人工智能（研究生）课程考核报告封面

<table>
<tr><td>院　　系</td><td colspan="2">计算机学院</td><td>专业</td><td colspan="3">计算机科学与技术</td></tr>
<tr><td>学生姓名</td><td colspan="2">尹达恒</td><td>学号</td><td colspan="3">201857</td></tr>
<tr><td>课程名称</td><td colspan="6">人工智能</td></tr>
<tr><td>授课时间</td><td colspan="2">2020 年 9 月至 2020 年 12 月</td><td>学时</td><td>32</td><td>学分</td><td>2</td></tr>
<tr><td>考核论题</td><td colspan="6">按要求撰写报告
1. 读书报告（见试卷纸试题一）。
2. 人工智能应用设计报告（见试卷纸试题二）。
提交说明：
1. 以此文档作为报告封面的正反面（每个人的两个报告装订在一起，用一个封面）；
2. 报告正文采用五号字体，除封面之外，其余各页正反打印；
3. 打印后，2020.2.25 前交给负责同学，统一提交给老师。</td></tr>
<tr><td>简
要
评
语</td><td colspan="6">完成度（分级 5、10、15、20）
规范性（分级 5、10、15、20）
条理性（分级 5、10、15、20）
应用场景具体程度（分级 5、10、15、20）
技术细致度和可行性（分级 5、10、15、20）
雷同或非自主设计程度（分级-0、-20、-40、-50）</td></tr>
<tr><td>总评成绩
（含平时成绩）</td><td colspan="6"></td></tr>
<tr><td>备注</td><td colspan="6"></td></tr>
</table>

任课教师签名：_____

日期： 2021.3.1

东 南 大 学 试 题 纸

课程 人工智能　　2020 —2021 学年第一学期

学 号 201857　　姓 名 尹达恒　　得 分＿＿＿＿＿＿

（本试卷共 2 页）

一、　读书报告（50 分）

1.从《Artificial Intelligence——A Modern Approach》(3e)中选择 1 章以上的内容进行详细深入的阅读，并撰写读书报告，要求列出

- 知识点
- 技术要点和算法，
- 对每一个技术/算法分析说明其适合的任务环境并给出理由。

二、　人工智能应用设计报告（50 分）

1．内容应包括：题目、摘要、主题词、正文、主要参考文献。

2．技术报告正文部分字数不少于４０００字，包括 8 个部分：

- 场景的描述，要具体到某个实际想定情景示例上；
- 智能化任务，说明该示例场景中可智能化解决的环节，及具体功能和非功能要求；
- 任务环境分析，给出 PEAS 和性质的详细分析；
- 智能 Agent 结构，给出适合的 Agent 结构，及具体模块的结构；
- 问题，说明上述智能化任务中面临的主要技术问题和非技术问题；
- 现状，上述问题的已有解决方法及其优缺点分析；
- 技术方案，给出采用一个课本中现成方法和算法的优缺点分析，结合上述优缺点分析，进一步给出本文的具体技术和算法方案；
- 方案分析，给出本文方案实际应用中的可行性分析。

3．题目自拟，建议从所了解的课题或日常生活中选择一个应用场景，以“面向 XXX 的智能 Agent 设计和关键技术”为题。

评分标准：

1. 见评语选项及分数等级；

《Artificial Intelligence——A Modern Approach》第 21 章强化学习读书报告

尹达恒
（东南大学，江苏　南京）

1　引言

1.1　何谓之“强化学习”?

1、“状态”：Agent 所处的环境的状况；

2、“动作”：Agent 对环境进行某种操作，进而使得“状态”发生改变；

3、“强化”，或称“回报”：Agent 从环境中获得什么是“好情况”和什么是“坏情况”的反馈；

4、“强化学习”：Agent 利用观察到的“回报”来学习针对某个“状态”的最优“动作”。

1.2　强化学习的应用领域

在许多复杂领域中，人很难对大量的数据给出精确一致的评价，例如在棋类运动中，人很难评价每种棋局的好坏。替代地，人可以告诉机器什么时候赢了或输了，使机器能运用类似的信息来学习评价函数，进而对从任何给定的棋局出发对获胜的概率进行精确的估计。

1.3　强化学习的 Agent 简单分类

1、 基于“效用”的 Agent：学习关于状态的效用函数，选择到达效用最大的状态的动作；

2、 基于“Q-learning”的 Agent：学习在给定状态下采取特定动作所产生的效用的函数（Q 函数），选择效用最大的动作；

3、 基于“反射”的 Agent：学习一组将状态直接映射到动作的策略。

1.4　强化学习的目标

对于系统的每个状态 $S_t(t \in \mathbb{N}_+)$，都对应一个回报值 $R(S_t) \in \mathbb{R}$；当采取某个策略 π 时，与非终止状态相关联的效用期望值可以定义为：

