MARLIN：基于软演员-评论家的真实网络拥塞控制强化学习

Raffaele Galliera[†,⋆]，Alessandro Morelli[†]，Roberto Fronteddu[†]，Niranjan Suri[†,⋆,‡]
[†]佛罗里达州人类与机器认知研究所（IHMC）[⋆]智能系统与机器人系——西佛罗里达大学（UWF），美国佛罗里达州彭萨科拉
[‡]美国陆军研究实验室（ARL），美国马里兰州阿德尔菲 {rgalliera, amorelli, rfronteddu,nsuri}@ihmc.org

arXiv:2302.01301v1 [cs.LG] 2 Feb 2023

摘要——快速高效的传输协议是日益分布式世界的基础。为了支持下一代应用和服务，持续提升通信性能的需求，加上系统和网络技术日益增强的异构性，推动了能够在特定环境下表现良好的拥塞控制（CC）算法的设计。如何设计一种能够适应广泛场景的通用拥塞控制算法，仍是一个尚未解决的研究问题。为应对这一挑战，我们提出采用一种新颖的强化学习（RL）方法。我们的解决方案 MARLIN 使用软演员-评论家算法，同时最大化熵和回报，并将学习过程建模为一项无限时域任务。我们在具有不同背景流量模式的真实网络上训练 MARLIN，以克服研究人员在将强化学习应用于拥塞控制时遇到的仿真到现实差距。我们在文件传输任务上评估了该方案，并将其与 TCP CUBIC 进行了比较。虽然仍需进一步研究，但结果表明，MARLIN 只需少量超参数调优，便能在与训练环境显著不同的任务中取得与 TCP 相当的结果。因此，我们认为，本研究是构建基于最大熵强化学习框架的拥塞控制算法的有前景的第一步。

索引词——计算机网络、通信协议、机器学习、拥塞控制、强化学习、软演员-评论家。

一、引言

网络通信是日益分布式的数字世界的支柱。如今，云/边缘/雾计算系统上的软件即服务、平台即服务和基础设施即服务、物联网、5G网络、空间通信与网络、点对点网络、区块链、自组织网络和网状网络等技术，每天都让无数组织和个人只需轻点屏幕，便能开展业务并完成已成为日常生活一部分的任务。这些技术各自不同的特性，加上基于它们构建的各类服务所提出的多样化需求，为网络协议设计者和研究人员带来了独特的挑战。

能够在不断变化的网络条件和连接频繁更替的情况下，实现较高网络资源利用率，同时维持较低拥塞（即端到端排队延迟）的传输协议，是现代分布式系统背后的关键使能技术。在传输协议设计中，这一双重目标（高信道利用率和低拥塞）正是拥塞控制（CC）算法的目标。拥塞控制对分布式应用和服务性能有着至关重要的影响，这促使研究人员和工程师设计了许多变体，每种变体都面向不同的目标场景，并在激进性、对丢包和拥塞的响应能力、公平性和友好性方面有所取舍 [1, 2, 3]。然而，设计一种能够在广泛的网络和流量场景中提供近乎最优性能的通用拥塞控制算法，仍然是一项尚未解决的挑战。

新型 CPU、GPU 和专用集成电路（ASIC）架构提升了计算能力，加之此类硬件加速器价格低廉却性能强大，推动机器学习（ML）技术在多个领域的成功应用呈爆发式增长，并开始满足边缘等资源受限环境的需求 [4]。这重新激发了人们对开发机器学习模型、强化学习（RL）智能体、算法和应用的研究兴趣，以应对真实世界中的独特挑战 [5, 6, 7, 8]。

与相对容易收集通信性能数据、且动作空间较小（通常只关注调整拥塞窗口（CWND）的大小）相比，设计高效的通用拥塞控制算法颇具挑战性，这促使研究人员探索将机器学习用于拥塞控制优化 [9, 10]。尽管许多研究已展现出令人鼓舞的成果，但在许多场景下，其表现仍不及现有的 TCP 拥塞控制算法。原因可能包括：难以利用真实网络和硬件训练强化学习智能体和/或机器学习模型；仿真和仿真环境的保真度较低；环境表示（状态）、动作空间和/或奖励函数的设计不够理想；以及通信延迟导致动作对状态的影响滞后。仍需开展更多研究来解决这些问题。

本文介绍了我们在 MARLIN 上的初步工作。MARLIN 是一种用于拥塞控制的强化学习智能体。我们的方法基于软演员-评论家（SAC） [11], , 这是一种离策略、熵正则化的强化学习算法，已成功应用于众多真实世界

©2023 IEEE。允许将本材料用于个人用途。对于任何其他用途，均须获得 IEEE 的许可；这些用途包括在任何当前或未来媒体中重印或再版本材料以用于广告或宣传目的、创作新的合集作品、转售或重新分发至服务器或列表，以及在其他作品中重用本作品中受版权保护的任何部分。

问题，尤其是机器人学领域的问题 [7]。我们介绍了 MARLIN 的设计与架构，并讨论了智能体的训练方式、所用超参数，以及设计阶段作出的主要假设和选择，同时将我们的工作与其他基于强化学习的拥塞控制方法进行了比较。此外，我们还探讨了这项研究的未来方向，而本文只是迈出的第一步。最后，我们展示了在真实网络场景中获得的初步实验结果，并在文件传输任务中考察 MARLIN 当前的性能，将其与传输控制协议（TCP）进行比较。

II. 设计用于拥塞控制的强化学习智能体

MARLIN 是一个用于拥塞控制的强化学习智能体，通过在真实网络中使用连续动作更新拥塞窗口进行训练。在本研究中，我们没有将相对于相互竞争的 MARLIN 流和/或其他协议的公平性纳入 MARLIN 的优化目标；我们将这一问题留待未来研究。

在 MARLIN 中，学习基于软演员-评论家 [11], ，该方法训练的是一种随机、离策略、熵正则化智能体。该智能体通过远程过程调用（RPC），以非阻塞、双向的方式与一个自定义传输协议交互，该协议将在第三节中介绍，如图1所示。采取动作的频率取决于对该路径估算出的最新平滑往返时间（SRTT）。奖励函数被设定为严格负值，以此内在地体现这样一条原则：传输信息时采取的每一步都是多余的一步。

尽管通常希望智能体尽可能快地完成任务，但这里提出的强化学习设置并不以终止状态为目标。若智能体在固定数量的待传输字节或固定回合长度上过拟合其轨迹，就会造成不便，因为它可能试图在一定数量的时间步内不惜一切代价找到终止状态。相反，我们旨在设计能够在“拥挤”链路上运行、且无论训练期间传输的数据量如何都能实现高信道利用率的强化学习智能体。因此，训练基于无限时域任务，不会遇到终止状态，并遵循 [12] 中介绍的无限时域训练部分回合自举（PEB）原则。

A. 将拥塞控制视为马尔可夫决策过程

在 TCP 等传输协议中实现的传统拥塞控制算法会采取动作，例如更新其拥塞窗口，以响应其感知到的环境变化，目标是实现最大吞吐量和最小拥塞。变化可能包括丢包、重复数据包或测得的往返时间（RTT）变化等。这种序贯决策设置可以自然地表述为马尔可夫决策过程（MDP） [13]。

马尔可夫决策过程框架通过 $\langle S, A, p, R\rangle$ 将智能体与环境按顺序进行的交互形式化为：在这一过程的每个时间步 t 交换三种不同的信号。

