

自动驾驶周围车辆行为预测与驾驶风格识别

5.1 概述

为实现安全舒适的自动驾驶,车辆行为预测任务需要对交通场景中的环境因素、历史观测以及社会交互等信息进行建模,以实现车辆未来时刻的运动状态预测。与弱势道路使用者(VRU)运动预测相比,由于驾驶规则和环境条件(道路几何形状)的约束,车辆行为随机性较小,例如,由于车身存在高惯性,车辆不能立即改变其轨迹,产生大的偏移。这些约束有助于降低行为预测问题的复杂度,使得车辆行为预测与 VRU 相比具有相对容易预测的一面。但车辆行为预测也有其自身的难点:首先是车辆行为之间相互依赖性强,场景中某一辆车行为的变化可能会影响其他车辆的行为;反之亦然。因此,进行车辆行为预测时需要特别关注周围车辆的行为。其次,交通规则和驾驶环境能够重塑车辆的行为。例如,在路上放置一个占道施工标志将完全改变接近它的车辆的行为。因此,车辆轨迹预测需要考虑交通规则和路网几何特征。此外,车辆的行为是多样的。例如,当车辆不改变方向减速行进到路口时,右转和左转都有可能发生。也就是说,车辆行为在本质上有着不确定性。

车辆本身的物理约束和环境因素的限制,车辆轨迹还与驾驶者的驾驶风格有关,因此本章将详细介绍车辆行为预测与驾驶风格识别的基本方法。对于周围车辆行为预测方法,本章将划分为传统方法和深度学习方法两部分进行介绍,前者将详细介绍基于卡尔曼滤波、概率分布以及隐马尔可夫的经典车辆行为预测方法的相关基础知识和建模方法;后者按场景将深度学习方法划分为静态单一场景和动态复杂场景方法,并基于编码-解码结构展开介绍不同方法。此外,本章还将介绍 3 种周围车辆驾驶风格识别方法,包括混合高斯模型、基于短期观测以及基于 LSTM-CNN 的识别方法。

5.2 基于统计模型的周围车辆行为预测

5.2.1 应用卡尔曼滤波的周围车辆行为预测

卡尔曼滤波算法(Kalman filter)是用于解决预测问题的最常用的递归滤波算法,它能够从过去的不准确和不确定的测量中,更准确地估计动态系统未来的状态。卡尔曼滤波的具体原理见附 A.2,本节不再赘述。下面介绍应用卡尔曼滤波进行车辆轨迹预测的具体步骤。

1. 轨迹表征

在车辆轨迹预测问题中,对于 N 个运动目标,目标 i 在不同时刻下的坐标位置信息用

坐标点集合 $O_{\text{obs}}^i = (\mathbf{p}_x^i, \mathbf{p}_y^i)$ 进行表征, 其中, $\mathbf{p}_x^i = (x_1^i, x_2^i, \dots, x_{t-1}^i, x_t^i)$ 与 $\mathbf{p}_y^i = (y_1^i, y_2^i, \dots, y_{t-1}^i, y_t^i)$ 分别表示第 i 条轨迹在欧氏空间中 x 方向和 y 方向上的坐标。

2. 状态方程

首先对车辆的理想状态进行估计, 已知车辆在 $t-1$ 时刻的位置 p 、速度 v 以及加速度 a , 利用加速运动模型对车辆在 t 时刻的位置和速度进行无误差的理想估计, 如式(5-1)和式(5-2)所示:

$$p_{t|t-1} = p_{t-1|t-1} + v_{t-1|t-1} \Delta t + \frac{1}{2} a \Delta t^2 \quad (5-1)$$

$$v_{t|t-1} = v_{t-1|t-1} + a \Delta t \quad (5-2)$$

其中, 位置 p 和速度 v 对应卡尔曼滤波的系统状态变量, 其状态方程用紧凑的矩阵形式可表示为

$$\mathbf{x}_{t|t-1} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} p_{t-1} \\ v_{t-1} \end{bmatrix} + \mathbf{B} \begin{bmatrix} \frac{1}{2} \Delta t^2 \\ \Delta t \end{bmatrix} \times \mathbf{u}_t = \mathbf{F} \mathbf{x}_{t-1|t-1} + \mathbf{B} \mathbf{u}_t \quad (5-3)$$

其中, $\mathbf{F} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$ 为状态变换矩阵, $\mathbf{B} = \begin{bmatrix} \frac{1}{2} \Delta t^2 \\ \Delta t \end{bmatrix}$ 称为控制矩阵, $\mathbf{u}_t = a$ 为对应的控制向量。

$\mathbf{x}_{t|t-1}$ 表示基于 $t-1$ 时刻理论预测的 t 时刻的状态, $\mathbf{x}_{t-1|t-1}$ 表示在 $t-1$ 时刻综合理论预测和实际测量得到的估计值, 其余下标含义类似。

与理论估计不同, 车辆很难严格遵循匀加速运动, 因此车辆在 t 时刻的真实状态应表示为基于 $t-1$ 时刻的真实状态经过 Δt 时间间隔后再加上噪声项 $\mathbf{w}_t$:

$$\mathbf{x}_t = \mathbf{F} \mathbf{x}_{t-1} + \mathbf{B} \mathbf{u}_t + \mathbf{w}_t \quad (5-4)$$

其中, $\mathbf{x}_t$ 表示 t 时刻的状态真实值, $\mathbf{x}_{t-1}$ 表示汽车在 $t-1$ 时刻的状态真实值。噪声 $\mathbf{w}_t$ 为服从均值为 0, 协方差为 $\mathbf{Q}_t$ 的二元独立高斯分布, 即 $\mathbf{Q}_t = \text{cov}(\mathbf{w}_t)$ 。

上述理论估计值 $\mathbf{x}_{t|t-1}$ 与真实状态 $\mathbf{x}_t$ 之间的误差可以用协方差矩阵来表示:

$$\mathbf{P}_{t|t-1} = \text{cov}(\mathbf{x}_t - \mathbf{x}_{t|t-1}) \quad (5-5)$$

对式(5-5)进行展开, 可以得到

$$\begin{aligned} \mathbf{P}_{t|t-1} &= \text{cov}(\mathbf{x}_t - \mathbf{x}_{t|t-1}) \\ &= \text{cov}(\mathbf{F} \mathbf{x}_{t-1} + \mathbf{B} \mathbf{u}_t + \mathbf{w}_t - \mathbf{F} \mathbf{x}_{t-1|t-1} - \mathbf{B} \mathbf{u}_t) \\ &= \text{cov}(\mathbf{F}(\mathbf{x}_{t-1} - \mathbf{x}_{t-1|t-1}) + \mathbf{w}_t) \\ &= \mathbf{F} \text{cov}(\mathbf{x}_{t-1} - \mathbf{x}_{t-1|t-1}) \mathbf{F}^T + \text{cov}(\mathbf{w}_t) \\ &= \mathbf{F} \mathbf{P}_{t-1|t-1} \mathbf{F}^T + \mathbf{Q}_t \end{aligned} \quad (5-6)$$

3. 观测方程

在车辆轨迹预测中, 可以利用车上的各类传感器或其他手段获取状态变量的测量值, 如加速度传感器、雷达点云检测器等。通过传感器测量得到的准确值可以表示为

$$\mathbf{z}_t = \mathbf{H} \mathbf{x}_t + \mathbf{v}_t \quad (5-7)$$

其中, $\mathbf{H}$ 是从真实状态空间到测量空间的映射, 如前面估计的位置与速度, 若传感器测量的是加速度, 则需要通过变换矩阵 $\mathbf{H}$ 将位置和速度映射到与加速度相同的空间。噪声 $\mathbf{v}_t$ 服从均值为 0、协方差为 $\mathbf{R}_t$ 的二元独立高斯分布, $\mathbf{R}_t = \text{cov}(\mathbf{v}_t)$ 。

理论估计与测量值之间的误差同样使用协方差矩阵来表示:

$$\mathbf{S}_t = \text{cov}(\mathbf{z}_t - \mathbf{H}\mathbf{x}_{t|t-1}) \quad (5-8)$$

式(5-8)可展开为

$$\mathbf{S}_t = \text{cov}(\mathbf{z}_t - \mathbf{H}\mathbf{x}_{t|t-1}) = \text{cov}(\mathbf{H}\mathbf{x}_t + \mathbf{v}_t - \mathbf{H}\mathbf{x}_{t|t-1}) = \mathbf{H}\mathbf{P}_{t|t-1}\mathbf{H}^T + \mathbf{R}_t \quad (5-9)$$

4. 迭代更新

上述步骤明确了 t 时刻的理论预测值和实际测量值, 并已知了理论预测值与 t 时刻真实值之间的估计误差, 因此, 卡尔曼滤波分别给预测值和实际值一个权重, 通过预测值和实际值的加权线性组合来得到估计值:

$$\mathbf{x}_{t|t} = \mathbf{K}_t \mathbf{z}_t + (\mathbf{I} - \mathbf{K}_t \mathbf{H}) \mathbf{x}_{t|t-1} \quad (5-10)$$

其中, $\mathbf{K}_t$ 称作卡尔曼增益, 其由最小化 $\mathbf{x}_{t|t}$ 与 $\mathbf{x}_t$ 间的均方误差求得, 在该约束下的权重因子即为最佳的。最小化均方误差为

$$\min_{\mathbf{K}_t} \|\mathbf{x}_t - \mathbf{x}_{t|t}\|^2 \Leftrightarrow \min_{\mathbf{K}_t} \text{Tr}(\text{cov}(\mathbf{x}_t - \mathbf{x}_{t|t})) = \min_{\mathbf{K}_t} \text{Tr}(\mathbf{P}_{t|t}) \quad (5-11)$$

对 $\mathbf{P}_{t|t}$ 展开后, 求一阶导等于零, 得到卡尔曼增益为

$$\mathbf{K}_t = \mathbf{P}_{t|t-1} \mathbf{H}^T \mathbf{S}_t^{-1} \quad (5-12)$$

再将 $\mathbf{K}_t$ 代回 $\mathbf{P}_{t|t}$ 中即可得到关于 $\mathbf{K}_t$ 的加权估计下的协方差矩阵:

$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t \mathbf{H}) \mathbf{P}_{t|t-1} \quad (5-13)$$

至此就完成了卡尔曼滤波的预测和更新两个步骤, 得到了未来时刻的逐时间步预测结果。随着预测时间的增加, 预测误差会逐步累积, 因此, 基于卡尔曼滤波的车辆轨迹对于长时间的预测效果欠佳。

5.2.2 基于动态贝叶斯网络的周围车辆行为预测

为了提升预测的准确性, 预测模型不仅需要最新的车辆状态, 同时需要考虑交通参与者周围环境及其对车辆轨迹产生的影响, 动态贝叶斯网络(Dynamic Bayesian Network, DBN)通过编码场景导航地图中的统计数据来更新目标状态, 从而实现目标的轨迹预测^[165]。

状态表征: 在第 k 个离散时间步上, 车辆状态变量定义为 $\mathbf{X}_k = (\mathbf{P}_k, \mathbf{V}_k)^T$, 其中, $\mathbf{P}_k = (X_k, Y_k)$ 为笛卡儿坐标系下的位置, $\mathbf{V}_k = (\Omega_k, \Theta_k)$ 为速度大小和角度。通过目标与导航地图 $\mathcal{M}$ 的交互, 可以从 $\mathcal{M}$ 中提取目标所在地图块的统计特征 (HoD, HoS, ρ, ξ), 其中方向直方图 (Histogram of Directions, HoD) 表示地图块中 N 个可行驶方向对应的概率 $p_i, i=1, 2, \dots, N$; 速度的直方图 (Histogram of Speed, HoS) 计算了 N 个离开该地图块的目标预期速度大小; $\rho \in [0, 1]$ 是遍历分数, 表示该地图块相对于其他地图块被目标占据的次数比重; $\xi \in [0, 1]$ 是路由分数, 表明该地图块成为路由点的概率。给定上述初始条件 $\mathbf{X}_0$, 动态贝叶斯网络将预测未来 T_p 个时间步的车辆状态 $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_{T_p}$, 则可以定义轨迹 ϕ_{T_p} 为目标状态的集合 $\{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_{T_p}\}$ 。

