基于深度强化学习的自动驾驶策略学习方法

夏 伟[1,2] 李慧云[1,2]

[1](中国科学院深圳先进技术研究院 深圳 518055)
[2](中国科学院大学深圳先进技术学院 深圳 518055)

摘 要 自动驾驶是人工智能研究的重要应用领域，文章提出了一种基于深度强化学习的自动驾驶策略模型学习方法。首先采用在线交互式学习方法对深度网络模型进行训练，并基于专业司机的经验数据对模型进行预训练，进而结合经验池回放技术提高模型训练收敛速度，通过对状态空间进行聚类再采样，提高其独立同分布特性以及策略模型的泛化能力。通过与神经网络拟和 Q-迭代算法的比较，所提方法的训练时间可缩短 90% 以上，稳定性能提高超过 30%。以复杂度略高于训练集的测试道路长度为基准，与经验过滤的 Q-学习算法相比，采用聚类再采样的方法可以使策略模型的平均行驶距离提高 70% 以上。

关键词 深度强化学习；自动驾驶；聚类；神经网络
中图分类号 TG 181 **文献标志码** A

Training Method of Automatic Driving Strategy Based on Deep Reinforcement Learning

XIA Wei[1,2] LI Huiyun[1,2]

[1](*Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen* 518055, *China*)
[2](*Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences, Shenzhen* 518055, *China*)

Abstract Automatic drive is an important application field of artificial intelligence. In this paper, a novel training strategy for self-driving vehicles was investigated based on the deep reinforcement learning model. The proposed method involves a Q-learning algorithm with filtered experience replay and pre-training with experiences from professional drivers, which accelerates the training process due to reduced exploration spaces. By resampling the input state after clustering, generalization ability of the strategy can be improved due to the individual and independent distribution of the samples. Experimental results show that, in comparison with conventional neural fitted Q-iteration algorithm, the training efficiency and controlling stability can be improved more than 90% and 30% respectively by the proposed approach. Experimental results with more complex testing tracks show that, average travel distance can be improved more than 70% in comparison with the Q-learning algorithm by the proposed method.

Keywords deep reinforcement learning; autonomous vehicle; cluster; neural network

收稿日期：2017-03-09 **修回日期**：2017-04-01
作者简介：夏伟，硕士，研究方向为深度强化学习在自动驾驶方向上的应用；李慧云(通讯作者)，博士，研究员，研究方向为智能驾驶，E-mail：hy.li@siat.ac.cn。

1 引 言

近年来，随着经济的发展和城镇化的推进，全球汽车保有量和道路里程逐步增加。中国市场调查网显示，截至 2016 年 6 月底，全国机动车保有量已达 2.85 亿辆，其中汽车为 1.84 亿辆[1]。随着群众机动出行的需求不断提高，我国汽车保有量保持快速增长趋势，诸如交通拥堵、事故、尾气排放、土地资源紧缺等一系列传统汽车无法妥善解决的问题日益凸显。据国家统计局统计，2014 年每万辆机动车的道路交通死亡率为 2.22%，全年交通事故造成高达 34 292 人的死亡[2]。然而，有研究表明，估计 90% 的交通事故是由于驾驶员的失误造成的，主要包括注意力不集中、判断失误和情境意识不足等因素[3]。

随着互联网技术的迅速发展，汽车工业科技含量水平得到了巨大的提高。虽然车辆中已经存在很多安全技术措施来减少交通事故发生时对驾驶员的伤害，但仍然不能从根本上解决人为因素造成的安全隐患[4]。自动驾驶技术被视为这一系列交通问题的有效解决方案，其发展备受瞩目。Google、特斯拉、百度和其他科技公司等在自动驾驶技术上均投入巨大的研发力量，并且有部分车辆模型已经在道路上进行测试。美国电气和电子工程师协会(IEEE)预测，至 2040 年自动驾驶车辆所占比例将达到 75%[5]。然而，就目前来说，开发出能够自如应对各种复杂多变路况下的自动驾驶系统仍然是一项巨大的挑战。

Q 学习是一种免模型的强化学习方法，其最通用的方式是构造一个列表存储所有状态-动作对的评价值，通过不断重复访问任意状态，尝试不同的动作来不断更新其状态-动作值。根据贝尔曼最优方程，其状态-动作值最终会收敛到一个最优策略[6]。其必要条件是需要不限次数访问所有的状态，而面对巨大的状态和动作空间的时候，这种方法是不能奏效的。Riedmiller[7]提出用神经网络拟和 Q-迭代算法(Neural Fitted Q-iteration，NFQ)代替列表的形式，来学习动作值函数。它通过存储每次实验的历史数据，重复计算奖励回报来对神经网络进行训练，这种方法在简单的控制任务，如倒立摆、登山车等，具有比较好的性能表现。但明显地，其对所有经验数据处理的资源消耗是不可避免的。在面对状态空间和动作空间较大的时候，其学习效率很低，而在稍微复杂的道路环境中，它并不能直接实现成功学习到驾驶策略模型。

2006 年，Hinton 等[8]提出的深度信念网络(Deep Belief Networks)开创了深度学习的一个新纪元。2013 年，Krizhevsky 等[9]在大规模视觉识别挑战赛(ImageNet Large Scale Visual Recognition Competition)使用卷积神经网络取得突出成绩之后，深度学习开始在计算机视觉、语音和文本分析等领域得到广泛应用[8,10,11]。鉴于强化学习方法所具有的普适性，研究人员开始结合深度学习和强化学习来解决控制和决策领域的难题[12,13]。其中，Mnih 等[10,13]提出的 DQN(Deep Q-Network)算法最具代表性，通过直接对雅达利 2600 游戏的图像信息进行学习，从而得到其控制策略，并在部分项目上远超人类游戏玩家。随后，DQN 算法在不同领域得到推广与应用。2016 年，Bojarski 等[14]提出使用卷积神经网络进行自动驾驶系统研究的方案。Sallab 等[15]提出使用 DQN 算法进行车道保持辅助系统的仿真研究，并对使用不同终止条件对学习到车道保持策略的影响进行了讨论，但文章仅给出了试验次数的变化情况，并没有给出具体的训练时间的对比。2017 年，Chae 等[16]提出使用 DQN 进行自动刹车系统研究的方法，在经过将近 7 万次模拟试验后，可以学习到自主刹车的能力。

