electronics

MDPI

Article

A Multi-Branch DQN-Based Transponder Resource Allocation Approach for Satellite Communications

Wenyu Sun [1], Weijia Zhang [1], Ning Ma [1] and Min Jia [2,*]

[1] The 54th Research Institute of China Electronics Technology Group Corporation, Shijiazhuang 050081, China
[2] School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin 150080, China
* Correspondence: jiamin@hit.edu.cn

Abstract: In light of the increasing scarcity of frequency spectrum resources for satellite communication systems based on the transparent transponder, fast and efficient satellite resource allocation algorithms have become key to improving the overall resource occupancy. In this paper, we propose a reinforcement learning-based Multi-Branch Deep Q-Network (MBDQN), which introduces TL-Branch and RP-Branch to extract features of satellite resource pool state and task state simultaneously, and Value-Branch to calculate the action-value function. On the one hand, MBDQN improves the average resource occupancy performance (AOP) through the selection of multiple actions, including task selection and resource priority actions. On the other hand, the trained MBDQN is more suitable for online deployment and significantly reduces the runtime overhead due to the fact that MBDQN does not need iteration in the test phase. Experiments on both non-zero waste and zero waste datasets demonstrate that our proposed method achieves superior performance compared to the greedy or heuristic methods on the generated task datasets.

Keywords: deep reinforcement learning (DRL); resource allocation; transparent transponder; satellite communications; Deep Q-Network (DQN)

Citation: Sun, W.; Zhang, W.; Ma, N.; Jia, M. A Multi-Branch DQN-Based Transponder Resource Allocation Approach for Satellite Communications. *Electronics* **2023**, *12*, 916. https://doi.org/10.3390/electronics12040916

Received: 2 December 2022
Revised: 15 December 2022
Accepted: 18 December 2022
Published: 11 February 2023

1. Introduction

Resource leasing is widely adopted in satellite communication systems based on transparent transponders, especially for frequency resources [1]. As shown in Figure 1, the transponder resource is shared among multiple satellite-terrestrial integrated networks. In general, the organizer of the satellite communication network and the lessor of the resource calculate the network frequency demand in advance and submit it together with the lease time of the frequency to the operation control center of the communication satellite who is the owner of the frequency. The operation control center maintains a table of allocated resources for each transponder of the satellite, and the lease time of each resource segment in the transponder is recorded in the table. Subsequently, the operators will find a free resource segment that satisfies the lessor's needs in the table, update the corresponding lease time period, and inform the lessor of the allocation result. Finally, the leaser uses the allocated satellite transponder resources to establish the satellite communication network.

The above process of transparent satellite transponder resource allocation mainly relies on manual labor, and is therefore only feasible when the maintenance leasing tasks are not laborious. However, with the development of the maintenance process, the intensity of the leasing business continues to increase, and the fragmentation of satellite transponder frequency caused by the leasing process is becoming increasingly serious, posing considerable challenges to the maintenance of resource leasing and the search for idle resource segments. These challenges have increased the task complexity and technical requirements for operators as well as the operating cost of the satellite system [2]. In addition, manual search struggles to guarantee optimal utilization of resources due to its empirical nature. Moreover, when the requirements of high-priority tasks cannot be

satisfied immediately, and the existing resources of low-priority tasks need to be preempted, how to select preemption objects to ensure the least impact is also a problem to be considered by the operators. The problem of resource allocation under multiple constraints has been proven to be NP-hard [3]. In order to solve the above problem, some traditional operations research algorithms [3–5], greedy algorithms [6] and heuristic algorithms [7–11] have been applied in the automation of transparent satellite transponders and intelligent resource allocation.

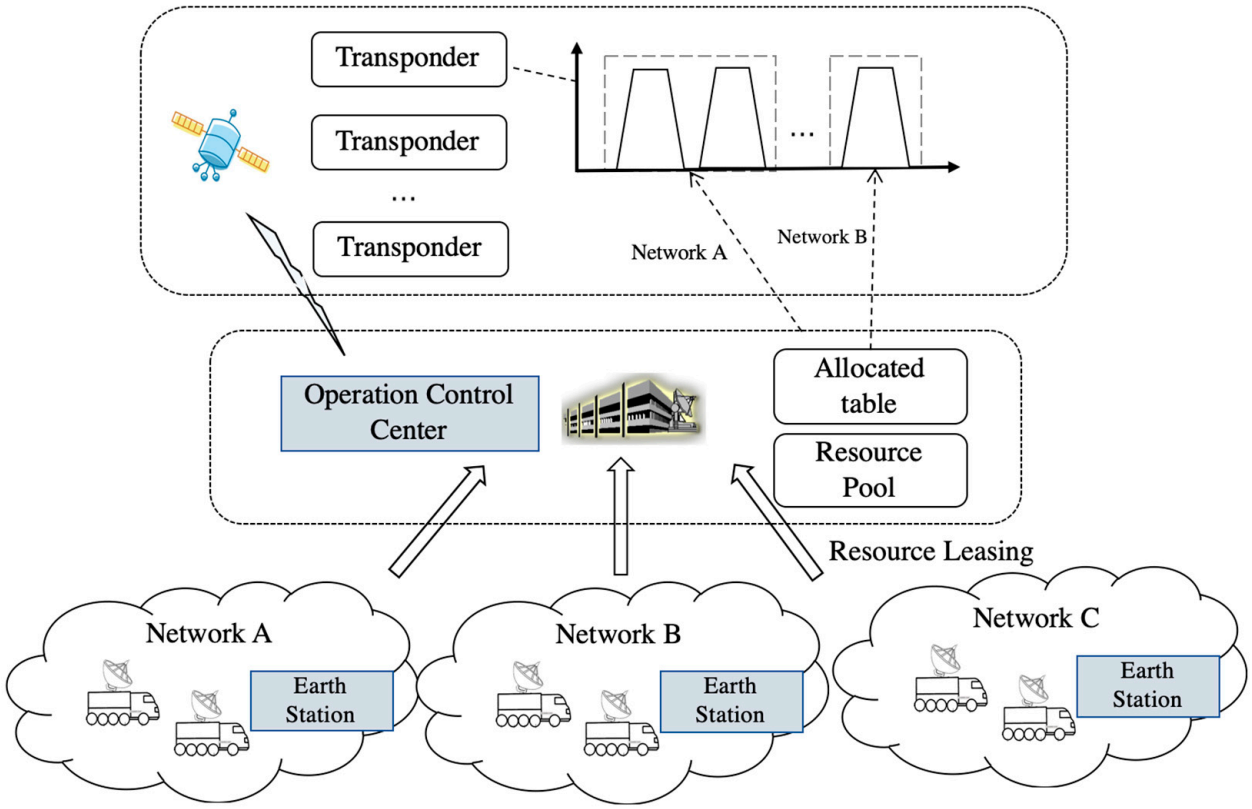

Figure 1. The process of transparent satellite transponder resource allocation.