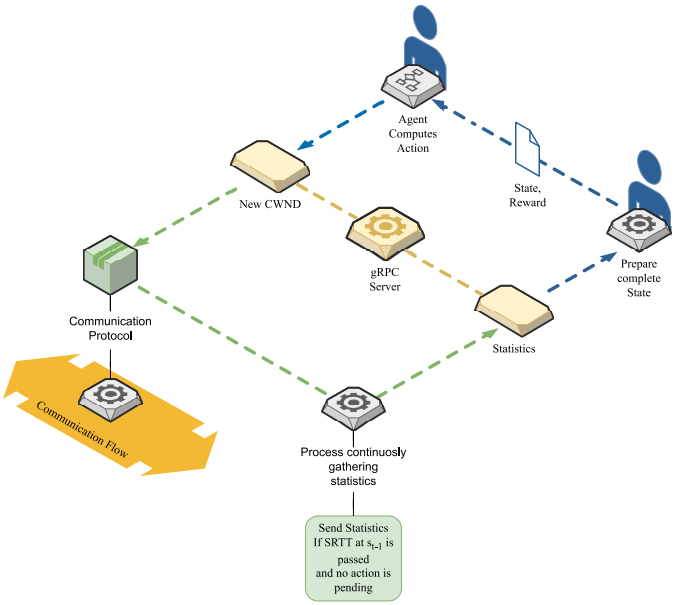


图1：智能体-协议接口。通信协议与智能体通过 gRPC服务器进行通信。

强化学习“从交互中学习”的基础 [14] 源于这一抽象：智能体在某一步 t 感知到状态 $s_t \in S$ 后所采取的动作 $a_t \in A(s_t)$，会使环境作出相应响应，产生奖励信号 $r_{t+1} \in R \subset \mathbb{R}$ 并转移到新的状态 s_{t+1}。这一过程遵循由概率函数 $p(s_{t+1}, r_{t+1}|s_t, a_t)$： $S \times R \times S \times A \longrightarrow [0,1]$ 描述的环境动力学。对状态转移动力学的这种表述也定义了马尔可夫性质：状态必须包含过去交互中会影响未来的所有相关信息，因为 $r_t \in R$ 和 $s_t \in S$的每个可能取值都仅取决于先前的状态和动作 s_{t-1}, a_{t-1}。

通信网络对强化学习智能体而言是一个微妙的环境。现代网络中的数据包通常间隔几毫秒传输一次，有时甚至只有几微秒。与此同时，在训练过程中运行随机梯度下降（SGD）来更新模型，甚至仅仅是计算一个新的动作，都可能耗费长得多的时间。因此，采取动作时，环境可能已经发生了显著变化，这可能使该动作过时，甚至变得错误。这被认为是将模拟环境中达到的性能迁移到真实世界部署后难以维持的主要原因之一 [15]。

此外，动作不会立即影响智能体的感知，因为改变拥塞窗口后，需要经过一段时间才会对网络产生影响，而这种影响还需要传回源端。采取动作后，新数据包会以不同的速率发送；它们首先需要到达目的地，然后接收方需要为这些数据包生成并发送新的确认（ACK）消息，最后该确认需要返回源端，才能获取有关影响的信息

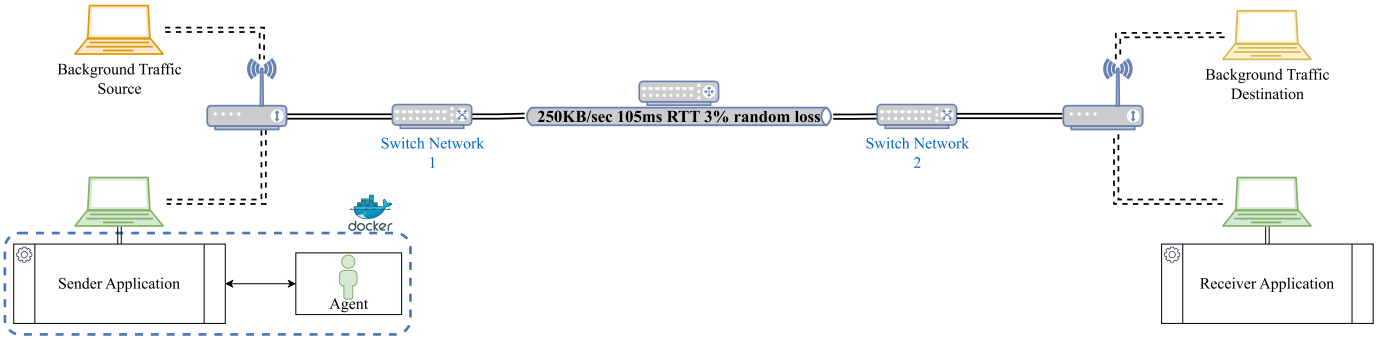


图 2：训练和测试使用哑铃网络。背景流量和智能体流量通过瓶颈链路从一个子网络传输到另一个子网络。发送端和接收方均连接到 WiFi接入点。

并推断所采取动作的影响。这些事件至少需要一个往返时间才能完成。这就引出了一个根本问题：对于想要将强化学习应用于拥塞控制问题的人来说，何时应采取下一个动作？其他类似方法（例如 [16, 17]，在固定时间内收集统计信息以计算状态并采取下一个动作）采用了不同的启发式算法。每当传输协议向 MARLIN 提供新的网络统计信息时，如果自上次采取动作以来至少已经过了最后一次报告的平滑往返时间，就会对截至该时刻收集的数据进行完善，分配与动作a_{t-1} 相关的奖励 r_t，将新状态 s_t 输入智能体，并采取新的动作 a_t。动作之间至少等待一个平滑往返时间，可以避免在智能体感知到前一个动作的影响之前就采取新动作。

B. 训练环境设置

MARLIN 的一个关键设计点是在真实网络上进行训练和评估。使用真实组件使我们能够了解拥塞控制领域给强化学习带来的挑战，避免因仿真模式不准确和假设过于简化而导致“仿真到现实”迁移失败的风险，代价是训练时间更长。依托真实硬件和协议实现，环境的随机性能够自然地丰富训练经验，并防止智能体对现实世界中不存在的人工模式过拟合。

为此，我们搭建了图 2 所示的网络，其中包括两个流量源和两个接收端、两个 WiFi接入点（AP）、两个网络交换机，以及一台按哑铃拓扑连接的路由器。流量源连接到位于哑铃一端的 WiFi接入点，而流量接收端连接到另一端的接入点。左侧的“背景流量源”负责生成背景流量，并将其发送到右侧的“背景流量接收端”节点。“发送端应用程序”通过一种自定义传输协议实现向“接收端应用程序”传输流量，该实现使用 MARLIN 作为其拥塞控制算法。

对实现灵活性的需求促使我们将 MARLIN 与 Mockets 集成。Mockets 是一种最初为通信环境而设计的协议，这类环境的带宽有限，通常见于战术网络和无线传感器网络。关于 Mockets 的更多详情，请参阅第 III 节。为避免 MARLIN 因协议实现而表现不佳，本次迭代中，我们围绕 Mockets 的典型使用场景构建了网络环境。我们在路由器上设置了智能队列策略，将两个子网之间的最大流量限制为 250 KB/s；此外，我们还在“发送端应用程序”节点上使用了 TC NetEm - 流量控制网络模拟器（TC-NETEM），引入 100 ms 的延迟和 3% 的随机丢包，这些影响仅作用于应用程序生成的流量。请注意，路由器的手册指出，启用智能队列后，实际速率会被有效限制为指定值的 95% [18]。