受前人工作的启发，本文提出一种基于深度强化学习的自动驾驶策略模型学习的新方法：采用强化学习的在线交互式学习方法对深度网络模

型进行训练，并基于专业司机的历史数据信息对模型进行预训练，然后结合经验池回放技术，有效提高收敛速度；同时对状态空间进行聚类再采样，提高其独立同分布的特性，因此提高了策略模型的泛化能力。最后，本文基于训练数据样本的不同处理方法，对策略模型的训练效率进行了更加详细的讨论。

2　基于强化学习的控制方法

2.1　强化学习

强化学习的灵感来源生物学中的动物行为训练，训练员通过强化与惩罚的方式让动物学会一种行为与状态之间的某种联系规则[17]。强化学习就是要解决这类问题：一个能够感知环境的智能体(Agent)怎样通过学习选择达到其目标的最优动作[18]。强化学习的基本框架可以描述如图 1 所示。

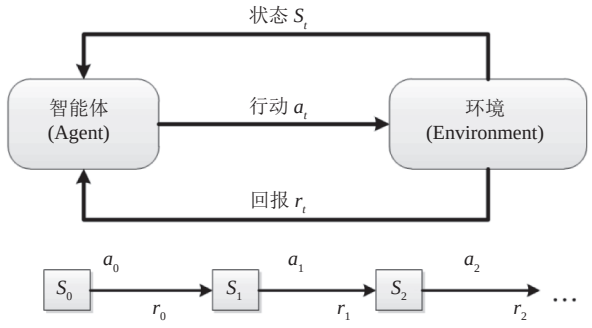

图 1　强化学习基本框架

Fig. 1　The basic framework of reinforcement learning

此 Agent 生存的环境描述为所有可能状态的集合 S，执行任意可能动作的集合 A。每次在某种状态 S_t 下，执行某一动作 a_t，此 Agent 会收到一个实值回报 r_t，它表示此“状态-动作”转换的立即值。如此产生一系列的“状态-动作-立即回报”，如上图 1 所示。Agent 的任务就是要学习一个控制策略 $\pi: S\rightarrow A$，使这些回报的累积和 $r_0+\gamma r_1+\gamma^2 r_2+\cdots+\gamma^n r_n (0\leqslant\gamma<1)$ 的期望最大化[19]。

2.2　基于表格式 Q 学习的控制方法

近年来，强化学习算法中一个重要的分支为 Q 学习的算法[6]，它可通过构造一个大表来表示其策略假设 $\hat{Q}$，其中每一个状态-动作对均有一个表项 $\hat{Q}(s,a)$。Agent 观测其当前的状态 s，选择某个动作 a，执行此动作，然后观测回报值 r 和下一状态 s'。Agent 不断重复此过程并以公式(1)更新 $\hat{Q}(s,a)$[18]。

$$\hat{Q}(s,a)\leftarrow r+\gamma\max_{a'}\hat{Q}(s',a') \qquad 0\leqslant\gamma<1 \tag{1}$$

在确定性回报和动作假设下的 Q 学习算法的流程描述如下：

(1) 对每个 (s,a) 初始化表项 $\hat{Q}(s,a)$ 为 0；

(2) 观察当前状态 s；

(3) 一直执行，直到 $\hat{Q}(s,a)$ 收敛：

(a) 选择一个动作 a 执行；

(b) 接收立即回报 r；

(c) 观察新状态 s'；

(d) 按照公式(1)更新表项 $\hat{Q}(s,a)$。

在任意状态-动作对被无限次数访问的条件下，Q 学习算法被证明是可以收敛的[18]。然而在面对高维度状态空间和动作密集型的复杂问题时，用表项存储状态-动作对的方法将很难收敛，并且在巨大的表项中，数据的存储、查找都将是极大的挑战。

2.3　基于神经网络拟和 Q-迭代的控制方法

基于显式查表式的 Q 学习方法是一种机械式的学习方法，并不会通过在已看到的状态-动作对中进行泛化，来估计未看到的状态-动作对的 Q 值。1995 年，Tesauro[20]使用神经网络和后向传播算法，结合时间差分(Temporal Difference，TD)训练法则成功实现了 TD-Gammon 算法，它使用了 150 万个对弈棋局来进行训练学习棋策略。Riedmiller[7]在 2005 年提出神经网络拟和 Q-迭代算法，通过存储和重复使用经验值，只需要使用较少次数的交互迭代，就可以让神经网络学会解决一级倒立摆、登山车等简单的控制问题。

然而，在自动驾驶复杂的场景下，历史数据的产生是巨大的，并且在单次试验中数据的相关性较强。故这种方法并不能解决自动驾驶的问题。

2.4 经验池回放

经验池回放(Experience Replay)的概念由Lin[21]在其 1993 年的博士论文里第一次提出。它的做法是，从以往的状态转移经验中随机采样数据进行训练，其作用在于克服经验数据的相关性和非平稳分布的问题，同时也增高了数据的利用率。2013 年 Mnih 等[13]提出的 DQN 算法，他们用经验池回放技术来解决深度强化学习中观测状态序列具有相关性的问题。Xia 等[22]通过构造具有两个参数的可变容器来实现 DQFE(Deep Q-learning with Filtered Experiences)算法，即具有状态序列对的最大数量 μ_{rms} 和试验(Episode)最大次数 K_{num} 两个参数约束的容器。目的是希望经验池保留一些较优的历史数据，Agent 的学习单元在每完成一次试验后进行学习，更新网络模型的权值。经验池的增量表达式可表示为：

$$\Delta d=\begin{cases}0 & len\left(DS_h\right)<\mu_{\text{rms}}\ \&\&\ num\left(DS_h\right)<k_{\text{num}}\\ -1 & \text{其他}\end{cases} \quad (2)$$