根据匀加速模型,目标运动状态的动态过程为

$$\mathbf{P}_{k+1} = \mathbf{P}_k + \left(\frac{\Omega_k \cos \Theta_k}{\Omega_k \sin \Theta_k} \right) \Delta_k + \mathbf{w}_k \quad (5-14)$$

$$\mathbf{V}_{k+1} = \phi(\mathbf{P}_k, \mathbf{V}_k, \mathcal{M}) \quad (5-15)$$

其中, Δ_k 是采样时间(假设 $\Delta_k = 1$), $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \sigma I_2)$ 是一个高斯白噪声。 ϕ 表示计算下一时刻速度向量 $\mathbf{V}_{k+1}$ 的函数。虽然匀加速模型是一个很强的假设,但 $\mathbf{V}_{k+1}$ 的计算和 $\mathcal{M}$ 中的学习期望值使得模型能产生非线性行为。

与求解 $\phi(\cdot)$ 的一个封闭解不同,动态贝叶斯网络(DBN)能够以概率的方式处理这个过程,其中目标的状态以高斯分布建模。DBN 由以下条件概率分布定义:

$$p(\mathbf{P}_{k+1} | \mathbf{P}_k, \mathbf{V}_k) = \mathcal{N}\left(\left(\frac{X_k + \Omega_k \cos \Theta_k}{Y_k + \Omega_k \sin \Theta_k}\right), \sigma I_2\right) \quad (5-16)$$

$$p(\mathbf{P}_{k+1} | \mathbf{P}_k, \mathbf{V}_k; \mathcal{M}) = \begin{cases} p(\Omega_k | \mathbf{P}_k, \mathbf{V}_k; \mathcal{M}) \\ p(\Theta_k | \mathbf{P}_k, \mathbf{V}_k; \mathcal{M}) \end{cases} \quad (5-17)$$

其中,

$$p(\Omega_k | \mathbf{P}_k, \mathbf{V}_k; \mathcal{M}) = \sum_{i=0}^N \Gamma(\mu_i, \sigma_i) p_{f_i} \quad (5-18)$$

$$p(\Theta_k | \mathbf{P}_k, \mathbf{V}_k; \mathcal{M}) = p_{f_i} \quad (5-19)$$

其中, $\Gamma(\mu_i, \sigma_i)$ 是伽马分布,它定义了地图块的 N 个速度,选择伽马分布的原因是它没有定义为负值,并且有一个峰值。概率 p_{f_i} 为

$$p_{f_i} = \frac{p_i e^{-\lambda d(\Theta_k, \Theta_i)}}{\sum_{j=0}^N p_j e^{-\lambda d(\Theta_k, \Theta_j)}} \quad (5-20)$$

其中, $d(\cdot)$ 是一个距离度量(L2 范数),而 λ 是一个平滑因子,指数 e 则用于将距离转化为概率。式(5-20)表明,DBN 考虑当前速度向量 Θ_k 的方向和所有其他可能的方向 Θ_i 之间的相似性,用地图分配给该方向 p_i 的概率进行加权。

此外,为了合并路由得分 $\xi \in [0, 1]$,式(5-20)中形成离散分布的离散值 p_{f_i} 需要修改。理想情况下,当 ξ 接近于 1 时,期望分布更“随机”(即更均匀),而当 ξ 趋于零时,则期待分布更“确定性”(即总是选择最可能的值)。这些性质都可以通过贝塔分布 $B(x; \alpha, \beta) \propto x^{\alpha-1} (1-x)^{\beta-1}$ 得到的,其中, $\alpha = \beta$ 。当 α 趋于零时,很容易验证贝塔分布是否为均匀的,当 α 趋于无穷时,该分布在最可能值达到峰值。为此,通过以下随机变量的变换:

$$\widetilde{p_{f_i}} \propto p_{f_i}^\alpha (1 - p_{f_i})^\alpha \quad (5-21)$$

其中, $\alpha = \frac{1-\xi}{\xi}$,通过这样的方式,地图块的路由得分被纳入模型中。

考虑到该算法的概率性质,通过运行几次即可得到了不同的轨迹。此外,算法利用目标与通过的每个地图块之间的遍历分数 ρ 进行轨迹选择,其中轨迹 Ψ_{T_g} 的概率计算为

$$p(\Psi_{T_g}) = \frac{1}{T_g} \sum_{i=1}^{T_g} \rho_i \quad (5-22)$$

其中, ρ_i 是目标与其通过的地图块之间的遍历分数。通过上述方式, 利用 DBN 可以预测车辆的多条不同轨迹及其概率。

5.2.3 应用隐马尔可夫模型的周围车辆行为预测

隐马尔可夫模型 (Hidden Markov Model, HMM) 是一个统计模型, 它可以用来描述一个含有隐含未知参数的马尔可夫过程, 常被用来解决有未知条件的数学问题。已知模型和观测序列, 在给定的观测序列下, HMM 可以求其最可能对应的状态序列。

因为隐马尔可夫预测算法是基于车辆运动的, 因此需要先将轨迹数据及 (X, Y) 转换为轨迹运动数据集 (X_m, Y_m) , 如式 (5-23) 所示。

$$D_V = \{(V_x^1, V_y^1), (V_x^2, V_y^2), \dots, (V_x^n, V_y^n)\} = \{X_v, Y_v\} \quad (5-23)$$

其中, $V_x^i = (x_1^i, x_2^i, \dots, x_T^i)$ 表明了第 i 条轨迹在 x 轴方向上的运动向量集合, $x_{t+1}^i - x_t^i$ 为相邻两时刻的 x 轴坐标差值, 同样 y 轴的方向运动向量集合为 $V_y^i = (y_1^i, y_2^i, \dots, y_T^i)$ 。在两个方向上的运动向量集合 (X_v, Y_v) 共同组成了轨迹运动向量集合 D_v 。

为了简化接下来对于 HMM 的介绍, 使用一条轨迹在 x 轴方向上的预测作为例子进行说明。此时, HMM 的观测序列为

$$V = \{V_1, V_2, \dots, V_t, \dots, V_T\} \quad (5-24)$$

其中, $V_t = x_t^i$ 为 t 时刻的观测。

HMM 的主要过程可以分为两部分: 第一部分是一阶马尔可夫过程, 模型在各个状态间进行转换。另一部分是处理获得的观测值到当前隐状态的转换关系。图 5-1 展示了 HMM 的具体结构。

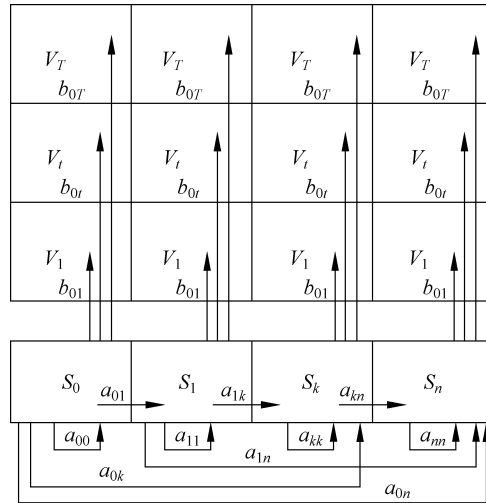


图 5-1 HMM 的具体结构

HMM 中的变量可以被分为两组。第一组是状态变量,表示当前时间下的系统状态。在 HMM 中,状态变量通常被假定为不可观测的,因此也被称作隐状态, $S = \{S_1, S_2, \dots, S_N\}$ 表示隐状态的集合, $Q = \{q_1, q_2, \dots, q_T\}$ 表示隐状态变量的序列。

第二组变量是观测变量,用于表示当前时间戳下的观测值。隐变量根据状态转移矩阵 $A = \{a_{ij}\}$ 从当前时间下的状态向下一时间状态转换。同时,发射矩阵 $B = \{b_j(V_t)\}$ 表示从当前的隐状态到获得观测序列的转移概率。此外假设隐变量的初始状态概率分布为 π 。因而 $\lambda = \{A, B, \pi\}$ 为 HMM 参数集。隐变量的状态转移过程通常是不可被观测的,因此模型变量只能通过观测序列进行推断。

综上,HMM 模型可以写作 $\bar{V} = \Psi_\lambda(V)$,其中 $\lambda = \{A, B, \pi\}$ 。

在预测问题中,首先需要对 HMM 轨迹模型进行训练,即给定一段历史观测轨迹序列 V ,在参数 λ 确定的情况下,求解最大概率的未来序列,以预测在 $t+k$ 时刻的观测值 $\bar{V}_{t+k}$,其中 k 代表预测的轨迹点数量。HMM 的预测问题可采用维比特算法来求解,即考虑前一个状态结点的概率、到达当前状态结点的转移矩阵,以及发射矩阵中当前观测和状态结点之间的关系,通过最大化每一步概率,从而得到最优预测路径。

需要注意的是,在上述模型处理过程中,HMM 需要所有数据点来进行轨迹预测,因此过去点的预测误差会被传递到下一个预测点,最终导致累积误差。此外,数据中的噪声也会增加这一预测误差。

5.3 基于深度学习的周围车辆轨迹预测

自动驾驶汽车传感器系统的快速发展产生了海量的实时场景数据,由于简单的物理模型或概率模型难以对复杂交通场景的理解与认知问题进行有效的建模,而神经网络因其高效的特征提取能力以及对复杂上下文关系的表征能力,使得有效建模场景中影响轨迹的多种因素成为可能,因而成为了车辆轨迹预测的主流方向。

目前基于深度学习的车辆轨迹预测模型层出不穷,如循环神经网络[包括长短时记忆网络(LSTM)和门控神经网络]^[166]、卷积神经网络^[167]、自注意力 Transformer 结构^[168]以及混合多种网络结构的复杂模型。由于研究者的不同偏好和轨迹预测任务本身的复杂性,基于深度学习的车辆轨迹预测模型很难进行结构上的分类。因此本节从场景和问题本身的视角出发,结合一部分经典方法,介绍了车辆轨迹预测的一些研究思路。

到目前为止,在基于深度神经网络的周围车辆轨迹预测模型中,最经典的是基于循环神经网络编解码模型,其常用于机器翻译任务,同时也可以用于轨迹预测。如图 5-2 所示,其中编码模块递归地输入源语言,解码模块递归地输出目标语言。相应地,在轨迹预测任务中,编码模块就负责输入和编码历史轨迹,而解码模块就输出未来预测轨迹。

上述模型较为简单,难以编码整个交通场景的动态与静态信息,因此需要更加复杂的结构以适应更加复杂的数据。但是不管模型如何复杂,它们的设计思路大都符合编码-解码结构,所以结合前面对场景中各个要素的分析,图 5-3 给出了更加一般化的网络结构设计思路与要点。

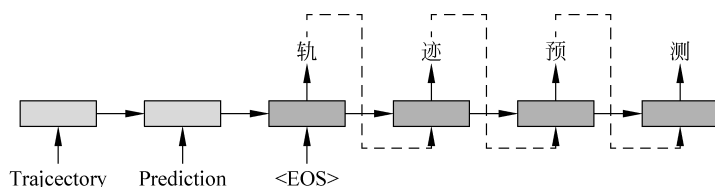


图 5-2 序列到序列的网络结构应用于机器翻译

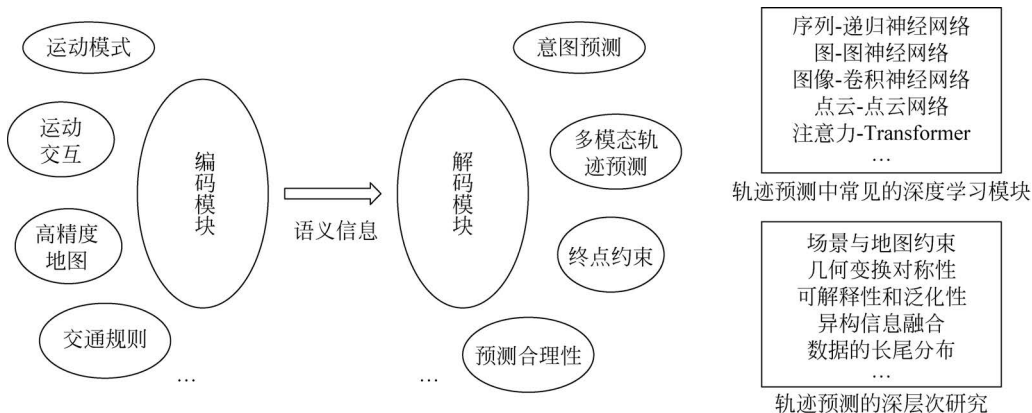


图 5-3 基于深度学习的轨迹预测模型框架

目前主流的轨迹预测算法基本都符合图 5-3 的设计策略,在具体的数据编码-解码结构以及数据的输入顺序上存在差异,但都需要满足上述轨迹预测的几个要点,如车辆交互、地图信息以及多模态预测等问题。

