameters for a given network.

5.2.1 Within-cluster activity hyperparameter tuning

Figure 5 shows how the performance is impacted with the change of the within-cluster activity factor for different number of D2D pairs. It is clear from the figure that the best performance is always obtained at the same value of α irrespective of the D2D pairs density (since the different number of D2D pairs are assumed within the same area). This means that the selection of α can be done once for a given network area and can still be used whenever there is a change in the users' density.

Also, the network size has low impact on the selection of the optimal α as shown in Fig. 6. This again asserts the fact that the tuning of the within-cluster activity percentage α can only be done once upon deployment.

5.2.2 Radius of interference hyperparameter tuning

Figure 7 shows how the radius of interference impacts the performance of the network for different number of D2D

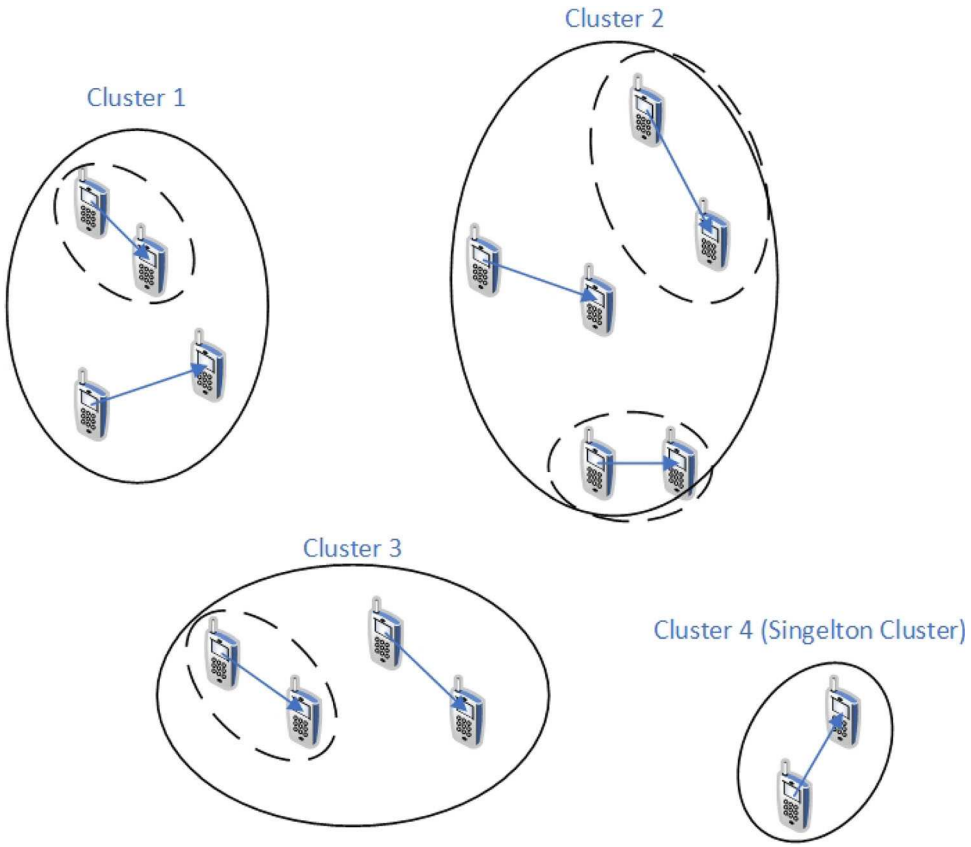

Fig. 2 DBSCHedule typical scheduling scene

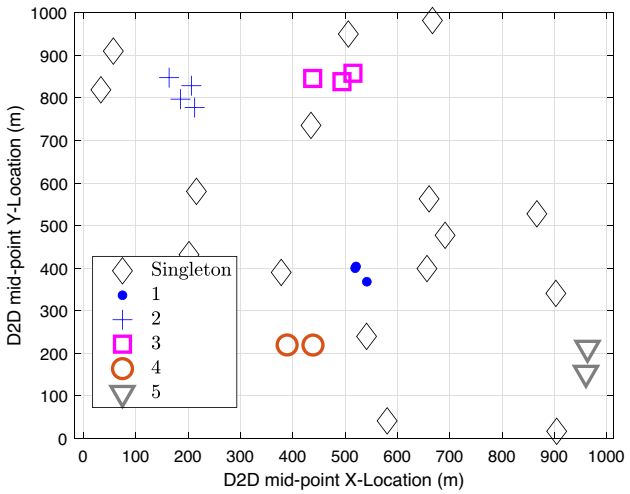

Fig. 3 A network of $N_d = 30$ D2D pairs clustered by DBSCAN

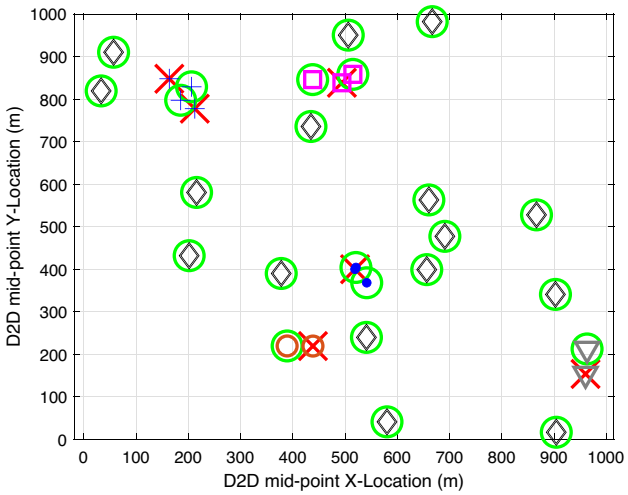

Fig. 4 A network of $N_d = 30$ D2D pairs clustered by DBSCAN and scheduled by Anti-clustering. The green circles mark scheduled D2D pairs and the red crosses mark the unscheduled pairs

pairs. It can be readily seen that the choice of the radius is almost the same when the area is kept constant irrespective of the number of D2D pairs.

As for Fig. 8, it shows the impact of changing the network size on the choice of the optimal radius of interference while maintaining the same number of D2D pairs. It is evident that the network size has a greater impact on the selection of this hyperparameter. This means that the radius of interference needs to be tuned for every coverage area. Accordingly, the initial estimation of α is sufficient for any given network setting. For our simulations, $\alpha = 0.14$ achieves approximately the best performance for different network sizes and densities.