In actual resource allocation, however, there exists a plethora of interdependent nodes, which result in a set of variables and constraints to be solved in operations research models and heuristic algorithms. Therefore, the high computational complexity renders it inapplicable to problems with strict timeliness requirements, especially in satellite resource allocation. With the development of the satellite communication field and the research of reinforcement learning, more and more empirical data are stored, and deep reinforcement learning (DRL) can exploit these data due to its characteristics to discover regular features and learn the policies. Hence, resource allocation methods based on reinforcement learning have started to receive increasing attention [12].

In this paper, we propose the Multi-Branch Deep Q-Network (MBDQN) for satellite communication resource allocation. By introducing TL-Branch and RP-Branch, this DRL-based model can simultaneously extract features for the satellite resource pool (RP) state and task list (TL) state. The feature embeddings are concatenated and fed into Value-Branch to calculate the action-value function. The task selection action and the priority action are calculated according to the optimal action-value function. Through well-planned and rapid allocation of leasing tasks, as shown in Figure 2, the satellite resource allocation method can improve the utilization rate of satellite transponder resources (i.e., frequency and its occupation time) under the condition of multiple constraints.

The MBDQN model possesses several excellent features. The model is highly efficient in terms of inference and decision-making. Despite its time-consuming training process,

MBDQN can have the training done offline, and the actual deployment and inference can be performed using the trained model, thus achieving near real-time decision-making.

The main contributions of this paper are summarized as follows:

(1) A reinforcement learning solution to the satellite resource allocation problem is proposed and established, where the satellite operation control system and the task requirements are defined as the agent and environment of DRL, respectively.

(2) Multi-Branch Deep Q-Network (MBDQN) is designed for resource allocation, which employs TL-Branch and RP-Branch to simultaneously extract features of the satellite resource pool state and task list state. Our proposed MBDQN achieves state-of-the-art performance on the generated task datasets.

(3) Two actions, namely frequency priority and time priority, are introduced into the action space, which further improves the average resource utilization rate of the system compared with when only time priority is involved. In contrast to traditional heuristic algorithms, our method is more suitable for online deployment and significantly reduces the time consumption during inference.

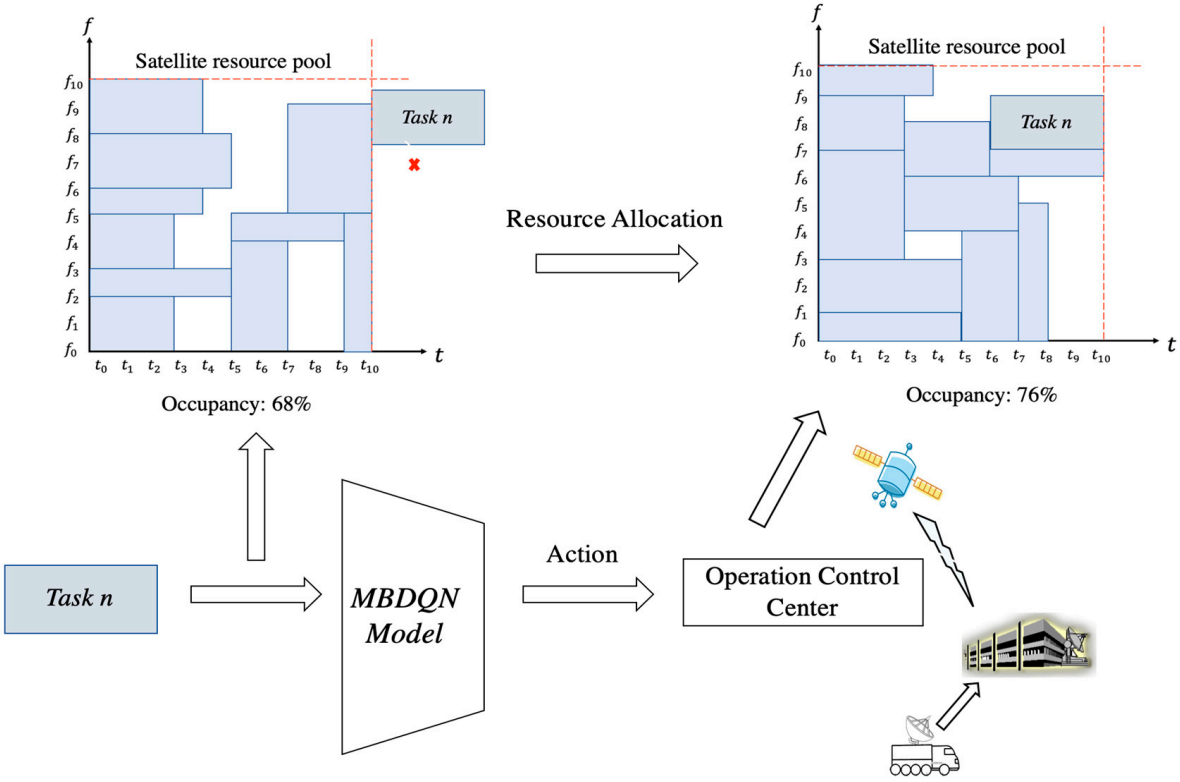

Figure 2. MBDQN is used to conduct efficient satellite resource allocation and improve resource occupancy.