其中，函数 *len*()和 *num*()分别用来计算经验池中状态序列的个数和试验的次数。在每次试验结束时，通过这两个函数去检测经验池中的数据，当同时满足这两个条件的时候，将新试验中的序列全部加入到经验池中，并且不删除经验池中已有的任何数据，否则剔除经验池中最差的一次试验数据。

2.5 自动驾驶仿真平台 TORCS

TORCS(The Open Racing Car Simulator)是一款高度可移植、跨多平台(Windows、Linux、FreeBSD、OpenSolaris 和 MacOSX)的多车竞技、开源游戏平台，它通过构建真实车辆的发动机模型、离合器模型、变速箱模型以及刹车模型等车辆物理动态模型，结合车辆的物理碰撞损耗、轮胎软硬特性和空气运动学等物理破环模型，以及基于不同道路阻尼系数和倾斜度等的赛道模型之上，来实现车与赛道环境的真实模拟交互，设计十分合理且逼真[23]。基于开源的特性，设计开发者可以获得仿真环境下所有车辆的真实数据，包括速度、位置和油耗等信息，可以较真实地反应不同车辆在各种现实环境下的驾驶动态。因此，TORCS 也成为一款较流行的 AI(Artificial Intelligence)博弈平台，受到了人工智能和智能控制等领域众多研究工作者的青睐。

通过 TORCS 的技术文档可以得知，在 TORCS 并不能直接去控制车辆的速度和加速度，实际是通过后台算法模拟驾驶员的实际驾驶特性，并通过输入命令直接去控制汽油引擎、油门和刹车等，从源代码中控制信息的数据结构体里，可以清晰地看到方向盘、油门和刹车等物理量的定义，这与真实环境下驾驶员的驾驶特性是完全一致的。因此，本文方法也是基于 TORCS 平台构建新的自动驾驶仿真系统来验证算法的可行性。

3 基于深度强化学习的自动驾驶策略

传统的智能车开发主要基于样车设计和控制算法，通过实车调试，再进行相应的修改，此类模式具有开发成本高、开发周期长和对硬件设备可靠性极度依赖的缺点。采用虚拟开发的模式，可以让开发设计师在考虑车辆运动学特征的同时，将注意力集中在车辆调度策略和控制算法上[24]。虚拟平台验证算法之后再进行实车开发，可以大大提高开发效率并降低成本。因此，本文采用的是虚拟开发调试算法的技术路线。

本文基于 TORCS 仿真平台，将深度强化学习、经验池回放以及聚类分析等技术结合应用到自动驾驶的技术研究上，实现自动驾驶的仿真系统，并验证本文提出的算法是有效可行的。

3.1　学习策略模型架构

本文基于 TORCS 仿真平台构建了新的策略模型的训练平台，然后将深度强化学习、经验池回放以及聚类分析等技术结合应用到自动驾驶的技术研究上，组成完整的自动驾驶策略模型的训练与测试一体的系统，具体架构如图 2 所示。以强化学习的框架来划分，包括环境和 Agent 两大主要板块；然后借鉴深度学习的思想，用预训练模块给网络模型设置一个较好的初始权值；最后构建一个测试模块在每次训练之后，进行模型性能的测试。

3.2　预训练网络的初始权值

受深度学习中预训练思想的启发，本文方法设置了一个预训练模块，通过采集一个赛车手代码控制行驶的状态和动作数据，来对我们的网络模型进行预训练[25]。实验对动作空间中的方向盘值进行离散化，以最接近赛车手方向盘量的离散值作为网络的输入，以有限次数的迭代方式使网络输出逼近当前状态与下一状态的变化量，其网络结构如图 3 所示。通过设置少量的训练次数，让网络去逼近两次状态之间的变化量。实验数据显示，预训练网络的权重平均值在 0～5 时，可以达到较好的训练加速效果。

