交互动力系统深度状态空间模型的低成本确定性推断

Andreas Look *andreas.look@bosch.com*
博世人工智能中心

Melih Kandemir *kandemir@imada.sdu.dk*
南丹麦大学

Barbara Rakitsch *barbara.rakitsch@bosch.com*
博世人工智能中心

Jan Peters *peters@ias.informatik.tu-darmstadt.de*
达姆施塔特工业大学

arXiv:2305.01773v1 [cs.LG] 2 May 2023

摘要

图神经网络常用于对相互作用的动力系统建模，因为它们能够灵活地扩展到智能体数量可变且较多的系统。尽管确定性相互作用系统方面已取得长足进展，但对随机系统进行建模要困难得多，因为我们希望获得未来轨迹的预测分布。现有方法要么依赖蒙特卡洛采样，因而计算速度较慢；要么采用简化假设，使预测分布成为单峰分布。在这项工作中，我们提出了一种深度状态空间模型，利用图神经网络对底层相互作用的动力系统进行建模。预测分布是多峰的，呈高斯混合模型形式，其中高斯分量的矩可通过确定性矩匹配规则计算。我们的矩匹配方案可用于无采样推断，与蒙特卡洛替代方法相比，能够实现更高效、更稳定的训练。此外，我们提出了高斯分量协方差矩阵的结构化近似方法，以扩展到包含众多智能体的系统。我们在两个具有挑战性的自动驾驶数据集上对这一新框架进行了基准测试。结果均证实，与最先进的方法相比，我们的方法具有优势。我们还通过精心设计的消融研究展示了各项独立贡献的价值，并详细分析了所提出的协方差近似方法的运行时间。最后，我们通过在未见过的场景中评估方法性能，实证展示了该方法的泛化能力。

1 引言

许多动力系统，如交通流（Li 等人，2018；Yu 等人，2018）、流体动力学（Ummenhofer 等人，2019）或人体运动（Jain 等人，2016），都涉及智能体之间的交互。图神经网络（图神经网络）（Battaglia 等人，2018）近年来已成为此类场景中的强大工具，因为它们能够仅从数据中学习交互系统的动力学。近期研究在确定性复杂系统建模方面取得了巨大进展，能够将智能体数量少、时间范围短的系统外推到智能体数量多、时间范围长的系统。这些方法已成功应用于涵盖流体、刚性固体和可变形材料的多种物理系统（Sanchez-Gonzalez 等人，2020）。然而，对于许多现实应用而言，为每个智能体预测一条未来轨迹并不足够，因为动力系统中的随机性会带来重大影响。例如，在自动驾驶中，驾驶员的意图（如超车、转弯、变道）是一个隐藏因素，可能导致驾驶轨迹呈现不同的模态。

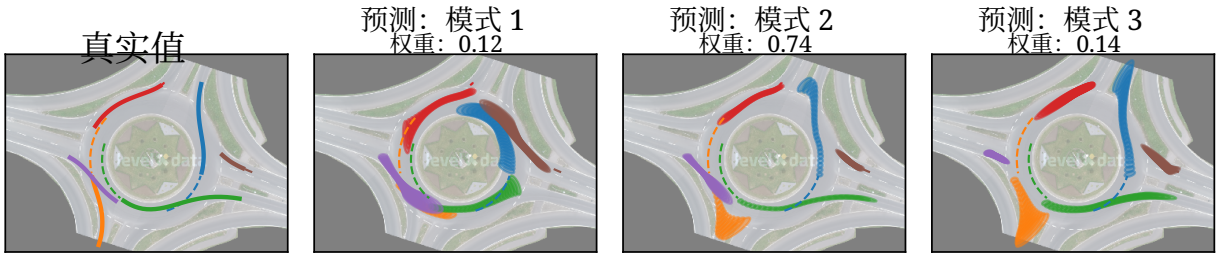


图 1：我们通过确定性矩匹配规则，将潜在图深度状态空间模型（GDSSM）的预测分布近似为高斯混合分布。给定以观测轨迹（虚线）形式呈现的历史信息，我们提出的 GDSSM 架构在预测未来动力学时会考虑交通参与者之间的交互。我们用实线表示真实的未来轨迹（最左侧图）。对于每种模式和每个交通参与者，我们展示模型预测的 95%置信区间（最右侧的三个图）。我们的模型会考虑交互，例如，只有当蓝色车辆留在环岛内时，棕色车辆才会驶入环岛（模式 1）。如果蓝色车辆正驶离环岛，蓝色车辆就会挡住棕色车辆的驶入车道（模式 2、模式 3），棕色车辆只能等待。真实数据和地图来自 rounD数据集（Krajewski等人，2020）。

在深度学习文献中，多种架构已成功应用于时间序列数据，其中两类尤为突出：a）循环方法，在每个时间步反复应用固定的转移模型（例如 Hochreiter（1991）、Chung 等人（2014））；b）基于历史记录的方法，通过卷积滤波器（例如 Bai 等人（2018）、Oord 等人（2016））或注意力模块（例如 Vaswani 等人（2017）、Li 等人（2019））汇总过去的信息。循环方法在潜在状态 x_t 中捕获系统在每个时间点 t 的内部状态。以这种方式预测输出遵循动力系统的因果顺序，即，要计算下一个时间点 $t+1$ 的潜在状态 x_{t+1}，需要先得到时间点 t 的潜在状态 x_t。状态空间模型（例如 Särkkä（2013））属于循环方法。这类模型由两个概率分布定义：转移模型 $p(x_{t+1}|x_t)$，用于随时间向前传播潜在状态；发射模型 $p(y_t|x_t)$，用于将潜在状态映射到观测空间。要获得状态空间模型的多步超前预测，需要在潜在空间中传播分布。对于一般的转移模型，这一操作无法以闭式形式完成，现有方法要么使用数值积分方案（例如 Solin 等人（2021）、Look 等人（2022）），要么使用蒙特卡洛方法（例如 Krishnan 等人（2017）、Bayer 等人（2021））。对于交互系统，潜在空间会随智能体数量线性增长，因此需要大量 MC 样本，这可能导致这些方法慢得难以实用。据我们所知，数值积分方案尚未用于多智能体动力系统。相比之下，概率式的基于历史记录的方法直接预测未来轨迹上的分布，从而减轻采样开销。然而，这种方法会使学习问题变得困难，因为模型需要学习多个未来时间步的分布。为此，通常需要复杂模型（例如大型神经网络）。这可能会妨碍它们在内存容量有限的嵌入式系统中使用。

在本研究中，我们提出了一种用于对具有交互智能体的随机动力学系统进行建模的新方法，该方法能够以高效的运行时间和内存开销，生成关于未来轨迹的丰富多模态预测分布。我们将未知的随机动力学系统建模为深度状态空间模型 （DSSM），其中所有智能体的共享动力学通过图神经网络在联合潜在空间中进行建模。我们的模型属于循环神经网络这一类别，并通过引入一种新颖的确定性矩匹配方案，替代训练和测试过程中计算开销较大的蒙特卡洛（MC）操作。此前，Look 等人（2022）针对动力学系统提出的矩匹配研究并未考虑交互系统，且仅适用于单峰过程。我们通过在转移模型中应用图神经网络克服了第一个局限，并通过在初始潜在状态上设置高斯混合模型 （GMM）克服了第二个局限。对于每个混合分量，我们分别应用矩匹配规则，从而得到关于未来轨迹的多模态预测分布。在自动驾驶领域，初始潜在状态

通常通过历史信息进行估计，并且能够提供有关驾驶员意图的信息（Tang & Salakhutdinov，2019）。在给定初始潜在状态的条件下，预测分布通常可以通过单峰分布进行准确建模（Cui 等人，2019；Chai 等人，2019）。最后，由于密集交通场景具有多样性，智能体数量众多可能导致高斯混合模型（GMM）的协方差矩阵规模过大，因为其矩阵大小会随着交通参与者数量的增加而呈二次增长。我们通过提出结构化协方差近似方法来解决这一问题。

我们将贡献总结如下：

- 我们推导了图神经网络层的输出矩，使图神经网络能够应用于矩匹配算法。由此，我们提出了首个适用于交互系统深度状态空间模型的确定性推断方案。
- 我们在初始潜在状态上引入高斯混合模型分布，从而得到关于未来轨迹的多模态预测分布。
- 我们提出了针对高斯混合模型协方差矩阵的结构化近似，可将我们方法的计算复杂度从 $O(M^3)$ 降至 $O(M^2)$，其中 M 表示智能体数量。

在实验中，我们在两个具有挑战性的自动驾驶数据集上对所提出的模型进行了基准测试。结果表明，与最先进的替代方法相比，我们的确定性模型具有出色的实证性能。图1展示了一个真实世界中多个智能体在环岛场景下的预测输出分布。由于交通参与者可以从多个出口驶离环岛，未来分布具有高度多模态的特征。我们的模型能够预测多个模态，并将其高效地近似为高斯混合模型，同时利用图神经网络考虑交互影响。

为了进一步深入了解我们的模型和推断方案，我们通过消融研究仔细考察了各项工作内容的单独影响。接下来，我们对协方差近似进行了实证运行时间研究。研究结果表明，稀疏协方差近似可将计算复杂度降低最多 100 倍，因此适用于计算资源有限的应用场景。最后，我们通过研究模型在分布外数据上的泛化能力来结束实验，例如在训练期间未曾观测到的交通环境。

2 背景

本章首先介绍单智能体系统的（深度）状态空间模型背景。随后，我们简要回顾矩匹配规则，这些规则能够以确定性方式将潜在动力学向前传播。接着，我们回顾用于交互建模的图神经网络。状态空间模型和图神经网络构成了我们针对随机动力学交互系统所提出的新模型的基础，我们将在第 3 节中介绍该模型。确定性矩匹配规则构成了训练和测试流程的基础，我们将在第 4 节中对这些流程进行描述。

2.1 深度状态空间模型

状态空间模型 （SSM）是一类用于动力学系统的模型（例如 Schön 等人（2011）、Särkkä（2013）），假设每个 D_y 维观测变量 $y_t \in \mathbb{R}^{D_y}$ 都由一个 D_x 维潜在变量 $x_t \in \mathbb{R}^{D_x}$ 生成。各潜在变量通过一阶马尔可夫动力学相互耦合，例如，时间点 x_t 的状态仅取决于前一时间点 x_{t-1} 的状态。通常，观测状态 y_{t-1} 不包含可靠预测下一个观测状态 y_t 所需的全部信息。以交通预测为例，观测状态 y_t 包含车辆的位置，却没有其速度或加速度数据。此时，我们可以将缺失的信息补充到潜在状态 x_t 中，从而准确预测下一个时间点。因此，状态空间模型是一类灵活的模型，可用于对复杂系统进行可靠预测。

一个深度状态空间模型（DSSM）是一种非线性状态空间模型（SSM），其中转移模型将潜在状态从当前时间点映射到下一个时间点，而发射模型则将潜在状态映射到 the

输出由神经网络实现。它们适用于底层真实动力学具有未知非线性函数形式、必须从数据中估计的应用场景。假设存在加性高斯噪声（例如 Krishnan 等（2017）），其生成模型可写为如下形式。

$$x_0 \sim p(x_0|\mathcal{I}), \tag{1}$$
$$x_t \sim \mathcal{N}\left(x_t|x_{t-1} + f(x_{t-1},\mathcal{I}), \operatorname{diag}\left(L(x_{t-1},\mathcal{I})\right)\right), \qquad t = 1,\dots,T \tag{2}$$
$$y_t \sim \mathcal{N}\left(y_t|g(x_t), \operatorname{diag}\left(\Gamma(x_t)\right)\right), \qquad t = 1,\dots,T \tag{3}$$