生成的背景流量模式可分为大象流和老鼠流。大象流是指持续时间较长的数据传输，占总流量的很大比例（例如大型文件传输或视频流媒体）；老鼠流则是指吞吐量较低、持续时间较短的数据传输（例如电子邮件或 RPC 调用）。在我们的测试平台中，背景流量约有 87% 来自大象流，其中包括四条彼此交替、每两秒在链路上传输一次的不同流，以及两条老鼠流。后者会持续产生持续时间极短（以毫秒计）的流量突发，其间隔符合泊松分布。

BackgroundTraffic 源使用多生成器网络测试工具（MGEN） [19]生成流量。大象流包括两条以 100 KB/s 传输数据的 UDP 通信、一条以 50 KB/s 传输数据的 UDP 通信，以及一条产生 200 KB/s 流量的 TCP 连接。一条 TCP 老鼠流和一条 UDP 老鼠流平均每秒在链路上额外产生 17 KB 流量。每条大象流都分配到一个持续 2 秒的不同时间槽中，并每 8 秒重复一次。老鼠流与第一条大象流同时启动，并持续运行，直到停止流量生成。

C. 智能体设计

MARLIN 与传输协议之间的接口如图1所示，使用 gRPC 实现。位于传输协议与 MAR-

	特征	描述		统计量
1	当前拥塞窗口	当前拥塞窗口	1	Last
2	已发送的 KB	已发送 KB 数量 *	2	Mean
3	新发送的 KB	已确认的 KB 数量 *	3	STD
4	已确认的 KB	已确认的 KB 数量 *	4	Min
5	已发送数据包	已发送数据包 *	5	Max
6	重传	重传数据包数量 *	6	EMA
7	瞬时吞吐量	吞吐量 *	7	与前一项的差值
8	瞬时有效吞吐量	有效吞吐量 *		
9	未确认的 KB	在途 KB 数量		
10	最后一次往返时间	检测到的最后一次往返时间 *		
12	最小往返时间	最小值往返时间 *		
12	最大值往返时间	最大值往返时间 *		
13	SRTT	平滑往返时间 *		
14	往返时间方差	RTT 方差 *		
		* 在最后一次 RTT 时段内		

表 I：构成状态的特征　　其所属环境。观测的时间范围还扩展为包含之前的 10 次观测 .

LIN 之间的 gRPC 中间件负责向智能体传递观测和奖励对，并将动作传回协议。

在给定的步骤 t，智能体通过 gRPC 中间件从传输协议接收网络统计信息。随后对统计信息进行处理，并与之前的 10 次观测堆叠，形成状态 s_t。智能体可以采取动作，按选定的因子增大、保持不变或减小传输协议的拥塞窗口。理想情况下，智能体应学会在尽可能短的时间内最大化传输数据量，同时密切关注不断增大的排队延迟；这会体现为测得的 RTT 增加。

MARLIN 基于 RL Baselines3 Zoo（RL-Zoo3） [20] 框架实现，该框架遵循 Stable-Baselines3（SB3） [21], 的最佳实践。Stable-Baselines3 是一个基于 PyTorch [22]的库，用于实现遵循 OpenAI Gym 接口 [23]的先进强化学习算法。

1) 状态：表 I 描述了 MARLIN 中编码的状态。在步骤 $t-1$ 执行动作后随后的时间段内，会收集 14 个特征。随后，MARLIN 为状态空间增加 7 个统计量，即最后一次、均值、标准差、最小值、最大值、指数移动平均（EMA）以及与前一状态 s_{t-1} 的差值；这些统计量会针对 14 个特征中的每一个进行计算。之前的 N 状态也会被叠加在一起，以形成先前观测的历史记录，从而试图遵循马尔可夫性质。最终提供给智能体的状态将包含 $N \times |Features| \times |Stats|$ 个特征。总计为 980 个不同的特征。

N 被视为该问题的一个超参数；表 II 完整列出了 MARLIN 使用的所有超参数。对于 N 的选择，我们遵循了 [24]，中提出的经验性考量，并在进行了一些初步试验和评估后，选择 10 作为历史记录长度。在训练过程中，观测会通过 SB3 中提供的 VecNormalize 环境包装器，利用移动平均进行处理和归一化。

2) 动作：MARLIN 的连续动作范围为 $[-1, 1]$，表示拥塞窗口大小的增益百分比。例如，如果智能体选择的动作是 1，则拥塞窗口会加倍；取值为 0 表示拥塞窗口不变；-0.5 会将窗口大小缩小 50%。回合开始时，初始拥塞窗口大小设为 4KB，这是传输协议实现的默认设置。

在本研究中，我们将拥塞窗口大小上限设为 50 KB；在我们用于训练和评估的配置中，若达到此值，吞吐量将是网络所能承载吞吐量的两倍。这一选择只会影响训练最初阶段，此时智能体会采取随机步骤来预热模型。过了这一阶段后（在 MARLIN 中，该阶段设为持续 10K 步），我们的实验表明，智能体确实不会将拥塞窗口填满至 50 KB。

3）奖励：我们设计了一种奖励函数，智能体越接近充分利用可用带宽，获得的奖励就越高：

$$r_t = -\frac{target_t}{target_t + acked_kilobytes_t^{cumulative}} \tag{1}$$

其中，$target_t$ 表示自回合开始以来，为了充分利用链路，智能体截至步骤 t 应已传输的字节数；$acked_kilobytes_t^{cumulative}$表示截至步骤 t接收方已确认的千字节数。

严格为负的奖励函数会促使智能体尽可能减少累计惩罚。在每个步骤中，智能体越接近充分利用链路的最佳能力，所受到的惩罚就越小。

细心的读者可能会注意到，这种奖励机制会鼓励智能体累积已确认字节，而不考虑其对往返时间的明确影响，从而可能优先选择能在近期产生更多已确认字节的动作，但代价是造成不理想的排队延迟。为避免这种风险，我们考虑了奖励函数的第二种形式，其中引入了一个基于往返时间的惩罚系数：

$$r_t = -\frac{target_t}{target_t + acked_kilobytes_t^{cumulative} * (1 - penalties)} \tag{2}$$

项 $penalties$ 取决于当前往返时间与最小指数移动平均往返时间之间的差值，定义如下：

$$penalties = \begin{cases} \alpha \frac{rtt_{diff}}{rtt_{min}^{ema}}, & \text{if } \frac{rtt_{diff}}{rtt_{min}^{ema}} < 1 \\ 0.99, & \text{otherwise} \end{cases} \tag{3}$$

在公式 3 中，α 取决于 rtt_{diff} 与 rtt_{min}^{ema} 之间差值的大小，其定义如下：

$$\alpha = \begin{cases} 1, & \text{if } |\frac{rtt_{diff}}{rtt_{min}^{ema}}| > 0.6 \\ 0.5, & \text{if } 0.1 < |\frac{rtt_{diff}}{rtt_{min}^{ema}}| \leq 0.6 \\ 0.3, & \text{if } 0.05 < |\frac{rtt_{diff}}{rtt_{min}^{ema}}| \leq 0.1 \\ 0.1, & \text{otherwise} \end{cases} \tag{4}$$