2. Related Work

Among traditional operations research methods, combinatorial optimization, linear programming, and non-convex optimization are widely adopted for resource allocation optimization problems [3–5]. With the growing complexity of actual resource allocation problems, the much-studied heuristic algorithms in combinatorial optimization problems have seen increasing use to address resource allocation problems. In [8], a downlink resource allocation model adapting to observation task increments is established, and a novel algorithm based on evolutionary computation is proposed. The simple ant colony optimization algorithm (SACO) [9] and Tabu search (TS) [10] have also been implemented to solve the satellite resource allocation problem. [11] proposes an implementation of the heuristic algorithm based on the genetic algorithm and particle swarm optimization (GA-PSO) to solve the joint power and frequency allocation problem.

At present, with the advancement of deep learning and reinforcement learning, deep reinforcement learning has seen growing applications in the field of combinatorial optimization, especially for resource allocation. According to the learning content of the agent in the process of interacting with the environment, model-free reinforcement learning algorithms can be divided into two categories: policy optimization algorithms (e.g., REINFORCE) that directly learn the action execution strategy, and value optimization algorithms (e.g., Q-learning) that learn a value function for action execution decisions. In addition to REINFORCE, policy optimization algorithms also include trust region policy optimization (TRPO) [13], proximal policy optimization (PPO) [14], advanced actor-critic (A2C) [15], asynchronous advantage actor-critic (A3C) [15], etc. Moreover, Deep Q-Network (DQN) [16] is a typical value optimization algorithm.

Furthermore, reinforcement learning algorithms designed for the allocation problem can search for a locally optimal solution in a shorter time than heuristic algorithms. Ref. [17] proposes a reinforcement learning heuristic optimization (RLHO) framework. It guides the PPO algorithm to obtain a better initial state with the reinforcement learning algorithm for solving the bin packing problem. In [18], an innovative resource management framework that applies DRL is proposed for the next-generation heterogeneous satellite networks. In [19], an improved graph-based minimum clique partition algorithm is proposed to revise the observation satellite scheduling problem for preprocessing in the task clustering phase by considering the maximum task priority and the minimum observation slewing angle under constraint conditions. Ref. [12] proposes a deep reinforcement learning-based framework (DRLF) to solve the problem of unknown dynamics and prohibitive computation for dynamic resource allocation.

3. Proposed Method

This section describes our proposed Multi-Branch DQN (See Figure 3 for an overview). We first define and formulate the resource allocation problem for satellite communications based on our model. Next, we discuss how to model the main elements of reinforcement learning defined in MBDQN. Afterward, the network structure of MBDQN is delineated in detail. Table 1 summarizes the notations used in this paper.

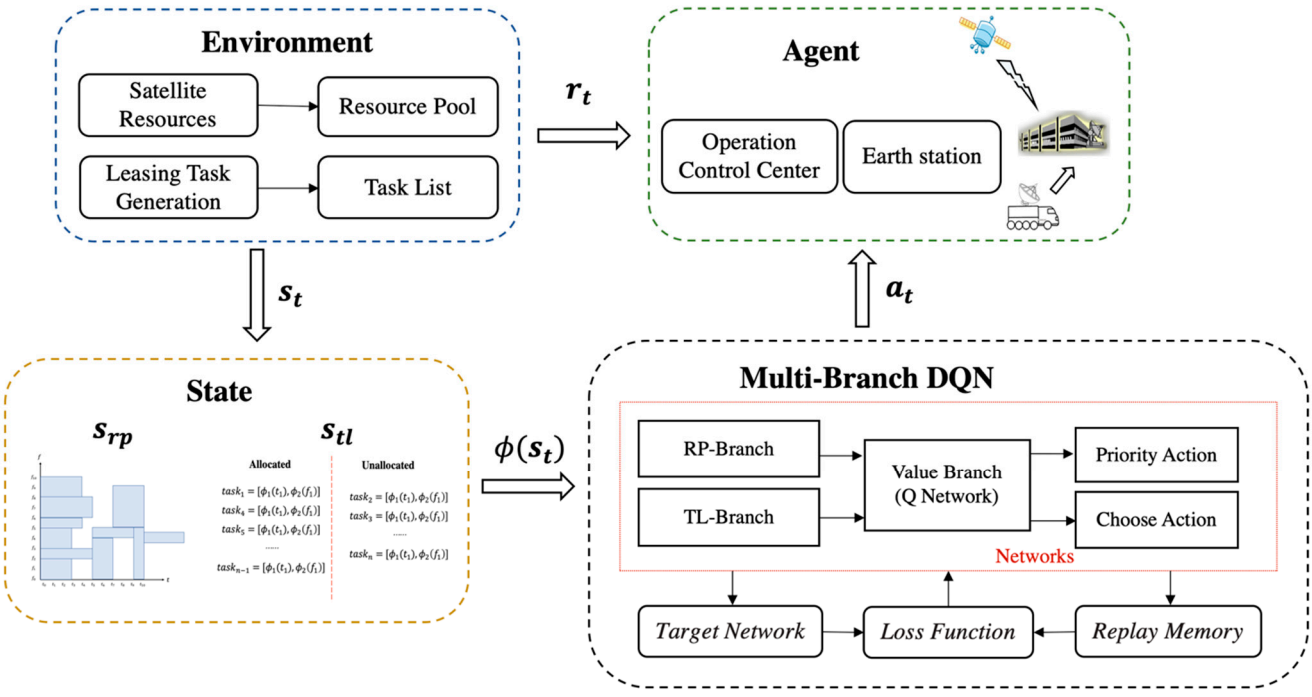

Figure 3. Overview of the proposed Multi-Branch DQN-based resource allocation method.

Table 1. Notation definitions.