其中，$\mathcal{I} \in \mathbb{R}^{D_\mathcal{I}}$ 是对辅助信息（如历史信息或关系信息）进行编码的上下文变量。均值更新 $f(x_t,\mathcal{I}): \mathbb{R}^{D_x} \times \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}^{D_x}$ 控制转移模型的确定性成分，并由具有任意架构的神经网络参数化。类似地，方差更新 $L(x_t,\mathcal{I}): \mathbb{R}^{D_x} \times \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}_+^{D_x}$ 由另一个神经网络参数化，用于对系统的随机性建模。 f 和 L 均为神经网络，分别由 θ_f 和 θ_L 参数化。发射模型服从高斯分布，其均值为 $g(x_t): \mathbb{R}^{D_x} \to \mathbb{R}^{D_y}$，方差为 $\Gamma(x_t): \mathbb{R}^{D_x} \to \mathbb{R}_+^{D_y}$；其中 g 和 Γ 均为架构和参数 θ_g、θ_Γ 任意的神经网络。

假设采用加性高斯噪声，我们可以将转移模型 [Eq. (2)] 解释为离散化的神经随机微分方程（Tzen & Raginsky, 2019; Look 等人, 2022）。尽管我们不再进一步探讨这一研究方向，但值得注意的是，这种联系使得将其直接扩展到不规则采样时间序列成为可能。最后，还有一些研究将状态空间模型与循环神经网络相结合（Chung et al., 2015; Fraccaro et al., 2016），其门控机制有助于学习长期效应。

2.2 转移核

本节介绍如何计算具有 $t > 1$ 的 t 步转移核 $p(x_t|x_0, \mathcal{I})$，以便在一般状态空间模型中将潜在状态向前传播。其定义如下递推关系

$$p(x_t|x_0,\mathcal{I}) = \int p(x_t|x_{t-1},\mathcal{I})p(x_{t-1}|x_0,\mathcal{I})dx_{t-1}, \tag{4}$$

其中 $p(x_0|\mathcal{I})$ 遵循式（1）。值得注意的是，式（4）无法以闭式形式计算，因为分布 $p(x_{t-1}|x_0, \mathcal{I})$ 必须经过非线性转移模型 $p(x_t|x_{t-1}, \mathcal{I})$ [式 2] 传播。

2.2.1 概述

针对转移核 [式（4）]，已有多种近似方法，大致可分为两类：（a）基于蒙特卡罗采样的方法（Brandt 与 Santa-Clara，2002 年；Pedersen，1995；Elerian 等人，2001）；（b）基于假设密度的确定性近似方法（Särkkä 等人，2015年）。蒙特卡罗方法在样本数趋于无穷时，能够以任意精度近似复杂分布，但在实际应用中往往速度较慢，且难以评估其收敛性。相比之下，确定性方法通常假设 t步转移核可以用高斯分布近似。如果转移模型可以进行局部线性近似，且观测值的采样足够密集，这一假设便有其合理性。随后，通过迭代方式计算转移核：在每个时间步应用高斯近似，并使用数值积分方案沿时间方向传播各阶矩（Särkkä 与 Sarmavuori，2013；Särkkä 等人，2015年）。

在神经随机微分方程的背景下，已有研究（Look 等人，2022年）提出，首先将微分方程离散化，随后通过神经网络层应用矩匹配，作为数值积分方案。由于所得算法通过时间和神经网络层传播矩，因此被称为 双维矩匹配 （BMM）。研究表明，与标准数值积分方案相比，该方法在计算量和准确性方面更具优势。由于状态空间模型中的转移模型可以解释为离散化随机微分方程（Särkkä 等人，2015年），我们基于他们的方法开展工作。此前，在期望传播（Hernandez-Lobato 和 Adams，2015年；Ghosh 等人，2016年）、确定性变分推断（Wu 等人，2019）以及证据深度学习（Haussmann 等人，2020）领域，也曾应用过通过神经网络层进行矩传播的方法。

2.2.2 二维矩匹配

在本节中，我们将重新介绍 Look 等人（2022年）提出的原始双维矩匹配 （BMM）算法，以及其在时间上向前传播潜在状态 [式（2）]中的应用。BMM 通过结合沿时间轴进行的水平矩匹配与跨神经网络层进行的垂直矩匹配，来近似转移核 $p(x_t|x_0, \mathcal{I})$。

水平矩匹配为了便于计算转移核，我们将所有时间步 $t = 1, \ldots, T$对应的 $p(x_t|x_0, \mathcal{I})$ 替换为高斯分布。

$$\begin{aligned} p(x_t|x_0, \mathcal{I}) &= \int p(x_t|x_{t-1}, \mathcal{I}) p(x_{t-1}|x_0, \mathcal{I}) dx_{t-1} \\ &\approx \mathcal{N}(x_t|\mu_t(\mathcal{I}), \Sigma_t(\mathcal{I})), \end{aligned} \tag{5}$$

均值为 $\mu_t(\mathcal{I})$、协方差为 $\Sigma_t(\mathcal{I})$。此近似将问题简化为计算转移核的前两个矩；若动力学可在局部近似为线性模型，则假设此方法效果良好，而许多应用都符合这一情况。假设单步转移核$p(x_t|x_{t-1}, \mathcal{I})$遵循式（2），则均值 $\mu_t(\mathcal{I})$ 和协方差 $\Sigma_t(\mathcal{I})$ 可根据先验矩 $\mu_{t-1}(\mathcal{I})$ 和 $\Sigma_{t-1}(\mathcal{I})$ 计算得出（Look 等人，2022年）

$$\begin{aligned} &\mu_t(\mathcal{I}) = \mu_{t-1}(\mathcal{I}) + \mathbb{E}[f(x_{t-1}, \mathcal{I})], \\ &\Sigma_t(\mathcal{I}) = \Sigma_{t-1}(\mathcal{I}) + \mathrm{Cov}[f(x_{t-1}, \mathcal{I})] + \mathrm{Cov}[x_{t-1}, f(x_{t-1}, \mathcal{I}),] + \mathrm{Cov}[x_{t-1}, f(x_{t-1}, \mathcal{I})]^T + \mathrm{diag}\left(\mathbb{E}[L(x_{t-1}, \mathcal{I})]\right), \end{aligned} \tag{6}$$

其中，$\mathrm{Cov}[x_{t-1}, f(x_{t-1}, \mathcal{I})]$ 表示参数中的随机向量之间的交叉协方差。

垂直矩匹配 转移函数的均值 $\mathbb{E}[f(x_{t-1}, \mathcal{I})]$ 和协方差 $\mathrm{Cov}[f_\theta(x_{t-1}, \mathcal{I})]$，以及期望方差更新 $\mathbb{E}[L(x_{t-1}, \mathcal{I})]$，均可通过神经网络层中的矩传播计算得出。对于许多常见的层，包括仿射变换和 ReLU 激活函数，相应的输出矩既可以通过闭式形式计算，也可以采用文献中已有的良好近似方法（Wu 等人，2019）。相比之下，无法使用矩匹配规则来近似交叉协方差 $\mathrm{Cov}[x_t, f_\theta(x_t, \mathcal{I})]$，因为我们无法将交叉协方差项分解为逐层运算。因此，我们转而使用 Stein 引理，借助

$$\mathrm{Cov}[x_t, f(x_t, \mathcal{I})] = \mathrm{Cov}[x_t] \mathbb{E}[\nabla_{x_t} f(x_t, \mathcal{I})], \tag{7}$$

其中，期望雅可比矩阵可近似为（Look 等人，2022年）

$$\mathbb{E}[\nabla_{x_t} f(x_t, \mathcal{I})] \approx \prod_{l=1}^{L} \mathbb{E}[J_t^l]. \tag{8}$$

上文中，J_t^l 表示层 l 在时间步 t处的雅可比矩阵。常见层类型的雅可比矩阵期望可解析求得，或可通过近似得到。

2.3 图神经网络

图神经网络 （图神经网络）已成为一种强大的交互建模方法（Battaglia 等，2018；Hamilton 等，2017；Gilmer 等，2017）。给定一组智能体及以图形式表示的关系信息，每个智能体对应图中的一个节点，并配有一组特征。智能体之间的关系通过边编码，智能体之间则通过沿边发送消息来交换信息。经过多轮消息传递，信息可以沿图流动。只要智能体之间存在路径，这种方式便可实现非相邻智能体之间的交互。

更正式地说，我们将 GNN 的结构定义如下。对于 M 个智能体，GNN 接收一组节点特征作为输入 $x = \{x^m\}_{m=1}^M$，其中 $x \in \mathbb{R}^{MD_x}$ 和 $x^m \in \mathbb{R}^{D_x}$，以及一组边 $\mathcal{E} = \{e^{m,m'}\}_{m,m'=1}^M$

它是上下文变量 $\mathcal{I} \in \mathbb{R}^{D_\mathcal{I}}$的一部分。边属性 $e^{m,m'}$ 采用二进制编码，其中$e^{m,m'} = 1$ 当智能体 m 与智能体 m' 相关联时。GNN 的输出是对节点特征的更新，即 $z = \text{GNN}(x, \mathcal{I})$，其中 $z \in \mathbb{R}^{MD_z}$，并由以下两个可重复多次的步骤组成：

1. 对于每个智能体 m，通过聚合邻近智能体的信息来接收消息 $x^{\mathcal{N}_m} \in \mathbb{R}^{D_x}$：

$$x^{\mathcal{N}_m} = \text{AGG}\left(\{x^{m'} | e^{m,m'} = 1\}\right) \in \mathbb{R}^{D_x}, \tag{9}$$

其中 $\{x^{m'} | e^{m,m'} = 1\}$ 表示节点 m的所有邻居构成的集合。聚合操作 AGG 具有置换不变性，即输入顺序互换时结果不会改变，并且适用于输入数量可变的情况。我们采用的一种常用聚合操作是均值函数。

2. 对于每个智能体 m，更新节点信息：

$$z^m = \text{UPDATE}(x^m, x^{\mathcal{N}_m}, \mathcal{I}), \tag{10}$$

其中 UPDATE($x^m, x^{\mathcal{N}_m}$,$\mathcal{I}$)： $\mathbb{R}^{D_x} \times \mathbb{R}^{D_x} \times \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}^{D_z}$ 通常由神经网络实现。

一种简单的交互动力系统形式会获取每个智能体在当前位置的特征，并将预定义半径范围内的智能体通过边连接起来。图神经网络操作通过考虑相邻交通参与者的信息，更新每个智能体的位置和速度信息。

3 图深度状态空间模型

我们的目标是对遵循复杂行为模式的智能体之间的随机动力交互进行建模，例如道路交通交互。我们通过提出图深度状态空间模型 （GDSSM），将深度状态空间模型扩展到交互系统中。该模型在转移模型中使用图神经网络来捕捉智能体之间的交互。本节将定义我们的概率模型，并在下一节中介绍一种用于高效、确定性训练和预测的新型方案。