值得注意的是，在 $rtt_{diff} < 0$的情况下， $penalties$项会变为负值，因此当检测到往返时间有所改善时，会对智能体给予奖励。

4）强化学习算法：MARLIN 采用 SAC [11], , 这是一种基于最大熵强化学习框架的离策略演员-评论家算法。SAC 在强化学习设定的基础——最大化奖励目标上，加入了熵最大化项。该项在探索与利用之间起到权衡作用，使智能体在力求最大化回报的同时，尽可能采取随机行为。当多个动作看起来同样具有吸引力时，也就是说，Q 值相等或接近时，学习策略会被鼓励为这些动作分配相等的概率。

在实践中，熵项（也称温度项）的作用会促使智能体舍弃不合适的轨迹，转而选择更有前景的轨迹，同时加快学习速度。随着训练步骤的推进，熵项可以保持固定，也可以进行优化或学习。然而，最优熵系数会因多种因素而异，例如任务的性质，甚至当前策略。因此，通常认为避免使用固定值是良好的实践，而应在优化演员、评论家和目标网络的同时更新该项 [7]。

D. 未来方向

上述工作是一个多年期研究项目的第一步。我们已经确定了后续自然开展的未来工作。虽然我们会依次介绍这些工作，但实际上它们彼此交织，我们很可能会在后续迭代中处理这些工作。

我们认为，MARLIN 可以受益于更具表现力的奖励函数。我们设想采用问题形式和奖励形式，截断那些偏离最优解过远、前景不佳的轨迹。我们相信，这能够显著加快解的收敛。此外，我们怀疑 MARLIN 当前的状态可能包含冗余信息，以及与问题相关性较低的特征。为此，我们计划研究更精简、更完善的状态表示，双管齐下地降低复杂度并提升收敛速度。我们计划

超参数	值
训练步数	1×10^6
历史长度	10
训练回合长度	200
学习率	3×10^{-4}
缓冲区大小	5×10^5
预热（开始学习）	1×10^4 步
批量大小	512
Tau	5×10^{-3}
Gamma	0.99
训练频率	$1/episode$
梯度步数	-1 （与回合长度相同）
熵正则化系数	“自动”（学习得到）
多层感知机策略隐藏层	$[400, 300]$

表 II：MARLIN 中使用的超参数。

在多样化的网络场景中训练这一新智能体，这些场景能够涵盖不同的流量模式和网络技术，以评估其泛化能力。最后，全面、自动化的超参数调优将进一步提升 MARLIN 的性能，并完成第一轮改进。

进一步发展需要改进智能体的设计，使其与拥塞控制面临的特定问题相契合。相关文献表明，机器学习模型能够准确区分由拥塞或信道错误导致的丢包。我们计划在 MARLIN 中集成类似的分类器，并研究采用类似方法识别由通往目的地的路径变化所引起的端到端时延变化是否可行。拥塞控制算法的基本构件是如何以及何时调整拥塞窗口大小。目前，MARLIN 主动控制“如何”调整，而将“何时”交由启发式方法决定。我们将研究基于学习的方法，把这一决策因素纳入 MARLIN，目标是使其成为一个更全面、更具响应性的系统，能够按照自身节奏做出合理决策。

迄今为止，我们一直从单个强化学习智能体的角度来构建拥塞控制问题。然而，传输协议需要拥塞控制算法，根源在于多智能体系统中缺乏协调：各个独立实体的行为完全以自身利益为中心。因此，我们认为，基于学习的拥塞控制下一步将是应用不断发展的多智能体强化学习算法，以优化智能体的协作和/或竞争行为。

III. Mockets 传输协议

我们面临的主要设计选择之一，是采用哪种传输协议来训练和评估 MARLIN。最终我们选择了自定义传输协议，因为它比 TCP 更容易与 MARLIN 集成。此外，自定义协议在获取编码智能体状态所需的信息方面具有更大的灵活性。

在本研究中，我们将 MARLIN 集成到 Mockets 中。Mockets 是一种基于 UDP 实现的消息型通信中间件，其设计理念与

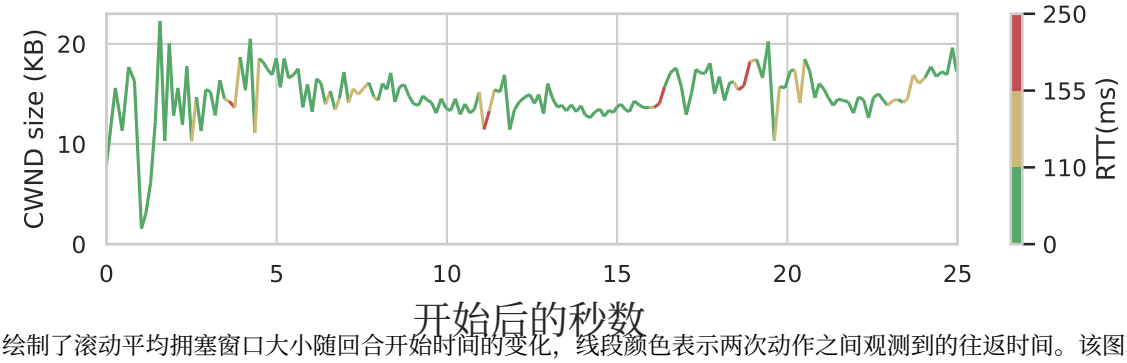


图 3：图中绘制了滚动平均拥塞窗口大小随回合开始时间的变化，线段颜色表示两次动作之间观测到的往返时间。该图取自第二个 MARLIN 模型（即 5b）的最佳运行结果，该模型在 24.84 秒内完成了传输。

QUIC 协议的设计理念相似 [25] （有关 Mockets 的更多详情，请参阅 [26]）。Mockets 实现了一种非常激进的拥塞控制算法，旨在通信延迟不断变化且丢包率较高时，充分利用可用带宽。尽管 Mockets 在网络环境恶劣时相比 TCP 和其他传输协议具有多项优势 [27]，，但其现有的拥塞控制机制无法与通过同一链路的其他通信流共享链路容量，也无法快速适应可用带宽的变化。因此，MARLIN 能为 Mockets 带来显著益处，使其也成为传统网络场景中的可行选择。

为了准确表示 MARLIN 中环境的状态，我们改进了 Mockets 中的往返时间估算。具体而言，发送端会跟踪每个数据包的发送时间。当收到确认时，发送端通过计算当前时间与接收方在生成确认消息前收到的最后一个数据包的发送时间之差，再减去确认处理时间，来计算往返时间。为了进行此项计算，接收方会在所有确认消息中附加最后收到的数据包标识符（严格递增的无符号整数）以及处理时间（以微秒为单位）。

IV. 实验结果

A. 所用硬件

图 2 所示的测试平台采用商用现成设备搭建。智能体在一台配备 Intel i7-8700 CPU、32GB DDR4 内存、NVIDIA GeForce RTX 2060 GPU 和 Intel Wireless-AC 9560 网络接口卡（NIC）的工作站上进行训练。“接收端应用程序”运行在一台 4GB 的 Raspberry Pi 4 Model B (RPi4B) 上。使用的两台 AP 分别是 TP-Link EAP245 和 Netgear WAC505，并连接到两台 Netgear GS108E 交换机。一台运行 macOS Catalina v10.15.6 的 2015 款 Apple MacBook Pro 生成背景流量，并将其发送到运行在 Dell Latitude 7000 笔记本电脑 (E7470) 上的接收端应用程序。台式机和 E7470 均运行 Ubuntu 20.04.5 LTS 操作系统。所有系统均按默认设置启用了所有与 NIC 相关的优化，例如 TCP 分段卸载 (TSO) 或大型接收卸载（LRO）。一台运行 EdgeOS 的 Ubiquiti Networks EdgeRouterX 路由器负责在两个子网之间路由数据包。