Symbol	Definition
N	Number of patches
M	Number of task lists
s_t	State of environment at time t
s_{rp}	State of resource pool
s_{tl}	State of task lists
a_t	Action selected at time t
a_p	Priority action selected
a_c	Choose action selected
$A(s)$	Action space of MBDQN
Δt	Effective time resource range
Δf	Effective frequency resource range
$p_{(i,j)}$	Patch (i, j) of resource pool
r_t	Reward at time t
ω	Parameters of MBDQN
ω^-	Parameters of target network
$Q(.)$	Action-value function of MBDQN
$Q'(.)$	Action-value function of target network
$\phi_1(t)$	State reformulation on time
$\phi_2(f)$	State reformulation on frequency
γ	Discount factor reward
y_t	TD target at time t
L	Loss function of the model
f_{rp}	Feature embedding of RP-Branch
f_{tl}	Feature embedding of TL-Branch
o_m	Status of the m-th task

3.1. Problem Formulation

A multi-branch DQN based resource allocation approach for satellite communications can be defined as follows. Given the satellite communication system conditions, the satellite resource and task demand generation of the satellite communication system is defined as the environment of reinforcement learning. In this environment, there is a state s_t, including the resource pool state s_{rp} and task list state s_{tl}, as well as a series of valid action sets $A(s)$ for decision-making. The agent controlled by MBDQN will select an effective action $a_t \in A(s)$ in the action set $A(s)$, and execute the action a_t in the environment to obtain reward r_t at time t, after which the state of the environment is transferred to s_{t+1}. In this method, an iteration will be executed until the environment reaches the termination condition, that is, when all allocations are completed and the resource pool has no further resources to allocate.

The MBDQN model essentially finds a better sort sequence than those by heuristic algorithms. Its ultimate goal is to minimize the resource occupation rate of the allocation result and shorten the total satellite task allocation time. An off-policy training method based on the DQN model [16], MBDQN is a typical value-optimized reinforcement learning algorithm.

The objective of DQN is to learn an action-value function $Q(s, a; \omega)$, where ω represents all parameters of the current action-value function, and $Q(s, a; \omega)$ the estimated value of action a under the parameters ω with state s. It can also be understood as the expected sum of all reward values to be obtained from the environment, still based on the parameters ω interacting with the environment. Finally, in accordance with the principle of maximizing the value of the action-value function, the action selected by the DQN algorithm is $a(s) = \arg\max_a Q(s, a; \omega)$. When the agent interacts with the environment and collects a certain amount of empirical data, the action-value function can be updated according to the principle of minimizing the estimation TD error.

3.2. Modeling of MBDQN

The primary goal of MBDQN is to model the satellite operation control program as the agent and model the task requirements and satellite communication resources as the environment. Furthermore, the main elements of reinforcement learning defined in the MBDQN model for the satellite resource allocation problem are discussed, including state space, action space, and reward. Meanwhile, the definitions of replay memory, target network, and loss function in the model training stage are also introduced in detail.

(1) State space

The state $s_t = \left(s_{rp}, s_{tl}\right)_t$ of the task requirements and satellite communication resources at time t is the observation based on both the state of resource pool s_{rp} and that of task list s_{tl}. Whenever a new resource lease requirement is generated, we format the task and update the state of task list s_{tl}, which can be expressed as

$$s_{tl} = \{\ [o_1, \phi_1(t_1), \phi_2(f_1)], \ldots, [o_m, \phi_1(t_m), \phi_2(f_m)]\ \} \tag{1}$$

where o_m represents the allocation status of the m-th task in the resource pool; t_m and f_m denote the satellite time and frequency resources occupied by the task, respectively. $\phi_1(t_m) = N * t_m/\Delta t$ and $\phi_2(f_m) = N * f_m/\Delta f$ are the state reformulation on t_m and f_m to make them conform to the tensor size of the model input. Both t_m and f_m are reformulated in range $[1, N]$. Δf and Δt represent the practical time resource and the frequency resource range in the satellite resource pool, respectively.

By dividing frequency resources and time resources N-1 times in each dimension, the resource pool can be divided into $N * N$ patch. The state matrix s_{rp} is used to represent the occupancy of each resource block in the resource pool. The representation of s_{rp} is as follows:

$$s_{rp} = \begin{bmatrix} p_{(1,1)} & \cdots & p_{(1,n)} \\ \vdots & \ddots & \vdots \\ p_{(n,1)} & \cdots & p_{(n,n)} \end{bmatrix}_{N*N} \tag{2}$$

where $p_{(n,n)}$ is the occupancy indication of the resource pool in the (n, n) patch.

(2) Action space

In the satellite resource allocation problem, the available action space $A(s)$ is the decision space of the satellite control system, and the action a_t is selected from $A(s)$ according to the current state s_t at time t. The available action space depends on the type of allocated resources and the limitation of resources. It includes the priority action space A_p and the chosen action space A_c. Formally, $A(s)$ can be defined as:

$$A(s) = \left\{ \left(A_c(i),\ A_p(j)\right) \middle| 1 \le i \le m,\ j = 0, 1 \right\} \tag{3}$$

where $a_c = A_c(i)$ is the action of task selection, which indicates the selected task number in the list in this turn; $a_p = A_p(j)$ is the action of resource search priority, which represents the priority of frequency search or time search during the allocation.

The exploration mechanism of action selection directly affects the sampling performance. As such, in the training phase, action selection adopts a ϵ-greedy strategy. MBDQN samples from $A(s)$ randomly with probability ϵ, which can be expressed as

$$a_t = \left(a_c, a_p\right)_t = \begin{cases} \text{sample from } A(s), & \text{with probability } \epsilon \\ \underset{a}{argmax} Q(s_t, a_t; \omega), & \text{others} \end{cases} \tag{4}$$

In the testing phase, MBDQN will select the a_t with the highest probability in the probability distribution $Q^*(s_t, a_t)$.

(3) Reward definition

In this paper, the resource occupancy rate is used as the optimization objective of the algorithm, that is, the reward value r_t of the network that represents the evaluation of the selected action a_t in state s_t. After the task is allocated, the higher the resource occupancy

rate, the higher the reward value. For task allocation with the same patch size, the closer the resource occupancy rate is to the upper limit, the larger the reward value should be. Based on the above principles, the reward is designed as Equation (5):

$$r = -\log\left(1 - \frac{\sum_{m=1}^{M} o_m * \phi_1(t_m) * \phi_2(f_m)}{\Delta t * \Delta f} + \varepsilon\right) \tag{5}$$

where the non-negative multiplier ε is used to avoid infinite values.

(4) Replay memory

For a satellite task allocation system, there exists a salient correlation between the order of task allocation and the final resource occupancy ratio. However, deep neural networks require that the input samples are uncorrelated. Hence, in the training phase, we adopt the replay memory trick to eliminate the correlation of the sample data generated by the interaction with the environment in the satellite task allocation system. Moreover, replay memory ensures that a sample can participate in training multiple times, thus improving learning efficiency. The transition $e_t = \langle s_t, a_t, s_{t+1}, r_t \rangle$ generated in each training step is saved in a replay memory pool.