我们感兴趣的是使用深度状态空间模型，通过耦合的潜在空间对 M 个相互作用的智能体的动力学进行建模。换句话说，我们不为每个智能体使用 D_x维的潜在空间，而是假设智能体共享一个大小为 MD_x的潜在空间。由于（i）不同场景中的智能体数量可能有所不同，（ii）转移模型应与智能体的顺序无关，并且（iii）高维潜在空间的参数化颇具挑战，因此我们在转移模型中使用图神经网络。

更正式地说，我们将智能体 m 在时间步 t 的状态记为 $x_t^m \in \mathbb{R}^{D_x}$，并将所有状态变量的集合记为 $x_t = \{x_t^m\}_{m=1}^M$。x_t 的动力学遵循公式 (2)，其中，均值更新 $f(x_t, \mathcal{I})$： $\mathbb{R}^{MD_x} \times \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}^{MD_x}$ 和方差更新 $L(x_t, \mathcal{I})$： $\mathbb{R}^{MD_x} \times \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}^{MD_x}$ 均借助图神经网络实现。

$$f(x_t, \mathcal{I}) = \begin{bmatrix} \tilde{f}(x_t^1, x_t^{\mathcal{N}_1}, \mathcal{I}) \\ \vdots \\ \tilde{f}(x_t^M, x_t^{\mathcal{N}_M}, \mathcal{I}) \end{bmatrix}, \qquad L(x_t, \mathcal{I}) = \begin{bmatrix} \tilde{L}(x_t^1, x_t^{\mathcal{N}_1}, \mathcal{I}) \\ \vdots \\ \tilde{L}(x_t^M, x_t^{\mathcal{N}_M}, \mathcal{I}) \end{bmatrix}. \tag{11}$$

智能体特定的均值更新记为 $\tilde{f}(x_t^m, x_t^{\mathcal{N}_m}, \mathcal{I})$： $\mathbb{R}^{D_x} \times \mathbb{R}^{D_x} \times \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}^{D_x}$，方差记为 $\tilde{L}(x_t^m, x_t^{\mathcal{N}_m}, \mathcal{I})$： $\mathbb{R}^{D_x} \times \mathbb{R}^{D_x} \times \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}_+^{D_x}$。两者均在一般图神经网络中实现更新函数 [式（10）],，而 $x_t^{\mathcal{N}_m}$包含来自所有邻近智能体状态的聚合信息[式（9）]。我们模型的确定性变体，例如设定 $L(x_t, \mathcal{I}) = 0$，已成功用于为复杂物理系统学习替代模型（Sanchez-Gonzalez 等人，2020）。我们进一步假设，只需在智能体之间耦合潜在动力学，并使发射模型 [式（3）]在各智能体之间保持独立即可。请注意，我们的转移模型由单个聚合和更新构成。

当数据采样足够密集，使得智能体之间的信息流动速度快于状态动力学的演化速度时，一步便已足够。不过，也可以通过在均值和方差更新函数中堆叠多个图神经网络层，轻松地将模型扩展为每个时间点执行多步消息传递。

此外，我们注意到，尽管转移噪声在各个智能体之间是因子化的，但智能体之间仍会产生相关性，因为均值和方差不仅取决于第 m 个智能体的状态，还取决于所有邻近智能体的状态。经过 a 次聚合步骤后，我们的图神经网络模型能够捕捉智能体 m 与智能体 m' 之间的相关性，前提是它们之间存在一条长度不超过 a 步的路径。相比之下，仅考虑第 m 个智能体状态的方法不会产生任何相关性，而考虑所有其他智能体状态的方法则会在一个时间步后产生一个完全相关的协方差矩阵。

为了完成对模型的概率描述，我们进一步指定初始潜在状态 $x_0 \in \mathbb{R}^{MD_x}$ 的分布，采用 高斯混合模型 （高斯混合模型）。

$$v \sim \mathrm{Cat}([\pi_1(\mathcal{I}), \ldots, \pi_V(\mathcal{I})]), \tag{12}$$
$$x_0 \sim \mathcal{N}(x_0 | \mu_{0,v}(\mathcal{I}), \mathrm{diag}(\Sigma_{0,v}(\mathcal{I}))),$$

其中 Cat($[\pi_1(\mathcal{I}) \ldots, \pi_V(\mathcal{I})]$) 是一个包含 V个混合分量的分类分布。每个分量由其权重 $\pi_v(\mathcal{I}) : \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}_+$、均值 $\mu_{0,v}(\mathcal{I}) : \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}^{MD_x}$ 和对角协方差$\Sigma_{0,v}(\mathcal{I}) : \mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}_+^{MD_x}$ 指定。权重 $\pi_{1:V}$构成一个标准的 V-单纯形。我们使用图神经网络（GNN），将其称为嵌入函数 h($\mathcal{I}$) : $\mathbb{R}^{D_\mathcal{I}} \to \mathbb{R}^{V+2VMD_x}$，以便对初始状态分布进行建模

$$\mathrm{h}(\mathcal{I}) = \begin{bmatrix} \pi_{1:V}(\mathcal{I}) \\ \mu_{0,1:V}(\mathcal{I}) \\ \Sigma_{0,1:V}(\mathcal{I}) \end{bmatrix}. \tag{13}$$

我们假设上下文变量 $\mathcal{I}$ 包含以边集合形式表示的关系信息，以及每个智能体以观测轨迹形式表示的历史信息。从某种意义上说，嵌入函数在此充当滤波器，根据过去的观测值学习初始潜在状态的分布。

在自动驾驶中，初始潜在状态可以与驾驶员的意图相联系（Tang & Salakhut- dinov，2019）。上下文信息 $\mathcal{I}$ 往往不足以排除对未来的不同假设，例如，后方的汽车是否打算在接下来的五秒内超车。采用混合模型可以让我们以合理的方式将不同假设纳入模型，最终得到高度多模态的预测分布。

Yang 等人（2020）此前已将状态空间模型与图神经网络相结合，用于多智能体轨迹预测。与我们的工作不同，作者（i）采用了不同的模型定义，在转移模型中使用循环神经网络和非高斯密度；（ii）在推理过程中执行蒙特卡洛采样，这可能导致收敛速度较慢。在下一章中，我们将展示，通过执行确定性矩匹配规则，我们的模型定义能够实现更高效的训练。我们将在实验中将其与蒙特卡洛替代方法进行比较。

4 GDSSM 的确定性近似

在本章中，我们介绍一种新颖的 GDSSM 推断方案，可实现高效且确定性的训练与预测。在第 4.1 节中，我们首先简要概述现有的推断技术，并将其与我们的方案进行比较。我们的方案旨在直接最大化未来轨迹的预测对数似然。预测对数似然无法以闭式形式计算。我们在第 4.2 节提出一种高效且确定性的近似方法，该方法依赖于双维矩匹配。我们的矩匹配规则需要计算图神经网络层的输出矩和期望雅可比矩阵。我们在第 4.3 节介绍常用层的输出矩。由于我们的算法在每个时间步都用涵盖所有智能体的高斯混合分布来近似输出分布，因此，当交通参与者数量较多时，每个混合分量对应的协方差矩阵可能会变得难以计算。我们在第 4.4 节提出协方差的稀疏近似，以解决这一难题。

4.1 参数推断

针对状态空间模型的经典推断方法旨在直接最大化数据的对数似然。

$$\log p(y_1, \ldots, y_T|\mathcal{I}) = \log \int p(x_0|\mathcal{I}) \prod_{t=1}^{T} p(x_t|x_{t-1}, \mathcal{I}) p(y_t|x_t) dx_0 \ldots dx_T, \tag{14}$$

其中， $p(x_t|x_{t-1},\mathcal{I})$ 在公式 (2) 中定义， $p(y_t|x_t)$ 在公式 (3) 中定义。只有当发射模型和转移模型均为线性高斯模型时，才能以闭式形式计算该量。在我们的情形中，转移模型由图神经网络参数化，发射模型则由标准神经网络参数化。这两个函数都具有高度非线性，因此无法解析求解公式 (14)。

因此，现有的大多数方法采用粒子滤波器（Schön 等人，2015年）、变分推断（Krishnan 等人，2017；Bayer 等人，2021）或两者结合的方法（Naesseth 等人，2018；Maddison 等人，2017；Le 等人，2018），以近似对数似然。这些方法的共同点是都需要学习一个提议分布 $q(x_0, \ldots, x_T|y_1, \ldots, y_T)$。在粒子滤波中，提议分布通过重要性函数 $q(x_t|x_0, \ldots, x_{t-1}, y_1, \ldots, y_t)$ 递归定义；令 $q(x_t|x_0, \ldots, x_{t-1}, y_1, \ldots, y_t) = p(x_t|x_{t-1}, y_t)$ 时，该分布在方差意义上是最优的（例如，Doucet 等人（2000））。变分推断旨在通过最小化真实后验与近似后验之间的 Kullback-Leibler 散度（KL 散度），在选定的变分族中找到对真实后验的最佳近似（Blei 等人，2017）；而变分序贯蒙特卡洛则旨在扩展后的采样空间上最小化 KL 散度（Le 等人，2018）。

然而，提议分布仅在推断过程中用作辅助工具。对于许多预测任务（Djuric 等人，2020；Jain 等人，2019；Chai 等人，2019），我们关注的量是预测对数似然（PLL）

$$\begin{aligned} \mathrm{PLL}(y_1, \ldots, y_T|\mathcal{I}) &= \sum_{t=1}^{T} \log p(y_t|\mathcal{I}) \\ &= \sum_{t=1}^{T} \log \int p(x_0|\mathcal{I}) p(x_t|x_0, \mathcal{I}) p(y_t|x_t) dx_0 dx_t, \end{aligned} \tag{15}$$

其中，转移核 $p(x_t|x_0, \mathcal{I})$ 定义见公式 (4)。

与标准训练目标不同，预测对数似然会在不接收任何来自观测值的反馈的情况下，将潜在状态向前传播至未来时刻，从而模拟测试时的行为。由于我们希望在训练时和测试时使用相同的目标，因此选择在训练过程中直接最大化预测对数似然。关于在训练过程中仅使用单步超前预测不足以在测试时获得可靠的多步超前预测这一观察，Bengio 等人（2015年）也曾提出过。该研究的作者提出了一种计划采样策略，以便在训练过程中逐步从单步超前预测切换到多步超前预测。多步超前训练也已成功应用于时空预测（Pal 等人，2021年）和基于模型的规划（Hafner 等人，2019年）。

PLL 旨在优化平均边缘对数似然 $\log p(y_t|\mathcal{I})$，而标准目标函数旨在优化联合似然 $\log p(y_1, \ldots, y_T|\mathcal{I})$。我们认为，两种目标函数的选择取决于具体应用：PLL 最适用于仅需评估边缘分布即可解决的任务。属于这一类别的一个重要且广泛的应用领域是自动驾驶，在该领域中，给定时间范围内相邻交通参与者的边缘分布通常足以用于控制车辆（例如，Herman 等人（2022年））。因此，自动驾驶领域的许多论文都将其方法在固定时间间隔上的性能作为评估指标进行报告，这与我们的 PLL 目标函数相一致（例如，Djuric 等人（2020年）；Jain 等人（2019年）；Chai 等人（2019年））。相比之下，对于需要采样生成逼真轨迹的任务，优化联合对数似然则更为合适，例如句子生成任务（Vaswani 等人，2017年）。