B. 训练

智能体在一个无限时域任务上训练 1M 步，每个部分回合持续 200 步。遇到部分回合的最后一步时，会发出截断信号并执行 PEB，以提醒智能体此后实际上还可以继续执行更多步并获得奖励 [12]。

受时间限制，优化步骤（针对演员、评论家、目标网络和学习得到的熵系数）在每个部分回合结束时执行。在真实网络上执行两个动作之间的训练步数，是我们无法承受的奢侈，因为我们观察到这会引入约 800–1000 ms 的延迟。在回合结束时训练可以避免这种影响，将开销限制在流水线本身；据我们报告，其耗时约为 25 至 30 ms。

在训练开始时，为了增加探索，先从均匀随机分布中抽取动作，持续 10K 步。随后，SAC 开始采用由其熵系数引导的常规探索—利用策略。

我们为 MARLIN 训练了三个不同的模型。对于第一个模型，我们使用了公式 1（5a）。对于第二个模型，我们使用了公式 2（5b）中的奖励函数。在这两种情况下，背景流量模式都是固定的，大象流按照以下顺序排列： [100 UDP、200 TCP、100 UDP、50 UDP]。对于最后一个模型（5c），我们将背景流量改为使用第二-B节中介绍的所有排列。

如图4所示，在三种设置中，均值训练奖励都呈现出令人鼓舞的上升趋势。以我们的理想目标作为参考，可以直观地解读训练期间表现出的性能；在本例中，该目标应产生 -100 的均值训练奖励。

C. 结果

训练结束后，我们在同一网络上使用背景流量模式 [100 UDP、200 TCP、100 UDP、50 UDP] 传输一个 3MB 文件，以测试全部三个 MARLIN 模型。需要注意的是，文件传输与智能体在训练过程中面临的问题有很大不同：

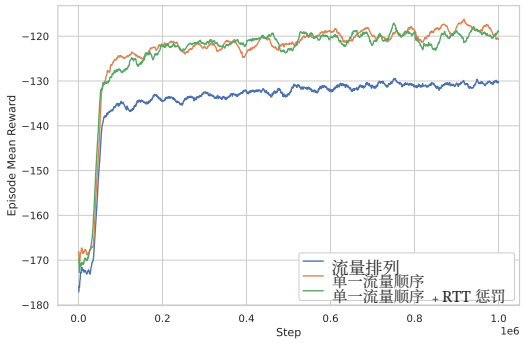


图4：三次训练运行中最后 100 个部分回合的均值训练奖励。

在这里，评估智能体能否完成一项时长超过训练期间所见部分回合的任务；训练期间的部分回合最多为 200 步，而完成得越快，结果就越好。

我们对三种训练后的智能体分别重复进行了 100 次实验。随后，将结果与安全复制协议（SCP）的性能进行比较。在我们的系统中，SCP 使用 TCP CUBIC 作为传输协议，并执行相同的文件传输任务。图 5 展示了根据背景流量的先验知识计算得出的最优行为，以及从训练后的智能体获得的一批结果。在相同条件下，同样重复传输 100 次后，还展示了 TCP CUBIC 的平均、最佳和最差性能。图 3 展示了 MARLIN 在三次实验中最佳运行期间拥塞窗口大小的变化。每个线段的颜色表示通信的往返时间，该往返时间是在使拥塞窗口大小达到线段最右端所表示数值的动作之后，紧接着的一个平滑往返时间周期内测得的。

出现显著性能下降的运行在 80 秒后中止，其中 5a 报告有 4 次实验中止，5b 有 20 次，5c 有 2 次。TCP CUBIC 平均用时 45.03 秒完成文件传输，最快用时为 33.97 秒，最慢则为 70.78 秒。排除中止的运行后，MARLIN 在 5a、5b 和 5c 中到达目标的平均用时分别为 46.1 秒、50.31 秒和 46.9 秒。所有智能体的最佳表现都快于 TCP CUBIC 的最佳表现。在 5a 中，48% 的实验表现与 TCP CUBIC 的平均表现持平或更好；5b 中这一比例为 31%，5c 中为 47%。最快的传输由 5b 完成，用时 24.84 秒，比 SCP 的最快传输快 27%，比其平均用时快 45%。作为比较，该链路的可用带宽足以在 25 秒内完成文件传输（见图 5）；如果 MARLIN 注入的流量导致其他流变慢，传输仍可能更快。

D. 讨论

尽管在像网络中的拥塞控制这样复杂的问题中，连续动作增加了决策复杂度，而训练预算为 100 万步，与其他基于强化学习的拥塞控制智能体相比并不算高，但结果仍然令人鼓舞，在我们的环境中已经可以媲美 TCP CUBIC。我们认为，这部分归功于软演员-评论家的样本效率。另一个原因可能在于将拥塞控制问题构建为一个奖励始终为负的无限时域任务，这使智能体能够将习得的经验应用于比训练期间所见部分回合更长、目标也不同的任务。

在训练期间对背景流量模式的顺序进行置换，并未导致评估期间的性能下降。事实上，与图 5a 所示的实验相比，评估运行的方差更低，最快传输速度更快，平均传输时间相近。此外，需要中止的实验也更少。这些结果表明，通过向算法提供来自多种场景的数据，或许能够进一步提升 MARLIN 的鲁棒性。

图 3 展示了一次评估运行期间的拥塞窗口大小。请注意，大多数情况下，在检测到往返时间增加后，智能体已学会在下一步通过减小拥塞窗口大小或降低其增长速度作出响应。这种行为与许多基于往返时间的拥塞控制算法相符。

最后，尽管评估期间获得的大多数轨迹都呈现出有希望且有价值的结果（图 5），但也表现出明显的不稳定性，值得进一步研究。这种行为在图 5b 中尤为明显，其中模型使用式（2）中的函数进行训练。与图 5a 和图 5c 中的模型以及使用 SCP 获得的结果相比，该模型有几次传输速度明显更快；然而，该模型的平均速度却慢得多。

五、相关工作

过去，TCP拥塞控制的设计与优化研究与本文所探讨的方法大相径庭。传统拥塞控制算法旨在充分利用瓶颈链路，通过采用各种启发式策略不断增大拥塞窗口大小，直到出现拥塞信号，例如丢包（如 NewReno [28]和 CUBIC [29]，后者是现代 Linux内核和近期 Windows操作系统中的默认拥塞控制算法），或延迟增加（如 Vegas [30] 以及较新的 BBR [31]）。事实证明，这些方法在特定网络条件下效果良好，但在其他场景中表现不佳，或会遇到其他类型的问题 [32, 33, 34]。

由于网络环境具有极高的异质性、复杂性和内在动态性，加之过去二十年间科学界对机器学习重燃兴趣，近期已有多项研究着眼于基于学习的传输协议优化技术。有些方法专注于解决传输协议中的某个特定问题，例如准确识别拥塞事件，但并不旨在取代拥塞控制算法。例如，研究人员已成功构建基于机器学习的模型，能够区分由拥塞导致的丢包

