Completion Time Minimization in NOMA Systems: Learning for Combinatorial Optimization

Anyue Wang, *Graduate Student Member, IEEE*, Lei Lei, *Member, IEEE*, Eva Lagunas, *Senior Member, IEEE*, Symeon Chatzinotas, *Senior Member, IEEE*, and Björn Ottersten, *Fellow, IEEE*

***Abstract*—In this letter, we study a completion-time minimization problem by jointly optimizing time slots (TSs) and power allocation for time-critical non-orthogonal multiple access (NOMA) systems. The original problem is non-linear/non-convex with discrete variables, leading to high computational complexity in conventional iterative methods. Towards an efficient solution, we train deep neural networks to perform fast and high-accuracy predictions to tackle the difficult combinatorial parts, i.e., determining the minimum consumed TSs and user-TS allocation. Based on the learning-based predictions, we develop a low-complexity post-process procedure to provide feasible power allocation. The numerical results demonstrate promising improvements of the proposed scheme compared to other baseline schemes in terms of computational efficiency, approximating optimum, and feasibility guarantee.**

***Index Terms*—NOMA, deep learning, resource optimization, mixed-integer exponential conic programming.**

I. Introduction

As an emerging technique, non-orthogonal multiple access (NOMA) has drawn considerable attention in various delay-sensitive and time-critical applications, e.g., NOMA in reducing transmission duration in IoT networks [1], delay minimization in NOMA-based mobile edge computing [2], and completion time minimization in NOMA-based unmanned aerial vehicle systems [3]. In addition, applying deep learning (DL) techniques to NOMA systems has been studied in various aspects. In [4], end-to-end learning for power allocation, i.e., relying on a learning model to output complete optimization results, was studied. In [5], a learning-based algorithm was proposed to solve a linear-programming scheduling problem in deadline-aware NOMA systems. In [6] and [7], deep reinforcement learning (DRL) was applied to address resource allocation problems.

It is worth noting that the existing solutions are mainly based on conventional iterative optimization approaches [1], [2], [3], [8], [9], or adopt straightforward end-to-end learning approaches [4], which might be limited in achieving a good trade-off between complexity and performance. In addition, RL-based approaches are prone to address unconstrained problems or with few constraints due to feasibility issues. DRL requires a Markov process to achieve a satisfactory performance [10]. However, for addressing the practical combinatorial optimization problems, the Markov process might not be satisfied and the problems could be complicated with numerous constraints. This can result in difficulties in reward function design, infeasibility issues, and potential degradation in overall performance.

Compared to the existing works, the novelty of this letter comes from the provided hybrid learning-optimization perspective for solving the NOMA time-minimization problems. Applying DL to address such challenging and complex problems encounters several difficulties, and is, however, studied to a limited extent in the literature. Therefore, in this letter, we intend to fill this gap and answer some key research questions, e.g., how to determine suited learning features; how to apply DL to tackle the high-complexity issue in discrete optimization meanwhile achieving good performance and satisfying all the constraints.

These motivate the letter with the following contributions. Firstly, we reformulate a mixed-integer non-linear programming (MINLP) problem (minimizing number of time slots (TSs) and optimizing user-TS-power allocation) to a mixed-integer exponential conic programming (MIECP) which remains computationally heavy but belongs to a solvable class of MINLP. By doing so, optimal labels in training (supervised) can be obtained through well-established optimization approaches.

Secondly, we propose a hybrid solution combined with DL and post-process optimization (DPO). The major advantage of the proposed DPO lies in its promising trade-off capability compared to other baseline suboptimal, optimal, and learning schemes. The core idea of DPO is to apply DL to tackle the high-complexity part in discrete optimization and rely on DNN predictions and a low-complexity post-process procedure to provide feasible power allocation, such that the overall algorithm can avoid a time-consuming iterative process (against high-complexity suboptimal/optimal algorithms), meanwhile is able to maintain satisfactory performance (against low-complexity suboptimal algorithms and end-to-end learning methods).