4.2 使用双维矩匹配近似预测对数似然

我们关注预测对数似然 PLL(y_1、.....、 y_T|$\mathcal{I}$)，它描述了截至时间步 T 的所有交通参与者的预测对数似然。为了计算它，我们需要求解公式（15）中给出的嵌套积分。BMM 算法允许我们在初始状态 x_0 具有高斯分布的情况下近似转移核$p(x_t|\mathcal{I}) = \int p(x_t|x_0、\mathcal{I})p(x_0|\mathcal{I})dx_0$ 。然而，在我们的模型表述中，初始潜在状态 x_0 遵循高斯混合模型，以实现多模态性。为此，我们将潜在边缘分布 $p(x_t|\mathcal{I})$ 近似为

$$p(x_t|\mathcal{I}) \approx \sum_{v=1}^{V} \pi_v(\mathcal{I}) p(x_{t,v}|\mathcal{I}), \tag{16}$$

其中，每个混合分量 $p(x_{t,\ v}|\mathcal{I})$ 都使用 BMM 算法近似为 $p(x_{t,\ v}|\mathcal{I}) \approx \mathcal{N}(\mu_{t,\ v}(\mathcal{I}) \Sigma_{t,\ v}(\mathcal{I}))$。假设初始状态服从高斯混合模型，我们便能以较高的计算效率高效获得多模态预测：通过显式地对混合分量进行边缘化来获得多模态分布，并通过应用 BMM 算法高效计算每个分量。最后，我们通过再次进行矩匹配，将边缘分布 $p(y_t|\mathcal{I})$ 近似为高斯混合模型

$$\begin{aligned} p(y_t|\mathcal{I}) &= \int p(y_t|g(x_t), \mathrm{diag}(\Gamma(x_t))) p(x_t|\mathcal{I}) dx_t \\ &\approx \sum_{v=1}^{V} \pi_v(\mathcal{I}) \mathcal{N}(a_{t,v}(\mathcal{I}), B_{t,v}(\mathcal{I})), \end{aligned} \tag{17}$$

其中，$a_{t,\ v}(\mathcal{I})$ 和 $B_{t,\ v}(\mathcal{I})$ 分别是第 v 个混合分量在第 t 个时间步的均值和协方差。这两个矩可表示为

$$a_{t,v}(\mathcal{I}) = \mathbb{E}[g(x_{t,v})], \qquad B_{t,v}(\mathcal{I}) = \mathrm{Cov}[g(x_{t,v})] + \mathrm{diag}\left(\mathbb{E}[\Gamma(x_t)]\right), \tag{18}$$

这是非意识统计学家定理的直接结果。我们在算法1中给出了使用我们的方法计算 $p(y_t|\mathcal{I})$ 的伪代码。算法 2 进一步展示了如何使用 PLL［式（15）］作为训练目标来优化模型参数。

在我们的方法中，我们通过对每个高斯混合分量分别应用矩匹配来近似 $p(x_{t,\ v}|\mathcal{I})$ 和 $p(y_t|\mathcal{I})$。我们指出，如定理 1 所述，对于局部线性的转移函数和发射函数，这种近似是精确的。

定理 1. 边缘分布 $p(y_t|\mathcal{I})$ 可通过解析计算得出

$$p(y_t|\mathcal{I}) = \sum_{v=1}^{V} \pi_v(\mathcal{I}) \mathcal{N}(a_{t,v}(\mathcal{I}), B_{t,v}(\mathcal{I})),$$

对于具有以下生成模型的 *GDSSM*

$$\begin{aligned} v &\sim Cat([\pi_1(\mathcal{I}), \ldots, \pi_V(\mathcal{I})]), \\ x_0 &\sim \mathcal{N}(\mu_{0,v}(\mathcal{I}), diag(\Sigma_{0,v}(\mathcal{I}))), \\ x_t &\sim \mathcal{N}\left(x_t|x_{t-1} + f(t,v,\mathcal{I})x_{t-1}, diag\left(L(t,v,\mathcal{I})\right)\right), && t = 1, \ldots, T \\ y_t &\sim \mathcal{N}\left(y_t|g(t,v,\mathcal{I})x_t, diag\left(\Gamma(t,v,\mathcal{I})\right)\right), && t = 1, \ldots, T \end{aligned}$$

其中，$f(t, v, I)$ $L(t, v, I)$ $g(t, v, I)$ $\Gamma(t, v, I)$ 是依赖于时间 t、分量 v 和上下文 I 的矩阵，其维度适当。

证明。 由于转移函数和发射函数的输出矩在解析上易于求解，因此证明很直接。详细内容见附录 A。

□

算法1 潜在空间中的双维矩匹配（BMMLS）

输入： T 。预测时域
$\mathcal{I}$ 。上下文变量
$f(x_t, \mathcal{I};\ \theta_f)$ ▷均值更新
$L(x_t,\ \mathcal{I};\theta_L)$ ▷协方差更新
$g(x_t;\ \theta_g)$ ▷ 均值发射
$\Gamma(x_t;\ \theta_\Gamma)$ ▷ 协方差发射
$h(\mathcal{I};\ \theta_h)$ 。 嵌入函数
输出： 近似边缘分布 $p(y_T|\mathcal{I})$

$\mu_0, \mathbf{1}{:}_V$(我) $\Sigma_0{}^{1:\ V}(\mathcal{I})\ \ \pi_v(\mathcal{I}) = h(\mathcal{I};\ \theta_h)$ 。初始步骤处的高斯混合模型，公式 13
对于 混合分量 $v \in \{1, \cdots\ \ V\}$ **执行**
对于 时间步 $t \in \{0, \cdots\ \ T-1\}$ **执行** ▷ 水平矩匹配
$\mu_{t+1,v} \leftarrow \mu_t(\mathcal{I}) + \mathbb{E}[f(x_t, \mathcal{I};\ \theta_f)]$ 。式（6）
$\Sigma_{t+1,\ v} \leftarrow \Sigma_t(\mathcal{I}) + \mathrm{Cov}[f(x_t,\ \mathcal{I};\ \theta_f)] + \mathrm{Cov}[x_t, f(x_t,\ \mathcal{I};\ \theta_f)\ ] + \mathrm{Cov}[x_t, f(x_t,\ \mathcal{I};\ \theta_f)]^T + \mathrm{diag}\,(\mathbb{E}[L(x_t,\ \mathcal{I};\ \theta_L)])$ 。公式 6
结束循环
$a_{T,\ v}(\mathcal{I}) \leftarrow \mathbb{E}[g(x_{T,\ v}; \theta_g)]$ 。公式 18
$B_{T,v}(\mathcal{I}) \leftarrow$ 协方差$[g(x_{T,v};\ \theta_g)]$ + 对角矩阵 $(\mathbb{E}[\Gamma(x_T;\ \theta_\Gamma)])$ ▷公式 18
结束循环
返回 $\sum_{v=1}^{V} \pi_v(\mathcal{I})\mathcal{N}(a_{T,v}(\mathcal{I})\ B_{T,v}(\mathcal{I}))$

算法 2 GDSSM 的确定性训练

输入： $\mathcal{D}$ ▷ 数据集
$f(x_t, \mathcal{I};\ \theta_f)$ ▷均值更新
$L(x_t,\ \mathcal{I};\theta_L)$ ▷协方差更新
$g(x_t;\ \theta_g)$ ▷ 均值发射
$\Gamma(x_t;\ \theta_\Gamma)$ ▷ 协方差发射
$h(\mathcal{I};\ \theta_h)$ ▷ 嵌入函数
输出： 优化后的权重 $\theta = \{\theta_f, \theta_L, \theta_g, \theta_\Gamma, \theta_h\}$

当尚未收敛**时**
I $y_{1:T} \sim D$ 。 从数据集中采样 $\mathcal{I}$ and $y_{1:T}$
对于 $y_t \in y_{1:T}$ **，执行**

。通过算法1近似边缘分布 $p(y_t|\mathcal{I})$
结束循环
$\mathrm{PLL}(y_1、\ \ldots\ldots、\ y_T|I) = \sum_{t=1}^{T} \log\ p(y_t|\mathcal{I})$ ▷计算预测对数似然 15
$\theta \leftarrow \theta + \frac{\partial PLL(y_{1:T}|\mathcal{I})}{\partial \theta}$ ▷更新权重 θ
结束循环
返回 θ

对于一般的非线性转移函数和发射函数，该方案的效果取决于两个因素：（a）转移函数和发射函数能否很好地通过局部线性方式近似；（b）在时间范围 t 内，协方差矩阵 $\Sigma_{t,\ v}$ 是否足够小。我们在实验中观察到，该近似效果良好，并在图 2 中通过一个小型玩具数据集进一步展示了其表现。最后，我们指出，高斯混合模型也已成功用于滤波问题（Alspach & Sorenson，1972）；与我们的工作类似，它们在预测步骤中对每个分量分别进行矩匹配。

4.3 图神经网络层的输出矩

为了使用 BMM框架，我们需要能够计算图神经网络层的前两个输出矩以及期望雅可比矩阵。下面，我们推导常见图神经网络层的输出矩和期望雅可比矩阵的解析表达式，包括：(i) 逐节点仿射变换，以及 (ii) 均值聚合。Wu 等人（2019）给出了 ReLU激活函数的输出矩。为完整起见，我们也在附录 B 中给出了 ReLU激活函数的输出矩。

令 $x^{l,}{}_t{}^m \in \mathbb{R}^{D_{x,^l}}$ 表示时间步 t 时节点 m 在第 l 层的节点特征，$x_t^l = \{x^{l,}{}_t{}^m\}_{m=1}^M$ 表示所有满足 $x_t^l \in \mathbb{R}^{MD_{x,^l}}$的节点特征集合。为简洁起见，这里省略了混合分量 v的索引。我们将时间步 t 时图中有 M 个节点的第 l 层的均值和协方差记为

$$\mathbb{E}[x_t^l] = \begin{bmatrix} \mathbb{E}[x_t^{l,1}] \\ \vdots \\ \mathbb{E}[x_t^{l,M}] \end{bmatrix}, \qquad \mathrm{Cov}[x_t^l] = \begin{bmatrix} \mathrm{Cov}[x_t^{l,1}, x_t^{l,1}] & \ldots & \mathrm{Cov}[x_t^{l,1}, x_t^{l,M}] \\ \vdots & \ddots & \vdots \\ \mathrm{Cov}[x_t^{l,M}, x_t^{l,1}] & \ldots & \mathrm{Cov}[x_t^{l,M}, x_t^{l,M}]. \end{bmatrix}, \tag{19}$$

