计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.210400057

基于深度强化学习的智能化渗透测试路径发现

周仕承　刘京菊　钟晓峰　卢灿举
国防科技大学电子对抗学院　合肥 230037
网络空间安全态势感知与评估安徽省重点实验室　合肥 230037
(zhoushicheng@nudt.edu.cn)

摘　要　渗透测试是通过模拟黑客攻击的方式对网络进行安全测试的通用方法,传统渗透测试方式主要依赖人工进行,具有较高的时间成本和人力成本。智能化渗透测试是未来的发展方向,旨在更加高效、低成本地进行网络安全防护,渗透测试路径发现是智能化渗透测试研究的关键问题,目的是及时发现网络中的脆弱节点以及攻击者可能的渗透路径,从而做到有针对性的防御。文中将深度强化学习与渗透测试问题相结合,将渗透测试过程建模为马尔可夫决策模型,在模拟网络环境中训练智能体完成智能化渗透测试路径发现;提出了一种改进的深度强化学习算法 Noisy-Double-Dueling DQN_{per},该算法融合了优先级经验回放机制、双重 Q 网络、竞争网络机制以及噪声网络机制,在不同规模的网络场景中进行了对比实验,该算法在收敛速度上优于传统 DQN(Deep Q Network)算法及其改进版本,并且适用于较大规模的网络场景。

关键词:网络安全;深度强化学习;渗透测试;路径发现;DQN 算法
中图法分类号　TP393

Intelligent Penetration Testing Path Discovery Based on Deep Reinforcement Learning

ZHOU Shi-cheng, LIU Jing-ju, ZHONG Xiao-feng and LU Can-ju
College of Electronic Engineering, National University of Defense Technology, Hefei 230037, China
Anhui Province Key Laboratory of Cyberspace Security Situation Awareness and Evaluation, Hefei 230037, China

Abstract　Penetration testing is a general method for network security testing by simulating hacker attacks. Traditional penetration testing methods mainly rely on manual operations, which have high time and labor costs. Intelligent penetration testing is the future direction of development, aiming at more efficient and low-cost network security protection. Penetration testing path discovery is a key issue in the research of intelligent penetration testing, the purpose of which is to discover vulnerabilities in the network and possible attackers' penetration testing path in time and achieve targeted defense. In this paper, deep reinforcement learning and penetration testing are combined, the agent is trained in simulated network scenarios, the penetration testing process is modeled as a Markov decision process model, and an improved deep reinforcement learning algorithm Noisy-Double-Dueling DQN_{per} is proposed. The algorithm integrates prioritized experience replay mechanism, double DQN, dueling DQN and noise net mechanism. Different scale network scenarios are used for comparative experiments. The algorithm is better than the traditional DQN (Deep Q Network) algorithm and its improved version in convergence speed and can be applied to larger scale network scenarios.

Keywords　Cybersecurity, Deep reinforcement learning, Penetration testing, Path discovery, DQN algorithm

1　引言

渗透测试是一种网络安全测试与评估方法,通过模拟黑客真实的攻击行为,来测试目标网络中可能存在的安全隐患,以达到清除隐患从而提高系统安全性的目的[1]。渗透测试路径是网络中主机以及它们存在的安全漏洞的特定组合,这些组合的搜索空间大小随着主机数量以及漏洞数量的增加呈指数增长,传统的渗透测试主要依赖人工方式,根据个人经验知识对目标网络进行信息获取,进而选取相应的攻击载荷,从而发现网络脆弱性。这一过程需要耗费大量的人力成本和时间成本,因此研究智能高效的渗透测试方法可以降低安全维护成本,帮助网络安全维护人员及时发现网络中存在的安全隐患。

强化学习是机器学习的一个分支,是与人类学习最接近的一种形式,可以通过探索和利用未知环境来学习经验。当前强化学习在游戏领域已经取得了超越人类表现的性能,特别是在 2016 年,AlphaGo 的出现打败了世界顶级围棋选手,其成为了人工智能历史上的里程碑,此外还有 OpenAI 研发的 OpenAI Five[2]、DeepMind 研发的 Alpha Star[3] 以及腾讯公司研发的 MOBA AI[4] 都在相应的游戏领域达到了人类顶尖选手水平。同许多游戏规则类似,渗透测试也是根据环境状态进行动态决策的过程,强化学习的特点无需依赖大量的

到稿日期:2021-04-06　返修日期:2021-05-26　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
通信作者:刘京菊(jingjul@aliyun.com)

静态数据，而是靠与环境的不断交互来学习获得最大奖励值，在学习机制上与渗透测试问题相符。将渗透测试过程游戏化或许可以成为实现智能化渗透测试的新思路，但实际上将人工智能与渗透测试相结合的想法并不新奇，早在 2016 年，美国国防部高级研究计划局（DARPA）举办了名叫网络大挑战（cyber grand challenge）的比赛，该比赛要求参赛队员使用机器学习技术训练智能体来完成网络攻防比赛，是一次真正的人机黑客大赛。