最后,对常见的车辆轨迹预测任务进行简要分类。按照场景可以划分为单一静态场景和动态复杂场景。单一静态场景主要是指地点固定的场景,比如在恒定地点的摄像头视角下或道路结构较为单一的高速公路场景。在这种较为理想的情况下,车辆行为较为统一,但应用范围有限,往往只能分析某个路口或某个路段的车辆行为。动态复杂场景是指地点不固定的场景,包括城市中车辆行驶在各种不同路况下的情景,需要对周围各种各样路况下的车辆行为和轨迹进行预测,这种场景更加符合实际,广泛应用于动态环境中的自动驾驶车辆,但是对模型的泛化能力提出了挑战。按照视角可以分为图像视角(前视视角)的轨迹预测以及 BEV 视角下的轨迹预测。目前轨迹预测主流研究以及自动驾驶应用都是基于 BEV 视角进行的,其实不单单是轨迹预测,在自动驾驶系统的感知模块,学术界与工业界也更加倾向于使用 BEV 框架进行构建。为了贴合主流研究与应用,本节介绍的方法都是基于 BEV 视角,图像视角的轨迹预测在第 4 章介绍的 VRU 轨迹预测中已有涉及,这里不再详细讨论。

因此,本节将首先对深度学习下的车辆轨迹预测问题进行深入分析,再基于编码-解码结构具体介绍静态单一场景和动态复杂场景下的深度学习轨迹预测方法。

5.3.1 深度学习下的车辆轨迹预测问题分析

基于传统方法的车辆行为预测问题往往只考虑被预测车辆本身的历史运动信息,而忽略了实际交通场景中的大部分信息,因此传统方法实际上是将车辆预测问题看成是车辆的动力学建模和简单的概率学预测问题。在实际交通场景中,环境往往复杂多变^[169],需要考虑诸多因素,如图 5-4 所示,为了实现尽可能准确的预测,必须考虑尽可能多的环境信息,而不仅仅局限于被预测车辆自身。从这个角度看,车辆轨迹预测问题就不单单是一种偏向物理的动力学的问题,而更倾向于是一种交通场景的理解与认知问题。

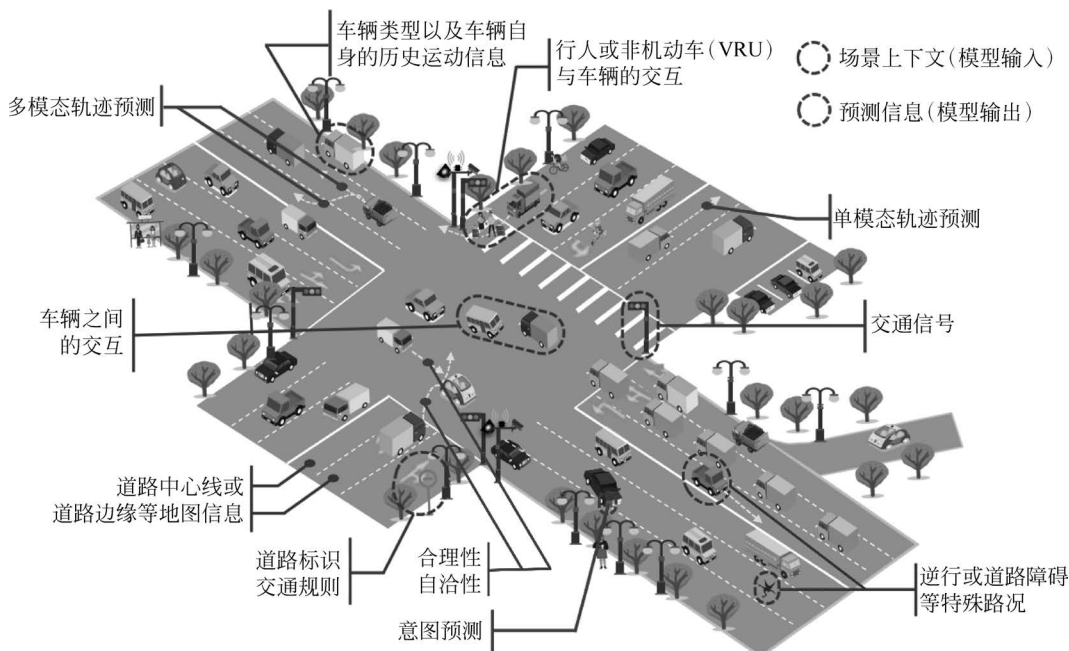


图 5-4 从场景理解与认知视角看待车辆轨迹预测问题

如图 5-4 所示,从场景理解与认知的角度,对车辆轨迹预测问题中的重要影响因素进行进一步的详细分析。这些因素大致可分为两类:一类是场景上下文信息,即可以收集到的包括历史轨迹在内的所有周边环境信息(即预测模型的输入部分);另一类是需要预测的信息,包括预测的轨迹性质和意图等(即预测模型的输出部分)。其中,场景上下文信息主要分为以下 3 种。

(1) 动力学因素:包含车辆类型以及车辆的历史运动信息,后者也是传统方法需要考虑的一个典型信息。车辆类型往往决定了车辆的动力学参数,比如车辆的速度上限和制动性等。而车辆的历史运动信息往往对车辆的未来运动有着很强的指导意义,比如一个车辆的历史轨迹有加速/减速趋势,那么它在短时间内的未来轨迹有较大概率保持这种加速/减速趋势。

(2) 交互因素:被预测车辆的未来行为不只依赖它的历史运动信息,还需要考虑周边

其他动态目标对它造成的影响。比如被预测车辆前方出现了横穿马路的行人,则车辆很可能会产生减速甚至紧急刹车避让的行为。需要注意的是,这里的交互不止局限于移动单位对车辆运动的“影响”,还可能是某种趋势的保持,即被预测车辆倾向于和周边车辆保持同步运动趋势,被预测前方车辆的运动模式很可能就是被预测车辆未来的行驶模式(也就是平时所说的“跟车”),此时周围车辆对预测便起到一种“提示”的作用。在实际复杂场景下,往往很难区分周围车辆对被预测车辆运动模式到底是“影响”还是“提示”,而是不加区分将它们统称为交互。车辆与车辆的交互以及车辆与VRU之间的交互也存在一定的区别,详细的分析已经在第4章提到,但现有研究针对这些区别并没有做深入的探讨,这也是车辆以及VRU轨迹预测需要攻克的难点,这里也只做简要的概述。

(3) 车辆与交通环境之间的交互: 车辆的行为往往与道路结构等环境信息高度相关,而环境信息中包含的一个重要内容就是道路结构或地图信息。车辆往往会沿着道路行驶,并且在直道上行驶的车辆在未来几秒内发生转弯的概率一定会远远小于在十字路口处发生转弯的概率,因此提前知道道路结构对后续的预测性能提升有着非常大的作用。另外,道路标识与交通规则对车辆的行为有一定的约束,车辆在一般情况下不会违反交通规则。当然上述提到的约束是大多数情况,还有一些极端情况(corner case),比如某些车辆逆行或道路损坏。这些特殊情况对车辆的预测鲁棒性有较大影响,同时也是自动驾驶领域本身要克服的问题。

因此,与图像识别等任务类似,周围车辆运动轨迹预测任务中使用深度学习的主要动机在于借助海量数据和深度神经网络强大的表征能力以及特征提取能力。这使得车辆行为预测任务中不再需要繁杂的特征工程,并且能够对场景复杂的上下文信息进行有效建模。与本书第4章介绍的VRU轨迹预测不同,由于车辆运动的自身特性,基于深度学习的周围车辆轨迹预测研究的侧重点与难点都有所不同。下面首先介绍基于深度学习的周围车辆轨迹预测中的主要研究问题。

(1) 意图预测与轨迹预测: 早期的周围车辆预测问题是意图预测^[170],即模型往往只要求给出一个粗略的意图,比如“左转”“直行”以及“变道”等,但这并不能体现车辆进一步的具体行为。除此之外,车辆的意图在简单的场景比如高速公路的场景下很容易被定义,而在纵横交错的复杂场景下,车辆行为的潜在空间也会变得复杂,难以被事先定义。后续研究进一步要求模型给出更加细粒度的轨迹预测,即给出被预测车辆在未来某个时间戳下的具体位置坐标。从这个角度看,粗粒度的意图是车辆行为的语义概括,而细粒度的轨迹则是车辆行为在真实物理空间下的具体表现,因此意图预测和轨迹预测是两个强关联的问题,本节后续也会介绍基于意图的轨迹预测解码模型。

(2) 单模态与多模态: 轨迹预测中另一个非常重要的问题就是多模态问题。轨迹预测问题中的多模态(multimodal)不是指多源数据,如图像和文本等多模态数据^[171],而是指轨迹预测问题中未来轨迹概率分布的多模态,因此轨迹预测的多模态性质又被称为多模概率性质。相较于模式识别中的检测、分类以及跟踪等识别类问题,预测问题的结果不确定性非

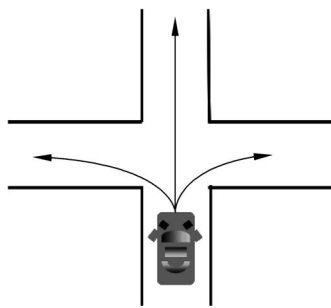


图 5-5 轨迹预测中的多模态

常大,因此研究者们提出了多模态的构想,即针对环境的不确定性,模型需要给出多种可能的预测轨迹以及每种轨迹的置信度。车辆轨迹预测的不确定性是由观测的不完备性导致的,比如司机的心理状态或天气情况等,这些难以观测到的因素都会对未来预测产生较大影响。图 5-5 给出了多模态的一个具体例子,对于一个刚好行驶在十字路口的车辆来说,它在某一次观测中会发生左转,在类似场景下的下一次观测中可能会发生右转。因此不管是物理模型还是交通场景中的一个预测模型来说,它需要给出的是一组尽可能完备的预测分布,而非单一可能的“观测结果”。这就是多模态预测的核心思想。

(3) 预测的合理性与自洽性:目前主流的周围车辆轨迹预测算法都是聚焦于对场景中的车辆进行独立或并行的预测,即只考虑每辆车自身的未来预测结果,而没有考虑车辆与车辆之间预测结果是否合理与自洽。实际上,正如前面提到的,轨迹预测本身是一种动态场景的理解问题,一个理想的预测算法不仅在历史信息编码层面上要尽可能考虑场景中尽可能多的信息,同时在预测层面上也要对整个场景做出一整套合理的预测而非局限于单独的车辆。如果进行独立预测,则可能会出现每辆车自身的预测轨迹单独看起来都没有问题,但是联合起来考虑时会产生两辆车预测轨迹距离非常近甚至相撞的情况,这当然在绝大多数情况下是一种不符合真实世界的预测(排除车祸这类偶然事件)。所以预测的合理性和自洽性也是轨迹预测中的一个重要问题,这就需要同时考虑整个场景中的所有移动单元,并且进行某种程度上的联合预测。在统计学概率上的解释就是给定整个历史场景的信息,所有车辆未来轨迹的条件概率并非是彼此独立同分布的,而是符合一种潜在的联合分布。这与前面提到的考虑车辆与周围移动单元的交互思想有些类似,只不过交互强调的是历史轨迹的联合编码,而合理性与自洽性考虑的是将车辆未来的状态联合解码预测。

基于上述基本的研究问题,接下来对基于深度学习的车辆轨迹预测中的难点问题进行总结。

(1) 编码方式:不同的信息需要采用不同的编码方式,比如历史轨迹序列通常使用一维卷积或递归神经网络进行编码,而静态的地图信息则需要使用图神经网络等非结构化算子进行编码或转为图像之后再使用卷积算子进行编码。

(2) 数据融合:对于场景中不同类型的数据进行融合所需要融合的算子,从最简单的池化策略到注意力机制等,由于每个单元类型从静态到动态并不统一,这类融合实际上是一种异构数据的融合。