与无线网络中的信道错误导致的丢包 [35, 36] ，或光突发交换网络中由介质竞争导致的丢包 [37]。正如上文所述，我们预计此类解决方案也会使我们的方法受益；但 MARLIN 的最终目标是训练一个智能体，使其能够在多种不同的网络场景中高效承担拥塞控制算法的任务。

更有意思的研究尝试将机器学习模型和强化学习智能体直接应用于拥塞控制，通过学习策略来优化调整发送速率 [38], 、拥塞窗口 [39], 或其他用于调节拥塞控制算法的参数 [40]。文献 [9, 10] 对机器学习在拥塞控制中的应用进行了全面综述。综述中提到的研究面临的常见问题包括参数选择、计算复杂度高、内存消耗大、训练效率低，以及与现有拥塞控制启发式方法的兼容性和公平性问题。

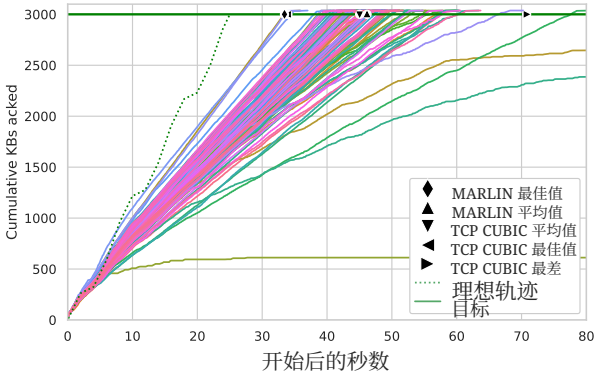
强化学习智能体能够完全自主地进行训练，因此将强化学习应用于拥塞控制优化具有很强的吸引力。OpenAI Gym 等广泛使用的强化学习库具有阻塞特性，这些库旨在解决 ATARI 游戏等强化学习问题 [41], ，在等待下一个动作时可能会被阻塞，这构成了主要挑战之一。研究人员通过在模拟环境中训练智能体来绕过这一障碍，并取得了非常有前景的成果，例如 DRL-CC [42] 和 Aurora [43]。然而，要在真实世界的网络中复现这些成果已被证明颇具挑战。依赖于在智能体计算下一个动作时进行阻塞的解决方案会引入显著延迟，从而损害整体通信性能，尤其是在高速网络中。相比之下，我们设计的 MARLIN 可与 Mockets 异步集成，避免传输协议在等待智能体采取下一个动作时发生阻塞。

其他研究人员提出使用非阻塞智能体来优化拥塞控制。MVFST-RL [16] 提出了一种基于 IMPALA 的非阻塞智能体， [44], 这是一种用于 QUIC 传输协议的 C++ 实现；Pantheon [39] 则用于网络仿真。智能体与系统之间的通信方式与 Park [17], 类似。Park 是一个基于 RPC、用于在计算机系统问题上试验强化学习智能体的平台。MARLIN 遵循类似的理念，以避免阻塞系统的缺点。然而，据我们所知，MARLIN 是唯一采用严格负奖励，并研究使用离策略、熵正则化强化学习算法（例如软演员-评论家）及连续动作空间的工作；该算法在真实网络上训练，其中真实的背景流量会竞争带宽。此外，我们在无限时域设置下训练了 MARLIN，并在常见的真实世界问题上评估了该模型，例如通过共享链路传输文件。

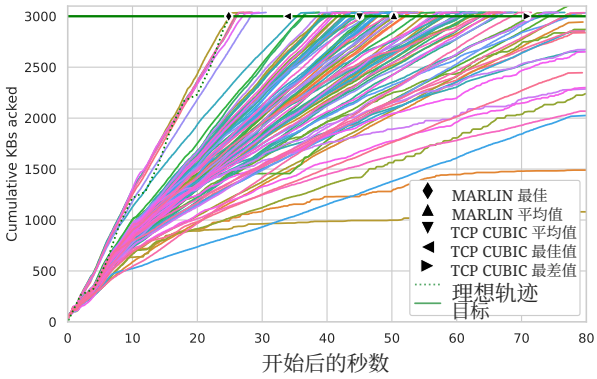
VI. 结论

本研究展示了如何通过训练基于离策略、熵正则化算法（如软演员-评论家）的强化学习智能体，获得有效策略。MARLIN 将拥塞控制问题构建为一个奖励始终为负的任务，并在存在竞争性动态背景流量的真实网络中执行连续动作。

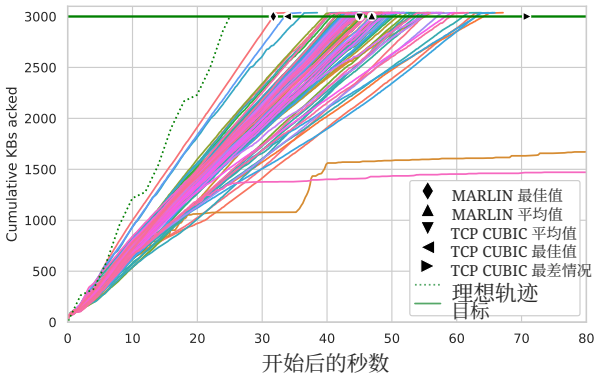
我们还提出了计划开展的未来研究方向，包括在更多异构环境中进行训练、探索多智能体强化学习设置、研究表达能力更强的奖励函数，以及设计一种能够自主决定何时采取下一步动作的智能体。