近年来，人工智能技术与渗透测试问题的结合成为了研究热点。Zang 等[5]对当前自动化渗透测试攻击路径发现的研究进展进行了归纳总结，并提出了下一步的研究方向。Zhou 等[6]提出了一种基于网络信息增益的攻击路径规划算法，该算法将渗透测试形式化为马尔可夫决策过程，利用网络信息获取奖赏，引导智能体选择最佳响应动作。为了真实地模拟现实世界中的黑客攻击行为，文献[7-9]将渗透测试过程建模为部分可观测马尔可夫决策（POMDP）模型，但是 POMDP 模型的计算求解复杂度较高，只适用于小规模网络环境。Zennaro 等[10]尝试将 Q-learning 应用到网络安全夺旗赛（CTF）中，实现了强化学习在简单渗透测试场景下的验证。Li 等[11]提出了一种基于 Q-learning 的最优攻击路径生成方法，该方法的局限性在于智能体需要提前获得网络拓扑。此外，由于在现阶段条件下难以使用真实的网络环境训练智能体，Schwartz 等[12]与 Baillie 等[13]提出了构建网络模拟环境的方法，用于强化学习训练并进行了测试。

综上，当前针对智能化渗透测试路径发现问题的研究仍处于初步阶段，POMDP 模型的优点是可以很好地建模渗透测试过程中的不确定性，但随之带来的是计算复杂度的提高，难以适用于大规模的网络场景；基于强化学习的方法以马尔可夫决策模型为基础，当前的研究仅在简单网络场景下进行了实验验证，在算法收敛速度和可扩展性上还存在较大的提升空间。

针对当前研究存在的问题，本文将渗透测试过程建模为马尔可夫决策模型，基于深度强化学习算法训练智能体生成渗透测试路径，本文训练的智能体不需要提前获得网络拓扑，这样学习效率更高，可以适用于较大规模的网络场景，主要工作如下：

（1）提出了一种改进的深度强化学习算法 Noisy-Double-Dueling DQN_{per}，该算法融合了优先级经验回放机制、双重 Q 网络、竞争网络以及噪声网络机制的优点，具有了更高的学习效率以及更好的鲁棒性。

（2）构建了典型的带安全防护的网络模拟环境，通过实验对比了不同算法的性能，随后改变了网络规模，在不同规模的网络场景下测试了算法的可扩展性。

2 模型与算法理论

2.1 马尔可夫决策模型

马尔可夫决策过程（Markov Decision Process，MDP）是建模离散决策问题的通用框架，是强化学习的理论基础。本文将渗透测试过程建模为马尔可夫决策过程，由四元组〈S，A，R，T〉定义。其中，S 表示当前网络状态的集合，是当前智能体已知网络中所有主机状态信息的排列组合，如该主机是否可访问、其操作系统的种类、是否已经获取权限、存在哪些脆弱性服务和进程等；A 表示智能体动作的集合，即智能体可以采取的对目标主机的扫描或漏洞利用操作的集合；R 表示奖励函数，指智能体采取某一动作后获得的即时奖励；T 为状态转移函数，指智能体采取某一动作后可能的状态分布。

强化学习（Reinforcement Learning，RL）以 MDP 为理论基石，不同于监督学习和无监督学习，其是一种从环境状态到动作映射的学习，解决的是贯序决策的问题[14]。使用强化学习的智能体可以通过与环境的反复交互，学会选择最优或近最优的行为以实现其目标，其原理如图 1 所示。

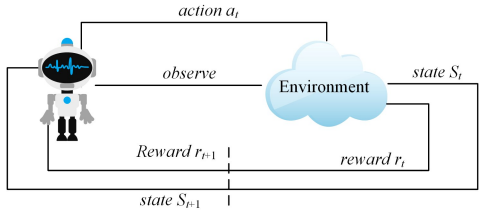

图 1 强化学习基本原理

Fig. 1 Basic principle of reinforcement learning

智能体在时刻 t 观测到环境状态 $s_{t+1}\in S$，根据策略 $\pi(a_t|s_t)$ 选取动作 $a_t\in A$ 后转移到状态 s_{t+1}，并获得实时奖励 r_t。智能体学习的目标是最大化累计奖励函数 R（见式（1）），其中 $\gamma\in(0,1)$ 为折扣因子，代表智能体学习时对未来奖励的关注程度。