Table 1 Summary of simulation and model parameters

Parameter	Value
Square area side length	500 m
D2D distance	$2-65$ m
Noise spectral density	-169 dBm/Hz
Bandwidth	5 MHz
Carrier frequency	2.4 GHz
Antenna height	1.5 m
Active link transmit power	40 dBm
D2D pair speed	5 km/h

5.2.3 Circle packing for estimating the DBSCHedule's hyperparameters

The experiments in this section suggest that both hyperparameters are not sensitive to the density of the D2D pairs in the network. On the other hand, for different network sizes, the radius of interference r showed to be more sensitive than the within-cluster activity factor α.

Circle packing is a geometrical problem for analyzing how many circles can fit within or around a certain area with the highest efficiency. The proposed DBSCHedule algorithm in the simulation under study can be ultimately considered as two circle-packing problems of the interference circles. The packed circles in this case would be of radius equal to the radius of interference r.

When estimating the within-cluster activity factor, we consider a central circle with radius r surrounded by circles also of radius r. The maximum number of circles of radius r that can be packed around the circumference of the central circle is 6. This scenario represents a dense network and the worst-case scenario. In this case, the central circle under consideration would be active only if all surrounding circles are inactive. Accordingly, only 1 out of 7 circles will be active, i.e., $\alpha \approx 0.14$, which is in accordance with our findings.