(5) Target network

The outcome of the allocated action in a satellite task allocation problem is usually a positive reward, even though the action is not necessarily optimal, and there will be an overestimation problem similar to DQN. Meanwhile, every time the binary $\langle s_t, a_t \rangle$ is used to update MBDQN parameters ω, MBDQN tends to overestimate the Q value of $\langle s_t, a_t \rangle$. Therefore, in the training phase, we use the target network to calculate the TD target, and SGD only updates the parameters of the DQN without updating those of the target network. TD target is calculated as

$$y_t = r_t + \gamma * \underset{a}{max} Q'\left(s_{t+1}, a; \omega^-\right) \tag{6}$$

where γ is the discount factor reward. The parameter of the target network ω^- is updated at regular intervals, with the target update interval empirically set to 100 in MBDQN. Parameters of the target network are updated by calculating the weighted average of ω and ω^- before assigning the value to ω^-.

(6) Loss function

Considering the self-consistency of the Q function, similar to DQN, we update the action-value function as per the principle of minimizing the estimation error of $Q(s_t, a_t; \omega)$ and y_t. A minibatch of data $\langle s_t, a_t, s_{t+1}, r_t \rangle$ is obtained by model inference in an iteration, that is, after obtaining the reward value r_t between states s_t and s_{t+1}. The loss function $L(\omega)$ is calculated by the following equation

$$L(\omega) = E\left[\left(y_t - Q(s_t, a_t; \omega)\right)^2\right] \tag{7}$$

where y_t is calculated by Equation (6).

MBDQN's overall operations are outlined in Table 2. During the training phase, the MBDQN model is initialized by Kaiming initialization [20], and parameters of the trained model are exported and saved at the end of a full episode. During testing, these saved parameters are loaded to initialize the MBDQN model, and the action-value function $Q(s_t, a_t; \omega)$ is calculated according to the current input state. The action $a_t = \underset{a}{argmax} Q(s_t, a_t; \omega)$ is selected in a deterministic way, and this loop is executed M times.

3.3. Structure of MBDQN

This section describes the network structure of the proposed MBDQN, including its RP-Branch, TL-Branch, and Value-Branch, as well as the detailed structure (as shown in Figure 4).

Table 2. Overall operations of the MBDQN model.

1.	**Training Process:**
2.	**Initialize** params of MBDQN and target network $\omega = \omega^-$
3.	**Initialize** hyperparams for training
4.	**For** step **in** episode **do:**
5.	Observe states s_{rp} and s_{tl} from environment at time t
6.	Format s_{rp} and s_{tl}, and obtain state s_t
7.	Evaluate MBDQN and obtain $Q(s_t, a_t; \omega)$
8.	Select action a_t by Equation (4)
9.	Execute action a_t and observe state s_{t+1}
10.	Calculate the reward r_t by Equation (5)
11.	Save transition $\langle s_t, a_t, s_{t+1}, r_t \rangle$ in replay memory
12.	Sample a minibatch of $\langle s_t, a_t, s_{t+1}, r_t \rangle$ from replay memory
13.	Calculate TD target y_t by Equation (6)
14.	Calculate loss function $L(\omega)$ by Equation (7)
15.	Update params ω of MBDQN through SGD optimizer
16.	For every T step, update params ω^- of target network
17.	**End For**
18.	**Save** and **export** params ω
	After training
1.	**Testing Process:**
2.	**Load** and **initialize** the trained params ω of MBDQN
3.	**For** m **in** number M **do:**
4.	Observe states s_{rp} and s_{tl} from environment at time t
5.	Format s_{rp} and s_{tl}, and obtain state s_t
6.	Evaluate MBDQN and obtain $Q(s_t, a_t; \omega)$
7.	Select action $a_t = \underset{a}{argmax} Q(s_t, a_t; \omega)$
8.	Execute action a_t and observe state s_{t+1}
9.	**End For**

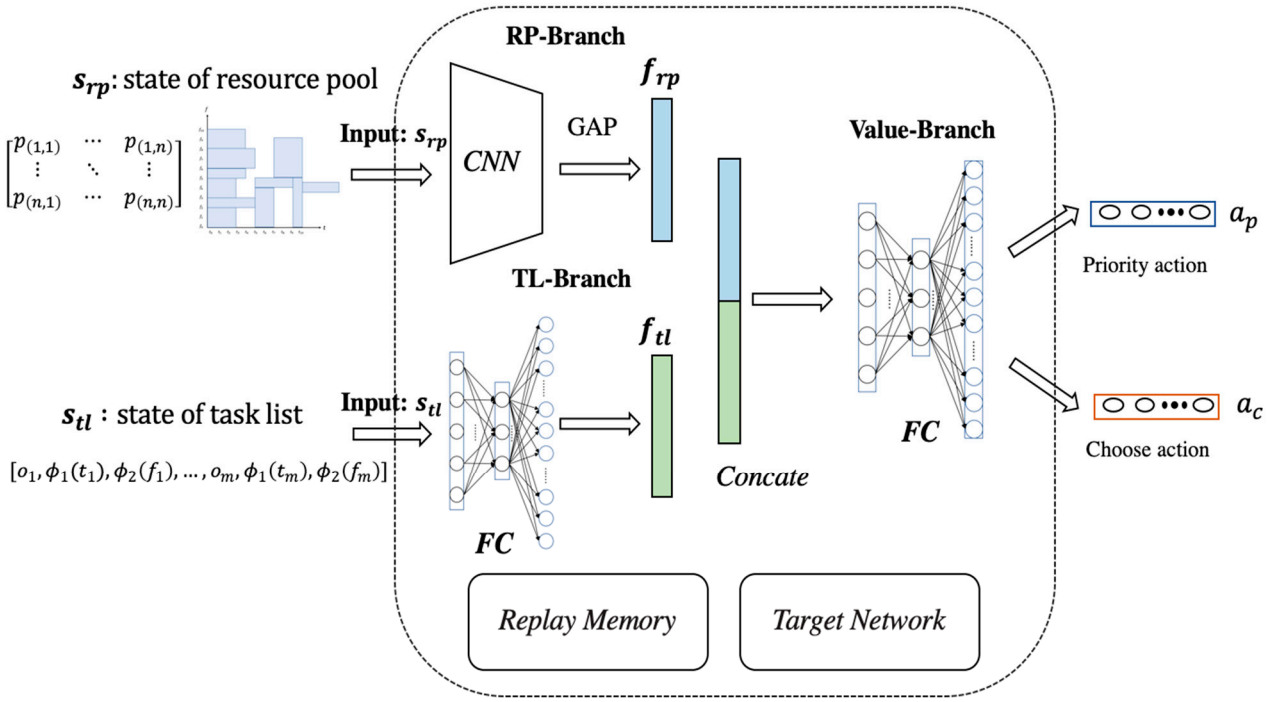

Figure 4. The structure of the proposed MBDQN resource allocation model.