(3) 多模态输出:神经网络需要针对被预测车辆给出多模态输出,即多个输出结果。但实际数据往往只有一个观测结果,因此在训练时需要精心设计这种多对一的训练策略。

(4) 输出约束:行为预测问题中最重要的是要给出细粒度的未来运动轨迹,回归是一种最常用的方式。回归方式虽然简单,但它对模型的输出并没有加以限制,在模型的推理过程中往往会产生一些不符合交通规则或物理规则的预测结果,比如车辆驶出可行驶区域之

外或者车辆之间产生碰撞。如何利用先验知识和规则对模型加以约束也是车辆预测问题中的难点。解决这类问题实际是在一定程度上提高深度模型的可解释性和泛化能力。

5.3.2 单一静态场景下车辆的轨迹预测

本节主要介绍几个单一静态场景下基于深度学习的车辆轨迹预测方法。单一静态场景主要是指地点相对固定的场景,比如笔直的高速公路或某个十字路口,模型在这种场景中只需要预测一个或几个路段场景下的车辆轨迹。在这类任务设定下,车辆的行为模型较为简单,便于归类。从编码的角度讲,模型实际上不需要对场景本身有泛化能力,因为所有数据都是来自一个或某几个场景,模型也不需要显式地编码地图结构。从解码的角度看,由于车辆的行为模式较为简单,因此可以事先根据道路结构定义几类意图状态,比如转弯、直行、变道等,并且可以根据事先划分好的行为进行多模态预测,比如车辆在直行的情况下未来会如何行驶,在转弯的情况下又会如何行驶。本节将按编码、解码的顺序介绍单一静态场景下的几种主流预测方法。

1. 编码部分

1) CS-LSTM

CS-LSTM^[172]主要应用于高速公路场景,即仅有几条笔直道路的情况,其同时考虑了车辆之间的交互信息。它的编码结构如图 5-6 所示,所有子模块都采用结构化编码方式。具体而言,对于每辆车来说,模型将其视为一维时间序列并用 LSTM 对每个序列进行编码,并且每辆车的 LSTM 权重是共享的。然后将每辆车对应的 LSTM 最后一个时刻的隐状态视为该车辆的编码。在场景部分,模型将整个场景划分为特定分辨率的网格,网格的高度约等于车辆的平均长度,网格的宽度约等于一个车道的宽度,对于场景中的每辆车的 LSTM 编码,模型根据车辆在当前时刻的位置将编码嵌入特定分辨率的网格中,从而得到场景特征图,通过卷积算子对特征图进行编码得到周围车辆的交互特征,最后将交互特征和被预测车辆的特征进行拼接得到被预测车辆编码的最终特征。CS-LSTM 实际上并没有显式地编码地图信息,而是将道路结构隐式地构造二维场景特征图,通过车辆所在车道等位置信息将其嵌入结构化的网格。

2) Grip 模型^[173]

Grip 模型既没有显式地考虑地图结构,也没有像 CS-LSTM 那样使用网格化方式对场景进行构建,而是使用图卷积网络对车辆之间的交互进行建模。整个模型编码部分的结构如图 5-7 所示。该模型首先针对场景中的车辆构造一张图,其中每辆车视为图上的一个节点。如果两个节点对应的两个车辆在零时刻的距离小于某个阈值,则这两个节点有边连接。然后交替地在时间维度上使用一维卷积,在空间维度上使用图卷积算子对邻居节点进行聚合。最后通过两次下采样得到车辆交互的时序特征,输出的节点特征就表示车辆最终的编码特征。Grip 相对于 CS-LSTM 的一个优势就是不需要地图结构的先验假设,而仅仅通过相对距离就可以建模车辆与车辆之间的交互。此外,图卷积算子是对整个场景进行编码,因此通过一次前馈就可以提取场景中所有车辆的运动特征。

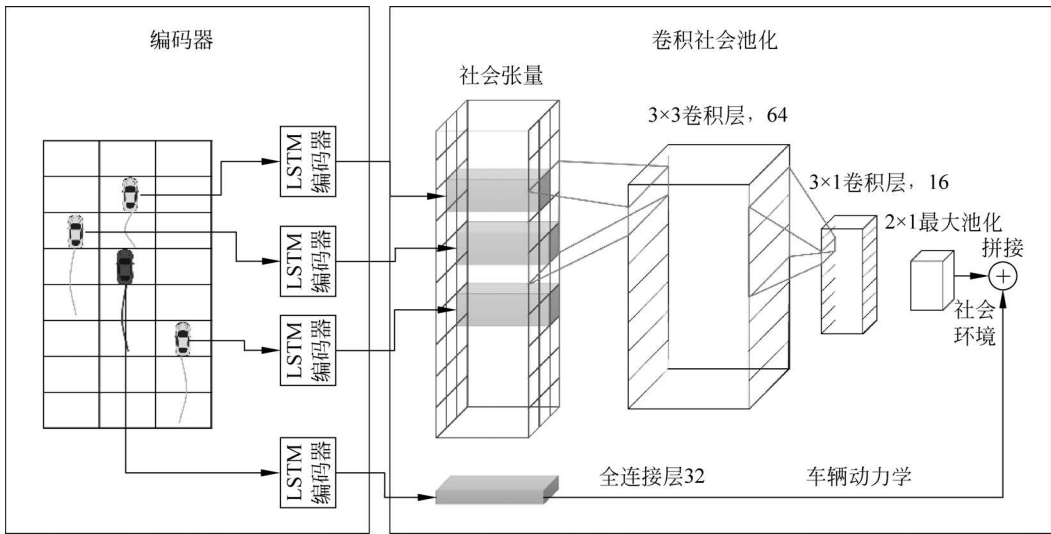


图 5-6 CS-LSTM 模型结构(编码部分)

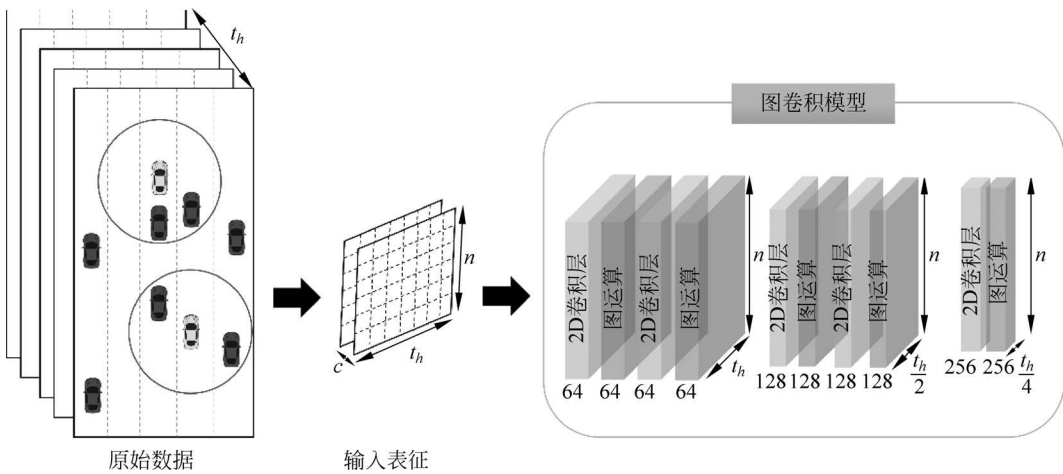


图 5-7 Grip 模型结构(编码部分)

2. 解码部分

在单一的静态场景下,行为的定义相对容易,因此在这类场景下的大多数方法都是针对车的行为意图进行定义,比如左转、直行和变道等。随后,根据给定的行为意图,并在给定该行为的条件下预测车辆的轨迹,这可以看成是一种给定行为情况下的条件概率预测。下面着重介绍两种典型场景下的行为定义方法。

1) M-LSTM^[174]

这是一种多模态轨迹预测方法,适用于高速公路场景。它针对高速公路场景对车辆在切向和法向的行为进行划分。在切向方向,车辆根据切向速度被分为加速和匀速;在法向方向,车辆被分为直行、左变道和右变道;最终整个行为意图模式共分为 6 种。整个模型结

构如图 5-8 所示,它的解码部分包括两部分:第一部分是意图预测,即预测车辆接下来的行为满足事先定义好的哪些轨迹;第二部分是基于意图的轨迹解码部分,它将行为编码(一般是 One-Hot 向量)以及编码好的车辆历史运动信息进行拼接后送入解码模块中得到输出的轨迹。对于不同的意图编码,模型可以输出不同的预测结果并根据意图预测模块给出对应行为轨迹的置信度,从而可以输出基于多个意图的多个结果。在训练时每辆车的真值往往只有一个,因此只要回归真值对应的行为下的预测结果即可,忽略其他行为下的预测轨迹。除此之外,M-LSTM 的两个分支也可以合并到一个编码模块上用多任务的范式进行联合训练,从而节省计算量。可以发现,M-LSTM 虽然编码部分简单,但它的解码部分可以迁移到上述介绍的 CS-LSTM 以及 GRIP 模型上。因此,多数情况下,用于轨迹预测的深度学习模型中的模型编码与解码部分可以拆开来独立研究,同时也可以排列组合进行使用,这也是将编码部分和解码部分分开介绍的原因。

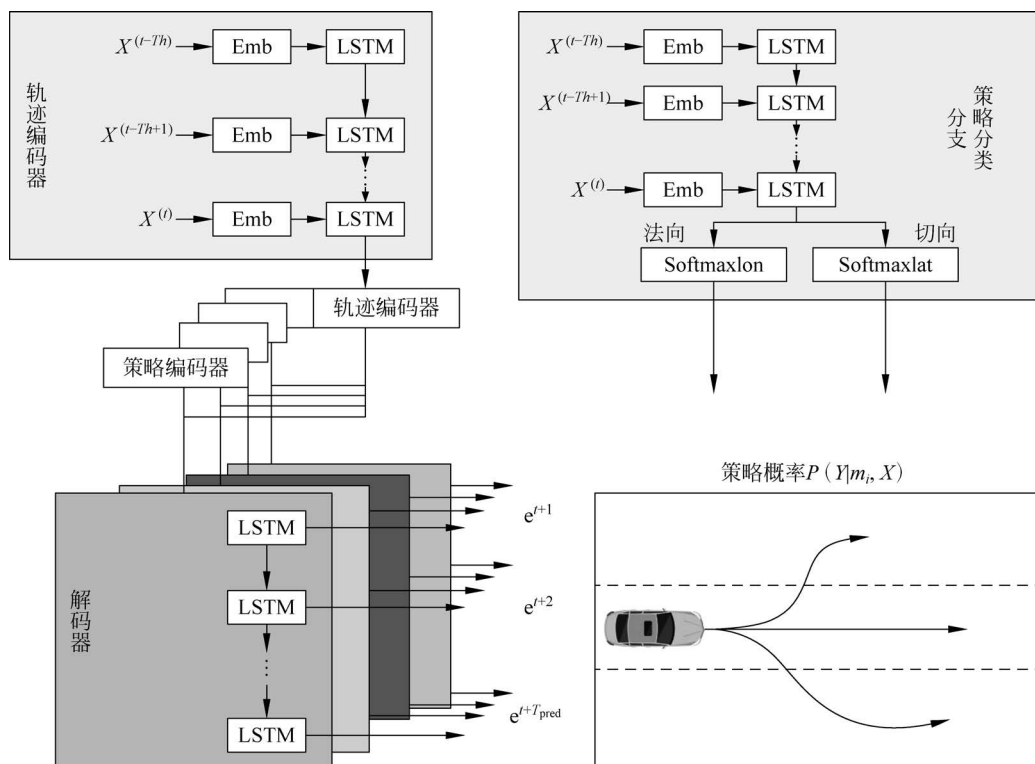


图 5-8 M-LSTM 模型结构与行为意图分类

2) 多模态意图预测(Multimodal Intent Prediction, MIP)模型^[175]

MIP 是一种静态场景下基于意图的多模态预测模型,它主要是针对十字路口的车辆轨迹进行预测。MIP 解码部分的结构和 M-LSTM 类似,此处不再赘述。MIP 根据车辆在当前时刻与未来轨迹终点的连线产生的线段与水平方向的夹角作为行为分类的依据。如果夹角绝对值超过某个阈值,则车辆的意图被定义为左转/右转,否则就被定义为直行。为了获

