基于强化学习和注意力机制的车辆换道研究

沈炼成，林镇阳，韩立君

***Abstract*— 这里是华丽丽的摘要**

I. 简介

II. 任务描述与分析

A. 任务描述

整体任务为根据输入的车辆周围情况，借助深度强化学习算法得到高层的指令规划。再借助仿真器内部的运动规划器，将高层指令转化为具体的轨迹让底层控制器有效跟踪。最终实现汽车的超车换道。评价标准如下:

- 尽可能少碰撞
- 尽量保持车道线内平稳行驶
- 速度越快越好
- 有前车阻挡能尝试安全换道

B. 仿真环境描述

整体实验基于 *highway_env* 开发，具有较强的灵活性。下面分别针对状态空间、动作空间等进行叙述。

1) 状态空间: 在环境中，状态空间可以选择底层的低维向量输入，也可以选择高维的图像输入和占用格作为输入。下面重点叙述使用低维向量输入和图片输入的基本情况。

当使用低维输入时，传入 15 辆车的坐标、速度、倾斜角度等信息表示出来，包括 $x, y, vx, vy, cos_h, sin_h$。传入一个大小为 $[15, 7]$ 的数组。方便后面网络进行处理，其中第一行表示的是本车的的各种信息。

使用图片输入时，传入大小为 $[600, 150]$ 的图片，同时将图片转化为灰度图。由于一张图片难以获得车辆的完整信息（如速度、运动方向等难以推测）, 因此将过去 4 帧叠放在一起作为状态观测输入，便于从观测推出状态。

后续实验证明，向量输入的归一化以及将两者信息整合同时输入可以显著提升换道效果。

2) 动作空间: 仿真环境可以将动作设置为连续的油门和方向盘角度, 但连续空间训练收敛难度会增大。考虑到算力限制，因此使用离散动作进行控制，再将离散动作送入规划器中生成连续动作。在仿真器中，可以分为 5 个离散动作——左转、保持车道、右转、加速、减速。

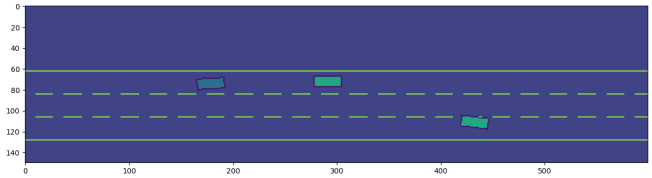

Fig. 1: 图像输入

分别用动作 a_0 到 a_4 表征。这五个动作共同构成了仿真环境的动作空间，下层规划器根据决策指令对车辆动作进行规划控制。

3) 奖励设置: 环境中奖励设置较为灵活，主要分为三个部分，碰撞惩罚、车速奖励、靠右奖励。系统默认碰撞惩罚为 -1，无碰撞为 0。车速奖励为 $r_v = 0.4\frac{(v_t - v_{min})}{(v_{max} - v_{min})}$ 其中，$v_{max} = 30m/s, v_{min} = 20m/s$。此外还有靠右奖励，奖励系数为 0.1。后面我们会讨论调整奖励系数对车辆的影响，实验发现，删除靠右奖励会导致车辆换道次数显著增加。

C. 任务分析

根据任务需要我们设计强化学习算法，将低维或高维输入转化为离散的汽车换道决策指令。即首先需要对观测量进行提取推测，得到状态量。再将状态量作为输入得到动作。而近来深度强化学习的兴起让这两个环节可以直接联合训练，提升了学习效率。

在学习算法方面，我们采用了 Dueling DDQN 技术加入了一些训练技巧，能够有效提升收敛速度和训练稳定性。具体方法我们会在下节进行详细叙述，同时与之前的 Dueling DQN，DQN 进行比较，证明我们方法的有效性。同时针对输入，我们分别探讨了对于向量输入、图片输入和混合输入的效果。此外，针对不同奖励设置进行了消融实验。

III. 强化学习算法

可以将超车换道问题转化为一个马尔可夫问题，由 $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ 来表示，分别代表了状态空间、动作空间、转移概率、奖励函数、折扣因子。