RP-Branch is a convolutional neural network (CNN) composed of convolutional modules, each consisting of convolutional layers, batch normalization (BN) layers [20], and ReLU activation. The resource pool state matrix s_{rp} calculated from Equation (2) has

an input size of $N * N$ that resembles a 2D image. Therefore, CNNs are more conducive to extracting discriminative features f_{rp} of the satellite resource pool state. The role of RP-Branch is to extract features from the input s_{rp} and leverage them as the prerequisite for deriving the action-value function. TL-Branch is a fully connected (FC) neural network. The satellite task list input sequence s_{tl} with an input size of $3 * M$ is calculated by Equation (1). Input data s_{tl} are serialized for FC layers to extract features from them. The objective of TL-Branch is to extract the features from the input s_{tl}, which are used to estimate the action-value function together with f_{rp}. RP-Branch and TL-Branch are in parallel, and their outputs are concatenated via a Concat layer. Then, f_{rp} and f_{tl} are concatenated to yield the global feature f for action-value function estimation, which is subsequently used as the input of Value-Branch to calculate $Q(s_t, a_t; \omega)$. Similar to TL-Branch, Value-Branch is also a neural network based on FC layers.

TL-Branch, RP-Branch, and Value-Branch jointly constitute MBDQN's overall structure, as depicted in Table 3. Compared with general reinforcement learning models, MBDQN combines the characteristics of multiple satellite resource types, is more suitable for different types of state inputs, and is more attentive to the resource state of the satellite communication system. In particular, a deeper and more complex network could be leveraged as the backbone of the MBDQN model, depending on the complexity of the satellite allocation problem to be solved (e.g., number of tasks, size of the resource pool block division). Nevertheless, the choice of backbone structure is beyond the scope of this paper. For simplicity, MBDQN only employs a relatively shallow network for investigation.

Table 3. Detailed network of Multi-Branch DQN.

Block	Module	Module Structure	Tensor Size
RP-Branch	Conv1	*Conv(K = 3)-BN-ReLU-MaxPool*	*(1,N,N)→ (32,N/2,N/2)*
	Conv2	*Conv(K = 1,3,1)-BN-ReLU,*	*(32,N/2,N/2)→ (64,N/4,N/4)*
	Conv3	*Conv(K = 1,3,1)-BN-ReLU,*	*(64,N/4,N/4)→ (128,N/4,N/4)*
	GAP	*Global Average Pooling*	*(128,N/4,N/4) → (128,1)*
TL-Branch	FC1	*FC(128)-BN-Dropout(0.5)-FC(256)*	*(3 * M,1)→ (256,1)*
	FC2	*FC(256)-BN-Dropout(0.5)-FC(128)*	*(256,1)→ (128,1)*
Value-Branch	Concat	*Concat*(GAP, *FC2)*	*[(128,1),(128,1)]→ (256,1)*
	FC3	*FC(256)-BN-Dropout(0.5)-FC(128)*	*(256,1)→ (128,1)*
	FC4	*FC(128)-BN-Dropout(0.5)-FC(128)*	*(256,1)→ (128,1)*
	Q-value	*[FC(2),FC(M)]*	*(128,1)→ [(2,1),(M,1)]*

The detailed network structure of the MBDQN model is presented in Table 3. The height and width of the input to the RP-Branch are both N. In addition, the trick of expanding the feature map is applied at the third convolutional layer (Conv3 in Table 3). Batch normalization [21] and dropout are also employed in MBDQN to alleviate internal covariate shift and overfitting problems. Once training is completed, the de-BN operation is performed to improve the inference speed of the model.

4. Experiments

In this section, the dataset generation and the detailed algorithm implementation of the experiments in this paper are illustrated in detail. The comparative experimental results and experimental analyses are also discussed.

4.1. Datasets and Implementation

4.1.1. Generation of Datasets

Due to the absence of standard satellite resource allocation datasets, the reinforcement learning model lacks a sufficient amount of data for training. In view of this, this paper adopts two generation methods to randomly generate the satellite task list dataset. The first method is the zero-waste satellite task list dataset (ZW-Dataset) generation method, which is more conducive to comparing the performance between different allocation methods. The second is the non-zero waste dataset (NZW-Dataset) generation method, which is more consistent with the actual condition of a satellite task list.

The ZW-Dataset generation method refers to the binned data generation algorithm proposed by [22]. Its core idea is to continuously divide a large task that fills the satellite resource pool, and randomly select subtasks with an area of not less than $2m/\gamma$ for cutting each time, where m is the maximum area of the task in the task sequence and γ is the aspect ratio parameter. Finally, it is divided into a list of several tasks that satisfy a certain area ratio γ and an aspect ratio ρ (both set to 3 in this paper). The optimal allocation result of the tasks can be known on the ZW-Dataset due to the zero-waste setting. The generation method of ZW-Dataset is demonstrated in Table 4.

Table 4. Zero-waste satellite task list dataset generation.