得更多模式和更加细粒度的意图,可以尝试对夹角划分更多区间,每个区间对应一种模式,从而可以获得更加细粒度的预测轨迹。

5.3.3 动态复杂城区环境中车辆的轨迹预测

本节主要介绍动态复杂城区场景下基于深度学习的车辆轨迹预测方法。在这种情况下,模型不仅要对不同车辆的不同运动模式有泛化能力,还需要对不同场景有良好的泛化,这对模型的容量和训练策略有着很高的要求。从编码角度来看,模型需要对各种各样的场景进行编码,比如二车道和三车道等不同数量的道路;直道和弯道等不同形状的道路;十字路口和 T 字路口等不同交通交汇场景。从解码的角度讲,车辆的行为意图在不同场景下很难进行统一的定义,比如 5.3.2 节在直道和十字路口这两种场景下,车辆的行为意图定义就有很大的不同。因此多模态的预测不能再依赖预先定义好的意图,必须寻找一种更加通用的方式。

1. 编码部分

面对结构复杂多变的场景,一类做法是使用栅格化(rasterization),即将整个场景“渲染”为图像,以便通过计算机视觉任务中的二维卷积神经网络对场景进行编码。这与前面介绍的 CS-LSTM 类似,只不过划分的栅格更加细粒度,语义更加丰富,但这类方法的缺陷在于计算冗余。与点云的体素化类似,车辆的可行驶区域占比很小,渲染后的图像中有很多无效区域,编码较为稀疏。因此针对上述缺点,第二类做法是使用一些非结构化的算子比如图神经网络或点云模型等去编码场景,这样的编码会更加紧凑,计算更为高效。

1) 栅格化编码模型

首先介绍两种基于栅格化的编码模型。第一个方法是 Multipath 模型^[176],该模型显式地考虑并编码了地图数据,其编码部分的基本结构如图 5-9 所示。由于原始数据尤其是地图数据是非结构化的信息,因此 Multipath 采用了与三维点云体素化类似的方式将场景信息绘制为结构化的图像。由于车辆运动是时序信息,因此 Multipath 将场景中每个时间戳的场景分别编码为一个图像,并把所有时间戳的图像按通道拼接,最后得到一个 $T \times H \times W$ 的结构化张量。随后,卷积神经网络用于对场景转化得到的结构化张量进行编码,此时再根据被预测车辆的位置和方向对特征图进行旋转裁剪,最后再通过一组 CNN 对裁剪后的区域进行二次编码就能够得到被预测车辆的特征。这个模型编码简单直观,但是从图 5-9 中可以看出,它的编码实际上是非常稀疏的。因为实际场

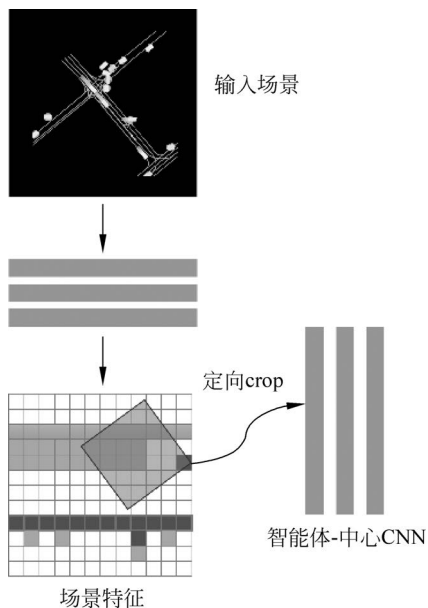


图 5-9 Multipath 模型结构(编码部分)

景中存在大量不可行驶区域(非车道部分),因此卷积计算在这些部分是冗余的。除此之外,它要对每个时间帧都生成一张图像,进一步增加了模型的计算复杂度。为了解决这个问题,实际上只需要将一些静态信息(如地图)栅格化为图像单独进行编码,而将其他动态信息(比如轨迹信息)另外编码,最后在高层语义上进行融合。这就是下面要介绍的 Home 模型^[177]。

Home 模型的结构与思想与 Multipath 类似,如图 5-10 所示。它也使用栅格化的手段,只不过 Home 仅仅将地图等静态信息栅格化为图像,因此不考虑帧序列问题。然后使用 CNN 和下采样对图像信息进行编码。对于车辆轨迹这类动态时序信息,Home 使用一维 CNN 模块和 GRU 模块对车辆轨迹进行编码。由于场景中周围车辆数目和位置并不固定,注意力机制对于处理这种变长问题有着天然的优势,因此将被预测车辆和周围车辆通过注意力模块进行融合得到交互特征,在这里,注意力机制已经开始有非结构化的思想。Home 将交互特征与地图特征进行拼接,得到最终的编码特征。这种对静态信息和动态信息分而治之的策略简化了模型的复杂度,模型不需要将时序信息纳入图像中,因此大大减小了模型复杂度。然而这类模型还是没有解决计算稀疏的问题,因此在计算上还存在一定的冗余。

2) 非结构化编码模型

上述两种典型的栅格化模型中的每个模块都是结构化算子。对于相对简单的场景,如高速公路,这种处理方式可以直接使用卷积算子进行编码。然而,对于非结构化场景,则需要使用栅格化技术将场景“绘制”为图像然后再使用卷积算子进行处理。对于后者,这种处理方式从理论上而言不够本质,因为它没能做到对原始数据直接进行处理。同时,绘制图像由于分辨率的限制会带来一定程度的信息损失。此外,由于可行驶区域的尺寸相比整个图像尺寸占比很小,因此编码结果是稀疏的,会引入一定的计算冗余,从而提升模型计算复杂度。对于这种需要二次转换的数据,实际上可以用图神经网络以及注意力等非结构化算子直接对非结构化数据进行建模。考虑到模型的复杂度、紧凑度以及数据提取效率等问题,研究如何直接对非结构化动态复杂场景进行特征提取很有必要。在其他领域,有很多直接处理非结构化数据的模型,比如图数据的 GCN 和 GAT 等模型;用于数据融合的注意力机制以及 Transformer 模型;以及用于点云的 PointNet、PV-RCNN 模型等。在轨迹预测问题中,车辆的轨迹往往是结构化数据,而车辆之间的交互以及高精度地图往往为非结构化的信息,因此相较于栅格化,直接使用非结构化的模型编码这类信息会使模型变得紧凑和高效。下面介绍几种经典的非结构化车辆轨迹预测模型。

第一类模型是谷歌提出的 VectorNet^[178],其设计旨在克服结构化 CNN 的诸多缺点。除计算低效外,CNN 由于感受野的限制也并不能很好地建模长距离依赖,比如一辆行驶速度非常快的车,过小的感受野并不能很好地提取它自身的运动特征以及和远处地图以及车辆的交互特征。VectorNet 的核心思想就是将场景中的一切元素都向量化,一段轨迹或者一段道路线可以看成是一组向量。栅格化和向量化的思想如图 5-11 所示。

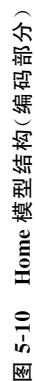


图 5-10 Home 模型结构(编码部分)

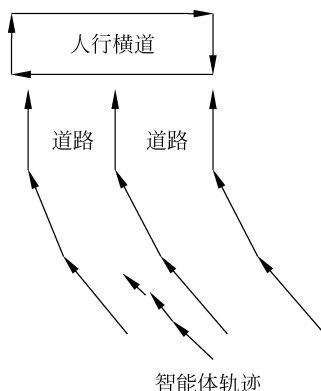


图 5-11 VectorNet 的向量化思想

VectorNet 的编码部分总体结构如图 5-12 所示。它采用分层结构,首先对场景中每个元素(如车辆、道路线)进行编码。对于某个特定的道路线或者某个特定的轨迹来说,它可以看成是由一组向量组成的子图。每个向量包含几组属性。首先是向量的起始点以及终止点,向量的其他属性包括轨迹的时间戳、车辆的类型、道路线的类型等,对每个向量编码之后,向量之间就呈无序性分布。需要注意的是,对于轨迹这种时序信息来说,原始数据具有时序性,但是将时间戳视为属性之后,新的表征数据理应具有无序性。其次,VectorNet 使用一组多层感知机和极大池化对每个单元的特征进行提取,然后将池化后的特征与多层感知机输出的每一个单元的特征进行拼接。最后,将拼接的特征再通过一组多层感知机和最大池化得到整个子图的特征。可以发现最大池化操作本身就满足置换不变性,很好地契合了前面提到的向量之间无序性这一特殊的性质。

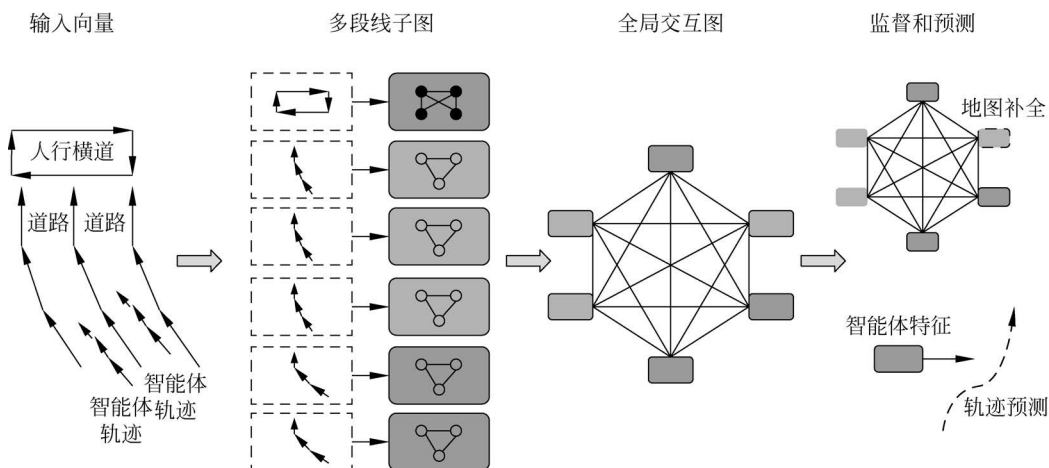


图 5-12 VectorNet 的结构

之后,VectorNet 将每辆车以及每个道路线对应的子图都视为一个基本单元,将整个场景构成一个图,并且每个单元都视为图上的一个节点。由于真实场景中这些点是没有拓扑

邻接关系的,因此 VectorNet 将整个图视作全连接图,即任意两个节点之间都有邻接关系。当然这样做有一些弊端,比如没有显式地把两个节点之间的关系考虑进去,因此 VectorNet 使用全局注意力机制对全图进行编码。任意两个节点之间的聚合权重都是通过节点本身的信息动态生成的。最终通过几层全联通注意力图聚合之后,得到的新的节点特征就被视为融合了场景信息的全局特征。然后对于要预测的车辆节点进行下一阶段的解码任务。由此可见,与 CNN 相比,VectorNet 能够利用所有信息进行高效编码,没有额外的计算量和开销,是一种紧凑的编码方式。

虽然 VectorNet 用全局注意力图可以建模长距离依赖信息,但是它并没有考虑细粒度的局部信息。此外,在表示所有单元的空间依赖关系的图上,其边是完全连接的,这使得它的性能受到一定的限制。在此基础上,LaneGCN 模型^[179]考虑了细粒度的局部信息。LaneGCN 的模型总体思路如图 5-13 所示,它分别对车辆运动信息以及地图拓扑结构信息编码,最后按照一定层次对车辆和地图进行细粒度融合。

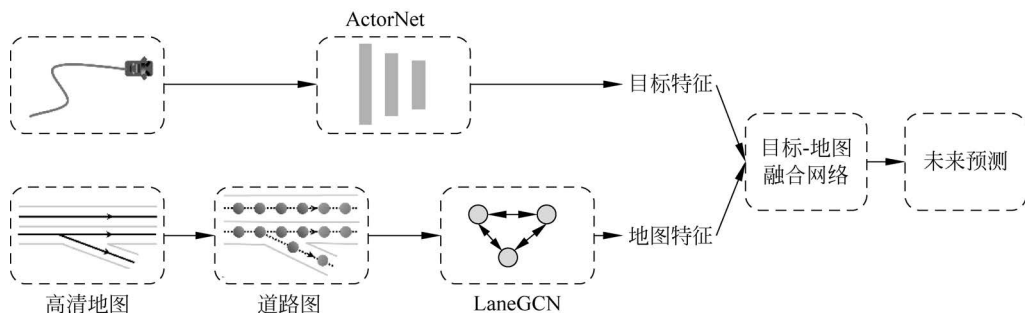


图 5-13 LaneGCN 的编码模块

LaneGCN 模型对于车辆的编码采用传统 CNN 的序列编码模式,它通过构建特征金字塔,将几组一维 CNN 残差模块堆叠起来。车辆运动信息在初始阶段已经编码完成,因此下一阶段的融合就不再考虑细粒度信息。