For estimating the radius of interference r, then given an expected number of D2D pairs N_d, the maximum r would be such that N_d circles can be packed in a square of side length equal to that of the network coverage area. Approximations of the radii that satisfy such conditions were studied in the literature in [26]. Using the highest packing factor in [26], $r \approx 0.1288$ assuming a unit square, which translates to $r \approx 64.4$ m when scaled up by 500, the square network side length, which is also in accordance with our findings from Fig. 7.

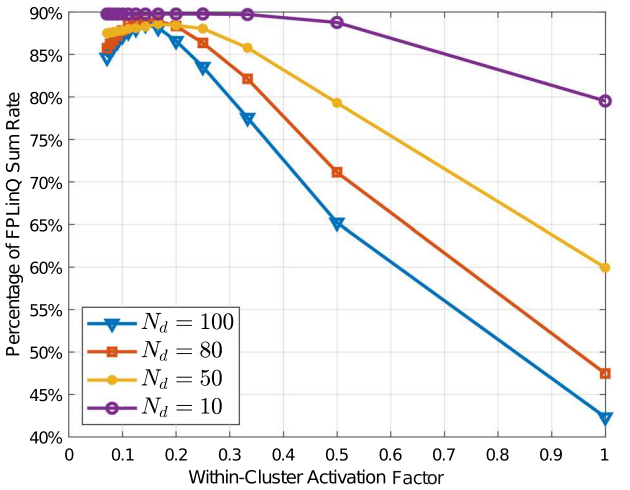

Fig. 5 Scheduling performance versus the within-cluster activity factor α for different number of D2D pairs

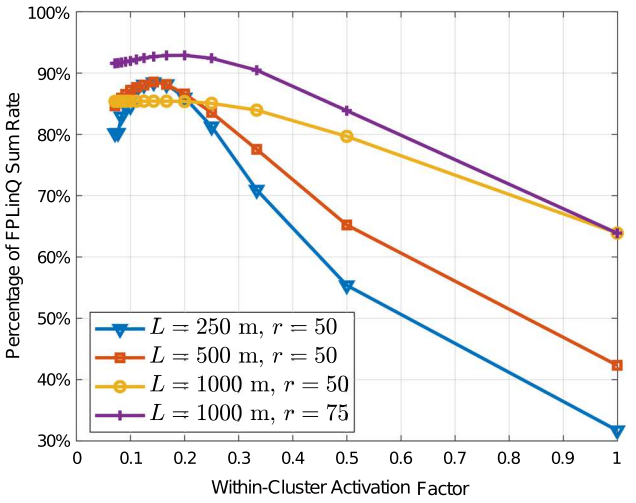

Fig. 6 The impact of the network size on the selection of the optimal α

5.3 Performance comparisons

Table 2 shows the performance of DBSCHedule compared to that of the unsupervised graph embedding as reported in [27] assuming different number of D2D pairs for a square area of 500 m side length. As before, the performance is expressed as a percentage of the sum-rate obtained from the optimal scheduler FPLinQ. Clearly, the graph embedding performance is, on average, slightly better than that of DBSCHedule, but the computational requirements for training and testing using a graph embedding and a neural network is extremely expensive compared to DBSCHedule. Moreover, DBSCHedule achieves this performance without

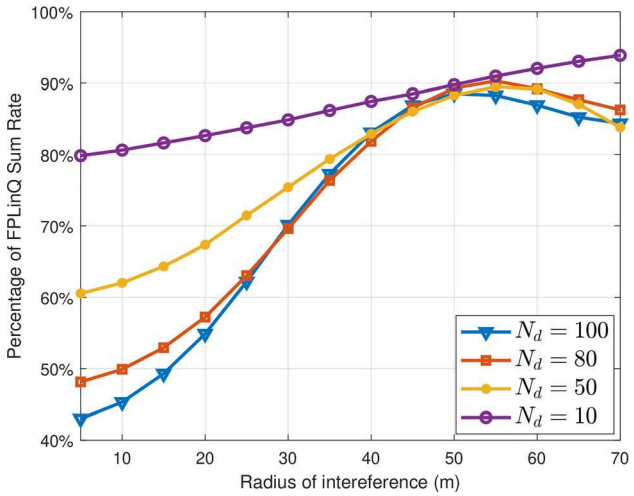

Fig. 7 Scheduling performance versus the radius of interference for different number of D2D pairs