根据前一节的分析，状态空间、动作空间、奖励函数均已得到。折扣因子 γ 设为 0.99。不妨设 t 时刻的回

报 G_t 为:

$$G_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i} \tag{1}$$

G_t 表示了从 t 时刻到结束的总的奖励，整个强化学习过程的目标即为最大化回报。但转移概率未知，因此采用无模型强化学习的方法，常用的无模型方法为 Q 学习算法。针对某个策略，定义 Q^π 为状态动作对好坏的评价标准，可以借助前面的回报定义为：

$$Q^\pi = \mathbb{E}_\pi[G_t | s_t = s, a_t = a] \tag{2}$$

同时使用经验回放池，使得训练数据满足独立同分布假设，以及充分利用过往探索经验，方便进行深度学习。使用目标网络，防止计算 TD 误差时目标变动导致更新困难。使用 $\epsilon - greedy$ 算法，ϵ 值逐渐衰减，来平衡整个利用与探索的过程。整个 DQN 算法流程如下所示：

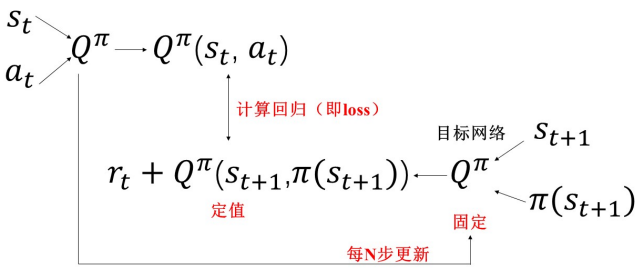

Fig. 2: DQN 算法框架

但是传统的 DQN 算法会存在 Q 值过高估计的问题，因此引入了 Double DQN, 将决策动作生成网络和状态动作对估值网络分开，可以一定程度缓解过估计问题。此外在训练数据中，很多状态并不需要估计每个动作的值，使用 Q 函数可能引入较多噪声。因此将 Q 函数分解为状态函数 V 和优势函数 A 最终 Q 值为两个函数的叠加。此外由于一个 Q 值可以有无数种分解方式，如果直接估计会导致难以收敛。因此对优势函数去均值保证分解的唯一性。最终 Q 值可以表示为：

$$Q(s,a) = V(s) + (A(s,a) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a')) \tag{3}$$

此外我们经过研究发现，获得终止信号有两种情况，撞车或达到仿真时间（50 步）。但这两种情况实际上是不一样的。撞车后，我们不学习如何从撞车后恢复（相当于车辆已死亡），此时设为终止没有问题。但是只是到达了仿真步数，此时车辆其实是还可以正常行驶的，如果直接将该状态存为终止状态，会导致 Q 值的估计震荡不容易收敛。因此我们将原有算法进行了修改，加入了死亡判断。即每次动作得到终止信号时，会判断信号来源于撞车还是到达仿真步数，如果是来源于撞车，则存入经验池的 $done = True$，否则仍设为 $False$。具体算法如算法 1 所示。

算法 1 加入死亡判断的 Dueling DDQN 算法

初始化大小为 N 的经验回放池 D，初始化随机选择概率 ϵ
使用随机参数 θ 初始化动作价值函数 Q
使用参数 θ^- 初始化目标动作价值函数 $\hat{Q}$, 其中 $\theta^- = \theta$
for episode=1,episode<=M,episode++ **do**
    初始化状态序列 $s_1 = x_1$
    for t=1,t<=T,t++ **do**
        以 ϵ 的概率进行随机动作选择 a_t，否则选择 $a_t = argmax_a \hat{Q}(s_t, a; \theta)$
        执行动作 a_t 得到奖励 r_t 和下一步的状态 s_{t+1}，以及是否结束 d_t
        if $d_t == True$ 且发生了撞车 **then**
            $\hat{d}_t = True$
        else
            $\hat{d}_t = False$
        end if
        将数据 $(s_t, a_t, r_t, s_{t+1}, \hat{d}_t)$ 存入经验回放池 D 中
        随机在经验回放池中选取一个 $batch$ 的 $(s_j, a_j, r_j, s_{j+1}, , \hat{d}_j)$
        令 $y_j = \begin{cases} r_j & \hat{d}_j = True \\ r_j + \gamma \max_{a'} \hat{Q}(s_{j+1}, a'; \theta^-) & \text{其余情况} \end{cases}$
        设定损失函数为 $(y_j - Q(s_j, a_j; \theta))^2$，采用随机梯度下降的方法更新参数 θ
        更新 ϵ，更新公式为 $\epsilon_t = \epsilon_{max} - (\epsilon_{max} - \epsilon_{min}) e^{-\frac{t_{sofar}}{n_{decay}}}$
        每 C 步令 $\hat{Q} = Q$
    end for
end for