II. System Model

We consider a general downlink NOMA system for time-constrained applications, consisting of one base station (BS), and K users sharing one frequency channel with bandwidth B. Let $\mathcal{K} = \{1, \ldots, k, \ldots, K\}$ and $\mathcal{T} = \{1, \ldots, t, \ldots, T\}$ be

Manuscript received December 9, 2020; accepted January 12, 2021. Date of publication January 19, 2021; date of current version February 22, 2021. This work was supported in part by the European Research Council (ERC) Project AGNOSTIC under Grant 742648; in part by the Fonds National de la Recherch CORE Projects ROSETTA under Grant C17/IS/11632107; and in part by the FlexSAT under Grant C19/IS/13696663; and in part by the ASWELL under Grant C19/IS/13718904. The associate editor coordinating the review of this article and approving it for publication was D. Michalopoulos. *(Corresponding author: Lei Lei.)*

The authors are with the Interdisciplinary Center for Security, Reliability and Trust, University of Luxembourg, 1855 Schuttrange, Luxembourg (e-mail: anyue.wang@uni.lu; lei.lei@uni.lu; eva.lagunas@uni.lu; symeon.chatzinotas@uni.lu; bjorn.ottersten@uni.lu).

Digital Object Identifier 10.1109/LNET.2021.3052891

the sets of users and TSs, respectively. All the data requests are delivered by scheduling users up to T TSs. In NOMA, more than one user can access the same TS. The BS superimposes and broadcasts multi-user signals to a group of users. A user receives the superimposed signal and performs successive interference cancellation (SIC) to decode and remove part of co-channel interference. The decoding order is determined based on the descending order of the channel gains [1], [2]. For simplicity, we define that users' indexes are consistent with the decoding order. The signal-to-interference-plus-noise ratio (SINR) of user k at TS t is expressed as:

$$\gamma_{kt} = \frac{g_k p_{kt}}{\sum_{j=1}^{k-1} g_k p_{jt} + \sigma^2}, \tag{1}$$

where g_k is user k's channel coefficient, p_{kt} is transmit power for user k at TS t, and σ^2 is the noise power. The channel condition of each user is assumed to be static over TSs. In (1), user k is unable to decode user j's signal when $j < k$, and user j's signal is thus treated as interference at user k's receiver. The rate of user k in TS t reads,

$$R_{kt} = B \log(1 + \gamma_{kt}). \tag{2}$$

Note that power is treated as optimization variables. Not all K users have to be allocated to each TS. When user j is not scheduled to t, i.e., $p_{jt} = 0$, no interference will be generated from user j to k at the denominator of γ_{kt}.

III. Problem Formulation and Characterizations

In $\mathcal{P}1$, we formulate a resource allocation problem to minimize the number of required TSs for completing a data-transmission task. We introduce binary variables x_{kt} to indicate whether user k occupies TS t ($x_{kt} = 1$), or not ($x_{kt} = 0$). We use binary variables y_t to represent whether the t-th TS is assigned by any user ($y_t = 1$), or not ($y_t = 0$). Vectors $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{p}$ collect all the $x_{kt},\ y_t,\ p_{kt}$ variables, respectively.

$$\mathcal{P}1:\ \min_{\mathbf{x},\mathbf{y},\mathbf{p}} \sum_{t\in\mathcal{T}} y_t \tag{3a}$$
$$\text{s.t.}\ \sum_{t\in\mathcal{T}} R_{kt} \geq D_k, \forall k \in \mathcal{K}, \tag{3b}$$
$$\sum_{k\in\mathcal{K}} p_{kt} \leq P_{\max}, \forall t \in \mathcal{T}, \tag{3c}$$
$$\sum_{k\in\mathcal{K}} x_{kt} \leq L, \forall t \in \mathcal{T}, \tag{3d}$$
$$\sum_{k\in\mathcal{K}} x_{kt} \leq K y_t, \forall t \in \mathcal{T}, \tag{3e}$$
$$p_{kt} \leq P_{\max} x_{kt}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \tag{3f}$$
$$\mathbf{x}, \mathbf{y} \in \{0,1\}, \mathbf{p} \succeq \mathbf{0}. \tag{3g}$$