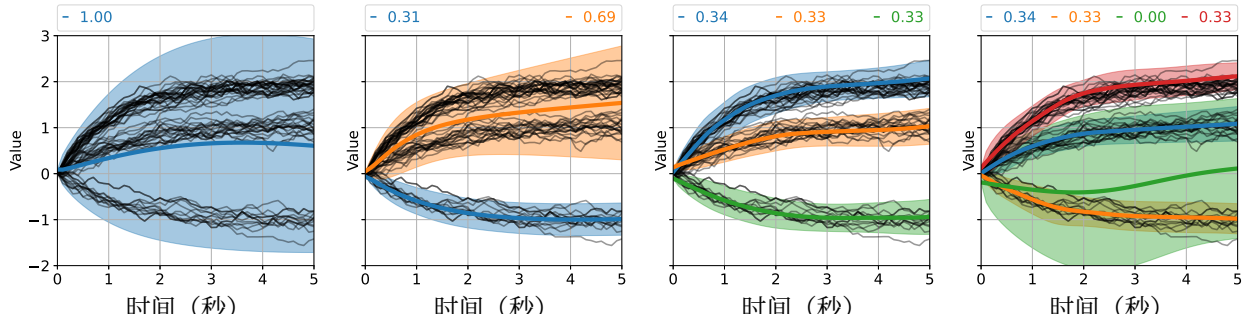


图 2：在一维多模态玩具问题上训练后，我们模型的预测结果。黑色实线表示具有三个等概率不同模式的真实动力学。对于我们的模型，我们报告每个预测模态的均值、95%置信区间和混合权重。我们将潜在空间的维度设为 $D_x = 3$，并将模型中的模态数 V 从左到右设为 1 到 4。对于 $V < 3$，模态数设得过少，因此我们观察到模态覆盖行为。对于$V \geq 3$，我们的模型能够恢复真实动力学。如果组件数过多（V=4），冗余组件的混合权重会被设为零。

其中 $\mathbb{E}[x_t^{l,m}] \in \mathbb{R}^{D_{x,l}}$ 和 $\text{Cov}[x_t^{l,m}, x_t^{l,m'}] \in \mathbb{R}^{D_{x,l} \times D_{x,l}}$。典型的图神经网络架构（见第 2.3 节）由交替进行的聚合步骤和更新步骤组成：聚合步骤收集所有邻居的信息，更新步骤则更新节点的特征。对于聚合步骤，我们在第 4.3.3 节推导了常用均值聚合操作的输出矩。对于更新步骤，我们假设神经网络由一系列仿射变换和非线性函数构成。非线性激活的输出矩是在各个智能体上独立应用的。因此，在图神经网络的设置中，其规则保持不变，我们可以采用 Wu 等人（2019）的推导。因此，仍需推导图神经网络中使用的仿射变换的输出矩，我们将在第 4.3.2 节处理这一问题。均值聚合操作以及更新步骤中使用的逐节点仿射操作，都是标准仿射层的特例，我们将在第 4.3.1 节回顾该仿射层。

4.3.1 标准仿射变换

假设我们对层 l 中状态为 $x^{l}{}_{,t}{}^{m}$ 的节点 m 应用仿射变换

$$x_t^{l+1,m} = W^l x_t^{l,m} + b^l, \tag{20}$$

权重矩阵为 W^l，偏置为 b^l。输出矩在解析上可得：

$$\mathbb{E}[x_t^{l+1,m}] = W^l \mathbb{E}[x_t^{l,m}] + b^l, \qquad \text{Cov}[x_t^{l+1,m}] = W^l \text{Cov}[x_t^{l,m}](W^l)^T.$$

仿射变换的期望雅可比矩阵为 $J_t^l = W^l$。

4.3.2 逐节点仿射变换

逐节点仿射变换会对层 m 中状态为 $x^{l}{}_{,t}{}^{m}$ 的每个节点 l 同时应用相同的变换，其权重矩阵为 W^l，偏置为 b^l。逐节点仿射变换可以解释为作用于所有节点集合 x_t^l 上的标准仿射变换，形式如下

$$\begin{bmatrix} W^l x_t^{l,1} + b^l \\ W^l x_t^{l,2} + b^l \\ \vdots \\ W^l x_t^{l,M} + b^l \end{bmatrix} = \begin{bmatrix} W^l & 0 & \dots & 0 \\ 0 & W^l & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & W^l \end{bmatrix} \begin{bmatrix} x_t^{l,1} \\ x_t^{l,2} \\ \vdots \\ x_t^{l,M} \end{bmatrix} + \begin{bmatrix} b^l \\ b^l \\ \vdots \\ b^l \end{bmatrix} = \underbrace{(I_M \otimes W^l)}_{\tilde{W}^l} x_t^l + \underbrace{(\mathbf{1}_M \otimes b^l)}_{\tilde{b}^l}, \tag{21}$$

其中， I_M 是形状为 $M \times M$的单位矩阵， $\mathbf{1}_M$ 是形状为 $M \times 1$的全 1 向量， $\otimes$ 是克罗内克积。逐节点仿射变换的输出矩为

$$\mathbb{E}[x_t^{l+1}] = \hat{W}^l \mathbb{E}[x_t^l] + \hat{b}^l, \qquad \mathrm{Cov}[x_t^{l+1}] = \hat{W}^l \mathrm{Cov}[x_t^l](\hat{W}^l)^T.$$

与标准仿射变换类似，逐节点仿射变换的期望雅可比矩阵可解析地表示为 $J_t^l = \hat{W}^l$。

4.3.3 均值聚合

一种常用的聚合操作是均值聚合器，它计算第 l 层在时间步 t 向节点 m 发送的消息 $x_t^{l,\mathcal{N}_m}$，具体如下

$$x_t^{l,\mathcal{N}_m} = \frac{1}{|\mathcal{N}_m|} \sum_{m' \in \mathcal{N}_m} x_t^{l,m'}. \tag{22}$$

令 $x_t^{l,\mathcal{N}}$ 为所有消息的集合，即 $x_t^{l,\mathcal{N}} = \{x_t^{l,\mathcal{N}_m}\}_{m=1}^M$。均值聚合可以等价地表示为一个线性变换

$$x_t^{l,\mathcal{N}} = \underbrace{(A \otimes I_{D_{x,l}})}_{\hat{A}} x_t^l. \tag{23}$$

上文中， $A \in \mathbb{R}^{M \times M}$ 表示行归一化邻接矩阵，它以矩阵形式汇总边信息$\mathcal{E}$ ，而 $I_{D_{x,l}}$ 表示维度为 $D_{x,l} \times D_{x,l}$ 的单位矩阵。克罗内克积相应地将邻接矩阵扩展至 $D_{x,l}$ 维节点特征。因此，均值聚合对应于一个权重矩阵由 $M \times M$ 个块构成的线性变换，其中每个块都是形状为 $D_x \times D_x$ 的对角矩阵。其矩可解析地表示为

$$\mathbb{E}[x_t^{l,\mathcal{N}}] = \hat{A}\mathbb{E}[x_t^l], \qquad \mathrm{Cov}[x_t^{l,\mathcal{N}}] = \hat{A}\mathrm{Cov}[x_t^l]\hat{A}^T. \tag{24}$$

期望雅可比矩阵可表示为 $J_t^l = \hat{A}$。

4.4 稀疏协方差近似

对于智能体数量 M 较多或状态维度 x^l较高的情形， t^m ，应用 BMM 算法的计算开销可能会很大。下面我们回顾 BMM 算法的计算复杂度。我们假设图神经网络模型由均值聚合步骤以及多个逐节点仿射变换和非线性函数组成，这与我们稍后在实验中采用的架构相同。均值聚合针对每个 D_x 潜在状态独立进行，我们用 H表示最大的隐藏层宽度。

计算非线性函数的开销很低，因为这些操作是逐元素进行的，在此分析中可以忽略它们对运行时间的影响。另两种操作（见第 4.3.2 节和第 4.3.3 节）可以表示为仿射操作，其权重矩阵具有高度结构化的形式；均值聚合步骤对应于由 $M \times M$ 个对角块组成的权重矩阵，而逐节点仿射变换对应于一个块对角权重矩阵，其包含 M 个形状为 $H \times H$ 的块。完整协方差矩阵在神经网络中的传播（前向计算开销）的计算复杂度为 $\mathcal{O}(M^2H^3 + M^2H^2D_x + M^2HD_x^2 + M^3D_x^2)$，其中第一项是隐藏层中 $H \times H$ 维逐节点仿射变换的开销，第二项和第三项是聚合操作之后 $H \times D$ 维逐节点仿射变换的开销，第四项则是聚合操作的开销。

期望雅可比矩阵的计算成本为 $\mathcal{O}(MH^3 + MH^2D_x + M^2D_x^2)$，其推导如下。设聚合操作的期望雅可比矩阵为 $\mathbb{E}[J_t^1]$ ，后续神经网络层的期望雅可比矩阵的乘积为 $\mathbb{E}[J_t^{net}] = \prod_{l=2}^{L} \mathbb{E}[J_t^l]$。神经网络层的期望雅可比矩阵 $\mathbb{E}[J_t^{net}]$ 是块对角矩阵，包含 M 个形状为 $D_x \times D_x$ 的块，其计算需要$\mathcal{O}(MH^3 + MH^2D_x)$ 时间。将 $\mathbb{E}[J_t^1]$ 与 $\mathbb{E}[J_t^{net}]$ 相乘会得到一个完全填充的矩阵，由于其因子的结构，每个元素都可以通过一次点积计算得出，这部分计算使运行时间增加 $\mathcal{O}(M^2D_x^2)$。

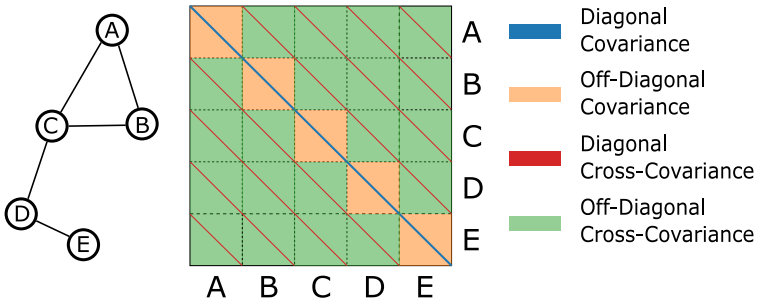


图 3：图结构数据的协方差矩阵分组。考虑左侧面板所示的包含$M = 5$ 个智能体的示例图。每个智能体的维度为 D_x。中间面板展示了所有智能体之间的协方差矩阵，该矩阵由 5×5 个块组成，每个块的维度为 $D_x \times D_x$。

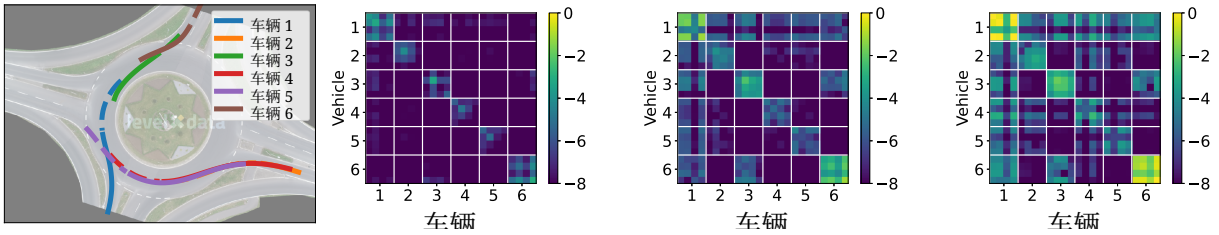
因此，使用完整协方差矩阵时，总成本主要由前向传播成本 $\mathcal{O}(M^2H^3+M^2H^2D_x+M^2HD_x^2+M^3D_x^2)$ 决定。[1] 由于前向传播的计算成本随智能体数量呈立方增长，很快便会变得难以处理，因此接下来我们提出针对协方差矩阵的不同稀疏近似。无论选择哪种近似，期望雅可比矩阵的计算成本都保持不变，因为它不依赖于协方差矩阵。