IV. 基于注意力机制的 Dueling DDQN 网络框架

在上述加入死亡判断的 Dueling DDQN 强化学习算法的指导下，我们结合了不同类别的状态作为输入，同

时对向量输入和图像输入分别引入了不同的注意力机制，整体网络框架如图 3所示：

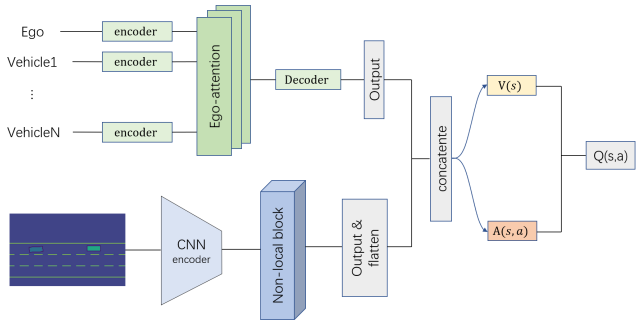

Fig. 3: 基于注意力机制的 Dueling DDQN 网络框架 [?]

A. *Ego-attention*

对于基于向量的状态输入，这里采用的注意力模型是 ego-attention [?]。这种注意力模型是社会注意力机制的一种变形，可以使得智能体更加关注距离其较近或者容易发生碰撞的车辆。

基于 ego-attention 注意力机制的模型如图 4所示，其可以用来表示 DQN 算法或者 Dueling DDQN 算法中的 Q 函数。网络首先由线性编码层组成，所有编码层

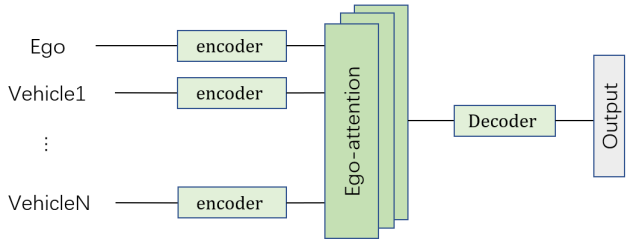

Fig. 4: 基于 ego-attention 的网络结构

的权重相同。经过线性编码层后每个实体的特征维度为 d_x，编码后的特征送入由多头堆叠而成的 ego-attention 层。该层类似于多头自注意力层 [?] 但是仅有关于当前车辆的单个输出，也即仅有当前车辆的询问编码。Ego-attention 头的结构如图 5所示，为了选择基于环境车辆的子集，当前实体首先发出单个询问 $Q = [q_0] \in \mathbb{R}^{1\times d_k}$，由当前实体编码特征经过线性投影 $L_q \in \mathbb{R}^{d_x\times d_k}$。这个询问之后与键的集合 $K = [k_0, \ldots, k_N] \in \mathbb{R}^{N\times d_k}$ 进行比较，其中包含每个实体的描述性特征 k_i，同样是由共享权重的线性映射 $L_k \in \mathbb{R}^{d_x\times d_k}$ 计算得到。询问 q_0 与任意键 k_i 间的相似性由它们之间的点积 $q_0 k_i^T$ 衡量，并由 $1/\sqrt{d_k}$ 进行放缩，最后由 softmax 算子 σ 进行归一化。得到的注意力矩阵用来对输出值的集合 $V = [v_0, \ldots, v_N]$

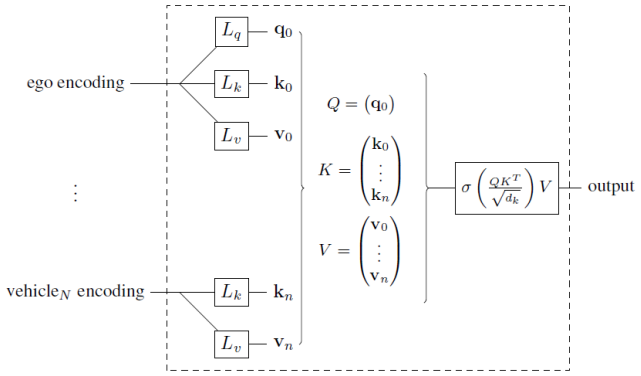

Fig. 5: ego-attention 头的结构

进行加权融合，其中每个值 v_i 是经过共享权重的线性映射 $L_v \in \mathbb{R}^{d_x\times d_v}$ 得到的特征。综上，每头的注意力计算可以写作：

$$output = \sigma\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

最后所有头的输出经过一个线性层进行结合，整个过程具有置换不变性：一个置换 τ 会改变公式中键 K 与值 V 行的顺序，但是两者的对应关系仍然得以保持。它也可以很自然地解释当前车辆与环境车辆的交互关系。