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right] \quad (S_0 = s)$$

强化学习的目标就是要学习这个效用函数的输出与状态每个 s 之间的对应关系

2 被动强化学习

2.1 直接效用估计

直接效用估计的核心思想是：一个状态的回报是从该状态往后的总回报期望。即贝尔曼方程：

$$U^*(s) = R(s) + \gamma \sum_{s'} P(s'|s, \pi(s))U^*(s')$$

在学习过程中，每次试验对于每个访问到的状态提供了一个样本每个样本都以状态为输入，以观察到的未来回报为输出，进而将强化学习问题简化为标准的归纳学习问题。

然而，直接效用估计只有在序列的最后才能计算经过的每个状态的未来回报，因为它没有考虑到不同状态效用之间的联系，而将效用视为孤立的值。实际上，从强化学习的目标可以明显看出，一个状态的效用值与其后所有状态的效用值都有关。从这个层面上讲，直接效用估计错失了很多学习的机会，直到序列的最后才能开始学习，因而收敛很慢。

2.2 自适应动态规划

自适应动态规划 Agent 通过学习连接状态的转移模型，并使用动态规划方法求解马尔可夫决策过程，以利用效用之间的约束。相当于把学到的转移模型 $P(s'|s, \pi(s)$ 以及观察到的回报 $R(s)$ 带入到贝尔曼方程中，以计算状态的效用。

由于环境的完全可观察性，学习模型本身的过程是容易的。这意味着我们面临一个有监督的学习任务，其输入是一个状态行动对，输出是结果状态，具体地，是对在状态 s 执行动作 a 后能够到达状态 s' 的转移概率 $P(s'|s, a)$ 的估计值。

对转移概率 $P(s'|s, a)$ 的估计有两种方法，分别对应于基于贝叶斯方法的强化学习和基于鲁棒控制理论的强化学习。

2.2.1 基于贝叶斯理论的强化学习

假设 u_h^π 是通过在模型 h 中执行策略 π 而获得的期望效用，$P(h|\boldsymbol{e})$ 是环境中出现 h 的概率，通常通过给定目前为止的观察用贝叶斯规则获得，那么基于贝叶斯理论的强化学习可以表示为：

$$\pi^* = \underset{\pi}{argmax} \sum_h P(h|\boldsymbol{e})u_h^\pi$$

2.2.2 基于鲁棒控制理论的强化学习

基于鲁棒控制理论的强化学习所输出的策略为在最坏情况下输出的最好策略：

$$\pi^* = \underset{\pi}{argmax}\ \underset{h}{min}\ u_h^\pi$$

2.3 时序差分学习

与求解贝尔曼方程的求解方法不同，时序差分学习使用观察到的转移来调整观察到的状态的效用，使得它们满足约束方程：

$$U^*(s) \leftarrow U^*(s) + \alpha(R(s) + \gamma(s') - U^\pi(s))$$

其中 α 是学习速度参数。

时序差分学习的基本思想是将效用估计朝着理想均衡方向调整，当效用估计正确时，理想均衡是局部成立的，其均衡就是贝尔曼方程。

自适应动态规划和时序差分实际上是紧密相关的，它们都试图对效用估计进行局部调整，以使得每一状态都与其后继状态相一致。区别在于，时序差分调整状态使其与已观察到的状态相一致，而自适应动态规划则调整状态使其使其与所有可能出现的后继状态相一致，并根据概率进行加权。此外，时序差分学习对每个观察到的转移都只进行单一的调整，而自适应动态规划为了重建效用估计和环境模型之间的一致性会进行尽可能多的调整。所以，时序差分学习可以看作是对自适应动态规划的一个粗略而有效的一阶近似。

3 主动强化学习

3.1 主动自适应动态规划学习 Agent

被动学习 Agent 有固定的策略决定其行为，而主动学习 Agent 必须决定自己应该采取什么行动。

一个显然的方案是，让 Agent 运用策略迭代，简单地执行最优策略所建议的行动，这类 Agent 称为贪婪 Agent。但这种贪婪 Agent 很容易陷入局部最优而无法找到真正的最优解，因为学校到的模型和真实环境并一定不相同。行动不仅仅根据当前学习到的模型提供回报，它们也通过影响所接收的感知信息对真实模型的学习做出贡献，通过改进模型，Agent 将在未来得到更高的回报。因此，一个 Agent 必须要在充分利用信息以最大化回报和探索以最大化长期利益之间进行折中。技术上，任何这样的方案在无穷探索的极限下都必然是贪婪的 (GLIE, greedy in the limit of infinite exploration)。一个 GLIE 方案必须对每个状态下的每个行动进行无限次数的尝试，以避免由于一系列不常见的糟糕结果而错过最优行动的有限概