LaneGCN 的细粒度体现在地图编码上。不同于 VectorNet 先编码一段道路线,然后再将一段长的道路线视为一个基本单元,LaneGCN 直接将道路线上的每个向量视为基本单元。更具体地说,LaneGCN 将整个地图视为一组图,而每个道路线向量视为节点,之后使用图卷积网络对道路线进行编码。可以看出,LaneGCN 的地图编码是从更加局部的细粒度入手进行编码。图卷积网络在前面已有介绍,其中最重要的两个部分就是节点和邻接关系。节点就是地图中的每个向量,而边的信息则需要靠地图的拓扑邻接关系来定义。原本 GCN 中定义的邻接关系是布尔型,即只区分两个节点有没有边,并不关心节点之间边的类型。而这样不加区分地将邻居节点聚合一定会损失一定的信息。

为了尽可能提高模型的表达能力,LaneGCN 使用多组边的定义。它们主要包含四大类,分别是前驱、后继、左邻以及右邻,分别如图 5-14 中的虚线所示。因此,从理论上来说,只需要考虑边的信息,性能就会有所提升。然而 GCN 还有另一个缺陷,就是和前面提到的考虑局部信息的 CNN 一样,存在感受野不足的问题。LaneGCN 拓展了边的定义,

由于车辆往往是沿着道路线前进,因此对道路线的切向感受野要求较高,而法线方向基本没有要求。LaneGCN 针对前驱和后继这两个邻接关系拓展了边的信息。具体而言,就是将第 k 个前驱或后继的关系也作为新的边,一般将 k 设置为 2、4、8、16 和 32 这几个值。

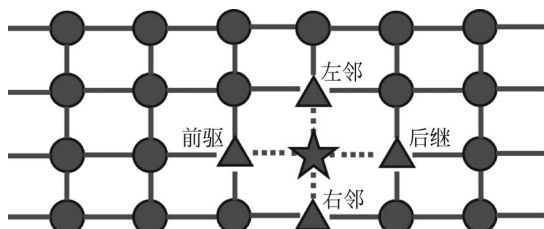


图 5-14 LaneGCN 对道路节点邻接关系的定义

首先, LaneGCN 对每一种邻接关系单独分配一组 GCN 模块,得到 14 个并行的 LaneGCN 模块,每一个模块按照各自的邻接关系对节点进行聚合,最后再将 14 个聚合的结果进行相加。这一操作类似于 CNN 的卷积,对上下左右几个位置分配不同的权重, LaneGCN 则是对不同关系的邻居分配不同的权重并采用类似空洞卷积的操作,有效地避免了过平滑和感受野过小的问题。

其次, LaneGCN 将地图节点特征和车辆特征进行融合。LaneGCN 采用轮转式的信息聚合顺序,即车辆到地图、地图到地图、地图到车辆以及车辆到车辆。融合过程如图 5-15 所示。地图节点本身不仅需要包含静态的拓扑信息,还要包含动态的交通信息。比如当前时刻有的车道十分拥挤,而有的车道则非常通畅,这对车辆的预测有着非常强的指导意义。随后是地图到地图,即再次使用 GCN 对道路节点的关系进行聚合,区别就是这次节点已经包含了动态场景的信息。

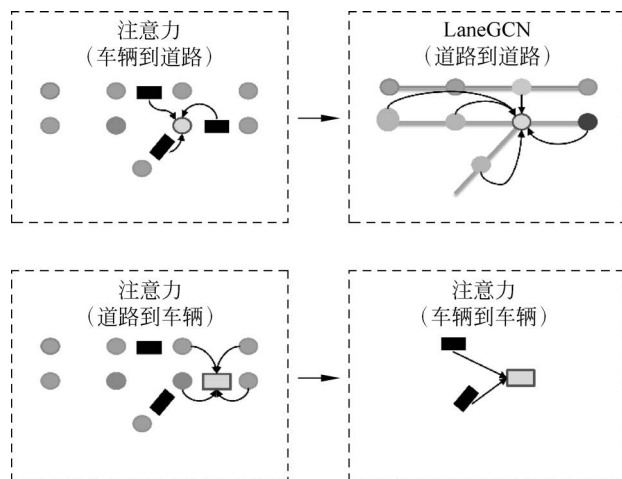


图 5-15 LaneGCN 的不同类型单元的聚合顺序

最后,需要进行地图到车辆的聚合,即地图节点再把道路静态拓扑结构信息以及动态信息反馈给车辆本身,让车辆能够“感知”周围的动态和静态环境。最后是车辆与车辆之间的聚合,即建模车辆之间的交互信息。上述 3 种交互方式都是局部化的,即将被聚合的节点周围一定范围内的节点视为邻居节点,并且聚合需要考虑节点之间的距离,因此采用节点-距离-节点这样的三元组拼接的形式,对邻居节点使用多层感知机进行编码再聚合。通过上述的信息交换过程,每个车辆的特征已经包含了场景特征,之后利用每个车辆的特征进行车辆的预测。

LaneGCN 直接将地图视为图的建模问题。除了图以外,还有使用点云的建模方式,比如 TPCN^[180]等。实际上无论是轨迹的动态时序点(点的属性中假设已经像 VectorNet 那样嵌入时间戳),还是道路线的采样点,它都可以视为二维空间下的一个点,点与点之间满足无序性和变长性等点云性质。因此整个场景可以视为包含一系列点的点云集合。点云编码一个非常重要的组成就是索引方式,如图 5-16 所示。有了索引之后,就可以用一些常见的点云模型,比如 PointNet++ 和 PV-RCNN 等对场景按点云的思路进行编码。限于篇幅,此处不再赘述。

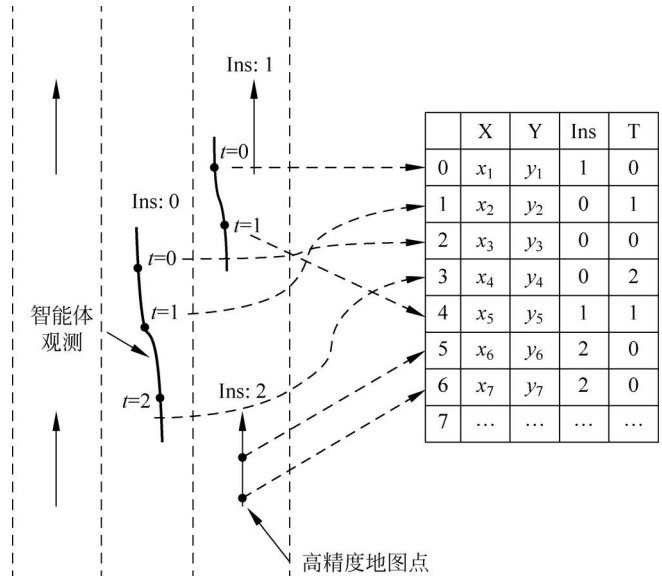


图 5-16 轨迹预测中点的索引方式

3) 几何对称性编码

最后讨论一下轨迹预测编码问题中的几何对称性问题。随着几何深度学习的兴起,越来越多的研究者开始在不同的领域研究数据的几何对称性质,轨迹预测领域也不例外。这里的几何对称性通常指旋转、平移、置换的等变性与不变性。等变性是指输入做了旋转或平移等某种变换时,输出也做出相等的变换。不变性是指无论输入做出任何变换,输出的结果都不变。以旋转为例,图 5-17 给出了轨迹预测问题下的几何对称性。可

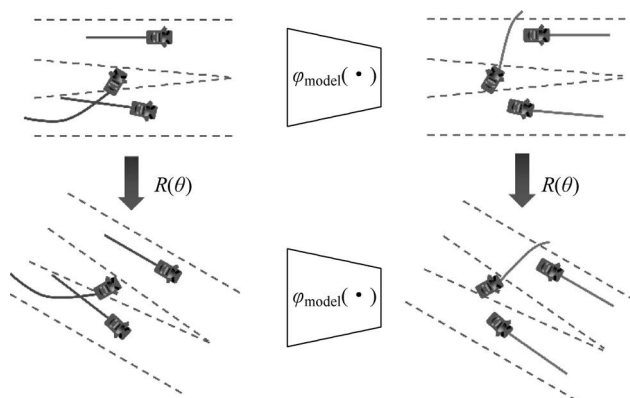


图 5-17 轨迹预测中点的旋转等变性

以观察到,当整个场景发生旋转时(即参考系发生一个变化),预测的轨迹也应该保证旋转相应的角度,即满足旋转的等变性。然而现有神经网络如果不经过人工干预,基本不可能做出一种对称的预测。因此对于轨迹预测模型引入几何对称性也是未来一个重要的研究方向。引入方式可以参考药物分子预测中采用的 N 元组形式^[181]或点云处理中采用的向量化神经网络等方法^[182]。

2. 解码部分

动态复杂场景下的多模态解码有两种主流方式,分别是基于回归的一阶段和基于分类的二阶段。与单一静态场景不同,复杂城区动态场景很难预先定义一组行为意图。所以在这种场景下,研究者们直接抛弃了行为的预定义,而是直接使用一种与意图无关的“赢者通吃”算法。这类解码的代表方法有 LaneGCN、TPCN 以及 Multipath 等,其主要思路如图 5-18 所示。解码模块包括一个置信度预测分支和 K 个模态的轨迹预测分支,而实际训练数据只有一个真值。然后将真值分别与 K 个模态进行比对,取误差或损失最小的作为匹配成功的结果。最后只训练匹配成功的分支,而忽略其他预测分支。除了上面提到的“赢者通吃”算法,还可以使用基于混合高斯分布假设的期望最大(Expectation-Maximization, EM)算法进行优化。

1) 一阶段解码模型

一阶段解码模型的优势是速度快,无须进行后处理,可以直接预测出结果。然而,其缺点在于解码阶段并没有显式地考虑地图约束信息。虽然模型在编码阶段会对地图有关的信息进行编码,但是由于神经网络的不可解释性,模型并不能保证在测试阶段预测的轨迹都在可行驶区域。因此在解码阶段需要额外的约束去保证这种预测的合理性与自洽性。由此出现了二阶段基于终点的解码方式。

2) 二阶段解码模型

二阶段解码按照预定义终点的疏密程度又分为稀疏终点解码(终点间隔一般大于 2 m)和密集终点解码(终点间隔为 0.25~1 m)。稀疏终点解码的代表方法有 LaneRCNN^[183]以

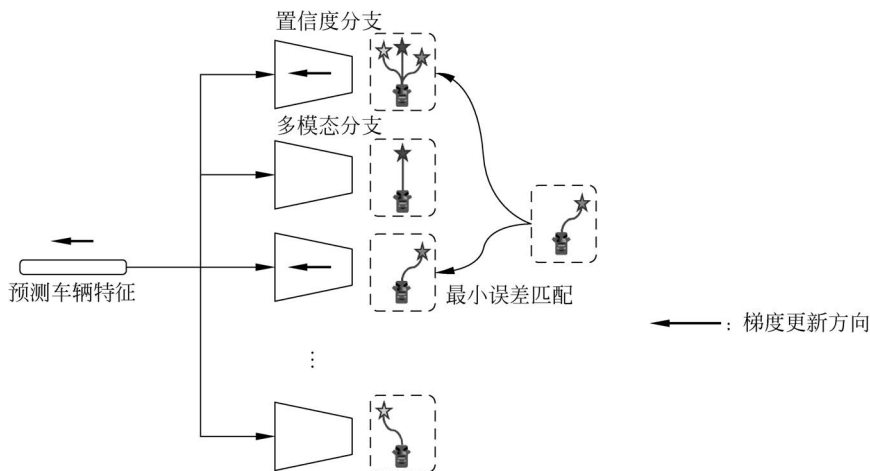


图 5-18 “赢者通吃”优化算法

及 TNT^[184]，图 5-19 给出了 TNT 算法中的终点采样思路。与一阶段直接回归整条轨迹不同，二阶段模型首先定义一系列终点（即要预测的最后一个时刻的位置），终点可以根据地图信息或道路拓扑结构进行启发式采样，然后再预测每个终点的置信度和偏移量（这里的终点就类似于目标检测的 Anchor），每一个终点对应一个模态。在训练时，离真值近的终点会有较大的置信度标签，反之则会较小。其次，根据预测好的每个终点补全轨迹。最后使用非极大抑制（NMS）算法对轨迹进行筛选，以保留预先设定的轨迹数量（即多模态的模态数量），得到多模态的预测轨迹。二阶段稀疏解码可以视为回归和分类问题，它的优势是提高了多模态轨迹命中率（保证至少有一个预测终点与真值小于某个阈值）的指标，即它通过人为扩大预测终点的覆盖范围尽可能保证较高的命中率。