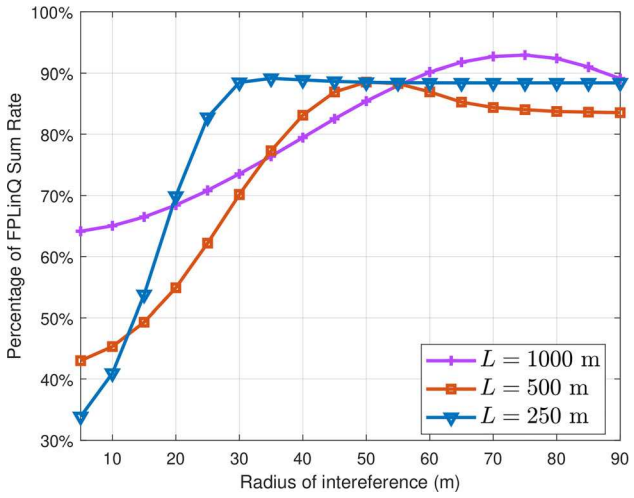

Fig. 8 The impact of the network size on the selection of the optimal radius of interference

any reference data unlike the graph-embedding model which requires an optimizer to train from. This highly limits its portability from one scene to another.

Consistency of DBSCHedule with the optimal scheduler FPLinQ in terms of the resulting average percentage of active links for the tested networks is shown in Table 3. As seen in the table, as the number of pairs in the network increases, the active percentage decreases for both models. Generally, DBSCHedule tends to result in more active links in less dense networks than FPLinQ and vice versa in more dense ones. DBSCHedule does not acquire any prior information about this percentage from FPLinQ as done in [20] to estimate the required number of clusters. The average difference between FPLinQ activity and DBSCHedule is 7.6%.

Table 2 Performance of DBSCHedule vs. the unsupervised graph embedding

Number of D2D pairs	10	50	80	100	500
Graph embedding	97.4	95.3	93.7	92.8	86.5
DBSCHedule	94	90.3	89.5	88.5	91.3

The importance of using the anti-clustering step in DBSCHedule can be deduced from Table 4. This table shows a performance comparison when using a simple random scheduling scheme versus using anti-clustering. Although the within-cluster active percentage is low (7–10%), yet for dense networks, the impact of random selection on performance is dramatic.

Finally, Table 5 shows a comparison between the performance of different clustering techniques when used in different scenarios as reported in [20] and that of DBSCHedule. Although these scenarios are not realistic in terms of having a fixed distance between the Tx and RX of the D2D pairs as well as imposing a large minimum separating distance of 5 m between the pairs, nevertheless, they are used here for comparison purposes. As seen from the results, the performance of DBSCHedule is always comparable to the other techniques except in the low density case. As mentioned previously, DBSCHedule does not require any information from the FPLinQ scheduler to perform the required scheduling while all the modified clustering techniques proposed in [20] require the resulting network activity from FPLinQ, which is not practical for a fully unsupervised approach. Also, it can be seen that the performance of DBSCHedule in the variable distance case does not vary much from the fixed case as shown in Table 2. This is unclear for the other techniques as it was not reported in [20].