3.3　状态信息聚类分析

在对状态信息进行深入分析后，我们发现经验池中具有很多冗余的数据，智能体在探索

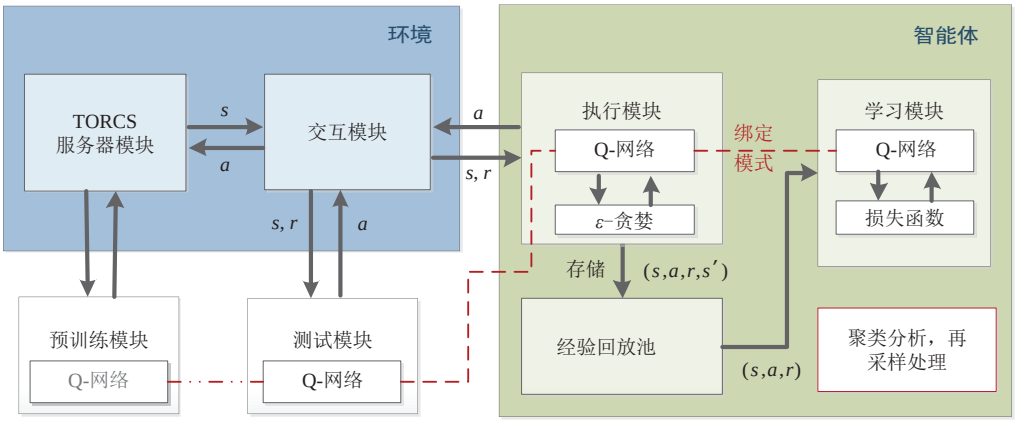

图 2　学习策略模型的架构

Fig. 2　The architecture of train strategy

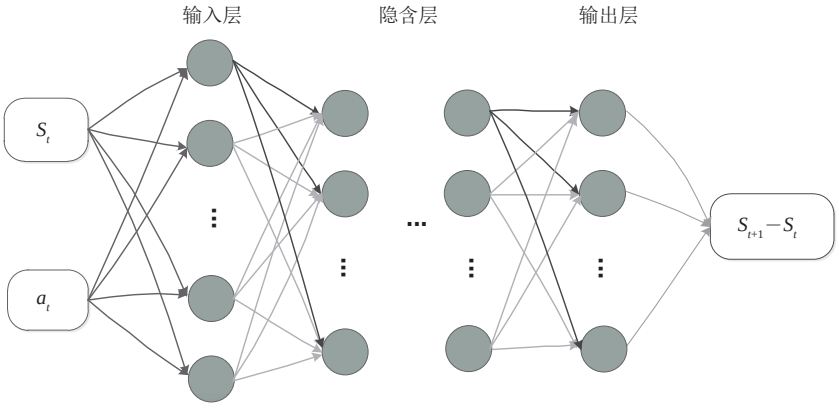

图 3　预训练网络的结构

Fig. 3　The structure of pre-training network

环境的过程中，有时也会产生很多没有实际意义的经验数据，如在有的试验过程中行驶的路程过短。过多冗余的数据和无实际意义的数据会给我们的交互学习带来很多障碍，因此本文方法对经验池中数据的处理做了进一步的改进。在尽量保证数据独立同分布的前提下，通过聚类再采样的方法，使用更少的经验池数据，更高效地训练我们的自动驾驶策略模型。首先，本文方法还是借用专业选手的数据，获得一个初步的分类模型。即将道路属性和驾驶动作放到同一空间进行归类。图 4 所示为将状态和动作数据分为 5 类的场景。由于是高维度的数据，不便于展示，于是只选取了一维连续的子空间 trackPos 进行展示。图 4 中 x 轴为状态序列编号，y 轴为 trackPos 传感器的值，这一维数据显示，其相近物理空间下的值是属于同一类，侧面论证本文的聚类分析是正确的。当 Agent 从经验池采样数据进行训练时，首先会根据状态和动作的值，计算与每一类别的距离，并将其分配到最近的一类。处理完所有试验序列后，再根据同类别下的数据按比例进行均匀间隔采样，最后再整合成一个新的子数据集，最后从这个子数据集内选取数据训练网络模型，该方法记为 DQFE-C 算法。聚类分析处理的详细算法步骤如表 1 所示。

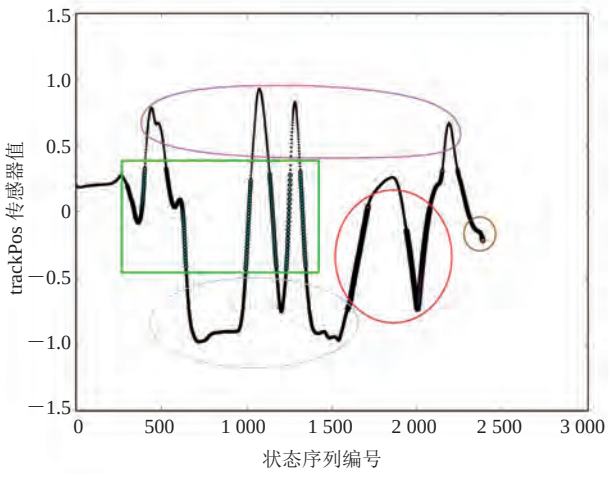

图 4 聚类分析示意图

Fig. 4 The illustration of clustering analysis

3.4 模型的交互式学习和测试

在每一个控制周期，本文方法先通过构建的交互模块观测 TORCS 服务器反馈的车辆和道路的最新状态。基于此状态，本文方法首先会计算上一状态对应控制动作的立即回报，然后再用我们的网络度量当前状态，并给出相对应的动作反馈给 TORCS 服务器，其交互式的操作过程如图 5 所示。

在每次试验结束的时候，再调用 Agent 中的学习单元用 Rprop[26] 算法对网络模型进行训练，训练时采用的数据集是由 3.3 节描述的聚类分析

表 1 聚类分析(DQFE-C)算法

Table 1 The algorithm of clustering analysis

步骤	具体操作
输入	初始化 K-means 分类模型 M_k 以及类别参数 N_C
过程	（1）对经验池中的实验序列进行类别信息分析，每个序列的处理过程如下： （a）根据 M_k 计算每个 state 的所属类别 （b）统计整个实验序列的类别信息 （c）根据每类的数量，基于均匀分配的原则，在每类里面均匀采样 M_{bs} 比例的样本序列，并对相应的状态-动作对计算 Q 值，将（s, a, Q）存入数据集 DS_s （2）用数据集 DS_s 用 Rprop[26] 训练算法对我们的网络进行训练 （3）用前文中经验池增量表达式（2）检查经验池是否超出范围，并做相应处理
输出	网络模型

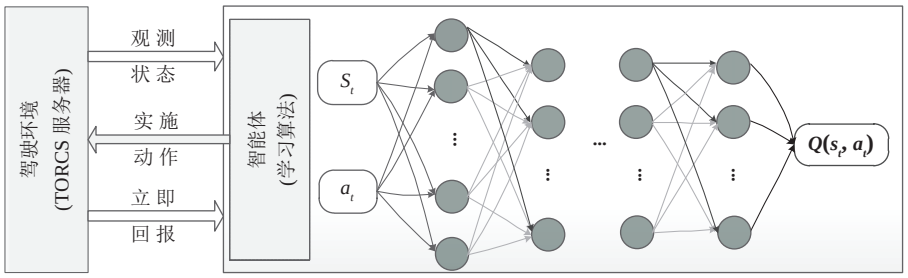

图 5　交互式学习模型

Fig. 5　The model of interactive learning

所得到的。每次训练之后调用测试模块来检测当次学习后模型的性能。在学习过程中，本文方法期望控制的车辆能够以较好的行驶速度跟踪道路的中线，且尽量不频繁的急速变换方向，于是构造了如式(3)所示的奖励回报函数 r。

$$r=\Delta d\times\cos(\alpha\times\theta)\times\mathrm{sgn}(\mathrm{trackPos}-\eta) \quad (3)$$

