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： 待定。

主题词： 策略，流量调节，联邦学习，强化学习，机器学习

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。

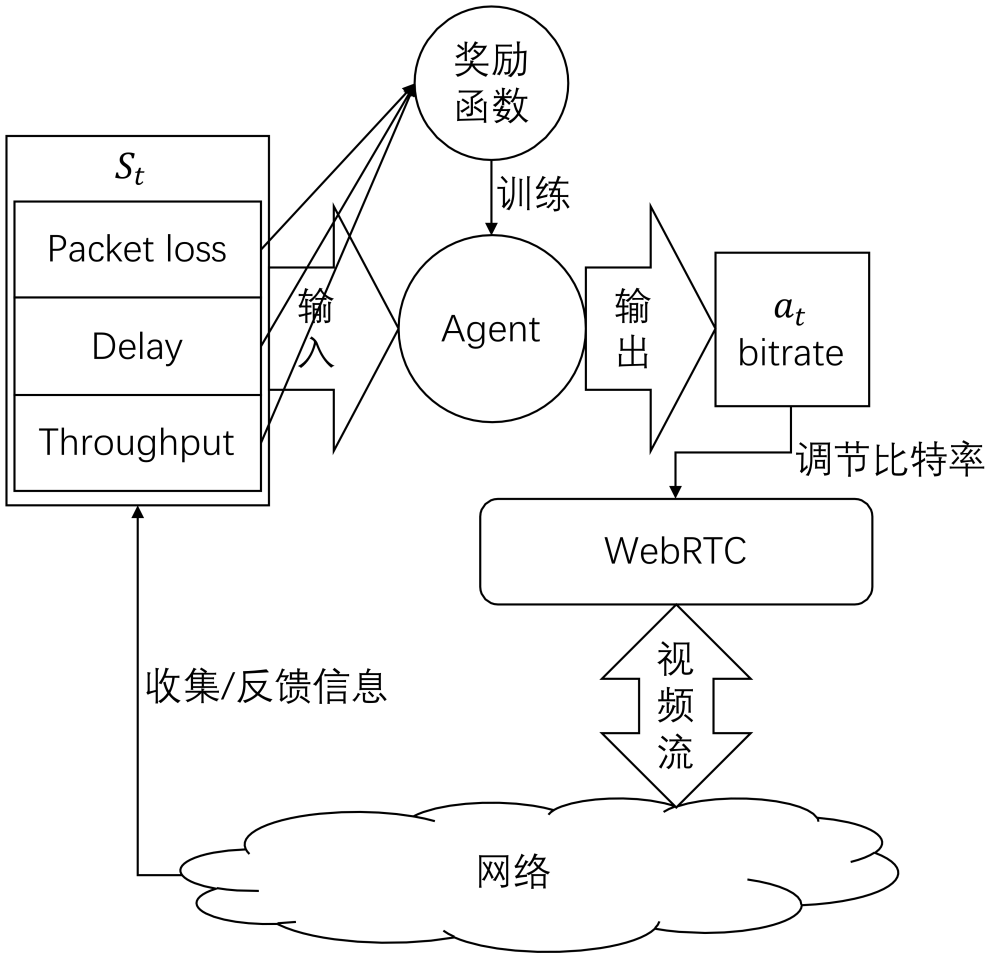

图 1. 智能 Agent 的任务环境

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；更进一步，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，可以使用强化学习模型进行求解。因此，本文将以强化学习模型为核心，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 神经网络结构

给出适合的 Agent 结构，及具体模块的结构；

5 问题

1、 如何适应动态的网络条件

2、 大量的 Agent 在各自的硬件上独立地运行，如何进行训练

3、 视频编解码器的调节具有滞后性

4、 掌握机器学习分类算法的性能提升方法；

5、 能编程实现一些机器学习分类算法并进行性能分析和改进；

6、 了解人工智能和分类算法的新进展、新应用。

6 现状

上述问题的已有解决方法及其优缺点分析；在虚拟环境中训练

7 技术分析

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；

8 方案分析

给出本文方案实际应用中的可行性分析。

具体来说，决策树的构建方法可以表示为算法 1[2]。

算法 1. 决策树学习的基本算法

Require: 训练集 $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)|1 \leq i \leq m\}$、属性集 $\mathcal{A} = \{a_i|1 \leq i \leq d\}$
Ensure: $\forall(\boldsymbol{x}_i, y_i) \in \mathcal{D}$，属性值 $a_i(\boldsymbol{x}_i)$ 存在

function DecisionTree($\mathcal{D}$,$\mathcal{A}$)
  if $\mathcal{D}$ 中的样本全部属于同一类别 C **then**
    return 单节点树，类别标记为 C
  end if
  if $\mathcal{A} = \varnothing$ **then**
    return 单节点树，类别标记为 $\mathcal{D}$ 中样本数最多的类
  end if
  计算 $\mathcal{A}$ 中每个特征的信息增益（ID3 算法）或信息增益比（C4.5 算法），从中选择最优属性 a_{best}
  for a_{best} 的每一可能值 a_{best}^t **do**
    令 $\mathcal{D}' = \{(\boldsymbol{x}_i, y_i)|a_{best}(\boldsymbol{x}_i, y_i) = a_{best}^t, (\boldsymbol{x}_i, y_i) \in \mathcal{D}\}$
    if $\mathcal{D}' = \varnothing$ **then**
      添加叶节点，类别标记为 $\mathcal{D}$ 中样本数最多的类
    else
      递归添加分支节点 DecisionTree($\mathcal{D}'$,$\mathcal{A}$)
    end if
  end for
end function