Finally, Table 6 compares the time complexity of the modified clustering algorithms shown in Table 5 with that of the proposed DBSCHedule. The pre-clustering and clustering steps both have a complexity of $\mathcal{O}(N_d^2)$ in [20] while DBSCHedule does not require a pre-clustering step and has a clustering time complexity of $\mathcal{O}(N_d \log N_d)$ [25] for DBSCAN. On the other hand, [20] uses round robin scheduling between the clusters, which has a complexity of $\mathcal{O}(N_d)$. The proposed scheduler, in contrast, uses Anti-clustering, which has the same time complexity as clustering, so its complexity is $\mathcal{O}(N_c^2)$, where N_c is the number of pairs in cluster c, and $N_c \leq N_d$. By comparing the total time complexity required by each algorithm, we find that the complexity of DBSCHedule is less than that of the clustering proposed in [20].

6 Conclusion

This paper tackled the resource allocation problem in D2D networks by proposing the DBSCHedule algorithm, which is an unsupervised approach to tackle the problem using only raw location information. DBSCHedule is based on both clustering and anti-clustering concepts. Based on the conducted experiments, DBSCHedule is shown to be resilient against changes in the network density and does not require any additional information from an external optimizer to perform the link scheduling problem. It achieves a performance that is comparable to other techniques in the literature. Moreover, it has the least time complexity. Further research can be done in future work to enable the clusters and anti-clusters to evolve in time as the network users move within the network.

Table 3 Percentage of active links in the network for FPLinQ and DBSCHedule

Number of D2D pairs	10	50	80	100	500
FPLinQ	68.8	38.1	32.5	28.2	18.8
DBSCHedule	85	47.3	29.1	23.5	14.4

Table 4 Performance of DBSCHedule with anti-clustering vs. random sub-cluster scheduling

Number of D2D pairs	10	50	80	100	500
DBSCHedule anti-clustering	94	90.3	89.5	88.5	91.3
DBSCHedule random scheduling	86	64.6	48.5	44.6	23

Funding This work is supported in part by Faculty Research Grants no. EN0281:SCRI18-07 and FRG22-C-E13 at the American University of Sharjah and in part by the National Science Foundation under Award No. CNS-2144297.

Data availability The datasets analyzed during the current study are available from the following repository https://github.com/willtop/Spatial_Deep_Learning_for_Wireless_Scheduling

References

1. Hussain, F., Hassan, S. A., Hussain, R., & Hossain, E. (2020). Machine learning for resource management in cellular and IoT networks: Potentials, current solutions, and open challenges. *IEEE Communications Surveys & Tutorials, 22*(2), 1251–1275.
2. Kai, C., Li, H., Xu, L., Li, Y., & Jiang, T. (2018). Energy-efficient device-to-device communications for green smart cities. *IEEE Transactions on Industrial Informatics, 14*(4), 1542–1551.
3. Shen, K. (2020). Fractional programming for communication system design. PhD thesis, University of Toronto (Canada).
4. Wu, X., Tavildar, S., Shakkottai, S., Richardson, T., Li, J., Laroia, R., & Jovicic, A. (2013). FlashLinQ: A synchronous distributed scheduler for peer-to-peer ad hoc networks. *IEEE/ACM Transactions on Networking, 21*(4), 1215–1228.
5. Zhuang, B., Guo, D., Wei, E., & Honig, M. L. (2017). Scalable spectrum allocation and user association in networks with many small cells. *IEEE Transactions on Communications, 65*(7), 2931–2942.
6. Lee, M., Yu, G., & Li, G. Y. (2019). Learning to branch: Accelerating resource allocation in wireless networks. *IEEE Transactions on Vehicular Technology, 69*(1), 958–970.
7. Shen, Y., Shi, Y., Zhang, J., & Letaief, K. B. (2019). LORM: Learning to optimize for resource management in wireless networks with few training samples. *IEEE Transactions on Wireless Communications, 19*(1), 665–679.
8. Soldani, D., Pentikousis, K., Tafazolli, R., & Franceschini, D. (2014). 5g networks: End-to-end architecture and infrastructure [guest editorial]. *IEEE Communications Magazine, 52*(11), 62–64.
9. Nasir, Y. S., & Guo, D. (2019). Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks.