$$R=\sum_{k=0}^{\infty}(\gamma^k r_{t+k}) \tag{1}$$

2.2 DQN 及其改进算法

强化学习根据是否用到环境的数学模型可以分为有模型（model-based）学习和无模型（model-free）学习，也可以根据是否需要定义价值函数分为基于价值（value-based）的和基于策略（policy-based）的，Q-learning 是基于价值的无模型学习中具有代表性的算法。Q 代表策略 π 的质量函数 $Q(s,a)$，指在某一时刻的 s 状态下，采取动作 a 能够获得收益的期望，该算法将所有状态 s 与动作 a 关联起来组成一张 Q 表，来存储状态动作对 (s,a) 的 Q 值，通过在每个状态下选择具有最高 Q 值的动作来形成策略，Q 值的更新方式如式（2）所示：

$$Q(s,a)\leftarrow Q(s,a)+\alpha[r+\gamma\max_{a'}Q(s',a')-Q(s,a)] \tag{2}$$

其中，α 为学习率。

Q-learning 采取表格的方式来存储 Q 值，这就限制了该算法解决状态空间和动作空间维度较大的问题。深度强化学习是强化学习与深度学习相结合的全新算法，实现了从感知到动作的学习[15]，可以很好地解决维度灾难问题。

Mnih 等提出了深度 Q 网络[16-17]（Deep Q Network，DQN），DQN 是 Q-learning 的扩展，DQN 采用神经网络来生成 Q 值，DQN 使用两个完全相同的神经网络生成目标 Q 值和估计 Q 值，使用经验回放机制打破数据之间的相关性，目标 Q 值的计算式如式（3）所示：

$$y_i=r+\gamma\max_{a'}Q(s',a';\theta_i^-) \tag{3}$$

其中，θ_i 和 θ_i^- 分别为第 i 步时的估计 Q 网络和目标 Q 网络参数，在间隔一定时间后将估计 Q 网络的参数赋值给目标网络。

DQN 的损失函数为：

$$L_i(\theta_i)=E_{\pi_{\theta_i}}[(y_i-Q(s,a;\theta_i))^2] \tag{4}$$

目前基于DQN算法的改进方法较多，如文献[18]提出了优先级经验回放(Prioritized Experience Replay，PER)机制。为保证数据独立同分布，DQN采用了经验回放机制，传统的经验回放机制采用随机采样的方式，忽视了样本的不同重要程度，优先级经验回放机制基于时间差分误差(TD-error)对经验回放池中的样本重要性进行排序，并根据重要性进行经验回放，该方法与传统DQN的随机采样方式相比，提高了训练过程的效率和稳定性，改善了模型的鲁棒性。

Double DQN算法[19]采用与DQN算法类似的网络结构，从改变目标Q值计算方式的角度，解耦了动作选择和目标Q值计算两个过程，即首先利用当前Q网络找到最大Q值对应的动作。

$$action=\arg\max_{a} Q(s',a;\theta_i) \tag{5}$$

再利用选出来的动作在目标网络中计算目标Q值。

$$y_i=r+\gamma Q(s',\arg\max_{a} Q(s',a;\theta_i);\theta_i^-) \tag{6}$$

实验证明，该方法有效解决了传统DQN中存在的Q值过估计问题。

Dueling DQN[20]从优化神经网络结构的角度出发来优化模型，将Q网络分为仅与状态有关的价值函数$V(s,\theta,\alpha)$以及与状态动作都有关的优势函数$A(s,a,\theta,\beta)$，将Q函数重新表示为：

$$Q(s,a,\theta,\alpha,\beta)=V(s,\theta,\alpha)+A(s,a,\theta,\beta) \tag{7}$$

其中，θ为神经网络的公共部分参数，α,β分别为价值函数和优势函数的特有参数。

“探索与利用”是强化学习的重要问题，传统的DQN采用Epsilon-greedy[21]算法，按照概率选择随机动作或者根据策略选择动作；Noisy DQN[22]从强化学习的“探索与利用”问题角度出发，将高斯噪声添加到Q网络最后的全连接层中，保证尽可能探索Q值高的动作，并通过实验证明了该方法相比传统探索方法而言提高了探索效率。

3 Noisy-Double-Dueling DQN$_{per}$

优先级经验回放机制通过改变经验回放池样本的采样方式，提高了对重要样本的利用率，Noisy DQN通过引入高斯噪声增强了智能体对环境的探索能力，Dueling DQN和Double DQN分别从优化神经网络结构和目标Q值计算方式的角度出发，提高了智能体的学习效率。本文将它们的优点相结合，提出了NDD-DQN$_{per}$(Noisy-Double-Dueling DQN$_{per}$)算法，旨在提高智能体在较大规模网络场景下的渗透测试路径发现的学习效率，提高了算法的可扩展性。