- **完整**：对完整协方差矩阵建模。前向计算开销：$\mathcal{O}(M^2H^3 + M^2H^2D_x + M^2HD_x^2 + M^3D_x^2)$。总开销：$\mathcal{O}(M^2H^3 + M^2H^2D_x + M^2HD_x^2 + M^3D_x^2)$。
- **主对角线**：保留协方差块中的对角线元素，这对应于图 3 中的蓝线。前向计算开销：$\mathcal{O}(MH^2 + MHD_x + M^2D_x)$。总开销：$\mathcal{O}(MH^3 + MH^2D_x + M^2D_x^2)$。
- **主块**：保留协方差矩阵中的块对角块，对应于图 3 中的橙色块和蓝色线条。前向计算成本：$\mathcal{O}(MH^3 + MH^2D_x + MHD_x^2 + M^2D_x^2)$。总计算成本：$\mathcal{O}(MH^3 + MH^2D_x + MHD_x^2 + M^2D_x^2)$。
- **所有对角线**：将协方差矩阵划分为形状为 $M \times M$ 的分块，并保留每个分块中的对角线元素，这对应于图 3 中的蓝色和红色线条。前向计算成本：$\mathcal{O}(M^2H^2 + M^2HD_x + M^3D_x)$。总成本：$\mathcal{O}(M^2H^2 + M^2HD_x + M^3D_x + MH^3 + MH^2D_x + M^2D_x^2)$。

请注意，将非对角块设为零（如在主块和主对角线中所做的那样）对应于智能体之间的独立性假设，并使运行时间从 $O(M^3)$ 降低到 $O(M^2)$。

最后需要注意的是，只有经过第一个时间步后，协方差矩阵才与图具有相同的结构。在时间步 t，未通过边相连的智能体仍可能具有非零交叉协方差，前提是它们之间存在一条长度不超过 t 步的路径。在我们的应用中，由于智能体数量相比时间范围较少，经过几个时间步后，协方差便不再稀疏。图 4 展示了一个示例场景在三个不同时间步下的协方差矩阵。对于 1 秒的短预测时域，协方差矩阵近似呈对角形。随着预测时域增加，协方差矩阵变得更加复杂，对角线元素不再占主导地位。我们指出，如有需要，还可以采用图神经网络模块中包含多个聚合步骤的架构，进一步扩大智能体之间的信息传播范围。

[1] 作为参考，采用蒙特卡洛方法的计算复杂度为$\mathcal{O}(SMH^2 + SMHD_x + SM^2D_x)$，其中$S$ 是蒙特卡洛样本数。



(a) 示例场景。 (b) 1 秒时的协方差。(c) 3 秒时的协方差。(d) 5 秒时的协方差。

图4：我们展示了来自rounD数据集（Krajewski等人，2020）的一个示例场景，以及单峰初始分布情况下潜在空间中不同时间步的协方差矩阵。在左图中，虚线表示观测历史，实线表示真实的未来轨迹。

5 实验

我们在两个具有挑战性的自动驾驶数据集上进行了实验。第一个实验（第 5.1 节）开展消融研究，第二个实验（第 5.2 节）则将我们的模型与最先进的方法进行基准测试。我们在第 5.3 节对所提出的稀疏协方差近似进行了运行时间分析和基准测试。在第 5.4 节，我们通过在全新的、未见过的交通环境中进行基准测试，分析了模型的泛化能力。

我们在附录C中详细介绍训练过程，并在附录H中介绍嵌入函数、转移函数和发射函数的架构。配套代码可在 `https://github.com/boschresearch/Deterministic-Graph-Deep-State-Space-Models`下获取。

作为评估指标，我们使用固定时间间隔（1s、2s、3s、4s、5s）下的均方根误差（RMSE）和预测负对数似然（NLL）。有关评估的更多详情，请参阅附录D。

5.1 rounD

rounD数据集（Krajewski等人，2020）由在德国不同环岛记录的车辆轨迹组成。由于环岛中车辆之间存在许多交互，我们预计预测分布将呈现多模态且高度复杂的特征。我们使用亚琛附近 Neuweiler 环岛的记录进行训练和测试。该数据集包含 22 个时段记录的 13,129 个跟踪目标，记录频率为 25 Hz，总记录时长为 6.6 小时。我们从数据集中移除行人、自行车以及停放车辆，因为它们对环岛中车辆行为模式的影响可以忽略不计。数据集整理完成后，剩余 12,715 个跟踪目标。我们将记录降采样至原来的五分之一，并构建由 8 秒片段组成、重叠率为 50 % 的数据集，共得到 5405 个片段。我们将最初 3 秒用作轨迹历史，它是上下文变量$\mathcal{Z}$的一部分；后续 5 秒则作为预测时域。最初 18 个记录时段（共 4,314 个片段）用于训练和验证。最后 4 个记录时段（共 1,091 个片段）用于测试。

我们通过连接欧几里得距离小于 30 米的车辆来构建连接图。该数据集由无人机录制，我们保留了原始全局坐标系。

5.1.1 基线方法

我们将我们的方法与多个基线模型进行比较。对于每个基线模型，我们移除了模型中的一项关键假设。在适当情况下，我们引用了采用类似思想的论文。我们没有重新实现这些工作，而是进行了消融研究，以替换模型中的特定组件，从而实现公平比较。此外，我们还比较了不同的训练策略，将其作为最大化 PLL 的替代方案，并分析了模型的多模态性。

(i) 模型：

(i.i) GDSSM：第 4 节所提出的模型。

(i.ii) 非循环图神经网络 （例如 Casas 等人（2020）；Herman 等人（2022））：该架构接收上下文变量$\mathcal{I}$，执行一轮消息传递，随后在不使用循环架构的情况下，为接下来的五秒分别输出一个正态分布。

(i.iii) 无潜在噪声 （例如 Sanchez-Gonzalez 等人（2020））：我们从潜在动力学[Eq. (2)]中移除噪声，同时保持发射模型不变。由于发射模型针对每个时间点独立运行，不确定性将无法再沿时间向前传播。

(i.iv) 线性 （例如 Li 等人（2020））：我们从潜在动力学中移除所有非线性。请注意，我们保留了非线性发射模型，以便将动力学映射到一个潜在空间，在该空间中可以对系统进行线性近似。

(i.v) 无相互作用 （例如 Krishnan 等（2017））：我们的模型采用对角邻接矩阵，即从图中移除所有边。该模型忽略了交通参与者之间的相互作用。

（*ii*）模态：

(ii.i) V 模式：我们改变 [方程 (12)] 中 V 高斯混合模型先验的混合分量数量，以测试多模态性的影响。

(ii.ii) ∞ 模态：该替代方法不遵循假定密度方法，即不将潜在边缘分布 $p(x_t|\mathcal{I})$ 近似为具有有限数量模态的高斯混合模型。相反$p(x_t|\mathcal{I})$ 通过模拟大量轨迹来进行近似，其中每条轨迹都可以遵循不同的模式（Brandt 与 Santa-Clara，2002年）。我们将粒子数量设为 100。

（*iii*）目标函数：

(iii.i) PLL/Det.：我们使用预测对数似然 [Eq.(15)] 训练模型，并采用第 4.2 节和第 4.3 节提出的 GDSSM 确定性近似方法。

(iii.ii) PLL/MC：我们使用预测对数似然 [Eq.(15)] 训练模型，并采用假定密度方法，将边缘分布 [Eq.(17)] 近似为高斯混合模型。难以解析求解的积分通过蒙特卡洛 （MC）积分求解。模型的一次前向传播，其计算成本约相当于 12 次蒙特卡洛模拟。为确保公平比较，训练时我们使用 16 个粒子，这比使用我们提出的矩传播算法进行训练的成本更高；测试时则使用 100 个粒子。所有基于 MC 的方法均使用相同数量的粒子。

(iii.iii) ELBO/MC （例如 Krishnan 等人，2017）：该模型通过最大化证据下界 （ELBO）进行训练。附录 E 中对这一损失函数进行了说明。近似后验是一个将潜在状态建模为正态分布的滤波分布。我们通过蒙特卡洛采样沿时间方向推进潜在状态。

(iii.iv) MCO/MC （例如 Maddison 等人，2017）：该模型通过最大化蒙特卡洛目标 （MCO）进行训练，该目标结合了粒子滤波器与变分推断。附录 E 中对这一损失函数进行了说明。提议分布是一个滤波分布，我们通过蒙特卡洛采样沿时间方向推进粒子。

5.1.2 结果

我们在表 1 中给出了所有方法的基准测试结果。首先，我们将确定性训练和测试方案（GDSSM PLL/Det.）与其蒙特卡洛替代方案（GDSSM PLL/MC）进行比较。尽管基于蒙特卡洛的训练比使用 BMM 进行训练成本更高，但其结果仍显著逊于我们的方法。我们的结果表明，我们的确定性方法能带来更有效的

表 1：rounD 结果。我们报告 RMSE 和负对数似然（10 次运行的均值 ± 标准误差）。

		非循环 GNN 1种模式 PLL/Det.	GDSSM ∞ 模态 ELBO/MC	GDSSM ∞ 模态 MCO/MC	GDSSM 无潜在噪声 1种模式 PLL/Det.	GDSSM 线性 1种模式 PLL/Det.	GDSSM 无相互作用 1种模式 PLL/Det.	GDSSM 1种模式 PLL/MC	GDSSM 1种模式 PLL/Det.	GDSSM 2种模式 PLL/Det.	GDSSM 3种模式 PLL/Det.	GDSSM 4种模式 PLL/Det.
RMSE	1 s	0.72 ± 0.02	1.53 ± 0.04	0.95 ± 0.04	1.22 ± 0.08	0.88 ± 0.02	0.91 ± 0.03	0.90 ± 0.02	0.79 ± 0.02	0.75 ± 0.02	0.74 ± 0.03	0.73 ± 0.03
	2 s	1.90 ± 0.02	3.42 ± 0.09	2.43 ± 0.09	2.33 ± 0.08	2.12 ± 0.03	2.10 ± 0.04	2.09 ± 0.02	1.87 ± 0.03	1.84 ± 0.03	1.82 ± 0.03	1.80 ± 0.04
	3 s	3.54 ± 0.03	5.99 ± 0.14	4.55 ± 0.16	3.88 ± 0.07	3.88 ± 0.05	3.69 ± 0.06	3.60 ± 0.04	3.36 ± 0.03	3.35 ± 0.03	3.35 ± 0.03	3.33 ± 0.04
	4 s	5.26 ± 0.04	9.24 ± 0.26	7.62 ± 0.27	5.58 ± 0.08	6.13 ± 0.09	5.39 ± 0.13	5.37 ± 0.05	5.08 ± 0.04	5.15 ± 0.09	5.14 ± 0.08	5.06 ± 0.04
	5 s	7.20 ± 0.05	11.59 ± 0.41	11.24 ± 0.41	7.65 ± 0.10	8.83 ± 0.13	7.56 ± 0.23	7.65 ± 0.06	7.24 ± 0.05	7.34 ± 0.12	7.35 ± 0.19	7.29 ± 0.09
NLL	1 s	1.90 ± 0.01	2.62 ± 0.04	2.79 ± 0.06	2.96 ± 0.08	1.95 ± 0.08	1.67 ± 0.07	2.82 ± 0.03	1.48 ± 0.05	1.34 ± 0.08	1.36 ± 0.04	1.21 ± 0.05
	2 s	3.25 ± 0.02	4.45 ± 0.07	4.21 ± 0.03	3.85 ± 0.10	3.93 ± 0.13	3.37 ± 0.08	4.11 ± 0.02	2.91 ± 0.03	2.93 ± 0.07	2.93 ± 0.06	2.79 ± 0.09
	3 s	4.40 ± 0.02	5.64 ± 0.09	5.21 ± 0.03	5.05 ± 0.13	5.11 ± 0.19	4.34 ± 0.08	4.67 ± 0.09	3.87 ± 0.02	4.01 ± 0.07	3.77 ± 0.10	3.83 ± 0.08
	4 s	5.13 ± 0.02	6.59 ± 0.13	6.01 ± 0.03	6.17 ± 0.16	6.01 ± 0.22	4.93 ± 0.09	5.01 ± 0.07	4.46 ± 0.03	4.52 ± 0.11	4.21 ± 0.15	4.34 ± 0.18
	5 s	5.71 ± 0.02	6.98 ± 0.14	6.73 ± 0.04	7.24 ± 0.20	6.80 ± 0.22	5.51 ± 0.10	5.41 ± 0.05	5.05 ± 0.04	4.82 ± 0.24	4.59 ± 0.15	4.27 ± 0.23