Table 5 Comparison between the performance of several modified clustering algorithms and DBSCHedule as percentages of FPLinQ performance

Network layout (N_d, Tx-Rx distance, L)	(50, 30, 500)	(30, 70, 1000)	(70, 20, 1000)
Spectral clustering	86.3	90.6	95.0
Hierarchical clustering	84.8	90.9	95.1
K-means clustering	81.8	89.4	94
DBSCHedule	75	88.8	93.4

Table 6 Comparison between the time complexity of the proposed DBSCHedule and the clustering algorithms in [20]

Point of comparison	DBSCHedule	Clustering in [20]
FPLinQ to identify number of clusters	Not required	$\mathcal{O}(N_d^2)$
Clustering complexity	$\mathcal{O}(N_d \log N_d)$ for DBSCAN	$\mathcal{O}(N_d^2)$
Scheduling complexity	$\mathcal{O}(N_c^2)$ for anti-clustering	$\mathcal{O}(N_d)$ using round robin

IEEE Journal on Selected Areas in Communications, 37(10), 2239–2250.
10. Liang, F., Shen, C., Yu, W., & Wu, F. (2019). Towards optimal power control via ensembling deep neural networks. *IEEE Transactions on Communications, 68*(3), 1760–1776.
11. Li, X., Fang, J., Cheng, W., Duan, H., Chen, Z., & Li, H. (2018). Intelligent power control for spectrum sharing in cognitive radios: A deep reinforcement learning approach. *IEEE Access, 6*, 25463–25473.
12. Sun, H., Chen, X., Shi, Q., Hong, M., Fu, X., & Sidiropoulos, N. D. (2018). Learning to optimize: Training deep neural networks for interference management. *IEEE Transactions on Signal Processing, 66*(20), 5438–5453.
13. Lee, M., Xiong, Y., Yu, G., & Li, G. Y. (2018). Deep neural networks for linear sum assignment problems. *IEEE Wireless Communications Letters, 7*(6), 962–965.
14. Cui, W., Shen, K., & Yu, W. (2019). Spatial Deep Learning for Wireless Scheduling. IEEE Journal on Selected Areas in Communications 37(6), 1248–1261. arXiv:1808.01486. https://doi.org/10.1109/JSAC.2019.2904352
15. Lee, M., Yu, G., & Li, G.Y. (2019). Graph Embedding based Wireless Link Scheduling with Few Training Samples, 1–27. arXiv:1906.02871.
16. Chen, T., Zhang, X., You, M., Zheng, G., & Lambotharan, S. (2023). Federated learning enabled link scheduling in D2D wireless networks. *IEEE Wireless Communications Letters*. https://doi.org/10.1109/LWC.2023.3321500
17. Shelim, R., & Ibrahim, A. S. (2023). Learning wireless power allocation through graph convolutional regression networks over Riemannian manifolds. *IEEE Transactions on Vehicular Technology*. https://doi.org/10.1109/TVT.2023.3325200
18. Elsheikh, A., Ibrahim, A. S., & Ismail, M. H. (2023). Sequence-to-sequence learning for link-scheduling in D2D communication networks. *Journal of Network and Computer Applications, 212*, 103567.
19. Shelim, R., & Ibrahim, A. S. (2022). Wireless link scheduling over recurrent Riemannian manifolds. *IEEE Transactions on Vehicular Technology*. https://doi.org/10.1109/TVT.2022.3228212
20. Cui, W., & Yu, W. (2020). A clustering approach to wireless scheduling. In *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, IEEE. pp. 1–5.
21. Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, 96, 226–231.
22. Späth, H. (1986). Anticlustering: Maximizing the variance criterion. *Control and Cybernetics, 15*(2), 213–218.
23. Brusco, M. J., Cradit, J. D., & Steinley, D. (2020). Combining diversity and dispersion criteria for anticlustering: A bicriterion approach. *British Journal of Mathematical and Statistical Psychology, 73*(3), 375–396.
24. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
25. Xu, D., & Tian, Y. (2015). A comprehensive survey of clustering algorithms. *Annals of Data Science, 2*(2), 165–193.
26. Peikert, R., Würtz, D., Monagan, M., & de Groot, C. (1992). Packing circles in a square: A review and new results. In *System Modelling and Optimization: Proceedings of the 15th IFIP Conference Zurich*, Switzerland, Springer, pp. 45–54 . Accessed 2–6 Sept 1991.
27. Prabhavalkar, R., Rao, K., Sainath, T., Li, B., Johnson, L., & Jaitly, N. (2017). A comparison of sequence-to-sequence models for speech recognition. http://www.isca-speech.org/archive/Interspeech_2017/pdfs/0233.PDF