In (3b), each user's demand D_k must be delivered. In (3c), the total allocated power on each TS cannot exceed the maximum transmit power $P_{\max}$. Constraints (3d) confine the maximum number of scheduled users at each TS up to L. The value of L is typically limited in practice [1], [2], e.g., $L = 2$. Constraints (3e) connect variables $\mathbf{x}$ and $\mathbf{y}$. That is, $\sum_{k\in\mathcal{K}} x_{kt} > 0$ results in $y_t = 1$, and $\sum_{k\in\mathcal{K}} x_{kt} = 0$ leads to $y_t = 0$. Analogously, constraints (3f) confine the dependence between p_{kt} and x_{kt}. Note that, we consider a general NOMA scenario without confining each user to be scheduled to only one TS.

$\mathcal{P}1$ is an MINLP due to binary variables $\mathbf{x}$, $\mathbf{y}$, and non-linear function R_{kt} in (3b). In general, the presence of non-linearity and non-convexity in MINLP could render an unsolvable problem. Thus it is necessary to identify the convexity of function R_{kt}. Based on (2), we can express each power p_{kt} by rate via performing a substituting procedure, then $\sum_{k\in\mathcal{K}} p_{kt}$ in (3c) reads,

$$\sum_{k\in\mathcal{K}} \left(\frac{\sigma^2}{g_k} - \frac{\sigma^2}{g_{k-1}}\right) \exp\left(\frac{\sum_{j\geq k} R_{jt}}{B}\right) - \frac{\sigma^2}{g_K}. \tag{4}$$

Substituting (4) in (3c), $\mathcal{P}1$ can be equivalently reformulated in $\mathcal{P}2$.

$$\mathcal{P}2:\ \min_{\mathbf{x},\mathbf{y},\mathbf{R}} \sum_{t\in\mathcal{T}} y_t \tag{5a}$$
$$\text{s.t.}\ \text{(3b), (3d), (3e),} \tag{5b}$$
$$\sum_{k\in\mathcal{K}} \left(\frac{\sigma^2}{g_k} - \frac{\sigma^2}{g_{k-1}}\right) \exp\left(\frac{\sum_{j\geq k} R_{jt}}{B}\right) - \frac{\sigma^2}{g_K} \leq P_{\max},\ \ \forall t \in \mathcal{T}, \tag{5c}$$
$$R_{kt} \leq R_{max} x_{kt}, \forall k \in \mathcal{K}, t \in \mathcal{T}, \tag{5d}$$
$$\mathbf{x}, \mathbf{y} \in \{0,1\}, \mathbf{R} \succeq \mathbf{0}. \tag{5e}$$

In $\mathcal{P}2$, the optimization variables are $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{R}$, where vector $\mathbf{R}$ collects all the rate variables R_{kt}. Since all the p-variables have been represented by R-variables, correspondingly, constraints (3f) become (5d), where $R_{max} > \max_{k\in\mathcal{K},t\in\mathcal{T}} R_{kt}$. We observe that the constraints (5c) are in fact with the format of non-symmetric exponential cones [11]. Thus $\mathcal{P}2$ is an MIECP.

Optimal or near-optimal solutions of small/medium-scale MIECP problems can be obtained by the state-of-the-art solver Mosek [11]. The procedure follows four steps: presolve, cut generation, starting-point initialization, and iterative search [11]. The first three steps aim to pre-optimize the problem and find a good initial point. In the iterative search, the optimum is bounded and approached by branch-and-bound and outer-approximation methods, where the upper bound is updated by finding a feasible solution, and the lower bound is updated by solving a relaxed problem via the outer approximation approach. However, the inherent difficulties in combinatorial optimization result in exponentially complexity, which might not be affordable in real-time applications.

DL techniques have demonstrated promising performance in classifier-like tasks, e.g., high accuracy in image recognition [10]. However, for the complicated combinatorial optimization problem $\mathcal{P}2$, DL may not be directly applied to output feasible power/rate values since any imperfect prediction can result in poor performance and incurring infeasibility issues. Hence effectively and appropriately addressing $\mathcal{P}2$ via DL techniques is non-trivial, and requires careful design in a tailored way.