率。一个 GLIE 方案最终还必须变得贪婪，以使得 Agent 的行动对于学习到的模型而言是最优的。

存在几种不同的 GLIE 方案：

1、 随机选择行动：收敛速度缓慢；

2、 给 Agent 尝试较少的行动加权，同时避免已确信具有低效用的行动：

$$U^+(s) \leftarrow R(s) + \gamma \max_a f\left(\sum_{s'} P(s'|s,a)U^+(s'), N(s,a)\right)$$

其中：

$$f(u,n) = \begin{cases} R^+ n < N_e \\ uothers \end{cases}$$

3.2 主动时序差分学习 Agent

与被动学习的情况相比，主动时序差分学习 Agent 最明显的差别是 Agent 不再具有固定的策略。和主动自适应动态规划学习 Agent 不同的是，由于主动时序差分学习 Agent 使用观察到的转移来调整观察到的状态的效用，在从被动 Agent 成为主动 Agent 时，其更新规则不需要做出任何修改。

3.3 Q-Learning

Q-Learning 是时序差分学习的一种，它学习一种行动-效用表示而不是学习效用。Q-Learning 的核心——Q 函数 $Q(s,a)$ 代表在状态 s 进行行动 a 的价值，其与效用值直接相关：

$$Q(s,a) = R(s) + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q(s',a')$$

可以看出，Q-Learning 不需要状态转移模型，而只需要 Q 值。Q-Learning 的更新公式为：

$$Q(s,a) \leftarrow Q(s,a) + \alpha(R(s) + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

4 策略搜索

一个策略 π 就是一个将状态映射到行动的函数，它具有比状态空间中的状态少得多的参数。例如，可以用一个参数化的 Q 函数集合表示 π，每个行动用一个函数，并选取具有最高预测值的行动：

$$\pi(s) = \max_a \hat{Q}_\theta(s,a)$$

每个 Q 函数都可以是诸如神经网络那样的一个非线性函数，然后策略搜索会调整参数 θ 来改进策略，直到找到一个 θ，使 $\hat{Q}_\theta$ 接近 Q。

离散的行动使得策略价值的变化不连续，令基于梯度的搜索变得困难，因此，策略搜索方法经常使用 $\pi_\theta(s,a)$ 的一种随机搜索表示方法，指定在状态 s 中选择行动 a 的概率。一个常用的表示是 softmax 函数：

$$\pi_\theta(s,a) = e^{\hat{Q}_\theta(s,a)} / \sum_{a'} \hat{Q}_\theta(s,a)$$

5 强化学习的应用

1、 Arthur Samuel 的下棋程序：加权线性函数用于形势评估，使用时序差分更新权值；

2、 Gerry Tesauro 的 TD-GAMMON 双陆棋程序：使用具有 80 个隐单元的神经网络，使用时序差分学习；

3、 小车连杆平衡：控制小车的位置以使连杆保持平衡，已有数千篇关于强化学习及相关控制理论的论文；

4、 直升机自动控制：通常使用策略搜索以及基于一个已经学习好的转移模型的进行仿真的算法。

人工智能（研究生）课程考核报告封面

<table>
<tr><td>院　　系</td><td colspan="2">计算机学院</td><td>专业</td><td colspan="3">计算机科学与技术</td></tr>
<tr><td>学生姓名</td><td colspan="2">尹达恒</td><td>学号</td><td colspan="3">201857</td></tr>
<tr><td>课程名称</td><td colspan="6">人工智能</td></tr>
<tr><td>授课时间</td><td colspan="2">2020 年 9 月至 2020 年 12 月</td><td>学时</td><td>32</td><td>学分</td><td>2</td></tr>
<tr><td>考核论题</td><td colspan="6">按要求撰写报告
1. 读书报告（见试卷纸试题一）。
2. 人工智能应用设计报告（见试卷纸试题二）。
提交说明：
1. 以此文档作为报告封面的正反面（每个人的两个报告装订在一起，用一个封面）；
2. 报告正文采用五号字体，除封面之外，其余各页正反打印；
3. 打印后，2020.2.25 前交给负责同学，统一提交给老师。</td></tr>
<tr><td>简
要
评
语</td><td colspan="6">完成度（分级 5、10、15、20）
规范性（分级 5、10、15、20）
条理性（分级 5、10、15、20）
应用场景具体程度（分级 5、10、15、20）
技术细致度和可行性（分级 5、10、15、20）
雷同或非自主设计程度（分级-0、-20、-40、-50）</td></tr>
<tr><td>总评成绩
（含平时成绩）</td><td colspan="6"></td></tr>
<tr><td>备注</td><td colspan="6"></td></tr>
</table>