其中，Δd 为相邻状态跑过的有效距离；θ 为行驶方向与道路切线夹角；α 为权重缩放因子；η 为车辆距离路沿的距离阈值；sgn 符号函数在 trackPos 的值大于 η 的时候，取值无穷小，意在表达车辆太靠近道路边界时的惩罚。

4　实验设置

4.1　自动驾驶仿真平台环境配置

基于 TORCS 开源平台及其发起的 SCR 竞赛(Simulated Car Racing Championship)所用的框架模型，本文构建了一个自动驾驶算法研究仿真平台，框架如图 6 所示。本文的算法同样以一个客户端的形式通过 UDP(User Datagram Protocol)通信方式与 TORCS 服务器建立联系，在每一个 20 ms 的控制周期，算法接收来自服务器感知的车辆状态和环境道路信息，然后算法计算出控制指令反馈给服务器。TORCS 服务器根据接收到的指令，模拟车辆和道路的物理变化过程，然后传递到下一个控制状态。以此类推，直到车辆完成预设的路程任务或者发生紧急情况(如碰撞损坏、油箱缺油等)重新开始。

TORCS 服务器提供 19 类模拟车辆状态信息的传感器信号，鉴于车辆自动驾驶仿真系统的普适性，本文构建了一个 TORCS 服务器的交互模块，用作智能体(Agent)与环境及其他模块的数据交互。如图 2 中策略学习模型的架构中所示。通过这个交互模块，我们选取了 4 类传感器信号来表征车辆的状态。如图 7 所示，trackPos、角度、测距传感器组 track 和 speedX 分别表示车与道路中线的距离、车前进方向与道路切向的夹角、安装在车前方的 19 个可配置位置的距离探测器的值和车辆在道路切向上的速度分量。所有输入到策略模型的传感器值均归一化到［－1，1］的范围。

4.2　训练和测试道路选择

训练自动驾驶策略模型时选用的道路是图 8(b)中展示的 CG Speedway number 1 地图，为更加全面地评估自动驾驶策略模型的性能，本文方法选取了三个不同复杂度的地图对模型进行测试。图 8 中展示的是测试用的三个地图，从(a)到(c)道路的路程越来越长，同时道路元素的复杂性也越来越高。

4.3　模型参数配置

模型中车辆方向盘的驾驶动作集合为[－0.5, －0.23, －0.13, －0.07, －0.02, 0, 0.02, 0.07, 0.13, 0.23, 0.5]，其中各值的大小是标准化后的数值。车前方测距传感器的配置角度分别为[－45, －19, －12, －7, －4, －2.5, －1.7, －1, －0.5, 0, 0.5, 1,