IV. The Proposed Solution

In this section, we propose a hybrid solution combined with DNNs and a post-process optimization (DPO) algorithm to tackle $\mathcal{P}2$. A complete solution for $\mathcal{P}2$ consists of

Algorithm 1 DPO

```
Input:
Trained DNNs by ${g, D; y^*}_n$, ${g, D; x^*}_n$ and ${g, D; r^*}_n$, $n = 1, \dots, N$.
Initialize unfinished-user set $\hat{\mathcal{K}} = \mathcal{K}$ and offset $\delta$ (integer).
Output: Power-user allocation
 1: Given a test set to the DNNs and obtain $\hat{y}$, $\hat{\mathbf{x}}$, $\hat{\mathbf{r}}$.
 2: for $t = 1$ to $\hat{y} + \delta$ do
 3:     Sort $\hat{x}_{1t}, \dots, \hat{x}_{Kt} \in \hat{\mathbf{x}}$ by a descending order.
 4:     Select up to $L$ highest-probability users $\{1, \dots, L\} \cap \hat{\mathcal{K}}$
        according to $\hat{x}_{1t}, \dots, \hat{x}_{Kt}$.
 5:     Power allocation $p_{kt} = P_{max} \times \hat{r}_{kt}$, $\forall \hat{r}_{kt} \in \hat{\mathbf{r}}$, $\forall k \in$
        $\{1, \dots, L\} \cap \hat{\mathcal{K}}$
 6:     for $k = 1$ to $L$ do
 7:         Calculate delivered demand $R_{kt}$ by (1) and (2).
 8:         Update residual demand $D_k = D_k - R_{kt}$
 9:         if $D_k \leq 0$ then
10:             $\hat{\mathcal{K}} = \hat{\mathcal{K}} \backslash \{k\}$.
11:     Break if $\hat{\mathcal{K}} = \varnothing$.
```

three terms, i.e., the minimum number of consumed TSs ($y^* = \min \sum_{t \in \mathcal{T}} y_t$), the user-TS allocation on y^* slots, and the corresponding rate/power allocation. The procedure is illustrated in Algorithm 1. As an input to DPO, the first DNN directly learns the mapping between channel-demand parameters and y^*. Note that we adopt a widely-used Softmax function as the activation function in the output layer, which normalizes the outputs to a categorical distribution [10]. For example, the *t*-th element in the output vector, y_t, represents the probability of consuming *t* TSs, which is derived by $y_t = \frac{\exp(\tilde{y}_t)}{\sum_{t' \neq t} \exp(\tilde{y}_{t'})}$, where $\tilde{y}_t$ denotes the output at *t*-th node in the last hidden layer. The predicted $\hat{y}$ is obtained by selecting the highest probability from output nodes, i.e., $\hat{y} = \max_{t \in \mathcal{T}} \{y_t\}$.

For training, we use a tuple $\{\mathbf{g}, \mathbf{D}; \mathbf{y}^*\}_n$, $n = 1, \dots, N$ to represent the *n*-th training set, where the input channel-coefficient vector is $\mathbf{g} = [g_1, \dots, g_K]$ (real values) and demand vector is $\mathbf{D} = [D_1, \dots, D_K]$. The corresponding optimal binary label is organized in $\mathbf{y}^* = [y_1, \dots, y_T]$. The second DNN-based predictions output an estimated user-TS allocation $\hat{\mathbf{x}}$ and power (or rate) splitting ratios $\hat{\mathbf{r}}$, with trained by two sets of tuples $\{\mathbf{g}, \mathbf{D}; \mathbf{x}^*\}_n$ and $\{\mathbf{g}, \mathbf{D}; \mathbf{r}^*\}_n$, respectively. The optimal labels in $\mathbf{x}^*$ are binary, and the labels in vector $\mathbf{r}^*$ are prepared by $r^*_{kt} = \frac{p^*_{kt}}{\sum_{k=1}^{K} p^*_{kt}}$, $\forall r^*_{kt} \in \mathbf{r}^*$.