任课教师签名：_____

日期： 2021.3.1

东 南 大 学 试 题 纸

课程 人工智能　　2020 —2021 学年第一学期

学 号 201857　　姓 名 尹达恒　　得 分＿＿＿＿＿＿

（本试卷共 2 页）

一、　读书报告（50 分）

1.从《Artificial Intelligence——A Modern Approach》(3e)中选择 1 章以上的内容进行详细深入的阅读，并撰写读书报告，要求列出

- 知识点
- 技术要点和算法，
- 对每一个技术/算法分析说明其适合的任务环境并给出理由。

二、　人工智能应用设计报告（50 分）

1．内容应包括：题目、摘要、主题词、正文、主要参考文献。

2．技术报告正文部分字数不少于４０００字，包括 8 个部分：

- 场景的描述，要具体到某个实际想定情景示例上；
- 智能化任务，说明该示例场景中可智能化解决的环节，及具体功能和非功能要求；
- 任务环境分析，给出 PEAS 和性质的详细分析；
- 智能 Agent 结构，给出适合的 Agent 结构，及具体模块的结构；
- 问题，说明上述智能化任务中面临的主要技术问题和非技术问题；
- 现状，上述问题的已有解决方法及其优缺点分析；
- 技术方案，给出采用一个课本中现成方法和算法的优缺点分析，结合上述优缺点分析，进一步给出本文的具体技术和算法方案；
- 方案分析，给出本文方案实际应用中的可行性分析。

3．题目自拟，建议从所了解的课题或日常生活中选择一个应用场景，以“面向 XXX 的智能 Agent 设计和关键技术”为题。

评分标准：

1. 见评语选项及分数等级；

面向实时交互式视频通信客户端侧流量调节的智能 Agent 设计和关键技术

尹达恒
（东南大学，江苏　南京）

摘要： 本文面向实时交互式视频通信领域，提出了一种基于强化学习和联邦学习的客户端侧流量调节的智能 Agent 设计方案，并针对关键问题和相关研究现状阐述了构建 Agent 所需的技术，最后对整体实现方案进行了可行性分析。

本文所设计的系统解决了实时交互式视频通信领域的几个重要问题，功能完善，为大规模实时交互式视频通信应用的构建提供了有力支持，具有较强的实用价值。

主题词：视频流，策略搜索，流量调节，联邦学习，强化学习

1 场景描述

2020 年的新冠肺炎疫情对传统的面对面“接触式”办公模式带来了巨大的冲击，作为“无接触式”办公模式的重要组成部分，视频会议软件得到的空前的发展，实时交互式视频通信应用迅速渗透到各行各业的生产活动中。

根据 Cisco 发布的年度网际网络报告 (Cisco Annual Internet Report)[1]，在当今的所有互联网流量中，实时交互式视频流量占据着主导地位。随着 LTE-Advanced 和 5G 的发展，新的低延迟应用也在迅速出现，例如实时视频/VR 广播、云游戏、手术机器人或车辆的远程操作等。这样的交互式视频应用比视频会议应用在带宽和延迟方面的要求更加苛刻。尽管电信基础设施努力满足需求，但基础设施仅能提供尽力而为的服务，因此，为了适应高度动态的网络条件和不同应用场景多样化的需求，在交互式视频通信客户端一侧的流量调节必不可少。

1、 应用领域：交互式视频通信；

2、 主要功能：在客户端一侧，根据网络环境实时地调节流量策略。

2 智能化任务

在上述在交互式视频通信客户端一侧调节流量的应用场景中，人工智能算法需要处理的智能化任务可以概括为：

- 输入：算法要能够及时地获取客户端一侧当前网络情况；
- 输出：算法要需要根据获取到的网络情况调节视频编码的比特率；
- 优化目标：视频编码的比特率应该调节到恰好使网路不发生拥塞。