1.	**Input**:
2.	Frequency range Δf and time range Δt
3.	Number of tasks N
4.	Area ratio γ and aspect ratio ρ
5.	**Initialize** set $task\_list = [[\Delta f, \Delta t]]$
6.	**For** n **in** $N-1$ **do:**
7.	Calculate max product $m = f_m * t_m$ in $task\_list$
8.	Randomly select tl in $task\_list$, satisfying $f_{tl} * t_{tl} > 2m/\gamma$
9.	Delete tl from $task\_list$
10.	Randomly select slicing dimension, satisfying $1/\rho < f_{tl}/t_{tl} < \rho$
11.	Slice $tl = [tl_1, tl_2]$ to subtasks
12.	Add tl_1, tl_2 in $task\_list$
13.	**End For**
14.	**Return** $task\_list$

By setting different parameters in the NZW-Dataset generation method, we can control the number of satellite tasks in the dataset, as well as the resource occupation in the time and frequency ranges. In this paper, the frequency and time in the resource pool are divided into ten segments and each task occupies at least one of them. The time range for the generated dataset is set to $t_i \in [\Delta t/N,\ \Delta t/2]$ and the time range to $f_i \in [\Delta f/N,\ \Delta f/2]$.

4.1.2. Implementation Details

In all experiments, the batch size was fixed at 32, using the SGD optimizer with momentum of 0.9 and weight decay of 5×10^{-4} for different episodes. We used BN and ReLU in all hidden layers and linear activation in the output layers, and initialized parameters by Kaiming initialization [20]. To eliminate random factors, the random seed is manually fixed when training the model to ensure that the initialized parameters of the model are consistent across different runs so as to avoid compromised validity and rationality of the comparative experiments. We use warmup [23] to update the learning rates, and the initial learning rates of TL-Branch, Value-Branch, and RL-Branch are set to 0.001, 0.001, and 0.01, respectively. The buffer size of replay memory is set to 1×10^5 and

the hyperparameters of replay memory are set as follows: $\alpha = 0.3$, $\beta = 0.5$, $\eta = 5 \times 10^{-6}$, and $\varepsilon = 1 \times 10^{-7}$, respectively. Gradient clipping is set to 20 and the discount factor reward γ to 0.95. The target network solves the overestimation problem in the training phase with a target update interval of 100. The nonnegative multiplier ε is empirically set to 0.1.

4.2. Results and Analysis

Based on the zero-waste and non-zero-waste datasets generated in 4.1.1, we conduct comparative experiments on the resource allocation performance and running time performance, which proves the effectiveness of the proposed MBDQN model. Specifically, the comparative experiments of this paper are shown in Table 5, and the performance of MBDQN and the traditional greedy and heuristic algorithms are mainly compared. In addition, the experimental results and how the reinforcement learning-based MBDQN model affects the algorithm for satellite resource allocation are analyzed.

Table 5. Comparisons of the proposed MBQDN with existing methods [6,7,11] on the zero-waste dataset.

Task Number	Bottom Left [6]		Bee Colony [7]		GA-PSO [11]		MBDQN (Ours)	
	AOP (%)	RO (s)	AOP (%)	RO (s)	AOP (%)	RO (s)	AOP (%)	RO (s)
$M = 10$	85.2	0.13	97.5	4.16	97.9	10.42	97.7	0.59
$M = 20$	86.9	0.22	93.5	9.32	95.1	18.96	95.3	0.83
$M = 30$	89.1	0.30	93.0	13.05	93.7	24.50	94.6	1.21
$M = 40$	91.3	0.37	92.6	19.41	93.9	32.54	95.1	1.47

As shown in Table 5, compared with the greedy algorithm and the heuristic algorithm, MBDQN gives rise to a significant improvement in the average resource occupancy performance (AOP) and runtime overhead (RO) when tested on the zero-waste dataset. MBDQN can substantially improve the AOP compared with the greedy algorithm (Bottom Left [5]) within the time limit. In addition, the performance improvement of MBDQN is more obvious with the increase of the satellite task number. For instance, when M is equal to 40, the performance of MBDQN is obviously higher than that of GA-PSO [10] (93.9% vs. 95.1%, AOP). This is due to MBDQN's ability to learn from prior experience, and different branches can accurately learn and extract multi-state features, rather than heuristic exploration for each prediction. In particular, in terms of time complexity, MBDQN significantly reduces the runtime overhead of a complete allocation (32.54 s vs. 1.47 s, $M = 40$). The reason is that RL-based methods get rid of the iterative process time compared with heuristic algorithms such as Bee Colony [6] and GA-PSO. MBDQN's training, albeit long, can be performed offline, and only the trained model is used for inference in practice.

To assess the performance of MBDQN in a more extensive and realistic setting, we also conduct experiments on the NZW-Dataset generation method. The parameters for the training and testing of each model are the same as the previous settings. The experimental results (presented in Table 6) are on par with those on ZW-Dataset. Due to the non-zero waste, the optimal allocation result on this dataset cannot be known, but AOP can still be used to compare the performance of each method. Compared with Bee Colony and GA-PSO, MBDQN also achieves the best performance in terms of both AOP (87.9 vs. 88.4 vs. 89.6, $M = 40$) and runtime overhead (21.39 vs. 31.81 s vs. 1.52 s, $M = 40$).

From Tables 5 and 6, it can be concluded that: (a) compared with the greedy (i.e., Bottom Left) and heuristic (i.e., Bee Colony, GA-PSO) algorithms, the proposed MBDQN has resulted in boosted AOP on both non-zero waste and zero waste datasets, and the improvement becomes more obvious as the number of tasks increases. (b) The MBDQN model has the advantage of shorter allocation time compared to heuristic algorithms such as Bee Colony and GA-PSO, which is due to the fact that MBDQN does not need iteration in the test phase. The network itself has a relatively small number of layers and the removal of BN further improves its inference speed. These results indicate that the trained multi-branch-based DQN model can effectively solve the problem of satellite resource allocation,

and TL-Branch and RP-Branch can accurately extract discriminative features of resource pool state and task state. In addition, the results also highlight an effective application of reinforcement learning to the satellite resource allocation problem.

Table 6. Comparisons of the proposed MBQDN with existing methods [6,7,11] on the non-zero waste dataset.