(a) 在单一流量模式下训练的智能体。



(b) 在单一流量模式下训练并采用 RTT 惩罚的智能体。



（c）在交通流的每种排列组合上训练的智能体。

图 5：每个模型在 100 次测试实验中的评估。

参考文献

[1] Josip Lorincz、Zvonimir Klarin 和 Julije Ože- govi´c。《5G网络中 TCP 拥塞控制的全面概述：研究挑战与未来展望》。载于：Sensors 21.13（2021年）。ISSN：1424-8220。DOI：10.3390/s21134510。Kormushev。《强化学习中的时间限制》。载于：（2017）。DOI：10.48550/ARXIV.1712.00378。URL：https://arxiv.org/abs/1712.00378。[13] Martin L. Puterman。《马尔可夫决策过程：离散随机动态规划》。第1版。美国：John URL：https://www.mdpi.com/1424-8220/21/13/4510。[2] Shiyao Ma、Jingjie Jiang、Wei Wang 和 Bo-chen Li。《基于拥塞的拥塞控制的公平性：实验评估与分析》。载于：arXiv：网络与互联网架构（2017）。[3] J. Widmer、R. Denda 和 M. Mauve。《TCP友好型拥塞控制综述》。载于：IEEE Network 15.3（2001），第28–37页。DOI：10.1109/65.923938。[4] Raffaele Galliera 和 Niranjan Suri。《边缘目标检测：现成的深度学习设备与加速器》。载于：Procedia Computer Science 205（2022）。2022年国际军事通信与信息系统会议（ICMCIS），第239–248页。ISSN：1877-0509。DOI：https://doi.org/10.1016/j.procs.2022.09.025。URL：https://www.sciencedirect.com/science/article/pii/S1877050922008900。[5] Nikita Rudin、David Hoeller、Philipp Reist 和 Marco Hutter。《利用大规模并行深度强化学习，在几分钟内学会行走》。Wiley & Sons, Inc.，1994。ISBN：0471619779。[14] Richard S. Sutton 和 Andrew G. Barto。《强化学习：导论》。美国马萨诸塞州剑桥：A Bradford Book, 2018。ISBN：0262039249。[15] Lei Zhang 等。《真实环境中的基于强化学习的拥塞控制》。载于：2020年第29届国际计算机通信与网络会议（ICCCN）。2020年，第1–9页。DOI：10.1109/ICCCN49398.2020.9209750。[16] Viswanath Sivakumar 等。《MVFST-RL：一种支持延迟动作的异步拥塞控制强化学习框架》。载于：（2019年10月）。URL：http://arxiv.org/abs/1910.04054。[17] Hongzi Mao 等。《Park：一个用于学习增强型计算机系统的开放平台》。载于：神经信息处理系统进展。H. Wallach 等编。第32卷。Curran Associates, Inc.，2019年。URL：https://proceedings.neurips.cc/paper/2019/file/f69e505b08403ad22 ing。2021年。DOI：10.48550/ARXIV.2109.11978。URL：https://arxiv.org/abs/2109.11978。[6] OpenAI：Marcin Andrychowicz 等。《学习灵巧的手内操作》。载于：The International Journal of Robotics Research 39.1（2020），第3–20页。DOI：10.1177/0278364919887447。eprint：https://doi.org/10.1177/0278364919887447。URL：https://doi.org/10.1177/0278364919887447。[7] Tuomas Haarnoja 等。《软演员-评论家算法及其应用》。2018年。DOI：10.48550/ARXIV.1812.05905。URL：https://arxiv.org/abs/1812.05905。[8] Tobias Jacob.、Raffaele Galliera.、Muddasar Ali. 和 Sikha Bagui。《使用单目相机进行船舶跟踪》。载于：第2届国际深度学习理论与应用会议论文集（DeLTA），INSTICC。SciTePress，2021年，第17–28页。ISBN：978-989-758-526-5。DOI：10.5220/0010516000170028。[9] Wenting Wei、Huaxi Gu、Baochun Li 和 Kai Chen。《拥塞控制：机器学习带来的复兴》。载于：IEEE Network 35（2021年7月4日），第262–269页。ISSN：1558-156X。DOI：10.1109/MNET.011.2000603。[10] Huiling Jiang 等。《当机器学习遇上拥塞控制：综述与比较》。2021年6月。DOI：10.1016/j.comnet.2021.108033。[11] Tuomas Haarnoja、Aurick Zhou、Pieter Abbeel 和 Sergey Levine。《软演员-评论家：具有随机演员的离策略最大熵深度强化学习》。载于：CoRR abs/1801.01290（2018）。arXiv：1801.01290。URL：http://arxiv.org/abs/1801.01290。

[12] Fabio Pardo、Arash Tavakoli、Vitaly Levdik 和 Petar Kormushev。《强化学习中的时间限制》。载于：（2017）。DOI：10.48550/ARXIV.1712.00378。URL：https://arxiv.org/abs/1712.00378。[13] Martin L. Puterman。《马尔可夫决策过程：离散随机动态规划》。第1版。美国：John URL：https://www.mdpi.com/1424-8220/21/13/4510。[2] Shiyao Ma、Jingjie Jiang、Wei Wang 和 Bo-chen Li。《基于拥塞的拥塞控制的公平性：实验评估与分析》。载于：arXiv：网络与互联网架构（2017）。[3] J. Widmer、R. Denda 和 M. Mauve。《TCP友好型拥塞控制综述》。载于：IEEE Network 15.3（2001），第28–37页。DOI：10.1109/65.923938。[4] Raffaele Galliera 和 Niranjan Suri。《边缘目标检测：现成的深度学习设备与加速器》。载于：Procedia Computer Science 205（2022）。2022年国际军事通信与信息系统会议（ICMCIS），第239–248页。ISSN：1877-0509。DOI：https://doi.org/10.1016/j.procs.2022.09.025。URL：https://www.sciencedirect.com/science/article/pii/S1877050922008900。[5] Nikita Rudin、David Hoeller、Philipp Reist 和 Marco Hutter。《利用大规模并行深度强化学习，在几分钟内学会行走》。Wiley & Sons, Inc.，1994。ISBN：0471619779。[14] Richard S. Sutton 和 Andrew G. Barto。《强化学习：导论》。美国马萨诸塞州剑桥：A Bradford Book, 2018。ISBN：0262039249。[15] Lei Zhang 等。《真实环境中的基于强化学习的拥塞控制》。载于：2020年第29届国际计算机通信与网络会议（ICCCN）。2020年，第1–9页。DOI：10.1109/ICCCN49398.2020.9209750。[16] Viswanath Sivakumar 等。《MVFST-RL：一种支持延迟动作的异步拥塞控制强化学习框架》。载于：（2019年10月）。URL：http://arxiv.org/abs/1910.04054。[17] Hongzi Mao 等。《Park：一个用于学习增强型计算机系统的开放平台》。载于：神经信息处理系统进展。H. Wallach 等编。第32卷。Curran Associates, Inc.，2019年。URL：https://proceedings.neurips.cc/paper/2019/file/f69e505b08403ad22 ing。2021年。DOI：10.48550/ARXIV.2109.11978。URL：https://arxiv.org/abs/2109.11978。[6] OpenAI：Marcin Andrychowicz 等。《学习灵巧的手内操作》。载于：The International Journal of Robotics Research 39.1（2020），第3–20页。DOI：10.1177/0278364919887447。eprint：https://doi.org/10.1177/0278364919887447。URL：https://doi.org/10.1177/0278364919887447。[7] Tuomas Haarnoja 等。《软演员-评论家算法及其应用》。2018年。DOI：10.48550/ARXIV.1812.05905。URL：https://arxiv.org/abs/1812.05905。[8] Tobias Jacob.、Raffaele Galliera.、Muddasar Ali. 和 Sikha Bagui。《使用单目相机进行船舶跟踪》。载于：第2届国际深度学习理论与应用会议论文集（DeLTA），INSTICC。SciTePress，2021年，第17–28页。ISBN：978-989-758-526-5。DOI：10.5220/0010516000170028。[9] Wenting Wei、Huaxi Gu、Baochun Li 和 Kai Chen。《拥塞控制：机器学习带来的复兴》。载于：IEEE Network 35（2021年7月4日），第262–269页。ISSN：1558-156X。DOI：10.1109/MNET.011.2000603。[10] Huiling Jiang 等。《当机器学习遇上拥塞控制：综述与比较》。2021年6月。DOI：10.1016/j.comnet.2021.108033。[11] Tuomas Haarnoja、Aurick Zhou、Pieter Abbeel 和 Sergey Levine。《软演员-评论家：具有随机演员的离策略最大熵深度强化学习》。载于：CoRR abs/1801.01290（2018）。arXiv：1801.01290。URL：http://arxiv.org/abs/1801.01290。