3 任务环境分析

智能 Agent 的任务环境图 1所示。本节将从 Agent 的输入输出和优化目标出发对 Agent 的任务环境进行分析。

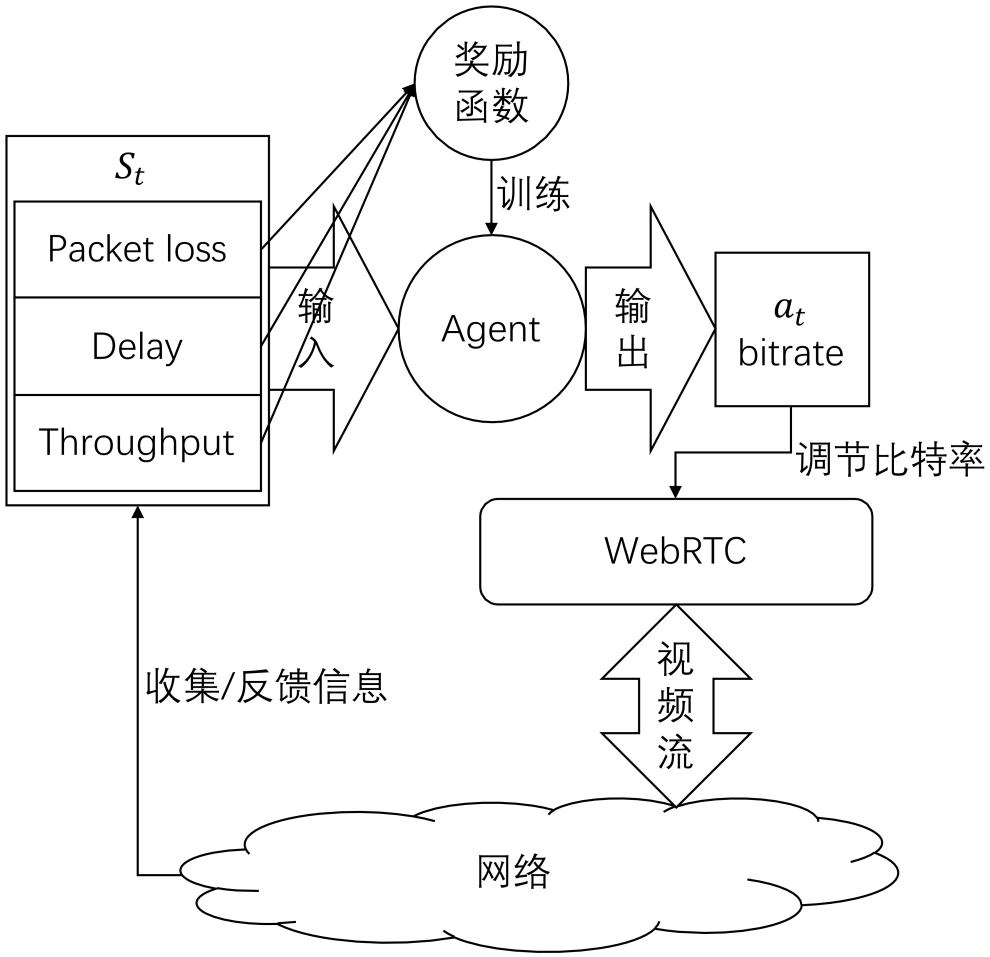

图 1. 智能 Agent 的任务环境

3.1 Agent 输入分析

网络情况所涵盖的变量很多，但大部分变量都记录在路由器、交换机等运营商侧的设备中。显然，实际情况下，运营商不可能在机房中大面积部署高能耗的人工智能应用，也不大可能将通信设备的状况信息开放给用户获取。因此，在客户端侧，Agent 所能获取到的信息是比较有限的。根据目前交互式视频通信常用的传输/应用层协议——WebRTC 的内容，Agent 可以通过协议中周期性的 RTCP ACK 消息收集到每个数据包的发送情况，继而计算出如下的网络状况信息：

1、 丢包率 (Packet loss)l_t：根据 TCP 协议中的超时重传规则，在 TCP 协议中对正常发送的包和重传包进行计数，即获取一定时间内的丢包率信息；

2、 延迟 (Delay)d_t：根据 TCP 协议的 ACK 机制，对包发送完成和收到确认 ACK 的时间间隔进行计数，即可获得发送过程的延迟信息；

3、 吞吐量 (Throughput)t_t：根据 TCP 协议中的发送机制，统计一定时间内的发送窗口和接收窗口大小，即可获得系统的吞吐量信息。

令 $S_t = (l_t, d_t, t_t)$ 表示第 t 个计时区间内 Agent 输入的网络状况信息。

3.2 Agent 输出分析

由于视频流需要连续发送的特点，在设计 Agent 时可以不必考虑包发送的时刻问题，而只用考虑一段时间内发送包的数据总量，即视频流的比特率。目前视频流应用常用的 WebRTC 框架即具有动态调节视频比特率的功能。