B. *Non-local block*

对于基于图像的状态输入，这里使用的注意力机制为 non-loacl block [?]。传统的卷积网络是处理局部区域的操作，为了从图像中获得较大范围的信息，需要通过叠加多层卷积来不断扩大感受野，这样的操作是复杂的，而且需要考验网络设计能力，最终得到的效果也不尽如人意，很难获得较大距离像素间的相对关系。受计算机视觉中非局部均值（non-local means）的启发，non-local block 被提出用于捕捉长距离依赖，与传统的卷积神经网络相比，它可以计算两个任意位置之间的交互，进而直接捕获长距离的依赖关系。另外，这一操作是非保持了输入输出的大小，可以很容易地作为一个模块与其他操作结合。

在本文中使用的 non-local block 如图 6所示，x 是输入，z 是 non-local block 的输出，y 是 non-local 操作的结果，其关系式为：

$$z_i = W_z * y_i + x_i \tag{5}$$

W 是对 y_i 的加权, 与输入 x 相加形成残差连接。

$$y_i = \frac{1}{C(x)}\sum_{\forall j} f(x_i, x_j)g(x_j) \tag{6}$$

$f(x_i, x_j)$ 用来计算 i 和所有可能关联的位置 j 之间的关系，其计算方式可以有多种，在本文中我们使用简单的

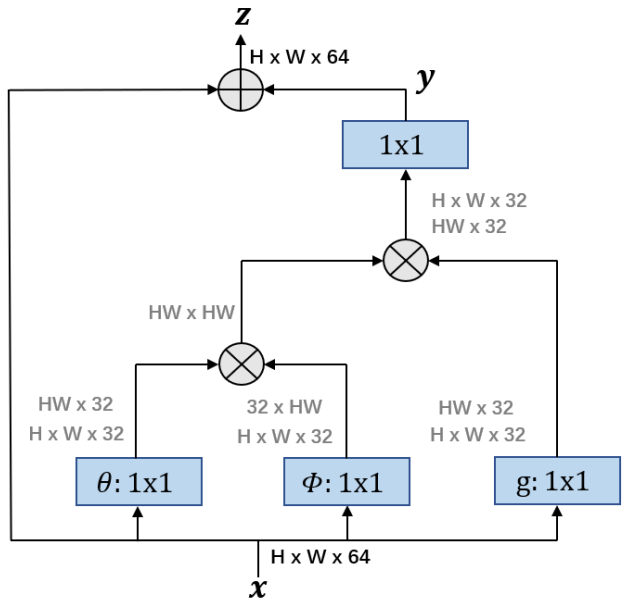

Fig. 6: non-local block 网络结构

点乘，如式 7；$g(x_j)$ 用于计算输入信号在 j 位置的特征值；$C(x)$ 是归一化参数。

$$f(x_i, x_j) = \theta(x_i)^T \phi(x_j) \tag{7}$$

$\theta(x_i)$、$phi(x_j)$、$g(x_j)$ 均为线性变换，通过空间 $1 * 1$ 的卷积实现。

基于 non-loacl block 机制，图片作为状态输入的网络模型如图 7所示，与上文的 ego-attention 模型相似，基于 non-local block 的网络可以用来拟合 DQN 算法或者 Dueling DDQN 算法中的 Q 函数。网络输入包括当前车辆和环境车辆的为图像信息，通过一个浅层的 CNN 网络初步提取图像特征，然后使用 non-local block 提取任意两个像素之间的依赖关系，从而获得当前车辆与环境车辆之间的关系。

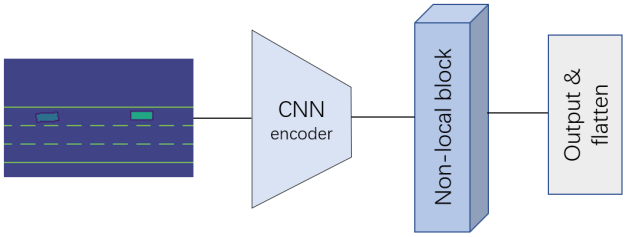

Fig. 7: 引入 non-local block 的网络结构

V. 实验分析

在实验部分，我们将奖励函数分为三个部分

1) 安全项：碰撞惩罚 -1，无碰撞为 0

2) 效率项：令 $r_v = 0.4\frac{(v_t - v_{min})}{(v_{max} - v_{min})}$ 其中，$v_{max} = 30m/s, v_{min} = 20m/s$