尽管我们的确定性矩匹配方案会产生近似误差，但与蒙特卡洛采样相比，这些近似方法的表现仍然更好。我们的这一发现可能有一个解释：由于多智能体空间的潜在空间会随智能体数量线性增长，蒙特卡洛方法会受到高方差的影响。

当训练目标从 PLL 更改为 ELBO/MC 或 MCO/MC 时，我们观察到，在预测时域为 1 秒的情况下，其性能与 PLL/MC 相当。对于更长的预测时域，采用 ELBO/MC 或 MCO/MC 进行训练的模型，其性能相较于采用 PLL 训练的模型下降得更快。我们认为，这种现象可以通过训练目标与测试目标之间的不匹配来解释。当使用 MCO/MC 或 ELBO/MC 进行训练时，模型通过提议分布从观测值中获取反馈；而在测试期间，模型必须在不接收任何观测值反馈的情况下，生成多步超前预测。

接下来，我们通过增加高斯混合模型先验中的分量数量，研究我们的方法能否捕捉多模态性。值得注意的是，只要分量数量足够多，高斯混合模型就能近似任意复杂的分布。在我们的实验中，我们观察到，增加分量数量会显著降低负对数似然，使高斯混合模型先验成为我们方法的关键组成部分，并表明真实的预测分布具有高度多模态性。如果分量数量选得太少，我们的模型会相应调整其不确定性预测。例如，我们在图1中观察到，当车辆接近环岛出口时，橙色智能体的不确定性会增大。较高的预测不确定性可以用两种可能的未来结果来解释：橙色车辆可能驶离环岛，也可能留在环岛内。因此，如果分量数量选得过少，我们的模型会学会进行补偿，从而使我们能够以精度换取计算运行时间。采用定制化实现时，由于计算过程可以轻松并行化，因此可以轻松扩展到更多分量。我们还注意到，均方根误差受模式数量选择的影响较小，且始终处于两个标准误以内。由于均方根误差衡量的是真实值与预测分布期望值之间的误差，这一结果表明，即使模式数量很少，预测分布也足以很好地建模期望值（实际上，一个模式似乎就够了）。我们还在附录F中给出了最小均方根误差值；该值会随着模式数量的增加而降低，表明不同模式对应着一组多样化的合理轨迹。

接下来，我们将模型与一种更简单的替代方案进行比较。在该方案中，假设动力学是线性的，因此可以精确地以闭式形式计算转移模型的矩（Särkkä & Solin, 2019）。相比之下，我们的方法 GDSSM 使用确定性矩匹配结果，以局部线性方式近似非线性动力学。我们发现，我们的方法取得了更低的均方根误差和负对数似然，这很可能归因于所提出模型类别更高的建模灵活性。

我们的消融研究进一步表明，舍弃潜在噪声并移除交通参与者之间的交互会导致更高的均方根误差和负对数似然。与以非循环方式对动力学进行建模相比，我们的模型实现了相近的均方根误差，并且在负对数似然方面优于其竞争者。

5.2 NGSIM

The下一代仿真 (NGSIM) 数据集（Halkias & Colyar，2007）由在美国两条不同的高速公路 US-101 和 I-80 上以 10 Hz 频率记录的车辆轨迹组成。该数据集通常用于交通预测方法的基准测试，使我们能够将自己提出的方法与现有方法进行比较。我们采用 Deo & Trivedi（2018）的实验设置，并使用这两种高速公路场景。按照原始出版物中的设置，我们采用以自车为中心的局部坐标系。我们将这些场景划分为三个各持续 15 分钟的时间段，分别对应轻度、中度和拥堵的交通状况，并将每条轨迹的采样频率降低一半。测试集由从两个位置随机抽取的四分之一轨迹组成。与 rounD 实验一样，我们将每条轨迹划分为长度为 8 秒的片段，其中前 3 秒用作轨迹历史，随后 5 秒用作预测时域。

同样，与 Diehl 等人（2019）；Lenz 等人（2017）；Wheeler & Kochenderfer（2016）类似，我们根据每辆车辆的车道位置引入连接图。每辆车辆最多与其他车辆有六条连接，即同一车道、左侧车道和右侧车道上位于其前方或后方最近的车辆。最外侧车道上的车辆最多有四个邻居，因为其左侧或右侧没有邻居。

5.2.1 基线方法

我们将我们的方法与以下方法进行比较：

(i) 恒定速度（*Mercat* 等人，*2019*）：该方法结合线性状态空间模型和卡尔曼滤波器进行预测。它不考虑智能体之间的交互。

(ii) 卷积社会 (CS)-LSTM（Deo & Trivedi，2018）：通过引入网格并在其上应用卷积层来建模车辆之间的交互。动力学由确定性长短期记忆网络建模，该网络在每个时间步预测正态分布的均值和方差。

(iv) 时空 (ST)-LSTM（Chen 等人，2020）：使用时空图结构对交互进行建模，即一种包含位置和时间相关组件的图。动力学由确定性长短期记忆网络建模，该网络在每个时间步预测正态分布的均值和方差。据我们所知，这是唯一一种同时报告负对数似然的基于图神经网络的方法。

(iv) 多未来预测 (MFP)（Tang & Salakhutdinov，2019）：一种具有确定性转移动力学的循环模型。通过初始状态以及反馈至动力学模型中的含噪观测值引入随机性。交互通过注意力模块进行建模。

5.2.2 结果

我们在表 2 中给出了所提出模型的基准测试结果。与第 5.1 节在 rounD数据集上的实验类似，我们观察到：(i) 确定性训练和测试比基于蒙特卡洛的方法更高效；(ii) 增加模式数量可以提升性能。

接下来，我们仅使用高斯混合模型先验中的一个分量，将我们的方法与所有其他单峰预测方法进行比较。可以看出，我们的方法在负对数似然方面优于其他方法。此外，我们的方法的均方根误差与其他方法相近，仅逊于 MFP。然而，与基准测试中的其他方法不同，MFP 报告的是 5 个蒙特卡洛样本中的最佳均方根误差，因此难以得出公平的结论。

随后，我们增加了模型和 MFP 的模态数量。从负对数似然（NLL）来看，在长期预测（3秒、4秒、5秒）方面，我们的方法表现更优，而 MFP 在短期预测（1秒、2秒）方面表现更好。对驾驶场景中的智能体进行准确的长期预测，对于高级自动驾驶至关重要，因为精准规划驾驶控制的前提是拥有准确的环境模型。例如，高级驾驶辅助系统通常采用 3 至 5 秒的时间范围来发出驾驶员警告并触发紧急制动，而自动驾驶汽车则以 5 秒或更长的时间范围为目标，以确保行驶安全舒适（Philipp & Goehring，2019）。我们在附录 F 中提供了不同模态数量下的最小均方根误差（minRMSE）值，作为预测多样性的衡量指标。

表 2：NGSIM 结果。我们报告 10 次运行的均方根误差和负对数似然的平均值及标准误。对于 MFP，我们报告 5 个蒙特卡洛样本中最佳的均方根误差值。

		恒定速度	CS-LSTM	ST-LSTM	MFP	MFP	GDSSM	GDSSM	GDSSM	GDSSM	GDSSM
		1种模式	1种模式	1种模式	1种模式	4 种模态	1种模式 PLL/MC	1种模式 PLL/Det.	2种模式 PLL/Det.	3 种模态 PLL/Det.	4 种模态 PLL/Det.
RMSE	1 s	0.75	0.61	0.51	0.54	0.54	0.56 ± 0.00	0.53 ± 0.01	0.55 ± 0.00	0.55 ± 0.01	0.57 ± 0.02
	2 s	1.81	1.27	1.21	1.16	1.17	1.27 ± 0.02	1.18 ± 0.01	1.22 ± 0.01	1.23 ± 0.02	1.24 ± 0.04
	3 s	3.16	2.09	2.01	1.90	1.91	2.11 ± 0.03	1.98 ± 0.02	2.02 ± 0.02	2.05 ± 0.03	2.06 ± 0.04
	4 s	4.80	3.10	3.01	2.78	2.75	3.16 ± 0.04	2.99 ± 0.03	3.03 ± 0.04	3.06 ± 0.05	3.05 ± 0.06
	5 s	6.70	4.37	4.31	3.83	3.78	4.47 ± 0.05	4.29 ± 0.04	4.33 ± 0.07	4.33 ± 0.07	4.35 ± 0.08
NLL	1 s	0.80	0.58	0.90	0.73	-0.65	0.64 ± 0.04	0.19 ± 0.02	-0.12 ± 0.02	-0.15 ± 0.03	-0.16 ± 0.03
	2 s	2.30	2.14	2.41	2.33	1.19	2.10 ± 0.10	1.61 ± 0.02	1.37 ± 0.02	1.35 ± 0.02	1.32 ± 0.02
	3 s	3.21	3.03	3.25	3.17	2.28	2.92 ± 0.11	2.42 ± 0.02	2.23 ± 0.02	2.23 ± 0.02	2.19 ± 0.02
	4 s	3.89	3.68	3.61	3.77	3.06	3.54 ± 0.10	3.02 ± 0.02	2.88 ± 0.02	2.88 ± 0.02	2.82 ± 0.02
	5 s	4.44	4.22	4.36	4.26	3.69	4.04 ± 0.10	3.50 ± 0.02	3.42 ± 0.02	3.41 ± 0.02	3.36 ± 0.02