在 WebRTC 框架的每个 RTCP 循环中，Agent 有机会修改下一个循环中的视频编解码器的目标输出比特率。因此，Agent 可以将 S_t 映射到一个可选比特率集合 A（例如 $\{0.1Mbps, 0.2Mbps, ..., 2.5Mbps\}$）经过一段时间的视频传输后，系统又能收集到一批新的网络状况信息，计算出新的状态 S_{t+1}，进而让 Agent 生成新的 a_{t+1}。通过这样的连续迭代，Agent 就能学会应对网络动态变化。

3.3 Agent 优化目标分析

根据 TCP/IP 协议的丢包规则，当网络发生拥塞时，数据包在设备中的排队时间将显著增大，网络中无法承载的数据包将被网络设备直接舍弃，在客户端一侧的表现就是丢包率和延迟的急剧增大。因此，如果 Agent 输出比特率 a_t 超过可用带宽，则将导致网络拥塞，进而在下一个状态 S_{t+1} 中出现较高的丢包率和较大的延迟。因此，通过观察从 S_t 到 S_{t+1} 的丢包率和延迟变化，就能判定网络是否出现拥塞，进而判定 Agent 输出的 a_t 是否合适。如果 a_t 超过可用带宽，那么当下次观察 S_t 或类似状态时，Agent 应输出较小的 a_t；反之，Agent 应输出较大的 a_t。由此，Agent 可以逐步逼近恰好使网路不发生拥塞的视频流比特率。

4 智能 Agent 结构

4.1 Agent 选型

根据第 3节的分析，直观上讲，本系统所需要的 Agent 就是一个简单的输入三个变量 (l_t, d_t, t_t)，输出一个变量 a_t 的函数。但也可以看出，在 Agent 的运行环境中输出变量 a_t 的取值会反过来影响 $t+1$ 时刻的网络状况，Agent 的优化目标也是从网络状况 S_t 中计算得到。综上所述，这是一个典型的强化学习模型的运行环境[2]，可以使用强化学习模型进行求解。更进一步，从第 3.2节的分析可以看出，Agent 所输出的 a_t 更接近于离散的策略输出。因此，本文将以强化学习模型中的策略搜索 Agent 为核心，介绍在实时交互式视频通信场景下的 Agent 设计。

4.2 Agent 运行过程

由第 3节的分析可知，Agent 的输入信息均为一段时间内的统计数据，输出也是接下来一段时间内的控制信息，因此输入和输出必然是在一个个时间片上进行的；而用于实时交互式视频传输的流量调节又要求 Agent 能细粒度地响应网络动态。通常来讲，具有动态调节比特率的视频视频编码器对视频质量的调节粒度均在帧级，即每一帧对应一种比特率，帧与帧之间的比特率可以不同。因此，对于用于实时交互式视频传输的流量调节的 Agent，其响应网络动态的粒度极限即是帧级，对应于数十毫秒的时间范围。进而可以得到 Agent 的运行过程：

1、 初始化：随机选一个较小值作为初始帧的比特率 a_0；

2、 传输帧：以比特率 a_{t-1} 对帧进行编码后发送；

3、 统计 S_t：在发送帧的过程中，系统收集每个数据包的活动（发送时刻、收到确认 ACK 的时刻、是否丢包等）；

4、 计算 S_t：根据统计得到的数据包的活动信息计算出在发送这一帧过程中的平均丢包率、平均延迟和平均吞吐量，作为 (l_t, d_t, t_t)；

5、 计算 a_t：Agent 接收 S_t，计算出下一帧的比特率 a_t；

6、 回到第 2 步，循环。

4.3 Agent 奖励函数

根据第 3.3节的分析，在 Agent 的运行环境中，Agent 需要控制 WebRTC 视频流的比特率实现最高的比特率且使得系统恰好不发生拥塞。因此，根据第 3.3节的判定方法，Agent 的奖励函数要能让系统产生最大的吞吐量以及最小的丢包率和延迟，因此易得奖励函数：

$$r_t = R(a_t) = -\alpha \times l_{t+1} - \beta \times d_{t+1} + \gamma \times t_{t+1}$$

其中 r_t 表示第 t 个时间段内产生的输出 a_t 应用于 $t+1$ 后所产生的奖励；α、β 和 γ 是训练前需要调节的超参数，表征网络情况的不同方面对系统的影响程度。