然而，基于稀疏定义的终点很难覆盖所有行驶区域，并且终点间隔较大会造成终点的预测误差增加。因此，研究者们提出了基于热力图或密集终点的预测，代表方法有 DenseTNT^[185] 以及 Home 系列^[186]。这里以 Home 为例介绍密集终点的两种解码过程。由于候选终点分布较为密集，因此可以不像稀疏终点那样去回归偏差值。模型仅对每个终点做分类，因此密集终点是一类纯分类问题，类似于 CV 任务中的语义分割。密集终点的优势除了可以预测更加细粒度的终点坐标外，还能利用不同的后处理采样方式优化不同的多模态轨迹预测评价指标，比如最小终点误差（Minimum Final Displacement Error, minFDE）以及最小命中率（Minimum Hit Rate, MR）等，图 5-20 给出了不同的采样思路，比如 minFDE 侧重于使所有预测终点尽可能接近真值，而 MR 则侧重于让终点分散，尽可能覆盖较大的范围。二阶段模型虽然精度高、采样灵活，但是由于需要 NMS 等后处理，因此计算速度较一阶段解码慢。除此之外，二阶段模型还需要人工启发式地根据道路场景和交通规则定义一系列终点。这会给二阶段模型的实际部署带一定来困难，需要后续研究去克服。

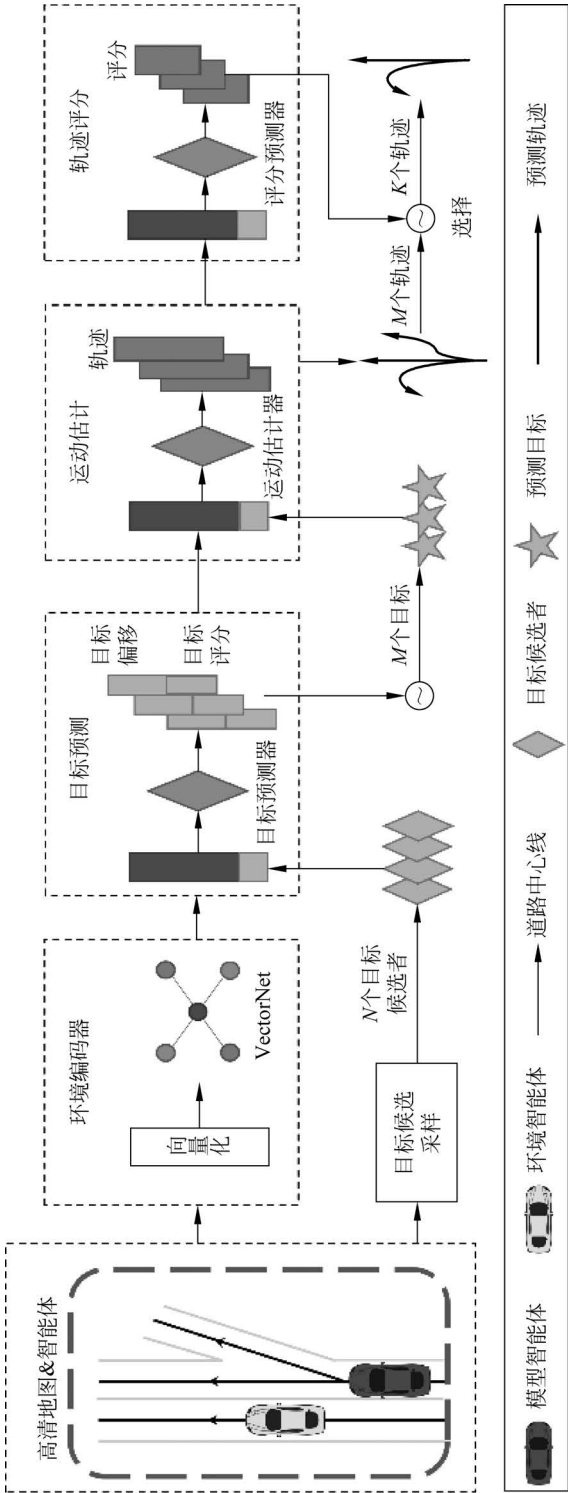


图 5-19 TNT 的解码结构

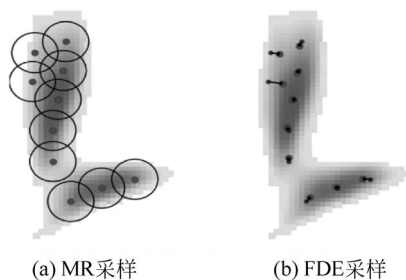


图 5-20 针对不同评价指标的不同采样方式

3) 解码自洽性

最后介绍解码的自洽性的问题。之前介绍的解码都是针对某辆特定的车,默认车与车之间的预测没有显式的依赖关系。然而这种假设在某些情况下是不合理的,可能会产生两辆车相撞的预测(在现实中是小概率,而在模型预测中会经常出现),与真实物理世界会发生矛盾,如图 5-21 所示。因此在解码时不能只考虑一辆车,而是要对整个场景所有车辆做出一组自洽的预测。这就需要模型的输入不仅是观测到的场景历史信息,还需要包含模型自身做出的预测,即多次地对模型自身的预测做出“反思”与修正。因此,不仅在编码端需要考虑车辆的历史信息交互,在解码端也需要考虑车辆的未来预测信息交互。考虑自洽性并且对预测进行多次修正的代表性方法有 M2I^[187] 以及 SceneTransformer^[188],它们在进行一次预测之后,再通过某些手段将预测作为输入反馈给模型自身做出二次或多次预测以得到更加合理与自洽的预测结果。

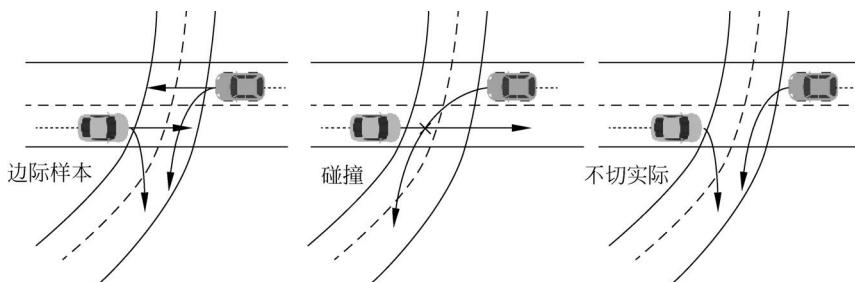


图 5-21 车辆间不合理的预测

5.4 周围车辆驾驶风格识别

驾驶风格是指驾驶员操纵车辆所表现出的相对稳定的行为特性^[189]。在车辆行驶过程中,不同驾驶风格(如保守的或者激进的)或不同的驾驶条件(如醉驾或赶路)会导致不同的行为倾向,进而影响车辆运动轨迹。5.3 节介绍的车辆轨迹预测方法大多没有考虑不同驾驶方式对预测的影响,而是将相同的预测模型应用于具有不同驾驶风格的车辆。因此,本节介绍 3 种驾驶风格分析和识别方法,以便于实现针对个体驾驶者的个性化的行为预测。

5.4.1 应用高斯混合模型的驾驶风格识别

驾驶风格的类别与其含义通常是根据研究者的主观或其他客观研究来定义的。目前大部分的驾驶风格识别的工作将驾驶风格划分为3种类型,包括谨慎型、正常型和激进型。本节首先介绍一种利用混合高斯的驾驶风格识别方法。

1. 高斯混合模型

高斯混合模型(GMM)是一个参数概率密度函数,由 k 个高斯子分布的密度线性加权组合而成。使用足够数量的子分布混合,可以逼近几乎任意一个连续概率密度函数,高斯混合密度定义为

$$p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x | \mu_k, \Sigma_k) \quad (5-25)$$

其中, x 是一个 d 维的随机变量, $\mathcal{N}(x | \mu_k, \Sigma_k)$ 是一个均值为 μ_k 、方差矩阵为 Σ_k 的多源正态分布。 π_k 为 k 个子分布的高斯混合系数,需要满足 $0 \leq \pi_k \leq 1$ 并且 $\sum_{k=1}^K \pi_k = 1$,以满足子分布间的组合是凸性的。

高斯回归过程(Gaussian Processes Regression, GPR)是使用高斯过程先验对数据进行回归分析的非参数模型。对于输入训练数据集 $D = \{(x_i, y_i)\}_{i=1}^N = (\mathbf{X}, \mathbf{Y})$,其中, $\mathbf{x}_i \in \mathbb{R}^d$ 为 d 维输入向量, $\mathbf{X} = [x_1, x_2, \dots, x_n]$ 为 $d \times n$ 维输入矩阵, $y_i \in \mathbb{R}$ 为相应的输出标量。输入数据集 X 通过给定函数映射至高维空间,可构成一个随机变量集合 $\{f(x_1), f(x_2), \dots, f(x_N)\}$,且具有联合高斯分布。该高斯过程全部统计数字特征可由均值函数 $m(x)$ 和协方差函数 $k(x, x')$ 确定,即

$$f(x) \sim \text{GP}(m(x), k(x, x')) \quad (5-26)$$

其中, $m(x) = E[f(x)]$, $k(x, x') = E[(f(x) - m(x))(f(x') - m(x'))]$,核函数为标准指数协方差函数,即

$$\text{cov}(f(x), f(x')) = k(x, x') = \theta_0^2 \exp\left(-\frac{(x - x')^2}{2\theta_1^2}\right) + \sigma^2 \delta_{ij} \quad (5-27)$$

其中, θ_0 为指数权重, θ_1 为长度规模, δ_{ij} 是狄拉克函数,当 $i=j$ 时,函数 $\delta_{ij}=1$,否则为0。

基于迭代两步法的EM算法是一种强大而高效的算法,用于从训练数据中获取高斯混合模型的参数,从而建立最大似然估计。E-step使用当前参数估计值计算对数似然期望,M-step估计最大化E-step发现的对数似然期望的参数。

2. 驾驶风格识别

除了依据数值关系将驾驶员的驾驶风格划分为固定的类别,也可以将驾驶员的驾驶行为参数化,用函数或概率分布的方式描述驾驶员的驾驶风格。如建模跟车行为与踏板行为的范式时,可以将每个驾驶员的跟车距离同速度的关系通过一个基于非线性最优逼近方法的速度模型或是一个基于高斯混合模型的统计方法映射到一个二维空间。踏板行为范式同样基于高斯混合模型进行建模,其代表了原始踏板信号或是从原始踏板信号中提取的频域特征的概率分布。跟车距离同速度之间的关系构成了对不同驾驶员在跟车驾驶风格上的描

述。驾驶员根据交通状况调整自己的阈值,从而与前方车辆保持舒适的间隔。轨迹被认为是在不同速度下使驾驶员感到舒适的跟车距离的运动路径。

Zichen Chen 等^[190]设计了一个包括因子分析、GMM 聚类 and BP 神经网络(BPNN)的驾驶风格识别策略,如图 5-22 所示。首先,因子分析阶段(包括可行性分析、共同因子确定和因子得分)旨在降低 11 个特征参数的数据维度,并获得公共因子(驾驶、制动和环境因素)。其次,GMM 聚类算法根据共同因素的概率分布,通过找到一个合适的高斯分布,使共同因素的可能性达到最大,从而准确地给每个驾驶员贴上风格标签(谨慎、正常或激进)。最后,使用经优化的三层 BP 神经网络训练一个合理的识别模型,以识别不同的驾驶风格。