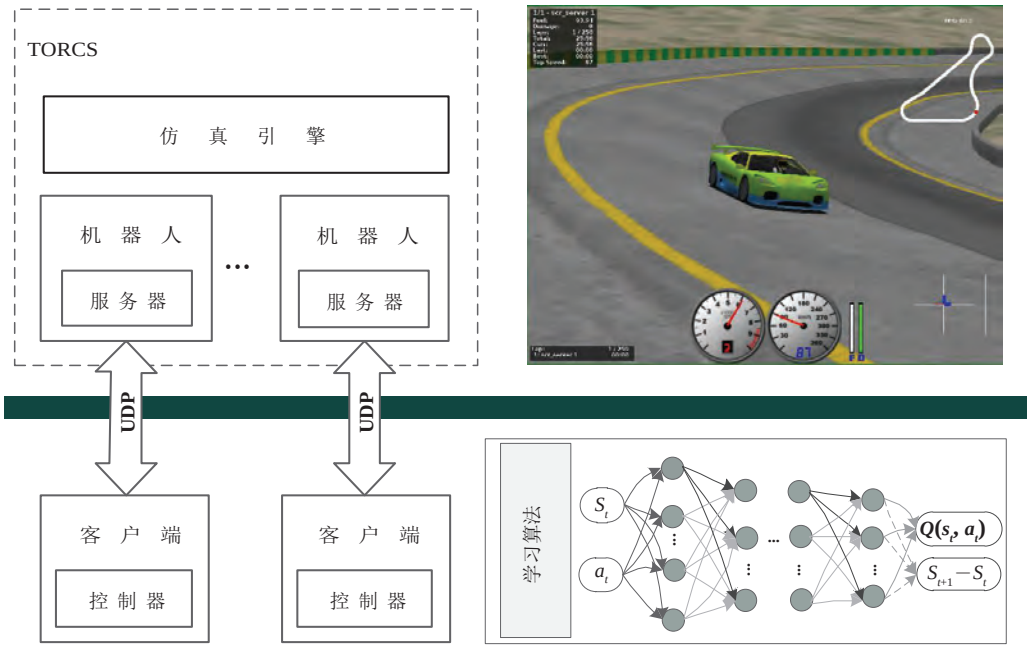

图 6 自动驾驶仿真平台架构

Fig. 6 The architecture of the self-driving simulation platform

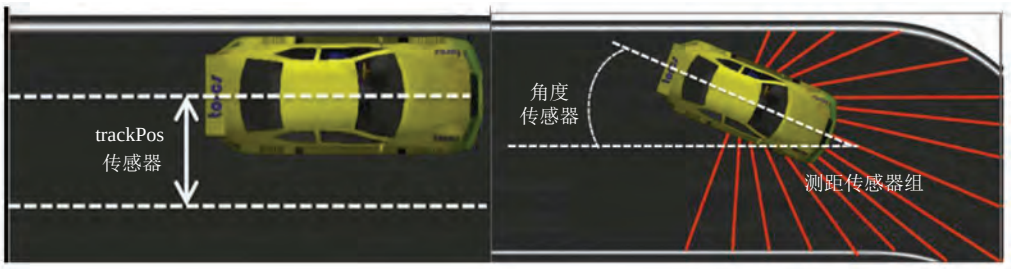

图 7 传感器示意图

Fig. 7 The illustration of sensor

(a) E-地图 5 (1 621.73 m) (b) CG Speedway number 1 (2 057.56 m) (c) E-地图 4 (7 041.68 m)

图 8 训练和测试用道路

Fig. 8 The demonstration of track in training and testing

表 2　参数配置

Table 2　Parameter configuration

参数名称	值	物理描述
τ_{prel}	80	预训练的迭代次数
M_{bs}	0.04	每次训练从经验池中采样数据占总量的比率
μ_{rms}	20 000	经验池中保留状态序列对个数的最大值
K_{num}	20	经验池中保留试验序列次数的最大值
N_{train}	20	交互学习中每次训练迭代的次数
Threshold	0.6	奖励回报函数中车辆距离路沿的距离阈值
K	5	聚类默认类别数
网络结构	20-11-10-1	网络中每层的节点数

1.7, 2.5, 4, 7, 12, 19, 45]（单位：度）。在本文训练与测试过程中，其他主要的参数配置如表 2 所示。

5　实验结果

5.1　计算资源消耗对比

从学习的时间上来看，随着训练的迭代，本文提出的方法单次训练时间是会收敛的，如图 9 所示。在 DQFE 算法中引入约束的经验池回放后，每次训练的迭代时间会收敛在 320 s 左右。本文进一步引入聚类分析再采样数据集训练的方法后，每次训练的迭代时间最终会稳定在 65 s 左右。然而，NFQ 算法随着交互数据的增加，所花费的计算时间也越来越多，300 次试验的平均训练时间为 820 s 左右，故本文提出的 DQFE-C 算法可以降低大约 92% 的时间消耗。实验数据表明，本文提出的算法有效，并且基于聚类思想的措施可以使算法的时间消耗进一步降低到 DQFE 算法的 1/5 左右。综上表明，本文方法所消耗训练时间的优势是明显的。

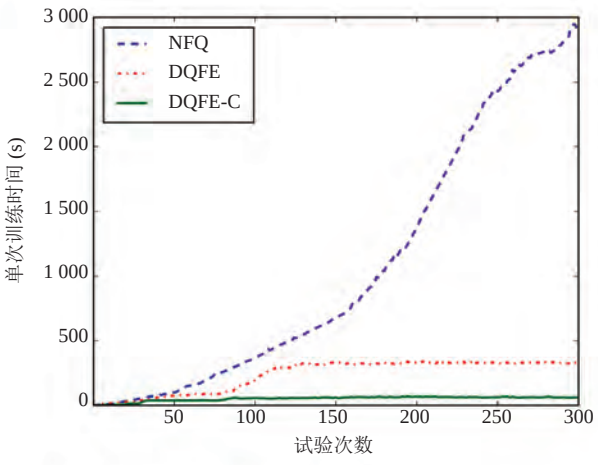

图 9　不同算法的单次训练时间对比

Fig. 9　The comparison of training time between different algorithms

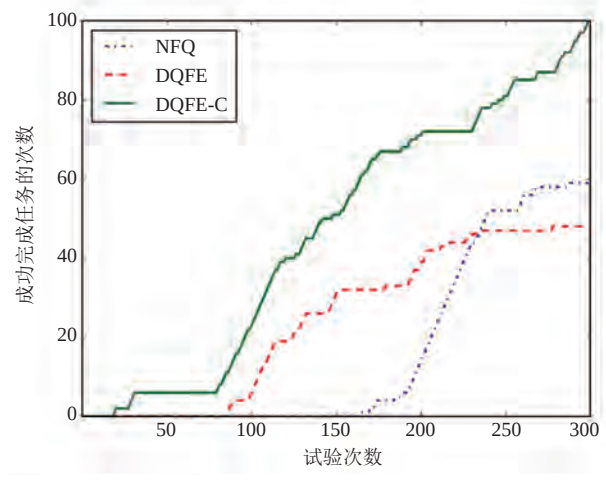

图 10　不同算法成功完成任务的对比

Fig. 10　The comparison of successfully finished tasks between different algorithms

5.2　模型的学习效率及控制性能对比

本文提出的算法在降低单次训练时间的基础上，训练的次数也得到保证，并且在引入聚类分析的改进方法中，学习效率也得到改善。分别用三种方法进行 300 次试验，我们记录了实验过程中获取跑完全程策略的次数，如图 10 所示。从

表 3 算法控制性能对比

Table 3 Performance comparison between different control algorithms

方法	距离道路中线的平均误差(m)	标准差(m)	误差范围（m）	50 次测试中完成全程的比率(%)
NFQ	0.251 3	0.125	[−0.210 8, 0.536 1]	66
DQFE	−0.088 4	0.153 7	[−0.372 0, 0.186 8]	98
本文方法（DQFE-C）	−0.278 1	0.013 7	[−0.597 7, −0.001 9]	100