4.4 Agent 内部结构

Agent 使用神经网络来表示带有一组参数 θ 的策略 π_θ。它采用简单的全连接层结构来提取隐藏在不同输入元素中的隐式特征。令 $\rho(\theta)$ 表示策略的价值，在视频流传输过程中，直接根据在参数 θ 下的执行结果获得一个对 θ 的梯度 $\nabla_\theta \rho(\theta)$ 的无偏估计。若在起始状态 S_0 下应用比特率 a 后，获得回报 $R(a)$，那么有：

$$\nabla_\theta \rho(\theta) = \nabla_\theta \sum_a \pi_\theta(S_0, a) R(a) = \sum_a \nabla_\theta \pi_\theta(S_0, a) R(a)$$

进而，在第 T 帧发送结束时，有：

$$\nabla_\theta \rho(\theta) = \sum_a \nabla_\theta \pi_\theta(S_0, a) R(a) \approx \frac{1}{T} \sum_{t=1}^{T} \frac{\sum_a \nabla_\theta \pi_\theta(S_0, a_t) R(a_t)}{\pi_\theta(S_0, a_t)} \approx \frac{1}{T} \sum_{t=1}^{T} \frac{\sum_a \nabla_\theta \pi_\theta(s, a_t) R_t(s)}{\pi_\theta(s, a_t)}$$

按照此梯度使用反向传播对参数 θ 进行更新，从而对 Agent 进行训练。

5 问题

从理论上讲，第 7节所述的 Agent 已经是一个比较完备的实时交互式视频通信客户端侧流量调节 Agent 了，其已经具备独立完成流量调节和训练的能力，对各种网络环境都有一定的适应力。但是，从现实角度出发，这样的 Agent 并不能很好地完成所需的流量调节功能，它还存在一些现实问题：

1、 延迟：第 7节所述的 Agent 在原本连续的帧发送流程中间插入了基于强化学习的调节操作，对视频流的发送有不利影响；

2、 孤立：实际情况下的网络环境中，不可能只有一个用户在使用视频流服务，调节网络流量的 Agent 本可以互相合作，但根据第 7节所述的 Agent 结构，每个 Agent 实例都只能在一个固定的客户端上运行，每个 Agent 都要处理（对网络来说）大量的网络环境数据，Agent 之间难以完成数据共享操作，进而也无法考虑系统中由其他 Agent 的情况（例如多 Agent 在从网络环境的变化规律中发现其他 Agent 的存在并进行协作[3]）；

3、 数据利用不充分：数据利用不充分的问题归根到底就是机器学习领域老生常谈的数据量问题。在实际操作中，尽管可以以集中式训练-分发的过程生成 Agent，但由于有适应动态网络环境的需求，Agent 必须有在客户端侧进行训练的过程。很显然，客户端侧的视频应用不可能保持长期运行，因此 Agent 在训练过程中智能接触到一个客户端的少量数据；但在现实场景下，一套服务器不可能只有一个用户在用，因此在一个 Agent 训练时，还会有许许多多个 Agent 也在同时进行训练，但它们都只能分别接触到单个个客户端内的少量数据，训练出各不相同的 Agent，而没能更进一步，将各个客户端的数据进行聚合，从而训练更强大的 Agent。

6 现状

6.1 延迟问题

在动态网络环境下自适应地调节在线视频流质量的问题由来已久，对于第 5节所述的延迟问题也有比较成熟的解决方案。例如，WebRTC 框架中就有将视频流量的调节和视频流的发送分开运行的功能[4]，在实际的开发过程中，视频流量的调节算法运行一轮后可以不直接调节视频发送参数，而是将目标发送参数写入内存，待视频流发送需要调节时直接读取，这样就能在不断连续视频流传输的情况下进行视频流发送参数的调节，使得调节算法的计算延迟对视频流传输的影响降到最小。

6.2 孤立

为解决现实情况下 Agent 训练时的障碍并非只有数据共享问题，在纷繁复杂的现实网络环境中，有许多问题制约着人工智能的应用，有些甚至涉及到不同国家的法律法规而非纯粹的技术障碍。跳脱出现实环境，将重要的问题抽象为虚拟环境是避免现实障碍的好方法，就像[5] 和[6] 做的那样。在虚拟环境中，抛却了现实网络的桎梏，我们就有可能在 Agent 的训练过程中进行大量的数据共享，进而训练一些具备协作能力的 Agent。

然而，在这种由模拟器构筑的虚拟环境中训练，其缺点也非常明显：

1、 复杂的现实世界中的互联网动态很难被准确模拟[7]。网络上的大量路由器和交换机有非常复杂功能和状态，例如多流竞争、数据包丢弃策略、负载引起的流量波动；终端设备上还有多个协议层之间的复杂交互，不同协议对边缘网络的不同影响等等。这些都是智能 Agent 需要考虑的。并且由于智能 Agent 的黑盒特性，稍有偏差就会导致 Agent 的训练结果与实际需要差别巨大。