The predicted $\hat{\mathbf{x}}$ and $\hat{\mathbf{r}}$ may not be feasible for $\mathcal{P}2$ since imperfect predictions may cause constraint violation. However, the predicted $\hat{\mathbf{x}}$ carries the information of which users with high probability to be scheduled on each TS. In addition, the approximated ratios in $\mathbf{r}^*$ give more flexibility in post-processing and performance scaling, compared to directly learning power values. We then rely on $\hat{\mathbf{x}}$, $\hat{\mathbf{r}}$, and $\hat{y}$ as useful guidance and develop a post-processing approach in lines 2 to 11. In line 2, to enable a higher successful rate in delivering feasible solutions, we introduce an integer offset δ as a scaling parameter to provide more chance to find a feasible power solution since with more TS resources, e.g., $\hat{y} + \delta$, all the users' demands can be satisfied more easily than using fewer TSs, e.g., $\hat{y}$. There are two inaccurate cases, $\hat{y} < y^*$ or $\hat{y} > y^*$. For the first case, $\hat{y} < y^*$ directly results in infeasible power allocation, and then δ provides more tolerance for the inaccurate $\hat{y}$ and finding a feasible solution. For the second case, we confine δ to be small, e.g., $\delta = 1$ or 2, in case of large performance degradation. In lines 3 to 4, by sorting $\hat{\mathbf{x}}$, we select up to L highest-probability users on each TS to satisfy constraints (3d), saying users $1, \dots, L$ for simplicity. Based on the user allocation, we assign power in line 5 to meet P_{max} in constraints (3c). We check if all the users' demands are delivered in line 9 to ensure constraints (3b). The algorithm terminates when $\hat{\mathcal{K}} = \varnothing$ or $\hat{y} + \delta$ is reached.

The complexity of a fully-connected DNN mainly comes from the operations of matrix multiplication, vector addition, and activation functions [10]. The number of floating-point operations (FLOPs) of a hidden layer is $(2N_l^{\text{in}} + 1)N_l^{\text{out}}$ [10], where N_l^{in} and N_l^{out} denote the number of input and output nodes at the *l*-th hidden layer, respectively. In line 1, the complexity for 3 DNNs in the testing phase is $\mathcal{O}(3\sum_{l=1}^{N^{\text{h}}}(2N_l^{\text{in}} + 1)N_l^{\text{out}})$, where N^{h} is the number of hidden layer. In the post-process, the complexity is dominated by line 3 and 7. In line 3, sorting *K* elements in $\hat{\mathbf{x}}$ on each TS costs $\mathcal{O}(K\log(K))$ complexity [12]. In line 7, the FLOPs number of SINR calculation for the selected L users is $(1 + L)L$. Thus the complexity of DPO can be expressed as $\mathcal{O}(\max\{3\sum_{l=1}^{N^{\text{h}}}(2N_l^{\text{in}} + 1)N_l^{\text{out}},\ TK\log(K) + TL(L + 1)\})$.

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Carrier frequency	2 GHz
Bandwidth	1 MHz
Cell radius	300m
Path loss	COST-231-HATA
Noise power spectral density	-173 dBm/Hz
P_{max}	33dBm
Demand D_k	3 to 7 bps/Hz [1]
K	4 to 10
L	2
N	6000 (1000 as testing sets)

V. PERFORMANCE EVALUATION

We evaluate the performance of the proposed DPO compared to the following four benchmarks.

- The optimal solution: by the Mosek solver (V.9.1) [11].
- Suboptimal "Best-worst+FTPC" [8]: "Best-worst grouping" principle for user grouping and fractional transmit power control (FTPC) for power allocation.
- Suboptimal "Matching+convex" [9]: Many-to-many matching for user grouping and interior-point method for the remaining convex power-allocation problem.
- End-to-end learning [4]: The scheme directly learns and predicts power allocation **p**.