记录的数据来看，本文改进的 DQFE-C 方法，在第 25 次试验处即学会行驶当前整个道路的控制策略，而 NFQ 算法却用了 160 次试验。

三种算法分别训练 300 次以后，将获得的最新策略在同一地图(图 8(b)中 CG Speed way number 1)下进行 50 次测试，分别计算车辆行驶的轨迹和道路中线的平均距离及标准差来进行对比。从表 3 的对比结果来看，与 NFQ 和 DQFE 算法对比，本文提出的算法均具有一定的优势。从数据上可以看到，本文改进的 DQFE-C 算法在控制车辆的稳定性上有很大提高，并且学习到驾驶策略模型的稳定性大大提高，学习到的策略模型在同一地图下可以 100% 的通过率驾驶完全程，而 NFQ 算法学习到的策略在 50 次测试中只有 66% 的测试通过率，DQFE 算法也有一次失误发生。由于训练和测试道路均为多处连续左转路段(如图 8(b)所示)，本文提出的 DQFE-C 算法学习到的控制策略具有靠右行驶的趋势，其行驶轨迹稳定在靠右边的 1/4 处，故导致了平均误差为−0.278 1。关于控制误差的范围，由于在左右转弯的时候，学习到的控制策略会考虑以某种切线角度过弯，而非一直跟踪中线，所以在过弯的时候可能会出现较大的控制误差。

6 讨 论

在这一部分，将讨论聚类数量对本文提出方法的影响，主要包括策略模型训练的效率以及经训练得到策略模型的控制效果和泛化能力。

6.1 聚类数量对模型控制效果的影响

为测量不同聚类数量对策略模型学习的影响，实验条件设置为聚类类别不同、其他参数均一致，进行策略网络模型的学习，均以第 10 次获取到成功跑完全程的策略模型为参考节点。在 50 次测试中，我们记录了该策略模型驾驶车辆的行驶轨迹与道路中线的平均误差值和标准差，结果如图 11 所示。并且记录了在多种不同聚类类别下的训练花费的总时间，如图 12 所示。然后，用获得的最新策略在同一个道路上进行测试，来评测其模型的控制性能。

随着聚类分析中采用类别数目的增加，得到的策略模型跟踪中线目标的控制效果呈调节式趋近的现象。然而，当聚类数目过多的时候，控制的抖动性呈增大趋势。并且从图 12 的训练时间

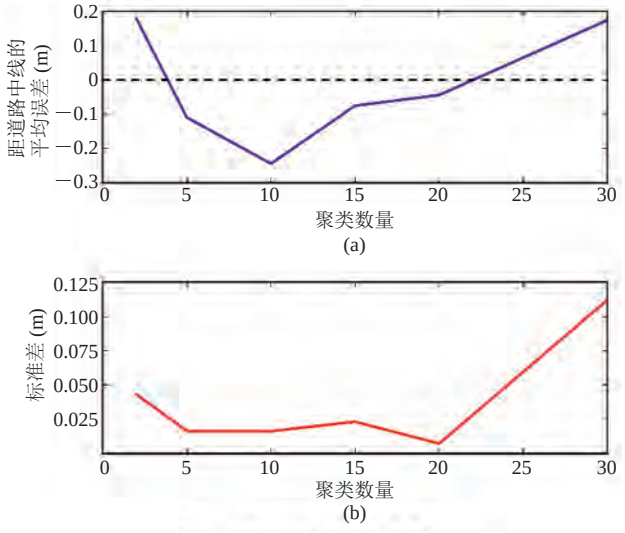

图 11 不同类别聚类的影响

Fig. 11 The effect from different number of clusters

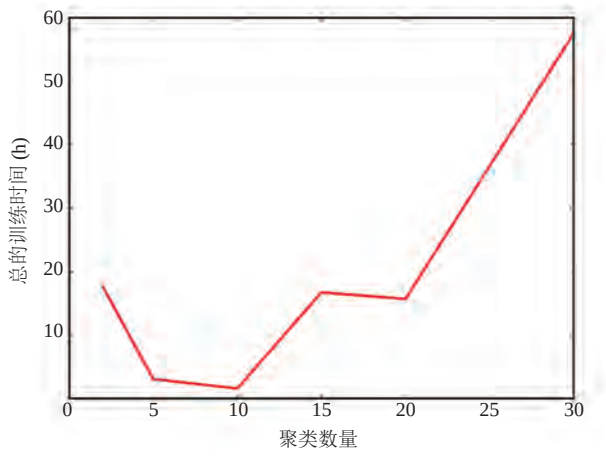

图 12　不同类别聚类情况下，第 10 次获得模型的总时间

Fig. 12　The entire training time of 10th strategy from different number of clusters

上来看，当聚类超过 20 的时候，训练时间的代价是巨大的。综上来说，聚类类别数在 5～15 的时候，可以获得比较好的控制性能和学习效率。

6.2　聚类数量对模型泛化能力的影响

前面实验的训练和测试均是在同一地图(图 8(b)中 CG Speed way number 1)进行的。为进一步评测聚类数量对模型泛化能力的影响，本实验将在图 8(b)中训练得到的模型，分别放在比训练地图稍简单和稍复杂的道路上进行泛化能力的测试，不同地图的具体参数如图 8 所示。

表 4 为 50 组测试的归一化数据结果，其中数值 1.000 表示通过了 50 次全部的测试。数据表明，聚类类别过少的情况下，模型的泛化能力相对要差，并且在类别为 10 的时候，DQFE-C 算法学习到的模型具有最好表现。在稍复杂的地图 3 上，经聚类分析得到策略模型最好的实验结果与没有采用聚类分析时的 DQFE 算法相比，平均行驶距离提高了 73.4%。分类过细的时候，在复杂地图并不能带来性能的提升，可能是由于在新的、复杂地图上，有的状态不能很好地归属到特定类别上去。

由于本文在实验中设置，当模型车即将碰上道路边沿的时候，即终止此次试验。故实验获取可行策略后，可能不会一直稳定收敛，即学习到一个驾驶策略后可能会再次出现不能行驶完预设任务的情况。并且由于本文方法在处理方向盘控制量的时候，使用的仍是离散的量，故导致模型车的驾驶动作可能会略微生硬，后续的优化过程中应该要更多地考虑到如何使用连续的驾驶动作进行控制。