Ahmed Elsheikh is currently an assistant professor at the Mathematics and Engineering Physics Department at Cairo University, Giza, Egypt. He received the B.Sc. degree in electronics and electrical communications engineering from Cairo University, Giza, Egypt, in 2011. His MSc. From the Mathematics and Engineering Physics, Cairo University, Giza, Egypt in 2014. He received the Ph.D. degree in Mathematics and Industrial engineering from the University of Montreal, Montreal, Quebec, Canada in 2018. Dr. Ahmed's research interests are focused on applied machine learning in various domains such as Engineering Physics, Industrial Engineering, and Communications.

Ahmed S. Ibrahim is currently an assistant professor at the Electrical and Computer Engineering Department at Florida International University (FIU), Miami, FL, USA. He received the B.S. (with highest honors) and M.S. degrees in electronics and electrical communications engineering from Cairo University, Giza, Egypt, in 2002 and 2004, respectively. He received the Ph.D. degree in electrical engineering from the University of Maryland, College Park, MD, USA, in 2009. Prior to joining FIU, Dr. Ibrahim was an assistant professor at Cairo University, wireless research scientist at Intel Corporation, and senior engineer at Interdigital Communications Inc. Dr. Ibrahim's research interests span various topics of next generation mobile communications and Internet of Things such as heterogeneous networks, drone-assisted millimeter wave communications, and vehicular networks.

Mahmoud H. Ismail (S'00-M'07-SM'15) received the B.Sc. degree (with highest honors) in Electronics and Electrical Communications Engineering, the M.Sc. degree in Communications Engineering both from Cairo University, Egypt, in 2000 and 2002, respectively, and the Ph.D. degree in Electrical Engineering from The University of Mississippi, MS, USA, in 2006. From August 2000 to August 2002, he was a Research and Teaching Assistant in the Department of Electronics and Electrical Communications Engineering at Cairo University. From 2004 to 2006, he was a Research Assistant in the Center for Wireless Communications (CWC) at the University of Mississippi. He is currently a Full Professor (on leave) at the Department of Electronics and Electrical Communications Engineering, Cairo University and an Associate Professor at the American University of Sharjah, Sharjah, UAE. He was also a Systems Engineering Consultant at Newport Media Inc. (now part of Microchip) Egypt Design Center in Cairo from 2006 - 2014. His research is in the general area of wireless communications with emphasis on performance evaluation of next-generation wireless systems and communications over fading channels. He is the recipient of the University of Mississippi Summer Assistantship Award in 2004 and 2005, The University of Mississippi Dissertation Fellowship Award in 2006, The University of Mississippi Graduate Achievement Award in Electrical Engineering in 2006 and the Best Paper Award presented at the 10th IEEE Symposium on Computers and Communications (ISCC 2005), La Manga del Mar Menor, Spain. He served as a reviewer for several refereed journals and conferences and he is a Member of Sigma Xi and Phi Kappa Phi.