The parameters settings are summarized in Table I. Detailed simulation settings, DNN hyperparameters, source codes, and training/testing data sets are online available at: *https://github.com/AnyueWang*. DNNs are implemented in TensorFlow, where ReLu, Mean Square Error, and Adam algorithm are adopted as the activation function, loss function, and optimizer, respectively, [10]. To prepare training sets $\{\mathbf{g}, \mathbf{D}; y^*\}_n$, $\{\mathbf{g}, \mathbf{D}; \mathbf{x}^*\}_n$, and $\{\mathbf{g}, \mathbf{D}; \mathbf{r}^*\}_n$, $n = 1, \dots, N$, the inputs **g** and **D** are generated from the adopted emulator based

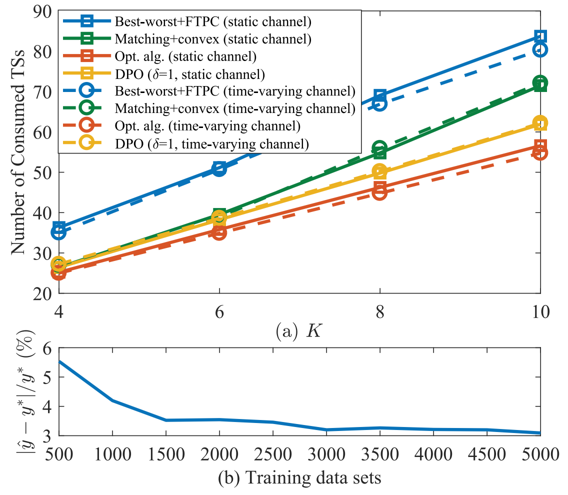

Fig. 1. (a) Minimum TSs with respect to K ($\delta = 1$ in DPO); (b) Predicted $\hat{y}$ in approximating optimal y^* with respect to training set size ($K = 8$).

TABLE II
AVERAGE COMPUTATIONAL TIME IN SIMULATION

Algorithm	$T = 4$	$T = 6$	$T = 8$	$T = 10$
Best-worst+FTPC (ms)	0.1408	0.1581	0.1886	0.1890
Matching+convex (s)	2.5	4.1	5.9	7.9
DPO (ms)	1.1	2.0	2.3	4.3
Optimal algorithm (s)	0.9	11.1	163.7	1130.2

on the parameter settings in Table I. Optimal labels y^*, $\mathbf{x}^*$, and $\mathbf{r}^*$ are obtained by solving $\mathcal{P}2$ via the Mosek solver.

1) DPO in Approximating the Optimum: In Fig. 1(a), we compare the minimum number of consumed TSs, in optimal, suboptimal, and DPO algorithms under both static and time-varying channels. The static model is consistent with the channel assumption in Section II. The time-varying channel is modeled as the correlated Rayleigh fading channel. Firstly, in average, the gaps between optimal and two suboptimal algorithms are 37.6% and 14.7%, respectively, whereas DPO keeps near-optimal with less than 8% gaps. This is because, in Fig. 1(b), when the DNN has been trained with sufficient data sets, e.g., 5000, the predicted $\hat{y}$ well approximates to the optimum, with only 3.07% gaps and $\hat{y} = y^*$ achieved in 92% of 1000 testing sets. Secondly, we observe that the proposed DPO is applicable to both channel conditions. The considerable gain of DPO over the other two suboptimal solutions remains in the time-varying channel, and the optimality gaps between DPO and the optimum keep almost the same magnitude.

2) Trade-Off Performance in DPO: We illustrate the average computational time in Table II. In the optimal algorithm, the computational time increases exponentially as K grows. "Best-worst+FTPC" costs the least time but leads to large performance degradation as shown in Fig. 1(a) and Fig. 2(a). "Matching+convex" consumes seconds, a higher magnitude than "Best-worst+FTPC" and DPO, to converge. DPO maintains at the millisecond level as K scales up, and is able to achieve near-optimal performance.