Task Number	Bottom Left [6]		Bee Colony [7]		GA-PSO [11]		MBDQN (Ours)	
	AOP (%)	RO (s)	AOP (%)	RO (s)	AOP (%)	RO (s)	AOP (%)	RO (s)
$M = 10$	76.3	0.12	85.1	4.52	85.2	11.44	85.5	0.63
$M = 20$	80.5	0.22	85.8	10.65	85.8	18.90	86.4	0.86
$M = 30$	82.4	0.31	86.2	14.20	87.0	24.43	87.7	1.24
$M = 40$	83.8	0.36	87.9	21.39	88.4	31.81	89.6	1.52

5. Conclusions

In this paper, MBDQN is proposed as the first reinforcement learning solution for the satellite resource allocation problem based on the transparent transponder. By introducing TL-Branch and RP-Branch into the network, the discriminative features of the resource pool and the task list are accurately extracted. Based on the structure of DQN, a multi-type action-value function, including priority and task index number, is selected through Value-Branch, which considerably improves the performance of average occupancy percentage and time consumption. Finally, comprehensive experiments demonstrated that our proposed MBDQN performed better than both traditional greedy methods and heuristic methods on both the generated zero-waste task dataset and non-zero waste task dataset. These experimental results highlight the effectiveness and reliability of MBDQN in solving satellite resource allocation problems.

Author Contributions: Conceptualization, M.J., N.M., W.Z. and W.S.; methodology, W.Z. and W.S.; software and validation, N.M. and W.S.; formal analysis, N.M. and W.S.; investigation, W.S.; resources, M.J.; data curation, N.M.; writing—original draft preparation, W.Z. and W.S.; writing—review and editing, N.M. and M.J.; visualization, W.S.; supervision, M.J.; project administration, N.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China No. 62231012, Natural Science Foundation for Outstanding Young Scholars of Heilongjiang Province under Grant YQ2020F001.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this research are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jia, M.; Zhang, X.; Sun, J.; Gu, X.; Guo, Q. Intelligent resource management for satellite and terrestrial spectrum shared networking toward B5G. *IEEE Wirel. Commun.* **2020**, *27*, 54–61. [CrossRef]
2. Yanlei, D.; Chunting, W.; Chenhua, S.; Yusheng, L.; Qing, X. Performance Evaluation for Satellite Communication Networks Based on AHP-BP Algorithm. In Proceedings of the 2018 10th International Conference on Communication Software and Networks (ICCSN), Chengdu, China, 6–9 July 2018; pp. 311–316.
3. Bai, Y.; Liang, C.; Chen, Q. Network Slice Admission Control and Resource Allocation in LEO Satellite Networks: A Robust Optimization Approach. In Proceedings of the 2022 27th Asia Pacific Conference on Communications (APCC), Jeju Island, Republic of Korea, 19–21 October 2022; pp. 1–6.
4. Guo, B.; Wang, H.; Wu, P. Application of constraint-based satellite mission planning model in forest fire monitoring. *AIP Conf. Proc.* **2017**, *1890*, 030012.
5. Lin, Z.; An, K.; Niu, H.; Hu, Y.; Chatzinotas, S.; Zheng, G.; Wang, J. SLNR-based secure energy efficient beamforming in Multibeam Satellite Systems. *IEEE Trans. Aerosp. Electron. Syst.* 2022; early access. [CrossRef]

6. Daoden, K.; Thaiupathump, T. Applying shuffled frog leaping algorithm and bottom left fill algorithm in rectangular packing problem. In Proceedings of the 2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC), Macau, China, 21–23 July 2017; pp. 136–139.
7. Karaboga, D.; Gorkemli, B.; Ozturk, C.; Karaboga, N. A comprehensive survey: Artificial bee colony (ABC) algorithm and applications. *Artif. Intell. Rev.* **2014**, *42*, 21–57. [CrossRef]
8. Chen, H.; Zhong, Z.; Wu, J.; Jing, N. Multi-satellite data downlink resource scheduling algorithm for incremental observation tasks based on evolutionary computation. In Proceedings of the 2015 Seventh International Conference on Advanced Computational Intelligence (ICACI), Wuyi, China, 27–29 March 2015; pp. 251–256.
9. Zhang, Z.; Hu, F.; Zhang, N. Ant colony algorithm for satellite control resource scheduling problem. *Appl. Intell.* **2018**, *48*, 3295–3305. [CrossRef]
10. Sarkheyli, A.; Bagheri, A.; Ghorbani-Vaghei, B.; Askari-Moghadam, R. Using an effective tabu search in interactive resources scheduling problem for LEO satellites missions. *Aerosp. Sci. Technol.* **2013**, *29*, 287–295. [CrossRef]
11. Pachler, N.; Luis, J.J.G.; Guerster, M.; Crawley, E.; Cameron, B. Allocating power and bandwidth in multibeam satellite systems using particle swarm optimization. In Proceedings of the 2020 IEEE Aerospace Conference, Big Sky, MT, USA, 7–14 March 2020; pp. 1–11.
12. Hu, X.; Liu, S.; Chen, R.; Wang, W.; Wang, C. A deep reinforcement learning-based framework for dynamic resource allocation in multibeam satellite systems. *IEEE Commun. Lett.* **2018**, *22*, 1612–1615. [CrossRef]
13. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust region policy optimization. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1889–1897.
14. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347.
15. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; pp. 1928–1937.
16. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
17. Cai, Q.; Hang, W.; Mirhoseini, A.; Tucker, G.; Wang, J.; Wei, W. Reinforcement learning driven heuristic optimization. *arXiv* **2019**, arXiv:1906.06639.
18. Deng, B.; Jiang, C.; Yao, H.; Guo, S.; Zhao, S. The next generation heterogeneous satellite communication networks: Integration of resource management and deep reinforcement learning. *IEEE Wirel. Commun.* **2019**, *27*, 105–111. [CrossRef]
19. Huang, Y.; Mu, Z.; Wu, S.; Cui, B.; Duan, Y. Revising the observation satellite scheduling problem based on deep reinforcement learning. *Remote Sens.* **2021**, *13*, 2377. [CrossRef]
20. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1026–1034.
21. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 448–456.
22. Bortfeldt, A.; Gehring, H. New Large benchmark instances for the two-dimensional strip packing problem with rectangular pieces. In Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06), Kauai, HI, USA, 4–7 January 2006; p. 30b.
23. Liu, L.; Jiang, H.; He, P.; Chen, W.; Liu, X.; Gao, J.; Han, J. On the variance of the adaptive learning rate and beyond. *arXiv* **2019**, arXiv:1908.03265.