3) 靠右项：如果靠右行驶，则奖励为 0.1，否则为 0

最终的奖励函数为三者的叠加。下面我们将针对不同输入，不同奖励函数设置，不同算法进行一系列的对比试验。同时将我们的带有死亡判断的 D3QN 进行消融实验，以验证我们算法的有效性。

A. 不同输入

我们分别尝试了向量输入、图像输入、和混合输入三种形式。并且针对他们是否加上注意力机制方法进行了广泛地比较。训练 1000 轮之后进行测试，测试结果取十次测试的平均成绩。最终结果如表 I所示：

TABLE I: 不同输入结果比较

输入类型	步长	奖励	换道次数	安全率
向量	49.1	38.19	16.4	0.9
图像	49.5	44.4	23.5	0.9
向量 + 注意力	50.0	36.5	46.1	1.0
图像 + 注意力	43.4	39.3	7.7	0.7
混合 + 注意力	50.0	43.1	9.5	1.0

经过对比我们可以发现在加入注意力机制之后，可能会导致某些指标改善（如安全率，换道次数），但是也同时会导致一些指标的下降。而且在单一输入情况下，控制效果不稳定，可能导致结果方差较大。比如纯图像输入虽然在此轮测试中奖励最高，但是不稳定，有时平均奖励会到 35 左右。

这也许是因为注意力机制本身会导致网络收敛震荡，而且单一输入情况下，信息不完全，导致这种震荡更加明显。混合输入的注意力机制方法具有较强的稳定性，同时能够显著提升回报成绩。

B. 更改奖励系数

我们针对奖励系数，分别将效率项奖励系数变化为 0.2，0.6。将靠右行驶奖励分别变为 0 和 0.2. 来验证奖励系数对于车辆控制效果的影响。训练方法采用我们上个实验表现最好的混合输入 + 注意力机制方法。同样我们训练 1000 轮之后测试 10 轮取平均成绩，值得提出的是，在测试过程中我们采用的是同一奖励函数，即最原始的奖励函数来保证我们的实验结果有可对照性。最终结果如表 II。

根据结果可以得到，单纯增大效率奖励系数会使整个车辆控制更加激进，显著减小换道次数，增大速度。但是与此同时，车辆的安全性也会降低。反之减小效率

TABLE II: 不同奖励函数结果比较（测试时奖励函数相同）

奖励函数设置	步长	奖励	换道次数	安全率
增大效率奖励	48.6	44.7	5.6	0.8
减小效率奖励	50.0	40.53	42.6	1.0
增大靠右奖励	42.0	38.92	16.4	0.5
移去靠右奖励	49.4	41.06	18.6	0.9
原方案对照	50.0	43.1	9.5	1.0

奖励会使整个控制的安全性增大，但是车辆控制会趋于保守，换道次数明显增加、速度也会变慢。

针对靠右奖励而言，移去靠右奖励之后，换道次数有小幅上升，但是由于靠右奖励本身系数较小，因此对于整体效果影响没有增大效率项这么明显。而如果过于增大靠右奖励，将导致网络在靠右和躲避车辆之间权衡被打破，反而会降低效果。

C. 更改训练算法

同时我们还验证了不同算法对于结果的影响，分别采用了 DQN，DDQN，Dueling DQN 与我们的算法进行比较，同样也是训练 1000 轮之后测试 10 轮取平均值，最终结果如表 III。

TABLE III: 不同训练算法结果比较

训练算法	步长	奖励	换道次数	安全率
DQN	49.5	46.44	23.7	0.9
DDQN	48.8	36.83	45.1	0.9
Dueling DQN	46.5	41.09	9.3	0.9
原方案对照	50.0	43.1	9.5	1.0

经过对比发现，D3QN 算法在稳定性和训练效果方面都是目前而言最佳的。其余方法可能导致换道次数显著上升。

D. 消融实验

为了验证死亡判断对于网络的影响，我们对比了是否加入死亡判断对于最终训练结果的影响。其余设置条件均相同，最终结果如表所示。

致谢