优先级经验回放机制的关键是判断样本的重要程度，采用样本的TD-error作为判断依据，TD-error代表了估计Q值与目标Q值的误差大小，误差越大，样本的价值就越大。样本i的TD-error表示为δ_i，样本i的优先级为p_i，本文采用比例优先级的方式计算p_i，为了防止p_i为零，ε是一个小的正数。

$$p_i=|\delta_i|+\varepsilon \tag{8}$$

则样本i的采样概率定义为：

$$P(i)=\frac{p_i^{\alpha}}{\sum_k p_k^{\alpha}} \tag{9}$$

其中，α代表的是TD-error对样本的影响程度，α越大，影响程度就越高。

在计算目标Q值时，采用Dueling DQN的神经网络结构，将最终Q值的输出变为价值函数和优势函数的线性组合。

$$Q(s,a;\theta,\alpha,\beta)=V(s,;\theta,\alpha)+(A(s,a;\theta,\beta)-\max_{a'\in A} A(s,a';\theta,\beta)) \tag{10}$$

在Dueling DQN神经网络结构的基础上，将噪声参数ξ加入网络的全连接层以增强模型的探索能力，使用ζ表示当前Q网络的参数，ζ^-表示目标Q网络的参数，将利用当前Q网络得到的最大Q值动作带入目标Q网络中，得到目标Q值的计算公式为：

$$y_i^{NDD}=\begin{cases} r, & \text{终止状态} \\ r+\gamma Q(s',\arg\max_{a} Q(s',a,\xi'';\xi),\xi';\xi^-, & \text{其他} \end{cases} \tag{11}$$

损失函数为：

$$L(\theta_i)=\mathbb{E}_{(s,a,r,s')\sim U(D)}[(y_i^{NDD}-Q(s,a,\xi;\theta_i,\alpha,\beta))^2] \tag{12}$$

NDD-DQN$_{per}$算法的模型结构如图2所示，神经网络由带噪声的全连接层组成，输入为环境状态的编码，输出为每个动作的Q值。算法的过程如算法1所示，在DQN算法的基础上，将随机采样的经验回放改变为优先级经验回放，修改目标Q值的计算公式以及损失函数。

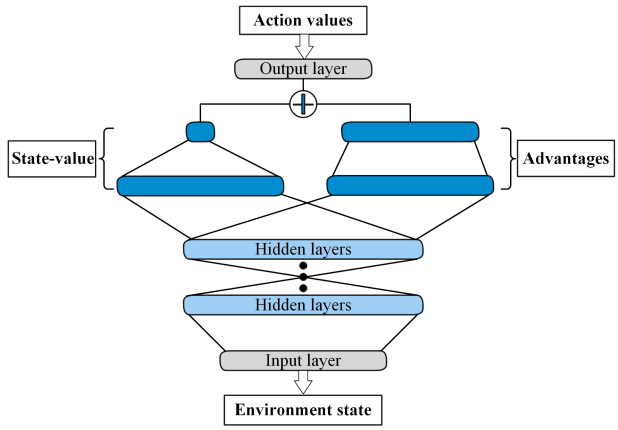

图2 模型结构

Fig. 2 Model architecture

算法1 Noisy-Double-Dueling DQN$_{per}$

输入：经验回放池D，容量N，训练回合数V，每回合训练步数T，目标网络更新频率C，随机噪声集合δ，Q网络参数ζ，目标网络参数$\zeta^-=\zeta$

输出：Q(s,a,ξ;ζ)动作值函数

1. For episode =1 to V do:
2. 初始化环境状态
3. For step=1 to T do:
4. Q网络噪声采样 ξ←δ
5. 选取动作 $a_t\leftarrow\arg\max_a Q(s_{t+1},a,\xi;\zeta)$
6. 执行动作 a_t，观测获得的奖励 r_t 以及下一状态 s_{t+1}
7. 存储序列 (s_t,a_t,r_t,s_{t+1})到经验回放池D，并更新优先级
8. 从经验回放池D中依据优先级采样序列(s_j,a_j,r_j,s_{j+1})
9. Q网络噪声采样 ξ′←δ，目标网络噪声采样 ξ″←δ
10. 依据式(11)计算目标Q值 y_j
11. 依据式(12)计算梯度下降以及损失函数

12. 每隔频率 C 更新目标网络参数 $\zeta^{-} \leftarrow \zeta$
13. End For
14. End For

4 实验

实验使用 Pytorch 作为算法的代码框架，硬件配置为 NVIDIA Geforce RTX3090 GPU，Inter Xeon Gold 6248R CPU。实验是基于文献[12]提出的构建网络模拟环境的方法，构建不同规模的网络场景，用于测试算法的性能。在构建的模拟网络环境的基础上对算法的收敛性和可扩展性进行测试：首先在同一网络场景中使用不同的深度强化学习算法，对比在规定训练步数上的收敛速度；其次在不同规模的网络场景下使用 NDD-DQN$_{per}$ 算法训练智能体，测试算法的可扩展性。