2、 本质上，数据驱动算法的能力严格受其学习环境的限制。在应对现实世界的互联网时，其在模拟器中通过反复试验获得的经验可能会过时。

6.3 数据利用不充分

直观上，“数据利用不充分”的解决之道就是“充分利用数据”，即通过客户端将用户侧的用于训练 Agent 的数据发送到云端进行集中，并在云端训练更强大的模型再分发给用户。这是解决数据量问题的通常解法，在中国当前互联网法律法规尚不完善的情况下，从用户侧大规模收集数据的行为也见怪不怪。但按照国际上互联网法规的发展道路看，从用户侧收集数据这种侵犯用户隐私还占用大量网络资源的行为总有被取缔的一天，因此从用户侧收集数据并非长久之计。

除此之外，即使成功地将大量的数据收集到的云端，Agent 的训练也仍旧需要在基于现实数据创建的虚拟环境中完成，同样有第 6.2节所述的那些问题。

7 技术分析

排除了虚拟环境和大规模收集敏感数据的可能，还有一条路就是直接从模型入手，在用户停止视频流后将训练好的 Agent 发送到云端，并使用联邦学习的方式整合各 Agent 的训练成果。

对于第节所示的 Agent，可以选用最简单的基于均值的加权模型聚合方法进行联邦学习[8-9]，即下一次分发的 Agent 的策略参数 θ' 是之前收集到的 K 个客户端侧训练好的模型的参数 θ_k 的加权平均：

$$\theta' = \sum_{k=1}^{K} \alpha_k \theta_k$$

其中 α_k 为各个客户端参数的权值，例如可以给最近上传的客户端参赋予较大权值以适应网络不断变化的情况。

8 方案分析

综合第 7节和第 7节的技术方案，可以得到本文所述的面向实时交互式视频通信客户端侧流量调节的智能 Agent 最终的运行过程：

1、 用户启动客户端，开启视频流应用；

2、 客户端从云端下载当前最新的 Agent；

3、 客户端开始视频流传输，使用 Agent 控制视频比特率，同时按照第 7节所述的训练方式对 Agent 进行训练，直到用户关闭视频流；

4、 将训练完的 Agent 发送到云端；

5、 云端使用第 7节介绍的方案对多个客户端发来的 Agent 进行联邦学习，生成新的 Agent。

根据现有的技术方案[4-5,10]，只需要在 WebRTC 框架中在 MediaStream 上添加下载和训练 Agent 代码，并通过 MediaStreamEvent 类和 MediaStreamTrack 类收集网络环境信息即可完成客户端侧的 Agent 训练；借助 FATE 框架即可搭建位于云端的联邦学习系统对上传的 Agent 进行更新和分发。项目可行。

参考文献

[1] Cisco Annual Internet Report - Cisco Annual Internet Report (2018–2023) White Paper - Cisco [Z].

[2] Russell S, Norvig P. Artificial intelligence: a modern approach[J]. 2002.

[3] Xu Z, Zhou L, Chi-Kin Chau S, et al. Collaborate or Separate? Distributed Service Caching in Mobile Edge Clouds[C/OL]//IEEE INFOCOM 2020 - IEEE Conference on Computer Communications. Toronto, ON, Canada: IEEE, 2020: 2066-2075. DOI: 10.1109/INFOCOM41043.2020.9155365.

[4] WebRTC 1.0: Real-Time Communication Between Browsers[Z].

[5] Mao H, Chen S, Dimmery D, et al. Real-world video adaptation with reinforcement learning [J]. arXiv preprint arXiv:2008.12858, 2020.

[6] Zhou A, Zhang H, Su G, et al. Learning to coordinate video codec with transport protocol for mobile video telephony[C]//The 25th Annual International Conference on Mobile Computing and Networking. 2019: 1-16.

[7] Yan F Y, Ma J, Hill G D, et al. Pantheon: the training ground for internet congestion-control research[C]//2018 {USENIX} Annual Technical Conference ({USENIX}{ATC} 18). 2018: 731-743.

[8] Goodfellow I J, Vinyals O, Saxe A M. Qualitatively characterizing neural network optimization problems[J]. arXiv preprint arXiv:1412.6544, 2014.

[9] McMahan B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data[C]//Artificial Intelligence and Statistics. PMLR, 2017: 1273-1282.

[10] FedAI.org – Federated AI Ecosystem[Z].