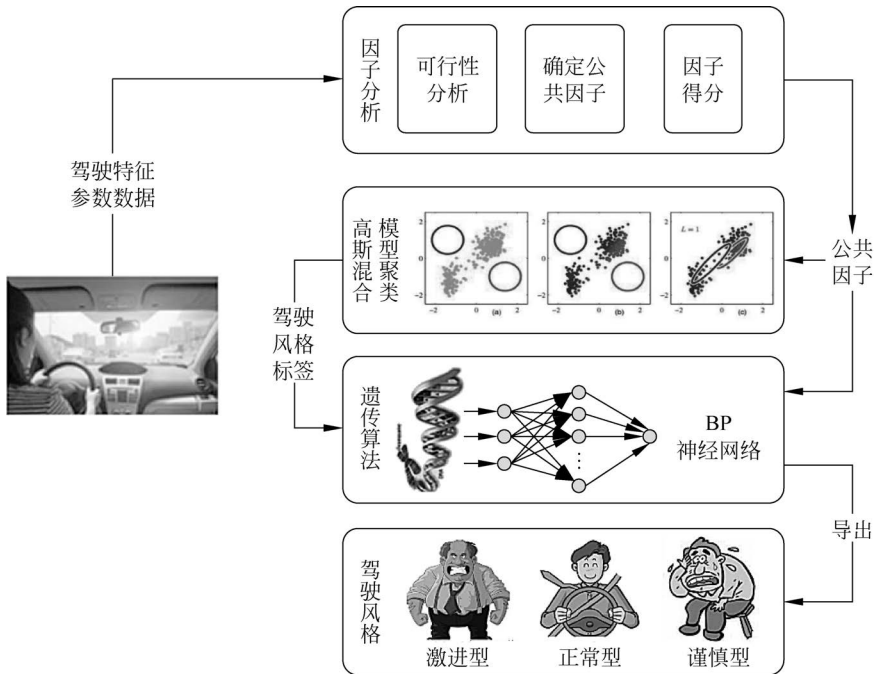


图 5-22 GMM 驾驶风格检测流程

此外,Nan Ding 等^[191]通过将驾驶风格同 GMM 混合模型结合,对不同的驾驶风格构建不同的影响因子,基于速度、加速度及距离等参数,实现了对异常数据的检测。

5.4.2 基于短期观测的实时驾驶风格分类方法

早期的工作使用高斯混合模型和层次聚类统计方法,通过无监督的方式让模型自行从预选驾驶参数或特征中挖掘不同的驾驶风格特征。然而,这些工作都需要对车辆和司机的历史数据进行相当长时间的观察,很难适用于复杂交通场景下在线的实时驾驶行为分析。因此,本节介绍一种基于短期观测的实时驾驶风格分类方法^[192]。

如图 5-23 所示,该方法由训练和推理两个阶段组成。在训练阶段,收集了某一特定交通系统(如一段高速公路)的历史数据,并采用无监督学习方法将数据聚类为不同的类别,这些类别对应不同的驾驶风格。同时,在推理阶段建立并应用由此产生的聚类策略,将同一系

统中任何新观察到的车辆划分为特定的驾驶风格。在训练和推理阶段,为了加速聚类过程,对高维传感器数据进行了预处理和主成分分析(Principal Component Analysis, PCA)等降维处理,从而充分利用与驾驶行为密切相关的特征。

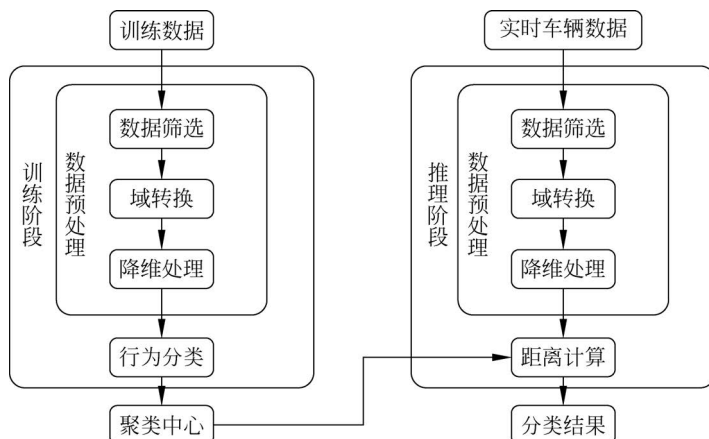


图 5-23 基于短期观测的实时驾驶风格分类方法框图

具体来说,数据预处理分为数据过滤和频域转换两个步骤。在数据过滤阶段,该方法对大量的传感器数据进行过滤,只选择与车辆动作(如位置、速度和加速度)或与其他车辆的时间和空间关系(如与前车的相对位置)相关的特征。由于驾驶员行为通常对其行为的绝对时间戳不敏感,且频域可以包含更多关于驾驶行为的信息,因此在进行聚类之前,该方法不是直接对时间数据进行处理,而是对每个 k 维时域特征进行傅里叶变换,进一步过滤那些与驾驶行为决策无关的时间戳数据。为减少数据噪声以及降低后续聚类的复杂度,随后对频域数据进行 PCA 降维处理。

数据处理完成后,训练阶段利用 k -means 方法对数据进行聚类,得到 k 个聚类中心,使得训练数据中的驾驶员得到不同的类别标签,每个标签对应不同的驾驶风格。在测试阶段,当观察到同一系统中的新驾驶员时,经过同样的数据处理之后,计算特征与训练得到的聚类中心的距离,以对该驾驶员进行分类,从而识别出驾驶风格的类型。

5.4.3 基于时序数据的 CNN-LSTM 驾驶风格识别模型

如 5.4.1 节和 5.4.2 节所述,传统的驾驶风格识别算法利用从原始驾驶数据中提取的手工特征,再应用不同的机器学习模型来识别驾驶行为风格。而这类解决方案往往受到所选特征集和所选模型的限制,考虑到驾驶风格分类任务中数据类型的缺乏,以及广泛使用的无监督聚类算法和单一卷积神经网络方法的识别精度不高,本节介绍一种基于 CNN 和 LSTM 的高精度、快速、可推广的驾驶风格识别方法^[193]。

首先,针对不完善的驾驶数据,如图 5-24 所示,该方法提出了一种收集驾驶员操作时间序列信息的方法,然后通过卷积神经网络提取驾驶员的风格特征,再针对收集到的时间数据,引入长短时记忆网络(LSTM)模块对驾驶特征进行编码和转换,实现驾驶风格分类。结果表明,驾驶风格识别的准确率达到了 93% 以上,同时速度也有了明显的提高。

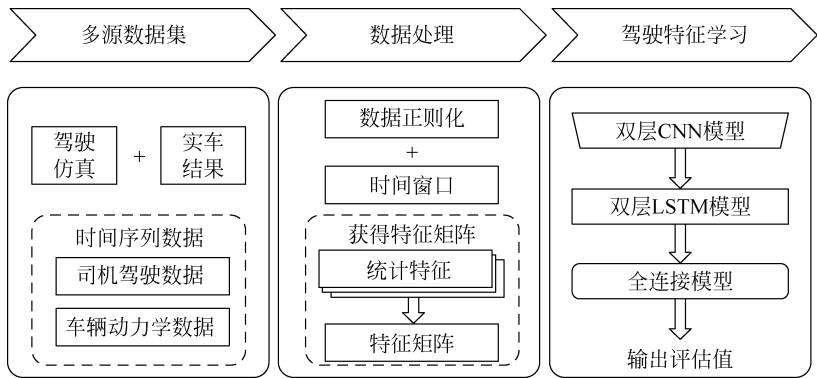


图 5-24 CNN-LSTM 驾驶风格识别框

1. 数据收集

现阶段用于驾驶风格分类的数据大部分来自感知数据,如驾驶员面部图像、眼球运动数据,以及包括 GPS 和车载三轴加速度计在内的轨迹数据。这些研究方法虽然取得了较高的精度,但由于车辆上装备了很多传感器,易受天气、振动等因素的影响,在真实环境中难以保证精度,同时也存在侵犯隐私的风险。作为车辆的直接操作者,驾驶员的操作数据最能直观地反映其驾驶风格,而车辆的运动学数据则能反映驾驶员对车辆运动状态的期望。为解决数据问题,该方法同时收集了模拟器数据和真车数据,形成了一个数据集。与真车相比,模拟器可以构建一个全面的驾驶环境,充分模拟驾驶员在城市道路、高速行驶等复杂环境的工作状态。同时,驾驶员的周围环境相对固定,减少了环境对驾驶风格的影响。然而,单独使用模拟器限制了数据的全面性,因此该方法采用了驾驶模拟器和真车数据的融合,其中模拟器数据作为网络训练的训练和验证集,真车数据在网络训练完成后用于评估泛化性能。采集的具体数据类型及其频率见表 5-1。

表 5-1 采集的数据类型及频率

数据类型	采样频率(Hz)
驾驶员身份码	100
数据采集时间跨度	100
方向盘角度	50
方向盘速度	50
油门踏板开度	100
刹车踏板开度	100
车辆偏航率	100
车辆纵向速度	100
车辆横向速度	100
车辆纵向加速度	100
车辆横向加速度	100
油门和刹车踏板开度组合	100

2. 数据处理

考虑到不同数据的维度差异,该方法首先对数据进行归一化处理,即根据数据的阈值范围将不同的数据投影到相同的固定区间。针对采集到的数据,该方法采用模糊 C 均值算法(Fuzzy C-Means, FCM)和谱聚类(Spectral Clustering, SC),得到相应的驾驶员驾驶风格标签。具体而言,设置聚类的类别数为 K ,应用 FCM 和 SC 进行模糊均值聚类,得到结果 $C_f(i)$ 、聚类中心 m_j 和谱聚类的聚类结果 $C_s(i)$,其中, i 为样本 ID。对于样本 i ,如果 FCM 和 SC 聚类结果 $C_f(i) = C_s(i)$,则样本 i 被标注为聚类的类别标签,并被划分为标注子集 X_i 。如果两次聚类结果不同,则根据驾驶员的主观评价得到该驾驶员对应的驾驶风格标签。

传统的聚类方法在驾驶员检测中的实时性较差,容易受到不均匀样本的影响而导致聚类的中心发生偏移。因此,该方法训练了一个用于驾驶员驾驶风格识别的网络模型,一旦网络训练完成,其结果就不会受到输入信息整体偏差的影响。驾驶风格作为一个包含在多维驾驶信号中的广泛特征,很难用一个特定的信号类型来明确表示。因此,需要对收集到的原始数据流进行处理,常见的做法是将数据拆分,并确定数据段中某一数据的统计参数,然后从整体数据的统计参数中得到该数据对应的驾驶风格。这项工作也采用了同样的方法,基于原始数据,使用上下文窗口的概念,将注意力集中在一个固定的时间窗口上。小窗口用于细化具体的数值大小,由小窗口组成的大范围特征矩阵用于捕捉瞬时变化的数值特征中包含的风格特征。双时间窗方法允许对瞬态驾驶行为的特征进行微观计数和转换,并在宏观层面上传达了驾驶员的驾驶风格信息。然而,这些信息只作为矩阵中的统计信息存在,并不以某一特征的数据形式表达。因此,深度学习网络能从这个矩阵输入中学习并表达司机的驾驶风格特征。

3. 特征提取

首先,对于处理过的数据,用卷积神经网络来提取驾驶数据中的驾驶特征。该方法使用的卷积神经网络由两个卷积层和两个最大池化层组成,每个卷积层后面都有一个最大池化层用于下采样。由第二个池化层产生的特征矩阵被叠加起来,作为网络下一阶段的输入。由于不能对时序的驾驶特征进行卷积操作,因此该方法在时间轴上应用一维卷积。

之前的研究证实了卷积神经网络用于驾驶风格分类任务的可行性,然而通常在 CNN 完成特征提取后直接利用全连接层进行分类,而不应将驾驶员的操作特征简单叠加而得到驾驶员的驾驶风格;在实际驾驶员驾驶车辆的时间维度上,每个不同的参数特征的出现都是随机的,全连接操作可能导致对驾驶风格的错误理解。从本质上讲,驾驶员操作数据和车辆动态数据具有很强的时间特性,因此,该方法使用 LSTM 模块对驾驶特征之间的依赖关系进行编码,并从最后的全连接层输出预测的驾驶风格。每个 LSTM 单元通过其中的门控状态控制传输状态,记住需要长期记忆的内容,如司机的驾驶风格;而忘记不重要的信息,如短时间内驾驶信息的变化。因此,相对于全连接模型而言,LSTM 模型更适合于驾驶风格分类的任务。

最后,在 LSTM 网络之后,通过一个两层全连接网络,并使用 Softmax 函数输出不同驾驶方式的预测概率值,从而实现驾驶风格识别。