4.1 实验场景

图 3 所示的网络为实验场景一，网络中有 7 个子网，17 台主机，其中子网 1、子网 4－子网 6 与内外网连接，子网 2、子网 3 为内网，子网 7 为蜜罐。子网之间设有防火墙，防火墙的作用是阻断特定流量。子网内主机地址由一个二元组（子网号，主机号）简化表示，子网内主机可以互相通信，每台主机定义其运行不同的服务和进程，服务表示主机上存在漏洞的软件，攻击者可以利用这些漏洞获取相应权限，获取权限之后可以根据主机运行的进程进行提权。

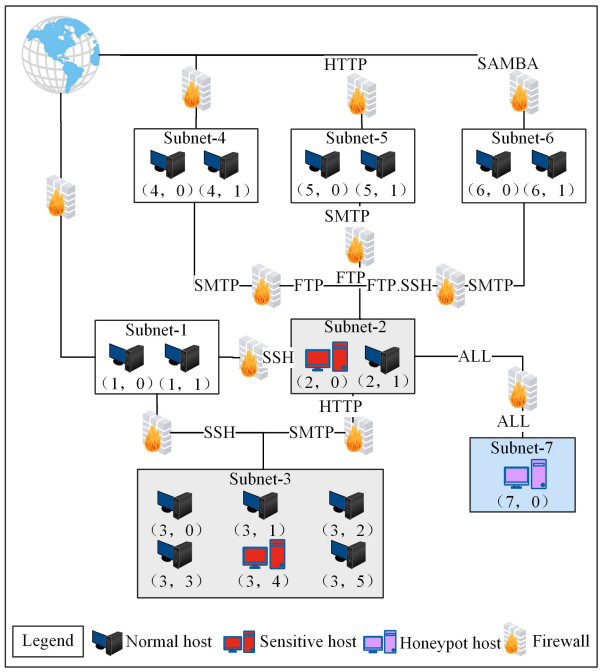

图 3 实验场景一

Fig. 3 Experiment scenario-1

为模拟真实的攻击者行为，设定智能体在探索过程中不能掌握网络拓扑信息以及主机配置信息，因此除漏洞利用（Exploit，Exp）和权限提升（Privilege Escalation，PE）动作外，智能体还可以采取扫描（scan）操作，以获取目标网络和主机的相关信息。

智能体在训练过程中的渗透路径的搜索空间大小随网络中主机的数量呈指数增长，可以表示为 $S \in O(N^H)$，其中，网络中主机数量为 H，对每台主机可进行的扫描、漏洞利用以及提权动作的数量为 N。

根据相应的服务选取具有代表性的具体漏洞来计算漏洞利用动作成功的概率，概率的确定方法参照文献[23]。根据通用漏洞评估方法（Common Vulnerability Scoring System，CVSS）访问复杂性（access complexity）指标的“高”“中”“低”值，分别设置为“0.2”“0.5”“0.8”的概率值，提权操作和扫描操作在本场景下假定概率值为 1。

奖励值是智能体学习的依据，奖励值定义为主机价值（value）减去动作代价（cost），动作的代价是时间、技能、金钱成本的综合量化。奖励值的计算式如式(13)所示：

$$Reward = \sum_{h \in H} value(h) - \sum_{a \in A} cost(a) \tag{13}$$

其中，H 表示所有被智能体入侵的主机的集合，A 表示智能体所有采取的动作的集合。基于这样的奖励值，智能体学习的目标就是尽可能用少的操作来攻击价值最大的主机。

实验场景一中，智能体可以采取的动作如表 1 所列。对于每一台主机，智能体可以采取表 1 中的动作来执行，实验选取内网渗透过程中经常被攻击者利用的服务和进程来指代漏洞，如智能体可以利用 FTP 漏洞获取主机的 ROOT 权限，利用 Tomcat 漏洞对获取到 USER 权限的主机进行提权。主机的配置信息如表 2 所列，每一台主机定义了地址、操作系统、主机价值、存在漏洞的服务和进程。

表 1 智能体动作列表

Table 1 Agent action list

Name	Type	Operating system	Cost	Probability	Access
SSH	Exp	Linux	3	0.8	USER
FTP	Exp	Windows	1	0.5	ROOT
HTTP	Exp	Linux	2	0.8	USER
SAMBA	Exp	Windows	2	0.2	ROOT
SMTP	Exp	Windows	3	0.5	USER
Tomcat	PE	Linux	1	1	ROOT
Daclsvc	PE	Windows	1	1	ROOT
Schtask	PE	Windows	1	1	ROOT
Subnet-scan	Scan	—	1	1	—
OS-scan	Scan	—	1	1	—
Service-scan	Scan	—	1	1	—
Process-scan	Scan	—	1	1	—