7　结　论

鉴于车辆驾驶所处环境的极其复杂和易变性，开发一个具有 L-4 等级的自动驾驶系统仍然是一个极具挑战性的工程。然而，深度强化学习的通用性能给自动驾驶提供了一个新方向。本文提出一种基于深度强化学习的自动驾驶策略模型学习的新方法：首先基于专业司机的历史数据信息

表 4　聚类数量对策略模型泛化能力的影响

Table 4　The effect on generalization ability with different cluster

算法	类别的数量	地图 1 (E-道路 5)	地图 2 (CG Speedway number 1)	地图 3 (E-道路 4)
DQFE	**1**	**1.000**	**0.989**	**0.241**
DQFE-C（本文算法）	2	0.740	0.987	0.116
	5	0.743	1.000	0.119
	10	**1.000**	**1.000**	**0.975**
	15	1.000	1.000	0.636
	20	1.000	0.993	0.168
	30	1.000	1.000	0.266

对模型进行预训练，然后结合经验池回放技术和聚类再采样的处理方式，采用强化学习的在线学习方法对深度网络模型进行训练，有效地提高了策略模型的泛化能力。实验结果显示，文章提出的方法与神经网络拟和 Q-迭代算法相比，在 300 次实验中降低 92% 左右的时间消耗，同时在 50 次测试中，稳定性能提高约 34%。泛化能力测试数据表明，以复杂度略高于训练集的测试道路长度为基准，与 DQFE 算法相比，采用聚类分析的方法可以使策略模型的平均行驶距离提高 73.4%。

参 考 文 献

[1] 汽车维修与保养. 2016 年底全国保有机动车达 2.9 亿辆 [J]. 汽车维修与保养, 2017 (02): 16.

[2] 中国交通事故赔偿网. 2014 年全国道路交通事故数据统计 [EB/OL]. [2017-03-08]. http://www.peichang.cn.

[3] Touran A, Brackstone MA, McDonald M. A collision model for safety evaluation of autonomous intelligent cruise control [J]. Accident Analysis & Prevention, 1999, 31(5): 567-578.

[4] 杨帆. 无人驾驶汽车的发展现状和展望 [J]. 上海汽车, 2014 (3): 35-40.

[5] 翁岳暄, 多尼米克•希伦布兰德. 汽车智能化的道路: 智能汽车、自动驾驶汽车安全监管研究 [J]. 科技与法律, 2014 (4): 632-655.

[6] Sutton RS, Barto AG. Reinforcement Learning: An Introduction [M]. Cambridge: MIT Press, 1998.

[7] Riedmiller M. Neural fitted q iteration-first experiences with a data efficient neural reinforcement learning method [C] // European Conference on Machine Learning, 2005: 317-328.

[8] Hinton GE, Osindero S, Teh YW. A fast learning algorithm for deep belief nets [J]. Neural Computation, 2006, 18(7): 1527-1554.

[9] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks [C] // Advances in Neural Information Processing Systems, 2012: 1097-1105.

[10] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning [J]. Nature, 2015, 518(7540): 529-533.

[11] 陈硕. 深度学习神经网络在语音识别中的应用研究 [D]. 广州: 华南理工大学, 2013.

[12] Hafner R, Riedmiller M. Reinforcement learning in feedback control [J]. Machine Learning, 2011, 84(1): 137-169.

[13] Mnih V, Kavukcuoglu K, Silver D, et al. Playing atari with deep reinforcement learning [J]. arXiv: 1312.5602, 2013.

[14] Bojarski M, Del Testa D, Dworakowski D, et al. End to end learning for self-driving cars [J]. arXiv: 1604.07316, 2016.

[15] Sallab AE, Abdou M, Perot E, et al. End-to-end deep reinforcement learning for lane keeping assist [J]. arXiv: 1612.04340, 2016.

[16] Chae H, Kang CM, Kim BD, et al. Autonomous braking system via deep reinforcement learning [J]. arXiv: 1702.02302, 2017.

[17] 刘赫. 动物行为训练的理论基础 [J]. 中国动物保健, 2014(2): 23-25.

[18] Mitchell T. Machine Learning (McGraw-Hill International Edit) [M]. New York: McGraw-Hill Education, 1997.

[19] Kaelbling LP, Littman ML, Moore AW. Reinforcement learning: a survey [J]. Journal of Artificial Intelligence Research, 1996, 4(1): 237-285.

[20] Tesauro G. Temporal difference learning and TD-Gammon [J]. Communications of the ACM, 1995, 38(3): 58-68.

[21] Lin LJ. Reinforcement learning for robots using neural networks [D]. Pittsburgh: Carnegie Mellon University, 1993.

[22] Xia W, Li H, Li B. A control strategy of autonomous vehicles based on deep reinforcement learning [C] // 2016 9th International Symposium on Computational Intelligence and Design (ISCID), 2016: 198-201.

[23] Wymann B, Espié E, Guionneau C, et al. TORCS, the open racing car simulator [OL]. [2017-03-08]. http://torcs. sourceforge. net.

[24] 何宁, 赵治国, 朱阳. 基于 TORCS 平台的虚拟车辆仿真系统开发 [J]. 机械设计与制造工程, 2010, 39(15): 37-41.

[25] Anderson CW, Lee M, Elliott DL. Faster reinforcement learning after pretraining deep networks to predict state dynamics [C] // 2015 International Joint Conference on Neural Networks (IJCNN), 2015: 1-7.

[26] Riedmiller M, Braun H. A direct adaptive method for faster backpropagation learning: the RPROP algorithm [C] // IEEE International Conference on Neural Networks, 1993: 586-591.