将这 4 个分类结果占测试集总量的比率共同呈现于表 3中，所形成的 2×2 矩阵就称为

混淆矩阵 (Confusion Matrix)。相比于分类正确率，混淆矩阵能反映有关分类器性能的更加具体的信息，是分类预测模型性能评价的常用指标。

表 1. 混淆矩阵

混淆矩阵 真实值 / 预测值	正	负
正	TP	FP
负	FN	TN

8.0.1 确定缺失项处理方法

缺失属性项确定之后，接下来就需要确定各缺失项的处理方法。首先使用 R 语言读取“diabetes.csv”的数据并对数据集中的数据缺失情况进行统计：

1、 统计“diabetes.csv”中的数据总行数，得到结果 768；

2、 使用 mice 包中的 md.pattern() 函数统计数据集中的数据缺失情况，数据缺失情况统计表（表 2）。

表 2. 数据缺失情况

是否缺失 列名 / 数据点数量	Glucose	BMI	BloodPressure	SkinThickness	Insulin
392	否	否	否	否	否
1	是	否	否	否	否
140	否	否	否	否	是
1	否	是	否	否	否
4	是	否	否	否	是
2	否	否	是	否	是
192	否	否	否	是	是
1	否	是	否	否	是
26	否	否	是	是	是
2	否	是	否	是	是
7	否	是	是	是	是
列缺失总量	5	11	35	227	374

9 实验结果与分析

9.1 实验结果

9.1.1 最高精度算法结果

- 算法最高精度：87.2611%
- 最高精度算法：人工神经网络，见**??**节
- 算法参数：见**??**节和**??**节
- 混淆矩阵：

表 3. 最高精度算法结果的混淆矩阵

%	估计样本 0	估计样本 1
真实样本 0	64.33	7.006
真实样本 1	5.732	22.93

9.2 实验结果分析

1、 由表 3可以看出：

a) 错误估计的样本中假正例率和假负例率基本保持平衡，说明该算法对于正例和负例的判定都有比较好的结果；

b) 正确估计的样本中主要都集中于真负例，说明该数据集的正负例非常不平衡；

c) 上面两个结论表明，该算法在一个正负例不平衡的数据集中训练得到了一个假正例率和假负例率基本保持平衡的分类器，说明该算法具有很好的鲁棒性。

2、 由图**??**、图**??**和图**??**可以看出：

a) 逻辑回归模型在该数据集上的准确性要略高于神经网络模型，且 Dropout 和 L2 正则化能使得神经网络模型的准确性和稳定性都有一定程度的提升，这说明本次实验中建立的神经网络模型在数据集上已经出现了过拟合现象，也印证了 Dropout 和 L2 正则化对神经网络模型过拟合的抑制作用；

b) 在三种使用不同核函数的 SVM 分类算法中，使用较为复杂的径向基核和多项式核构造 SVM 的分类正确率都不及最简单的线性核 SVM，表明所使用的数据集中数据分布结构比较简单，复杂的核函数并不能对 SVM 分类准确率的提升有帮助；

c) 两种决策树算法在不同预处理数据集上的预测正确率相差很大，这表明决策树算法是一种不稳定的分类算法，在实际使用时需要一些后剪枝算法进行辅助；

d) 高斯朴素贝叶斯算法在不同的预处理数据集上都有大致相近的正确率，表明基于贝叶斯的分类算法具有较好的稳定性；

e) Stacking 集成学习在多次重复实验中具有最高的稳定性，随机森林和 AdaBoost 集成学习算法在不同预处理数据集上的结果要比两种决策树算法都更接近，且随机森林算法的正确率要高于两种决策树算法，表明集成学习策略能提高机器学习算法的鲁棒性和准确性。

3、 由图**??**和图**??**可以看出：

a) 经过 EM 填补的训练集训练出的机器学习算法在测试集上的准确率较为接近，且该训练集训练出的逻辑回归与神经网络算法在该测试集上的结果波动较小，表明 EM 填补相对于其他方法填补得到的训练集具有更好的分类特征，且不会与原数据集产生较大差异，对于有缺失数据的数据集使用 EM 算法能在一定程度上有提高训练出的机器学习算法的稳定性；

b) 删除“SkinThickness”列后的训练集训练出的逻辑回归与神经网络算法的测试集正确率较高但是波动很大，表明 SkinThickness 列并非是和糖尿病直接相关的特征，但是在一定程度上有助于糖尿病诊断；

c) 当删除训练集“Insulin”列或是“Insulin”和“SkinThickness”列都被删除时，逻辑回归与神经网络算法的测试集正确率都有所下降，这表明胰岛素确实时糖尿病诊断的重要特征。

参考文献

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

[2] 周志华. 机器学习[M]. 北京: 清华大学出版社, 2016.