表 2 主机配置列表

Table 2 Host configuration list

Address	Operating system	Host value	Service	Process
(1,0)(1,1)	Linux	0	SSH	Tomcat
(3,0)(6,0)	Linux	0	SSH	Tomcat
(3,2)(3,3)	Linux	0	SSH	—
(4,0)(4,1)	Windows	0	FTP	Daclsvc
(2,0)	Windows	100	SMTP	—
(2,1)	Windows	0	SMTP	Schtask
(3,1)	Linux	0	SSH,HTTP	—
(3,4)	Linux	100	SSH	Tomcat
(5,0)	Windows	0	FTP	Daclsvc, Schtask
(5,1)	Windows	0	FTP,HTTP	—
(6,1)	Windows	0	SSH,SAMBA	—
(7,0)	Windows	-500	ALL	Daclsvc, Schtask

初始状态时，设定敏感主机奖励值为正值，蜜罐主机的奖励值为负值，智能体位于互联网中，只知道与互联网相连的子网拓扑。智能体在网络中通过一轮轮探索来学习攻击者行为

的行为策略，每一轮探索的终止条件可以是下列3种情况：

(1)获取所有的敏感主机的ROOT权限；

(2)训练步数达到设定的最大值；

(3)入侵到蜜罐主机。

在奖励值的驱动下，智能体通过不断试错来学习最大累积奖励值策略。初始时没有经过学习，智能体对未知环境倾向于使用随机策略来选择动作进行执行，平均每回合使用的训练步数较大，所获得的累计奖励值较小，而且会入侵蜜罐主机，但随着不断的探索和学习，导致累积奖励值减小的行为(如过多的动作或入侵到蜜罐主机)将不会被学习，最终智能体将学会用最少的动作去攻击有正值奖励值的敏感主机。

为测试不同规模网络下算法的可扩展性，在场景一的基础上增加网络中的主机数量，构建的其余6个网络场景如表3所列。

表3 实验场景列表

Table 3 Experiment scenarios list

Scenario	Scenario-2	Scenario-3	Scenario-4
Scale(Host number)	20	40	60
Scenario	Scenario-5	Scenario-6	Scenario-7
Scale(Host number)	80	100	120

4.2 实验结果

首先是在实验场景一下使用同样的一组超参数测试DQN，DQN_{Per}，$Double\text{-}DQN_{Per}$，$Noisy\text{-}DQN_{Per}$，$Dueling\text{-}DQN_{Per}$和$NDD\text{-}DQN_{Per}$这6种算法，以DQN为基准对比衡量算法在规定训练步数(steps)上平均奖励值(mean reward)的变化(见图4)、蜜罐主机被入侵的概率(见图5)以及每回合所用训练步数的变化(见图6)。

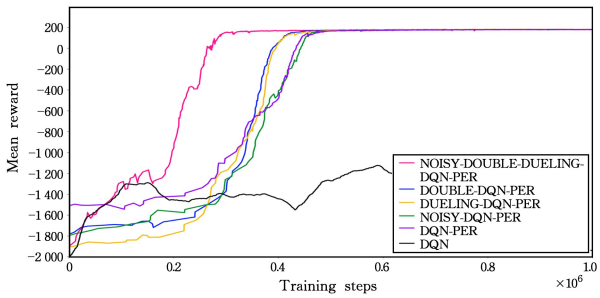

图4 平均累积奖励值随训练步数的变化

Fig. 4 Mean reward versus training steps

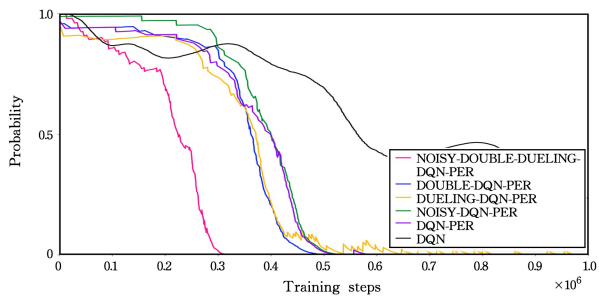

图5 蜜罐主机被入侵概率随训练步数的变化

Fig. 5 Probability of honeypot host been invaded versus training steps

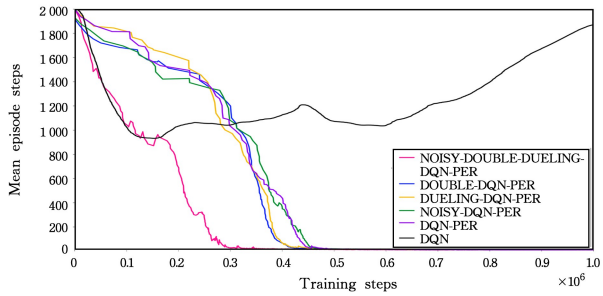

图6 每回合训练所用步数的变化

Fig. 6 Mean episode steps versus training steps

设定总的训练步数为10^6，每回合允许的最大训练步数为2000，表示智能体在每回合只能探索2000次，若2000次探索没有完成目标，则该回合结束，训练过程的其他超参数如表4所列。