[18] Ubiquiti：EdgeOS。网址：https://dl.ubnt.com/guides/edgemax/EdgeOS_UG.pdf。[19] 美国海军研究实验室（NRL）协议工程高级网络（PROTEAN）研究组。多生成器（MGEN）网络测试工具。《使用单目相机进行船舶跟踪》。载于：第2届国际深度学习理论与应用会议论文集（DeLTA），INSTICC。SciTePress，2021年，第17–28页。ISBN：978-989-758-526-5。https://www.nrl.navy.mil/Our-Work/Areas-of-Research/Information。[20] Anonin Raffin。《Stable Baselines3 zoo》。2020年。网址：https://github.com/DLR-RM/stable-baselines3。[21] Antonin Raffin 等人。《Stable Baselines3 - 可靠的强化学习实现》。载于：《机器学习研究杂志》22.268（2021年），第1–8页。网址：http://jmlr.org/papers/v22/20-1364.html。[22] Adam Paszke 等人。《PyTorch：一种命令式风格的高性能深度学习库》。载于：第33届神经信息处理系统国际会议论文集。美国纽约州雷德胡克：Curran Associates Inc.，2019年。[23] Greg Brockman 等人。OpenAI Gym。2016年。预印本：arXiv:1606.01540。[24] Nathan Jay、Noga Rotman、Brighten Godfrey、Michael Schapira 和 Aviv Tamar。《从深度强化学习视角看互联网拥塞控制》。载于：第36届国际机器学习会议论文集。Kamalika Chaudhuri 和 Ruslan Salakhutdinov 编。第97卷。机器学习研究论文集。PMLR，2019年9月，第3050–3059页。网址：https://proceedings.mlr.press/v97/jay19a.html。[25] Martin Thomson、Jana Iyengar。《QUIC：一种基于UDP 的多路复用安全传

输协议》。RFC 9000。互联网工程任务组，2022年2月。网址：https://datatracker.ietf.org/doc/rfc9000/。[26] Erika Benvegnù、Niranjan Suri、Mauro Tortonesi 和 Tomás Esterrich。《使用 Mockets 通信中间件实现无缝网络迁移》。载于：2010 - MILCOM 2010 军事通信会议。2010年，第2298–2303页。DOI：10.1109/MILCOM.2010.5680364。[27]Alessandro Morelli、Michel Provosty、Roberto Fronteddu 和 Niranjan Suri。《战术网络环境中的传输协议性能评估》。载于：MILCOM 2019 - 2019年 IEEE 军事通信会议（MILCOM）。2019年，第30–36页。DOI：10.1109/MILCOM47813.2019.9021047。[28] Andrei Gurtov、Tom Henderson、Sally Floyd 和 Yoshifumi Nishida。《TCP 快速恢复算法的 NewReno 修订版》。RFC 6582。2012年4月。DOI：10.17487/RFC6582。网址：https://www.rfc-editor.org/info/rfc6582。[29] Sangtae Ha、Injong Rhee 和 Lisong Xu。《CUBIC：一种新的 TCP 友好型高速 TCP 变体》。载于：SIGOPS 操作系统评论 42.5（2008年7月），第64–74页。ISSN：0163-5980。DOI：10.1145/1400097.1400105。网址：https://doi.org/10.1145/1400097.1400105。[30] Lawrence S. Brakmo、Sean W. O'Malley 和 Larry L. Peterson。《TCP Vegas：拥塞检测与避免的新技术》。载于：SIGCOMM。1994年。[31] Neal Cardwell、Yuchung Cheng、C. Stephen Gunn、Soheil Hassas Yeganeh 和 Van Jacobson。《BBR：基于拥塞的拥塞控制》。载于：ACM Queue 14，9–10月（2016年），第20–53页。网址：http://queue.acm.org/detail.cfm?id=3022184。[32] Philipp Bruhn、Mirja Kuehlewind 和 Maciej Muehleisen。《低时延蜂窝网络中 TCP CUBIC 的性能与改进》。载于：2022年 IFIP 网络会议（IFIP Networking）。2022年，第1–9页。DOI：10.23919/IFIPNetworking55013.2022.9829781。[33] Marko Šoši´c 和 Vladimir Stojanovi´c。《解决高速长距离链路上的 TCP 性能不佳问题——BIC、CUBIC 和 Hybla 的概述与比较》。载于：2013年 IEEE 第11届智能系统与信息学国际研讨会（SISY）。2013年，第325–330页。DOI：10.1109/SISY.2013.6662595。[34] Phuong Ha、Minh Vu、Tuan-Anh Le 和 Lisong Xu。《云网络中的 TCP BBR：挑战、分析与解决方案》。载于：2021年 IEEE 第41届分布式计算系统国际会议（ICDCS）。2021年，第943–953页。DOI：10.1109/ICDCS51616.2021.00094。[35] Kimoon Han、Jae Yong Lee 和 Byung Chul Kim。《用于无线 TCP 拥塞控制的基于机器学习的丢包判别算法》。载于：2019年国际电子、信息与通信（ICEIC）。2019年，第1–2页。DOI：10.23919/ELINFOCOM.2019.8706382。

[36] P. Geurts、I. El Khayat 和 G. Leduc。"一种改进无线计算机网络拥塞控制的机器学习方法"。载于：第四届 IEEE 数据挖掘国际会议（ICDM'04）。2004 年，第 383–386页。DOI：10.1109/ICDM.2004.10063。

[37] A. Jayaraj、T. Venkatesh 和 C. Siva Ram Murthy。"使用机器学习技术对光突发交换网络中的丢包进行分类：提升 TCP 的性能"。载于：IEEE《通信领域精选期刊》26.6（2008年），第45–54页。DOI：10.1109/JSACOCN.2008.033508。

[38] Mo Dong、Qingxi Li、Doron Zarchy、P. Brighten God- frey 和 Michael Schapira。"PCC：重新架构拥塞控制，以实现稳定的高性能"。载于：第 12 届 USENIX 网络系统设计与实现会议论文集。NSDI'15。加利福尼亚州奥克兰：USENIX协会，2015 年，第 395–408 页。ISBN：9781931971218。

[39] Francis Y. Yan 等。"Pantheon：互联网拥塞控制研究的训练场"。载于：2018 年 USENIX年度技术会议（USENIX ATC 18）。马萨诸塞州波士顿：USENIX协会，2018 年 7 月，第 731–743 页。ISBN：978-1-939133-01-4。URL：https://www.usenix.org/conference/atc18/presentation/yan-fr ancis。

[40]Keith Winstein 和 Hari Balakrishnan。"TCP Ex Machina：计算机生成的拥塞控制"。载于：SIGCOMM Comput. Commun.Rev. 43.4（2013 年 8 月），第 123–134 页。ISSN：0146- 4833。DOI：10.1145/2534169.2486020。URL：https://doi.org/10.1145/2534169.2486020。

[41] VolodymyrMnih 等。"使用深度强化学习玩雅达利游戏"。载于：CoRR abs/1312.5602（2013）。arXiv：1312.5602。网址：http://arxiv.org/abs/1312.5602。

[42] ZhiyuanXu、Jian Tang、Chengxiang Yin、Yanzhi Wang 和 Guoliang Xue。"经验驱动的拥塞控制：当多路径TCP遇上深度强化学习"。载于：IEEE通信领域精选期刊 37.6 （2019 年），第 1325–1336 页。DOI：10.1109/JSAC.2019.2904358。

[43] Nathan Jay、Noga Rotman、Brighten Godfrey、Michael Schapira 和 Aviv Tamar。"从深度强化学习视角看互联网拥塞控制"。载于：第36届国际机器学习会议论文集。Kamalika Chaudhuri 和 Ruslan Salakhutdinov 编。第97卷。机器学习研究论文集。PMLR，2019年9月，第3050–3059页。网址：https://proceedings.mlr.press/v97/jay19a.html。

[44] Lasse Espeholt 等人：《IMPALA：采用重要性加权演员-学习器架构的可扩展分布式深度强化学习》。载于：CoRR abs/1802.01561（2018）。arXiv：1802.01561。网址：http://arxiv.org/abs/1802.01561。