3) Post-Processing Design in Addressing Feasibility Issues: To enable a fair feasibility comparison, the number of consumed TSs is uniform for all the algorithms in Fig. 2(a). When the TS resources are limited, the "Best-worst+FTPC" and "matching+convex", and end-to-end learning schemes lead

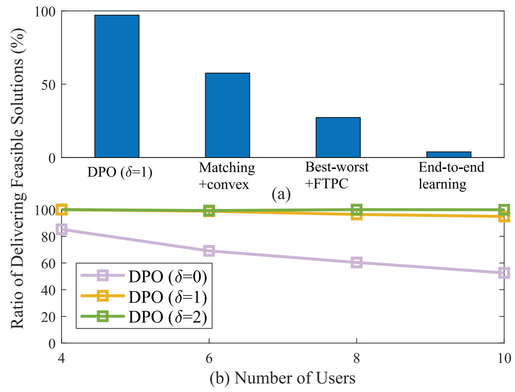

Fig. 2. (a) Ratio of delivering feasible solutions in 1000 testing sets ($K = 8$); (b) DPO in delivering feasible solutions with respect to K and δ.

to feasible solutions in less than 30%, 60%, and 10% instances, respectively. In contrast, with $\delta = 1$, DPO successfully obtains feasible solutions in 98% of 1000 testing sets. As illustrated in Fig. 2(b), this feasibility rate can be further improved to 100% by scaling up δ.

VI. Conclusion

We have addressed a combinatorial optimization problem in NOMA resource allocation. To deal with the high complexity, we have proposed the DPO combining learning and optimization to provide an efficient, feasible, and near-optimal solution. As a future extension, the proposed DPO can be further extended to adapt dynamic environments, e.g., bursty traffic, via, e.g., transfer learning, such that data recollection and retraining in DNNs can become more efficient.

References

[1] D. Zhai, R. Zhang, L. Cai, and F. R. Yu, "Delay minimization for massive Internet of things with non-orthogonal multiple access," *IEEE J. Sel. Topics Signal Process.*, vol. 13, no. 3, pp. 553–566, Jun. 2019.
[2] Z. Ding, D. W. K. Ng, R. Schober, and H. V. Poor, "Delay minimization for NOMA-MEC offloading," *IEEE Signal Process. Lett.*, vol. 25, no. 12, pp. 1875–1879, Dec. 2018.
[3] X. Mu, Y. Liu, L. Guo, and J. Lin, "Non-orthogonal multiple access for air-to-ground communication," *IEEE Trans. Commun.*, vol. 68, no. 5, pp. 2934–2949, May 2020.
[4] J. Luo, J. Tang, D. K. C. So, G. Chen, K. Cumanan, and J. A. Chambers, "A deep learning-based approach to power minimization in multi-carrier NOMA with SWIPT," *IEEE Access*, vol. 7, pp. 17450–17460, 2019.
[5] L. Lei *et al.*, "Learning-assisted optimization for energy-efficient scheduling in deadline-aware NOMA systems," *IEEE Trans. Green Commun. Netw.*, vol. 3, no. 3, pp. 615–627, Sep. 2019.
[6] C. He, Y. Hu, Y. Chen, and B. Zeng, "Joint power allocation and channel assignment for NOMA with deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2200–2210, Oct. 2019.
[7] L. Qian, Y. Wu, F. Jiang, N. Yu, W. Lu, and B. Lin, "NOMA assisted multi-task multi-access mobile edge computing via deep reinforcement learning for industrial Internet of Things," *IEEE Trans. Ind. Informat.*, early access, Jun. 10, 2020, doi: 10.1109/TII.2020.3001355.
[8] M. S. Ali, H. Tabassum, and E. Hossain, "Dynamic user clustering and power allocation for uplink and downlink non-orthogonal multiple access (NOMA) systems," *IEEE Access*, vol. 4, pp. 6325–6343, 2016.
[9] B. Di, L. Song, and Y. Li, "Sub-channel assignment, power allocation, and user scheduling for non-orthogonal multiple access networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 11, pp. 7686–7698, Nov. 2016.
[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT press, 2016.
[11] Mosek, *MOSEK Optimization Toolbox for MATLAB: User's Guide and Reference Manual, Version 4*. Copenhagen, Denmark: Mosek ApS, 2019.
[12] T. H. Cormen *et al.*, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2009.