表4 超参数列表

Table 4 Hyperparameter list

Hyperparameter	Meaning	Value
Max steps	总训练步数	10^6
Step limit	每回合允许的最大训练步数	2000
Learning rate, lr	学习率	0.0001
Batch size	训练所选取的样本批次大小	64
Discount factor, γ	折扣因子	0.99
Hidden layer size	隐藏层单元数	128
Replay memory size	经验回放池大小	50000
Target network update frequency	目标网络更新频率	1000

为验证$NDD\text{-}DQN_{per}$算法的可扩展性，依据表3所列的网络场景，测试算法在规定训练回合数(episode)上平均奖励值的变化。实验结果如图7所示，实验规定训练回合数为400，每一回合的最大训练步数为5000，其余超参数与表4所列内容相同。

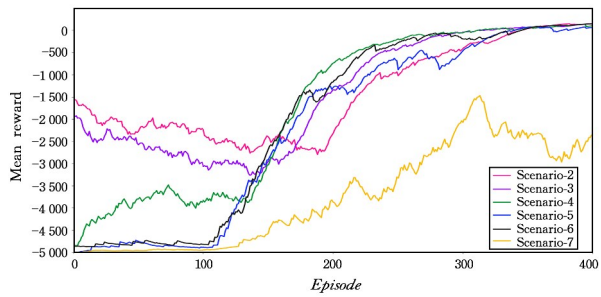

图7 不同规模场景下平均累积奖励值随训练回合数的变化

Fig. 7 Mean reward on different scenarios versus training episodes

随着训练的进行，除DQN外其余算法均能稳定收敛到最优值，累积奖励值不断增加，智能体入侵蜜罐的概率以及每回合探索所用的步数不断减小，其中$NDD\text{-}DQN_{Per}$算法的收敛速度最快。这是因为，智能体在网络中进行渗透路径的搜索面临两个问题：1)大规模离散动作空间的问题，智能体每一步的可选动作空间大小与网络中主机数量和漏洞数量相关；2)稀疏奖励值的问题，整个网络中的正向奖励只有少量的敏

感主机,意味着智能体的绝大部分探索都无法获取奖励,而在强化学习中,奖励是智能体学习的依据,起到监督信号的作用,稀疏奖励会导致强化学习算法迭代缓慢,难以收敛[24]。优先级经验回放机制的加入极大地提高了样本的利用效率,使得智能体优先选择历史经验中高奖励值的样本来学习;噪声网络的引入提高了智能体的探索效率,有助于智能体在大规模状态空间中学习到最佳策略; Double DQN 和 Dueling DQN 的引入使得 Q 值的计算更加准确,提高了算法的收敛速度和稳定性。

在扩大网络规模时,路径的搜索空间大小随网络中主机的数量呈指数增长,在本实验中,当网络中的主机数量不超过 100 时,在规定的每回合最大训练步数以及训练回合数内算法可以收敛到最优值。

4.3 结论

实验在 7 个网络场景上进行了测试,在实验场景一下,使用相同的超参数对比了不同的算法收敛速度上的差异,本文提出的 NDD-DQN$_{per}$ 算法的收敛速度最快,能在更短的时间内得到最佳渗透路径;在实验场景二至实验场景七上测试了不同规模下算法的可扩展性,在规定的训练步数内,算法可以在主机数量不超过 100 时收敛到最优值,说明 NDD-DQN$_{per}$ 算法可以适用于较大规模的网络场景,当规模增大时,算法的收敛速度受限于智能体对环境的探索程度,增加训练回合数或者增加每回合的最大训练步数可以保证智能体对环境进行更充分地探索,进而学习到最佳策略。

结束语 本文从 DQN 算法的经验回放机制、目标 Q 值计算方式、神经网络结构以及强化学习"探索-利用"问题的改进出发,融合了多种 DQN 改进版本,总结提出了 NDD-DQN$_{per}$ 算法,基于此算法训练智能体模拟黑客对网络进行渗透测试,验证了算法在不同网络规模下的性能,本文算法在收敛速度上优于传统 DQN 算法及其改进版本,同时可以适用于较大规模的网络场景。

受限于强化学习与环境的强交互性,当前应用强化学习的智能化渗透测试仍处于模拟验证阶段,应用虚拟化技术在仿真网络甚至真实网络中训练智能体是未来的发展方向,同时,为了能适用于更大规模的网络场景,更好地解决大规模离散动作空间和稀疏奖励值的问题,在未来可以优化奖励函数设置方式以及将分层强化学习(Hierarchical Reinforcement Learning)应用到智能体的训练中,将目标的选取与动作的选择分离,进而提高训练效率。

参 考 文 献

[1] XIONG Y. Design and Implementation of Automatic Penetration Testing Platform[D]. Beijing: Beijing University of Posts and Telecommunications, 2019.

[2] BERNER C, BROCKMAN G, CHAN B, et al. Dota 2 with large scale deep reinforcement learning[J]. arXiv: 1912.06680, 2019.

[3] VINYALS O, BABUSCHKIN I, CZARNECKI W M, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning[J]. Nature, 2019, 575(7782): 350-354.

[4] YE D, CHEN G, ZHANG W, et al. Towards playing full moba games with deep reinforcement learning[J]. arXiv: 2011.12692, 2020.

[5] ZANG Y C, ZHOU T Y, ZHU J H, et al. Domain-Independent Intelligent Planning Technology and Its Application to Automated Penetration Testing Oriented Attack Path Discovery[J]. Journal of Electronics & Information Technology, 2020, 42(9): 2095-2107.

[6] ZHOU T, ZANG Y, ZHU J, et al. NIG-AP: a new method for automated penetration testing[J]. Frontiers of Information Technology & Electronic Engineering, 2019, 20(9): 1277-1288.

[7] SHMARYAHU D, SHANI G, HOFFMANN J, et al. Simulated penetration testing as contingent planning[C]// Proceedings of the International Conference on Automated Planning and Scheduling. 2018.

[8] SARRAUTE C, BUFFET O, HOFFMANN J. POMDPs make better hackers: Accounting for uncertainty in penetration testing[C]// Proceedings of the AAAI Conference on Artificial Intelligence. 2012.

[9] SCHWARTZ J, KURNIAWATI H, EL-MAHASSNI E. POMDP+ Information-Decay: Incorporating Defender's Behaviour in Autonomous Penetration Testing[C]// Proceedings of the International Conference on Automated Planning and Scheduling. 2020: 235-243.

[10] ZENNARO F M, ERDODI L. Modeling penetration testing with reinforcement learning using capture-the-flag challenges and tabular Q-learning[J]. arXiv: 2005.12632, 2020.

[11] LI T, CAO S J, YIN S W, et al. Optimal method for the generation of the attack path based on the Q-Learning decision[J]. Journal of Xidian University, 2021, 48(1): 160-167.

[12] SCHWARTZ J, KURNIAWATI H. Autonomous penetration testing using reinforcement learning[J]. arXiv: 1905.05965, 2019.

[13] BAILLIE C, STANDEN M, SCHWARTZ J, et al. Cyborg: An autonomous cyber operations research gym[J]. arXiv: 2002.10667, 2020.

[14] SUTTON R S, BARTO A G. Reinforcement learning: An introduction[M]. MIT press, 2018.

[15] ZHAO X Y, DING S F. Research on Deep Reinforcement Learning[J]. Computer Science, 2018, 45(7): 1-6.

[16] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Playing atari with deep reinforcement learning[J]. arXiv: 1312.5602, 2013.

[17] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529-533.

[18] SCHAUL T, QUAN J, ANTONOGLOU I, et al. Prioritized experience replay[J]. arXiv: 1511.05952, 2015.

[19] VAN HASSELT H, GUEZ A, SILVER D. Deep reinforcement learning with double Q-learning[C]// Proceedings of the AAAI Conference on Artificial Intelligence. 2016.

[20] WANG Z, SCHAUL T, HESSEL M, et al. Dueling network ar-

chitectures for deep reinforcement learning[C]// International Conference on Machine Learning. PMLR, 2016: 1995-2003.
[21] WUNDER M, LITTMAN M L, BABES M. Classes of multiagent q-learning dynamics with epsilon-greedy exploration[C]// ICML. 2010.
[22] FORTUNATO M, AZAR M G, PIOT B, et al. Noisy networks for exploration[J]. arXiv: 1706. 10295, 2017.
[23] BACKES M, HOFFMANN J, KÜNNEMANN R, et al. Simulated penetration testing and mitigation analysis[J]. ArXiv, abs/1705. 05088.
[24] YANG W Y, BAI C J, CAI C, et al. Survey on Sparse Reward in Deep Reinforcement Learning[J]. Computer Science, 2020, 47(3): 182-191.

ZHOU Shi-cheng, born in 1995, postgraduate. His main research interests include cyberspace security and reinforcement learning.

LIU Jing-ju, born in 1974, professor. Her main research interests include cyberspace security and machine learning.