5.3 协方差近似

我们在第 5.3.1 节详细分析了不同协方差近似的运行时间，并在第 5.3.2 节研究了它们对预测性能的影响。

5.3.1 运行时间

我们在图 5 中将不同协方差近似的运行时间以及蒙特卡洛替代方案的运行时间分别作为输入维度和智能体数量的函数进行可视化。我们使用与 rounD 和 NGSIM 数据集实验中相同的 NSDE 架构。对于蒙特卡洛替代方案，我们展示了 16 个粒子的运行时间，因为在第 5.1 节和第 5.2 节的训练中使用了相同数量的粒子。

我们首先确认，传播完整协方差矩阵的计算成本高于任何一种所提出的近似。事实上，对于智能体数量较多且输入维度较高的系统，我们的稀疏协方差近似最多可将运行时间缩短至原来的百分之一。正如我们在第 4.4 节中推导的，使用 BMM 算法并采用完整协方差矩阵或全对角线近似时，计算成本与智能体数量呈三次依赖关系。相比之下，主对角线近似、主块近似以及基于 MC 的预测与智能体数量呈二次依赖关系。当需要在有限的计算预算下对智能体数量较多的系统进行建模时，这两种近似因此成为基于 MC 的预测的一种颇具吸引力的替代方案。

对于输入维度（≈ 8）和智能体数量（≈ 16）适中的系统，我们提出的所有近似方法计算下一时间点的分布所需时间均不超过 5ms，相当于进行 5–10 次蒙特卡洛模拟的开销。

5.3.2 基准测试

接下来，我们研究不同协方差近似对性能的影响，并报告单峰 GDSSM 情况下的结果。结果见表 3。以下是我们的主要发现。

首先，对完整协方差矩阵进行建模，在均方根误差和负对数似然方面都取得了最佳性能。其次，在各种协方差近似中，全对角协方差近似表现最佳。其均方根误差与完整解相当，但当系统具有高度交互性时（见 rounD数据集），其负对数似然略逊一筹。在这种情况下，就负对数似然而言，它优于另外两种稀疏近似。这种表现或许可以用协方差近似中的假设来解释：它是唯一允许智能体之间存在相关性的近似方法，因为其结构仅忽略了潜在特征之间的依赖关系。

第三，完整解与不同近似方法在均方根误差方面的性能差异介于一个至两个标准误之间。因此，仅对主对角线进行建模也可能足以满足

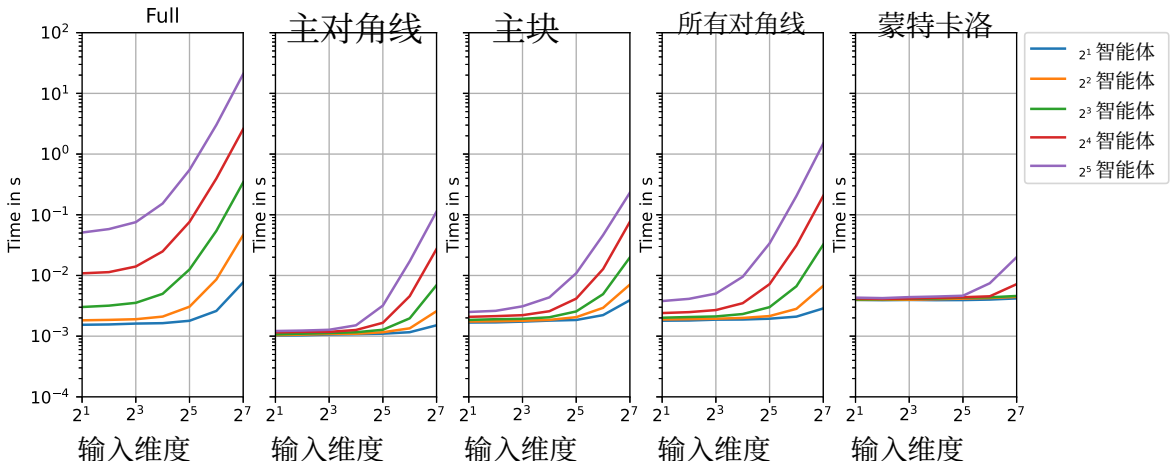


图 5：使用具有三个隐藏层（每层大小为 24）的图神经网络计算潜在动力学输出矩的墙钟时间，比较不同协方差近似方法（从左到右）。对于每种近似方法，我们绘制运行时间随输入维度 D_x 和智能体数量 M变化的情况。图神经网络采用随机初始化，并报告 100 次重复实验的平均运行时间。

计算资源有限且对准确性要求较高的应用场景，前提是能够接受校准效果略有下降。对于这类应用，运行时间可从 $O(M^3)$ 缩短至 $O(M^2)$。

表 3：单峰情形下，不同协方差近似方法在 rounD 和 NGSIM 数据集上的测试性能。我们给出 10 次运行的平均值和标准误。

		rounD				NGSIM			
		Full	主对角线	主块	所有对角线	Full	主对角线	主块	所有对角线
RMSE	1 s	0.79 ± 0.02	0.82 ± 0.02	0.83 ± 0.04	0.78 ± 0.02	0.53 ± 0.01	0.54 ± 0.01	0.55 ± 0.01	0.53 ± 0.01
	2 s	1.87 ± 0.02	1.88 ± 0.02	1.89 ± 0.05	1.87 ± 0.02	1.18 ± 0.01	1.18 ± 0.02	1.19 ± 0.02	1.19 ± 0.02
	3 s	3.36 ± 0.03	3.40 ± 0.02	3.38 ± 0.06	3.34 ± 0.02	1.98 ± 0.02	1.99 ± 0.03	2.05 ± 0.03	1.99 ± 0.02
	4 s	5.08 ± 0.04	5.07 ± 0.04	5.09 ± 0.07	5.05 ± 0.03	2.99 ± 0.03	2.98 ± 0.04	3.07 ± 0.04	2.97 ± 0.03
	5 s	7.24 ± 0.05	7.25 ± 0.06	7.30 ± 0.08	7.25 ± 0.05	4.29 ± 0.04	4.34 ± 0.05	4.54 ± 0.06	4.32 ± 0.04
NLL	1 s	1.48 ± 0.05	1.77 ± 0.06	1.79 ± 0.05	1.69 ± 0.03	0.19 ± 0.02	0.24 ± 0.04	0.22 ± 0.03	0.18 ± 0.03
	2 s	2.91 ± 0.03	3.34 ± 0.04	3.35 ± 0.06	3.25 ± 0.02	1.61 ± 0.02	1.63 ± 0.03	1.64 ± 0.03	1.59 ± 0.03
	3 s	3.87 ± 0.02	4.27 ± 0.03	4.28 ± 0.05	4.18 ± 0.02	2.42 ± 0.02	2.44 ± 0.02	2.46 ± 0.03	2.43 ± 0.02
	4 s	4.46 ± 0.03	5.00 ± 0.02	5.01 ± 0.05	4.92 ± 0.02	3.02 ± 0.02	3.04 ± 0.02	3.06 ± 0.04	3.01 ± 0.02
	5 s	5.05 ± 0.04	5.69 ± 0.02	5.68 ± 0.06	5.64 ± 0.05	3.50 ± 0.02	3.59 ± 0.02	3.62 ± 0.03	3.57 ± 0.02

5.4 分布外测试

我们通过在分布外数据上测试模型，分析其泛化能力，例如测试模型在训练过程中未曾见过的交通环境中的表现。

在这项实验中，我们重复使用 rounD数据集（Krajewski等人，2020），因为它包含三个不同环岛的记录。这里，每个环岛对应一个独立的交通环境。我们从每个环岛各选取一段记录（亚琛的 Kackerstraße、Alsdorf 的 Thiergarten，以及亚琛附近的 Neuweiler），并采用与第 5.1 节相同的数据整理步骤。为了在不同交通环境之间实现泛化，我们对实验设置作如下调整：

- **局部坐标系：**我们将数据转换到局部坐标系中，该坐标系以自车为中心，并按照自车的航向方向定向。

- **将地图信息加入上下文变量$\mathcal{I}$。** 我们提取自车周围的局部地图，其范围是长 74 米、宽 44 米的矩形。随后，我们

对地图进行二值掩码处理，将图像划分为可行驶区域和不可行驶区域。我们的任务是对位于该矩形内的所有车辆进行联合建模。

有关预处理的更多细节请参见附录 G。

5.4.1 结果

接下来，我们通过比较以下策略来分析模型的泛化能力：（1）在两种交通环境中训练，并在第三种不同的交通环境中测试；（2）直接在测试交通环境中训练模型；（3）在所有交通环境中训练。为确保比较公平，每种交通环境的数据均被划分为互不重叠的训练集和测试集。

我们在表 4 中展示了不同交通环境下的结果。表 4 的第一列描述了我们的模型在交通环境 Thiergarten (T) 和 Neuweiler (N) 上进行训练，并在交通环境 Kackertstraße (K) 上进行测试时的均方根误差（RMSE）和负对数似然（NLL）。第二列描述了使用来自同一交通环境 (K) 的场景进行训练和测试时的预测性能。第三列描述了在全部三个交通环境 (KNT) 上进行训练，并在交通环境 K 上进行测试时的预测性能。其余各列以类似的方式，对我们的模型在交通环境 T 和 N 中的泛化能力进行了基准测试。

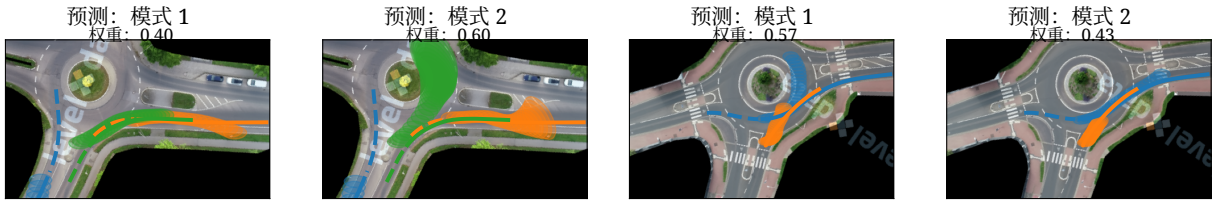
在所有实验中，我们观察到性能依次提升：（1）训练和测试使用不同的训练环境；（2）训练和测试使用相同的环境；（3）训练时使用所有环境。对于位置 K 和 T，（1）与（2）之间的性能差异适中，这表明我们的模型能够在测试时泛化到未见过的交通环境。对于位置 N，性能差异更大。进一步研究这些位置可以解释这一现象：N 是一个交通拥堵的多车道环岛，而 K 和 T 是交通流量适中的单车道环岛。因此，在交通环境 N 上进行分布外测试更具挑战性。图 6 展示了 GDSSM 模型的一些预测示例。

表 4：使用我们的方法 GDSSM（1种模式）在 rounD数据集的不同交通环境下的测试性能。对于每种测试交通环境，我们改变用于训练的交通环境。我们提供了 10 次运行的平均值和标准误。Kackertstraße=K、Thiergarten=T、Neuweiler=N。

	Test	K			T			N		
	训练	TN	K	KTN	KN	T	KTN	KT	N	KTN
RMSE	1 s	2.12 ± 0.09	1.88 ± 0.05	1.55 ± 0.04	1.42 ± 0.04	1.49 ± 0.05	1.21 ± 0.04	2.98± 0.12	2.02 ± 0.07	1.55 ± 0.04
	2 s	4.56 ± 0.13	3.73 ± 0.05	3.34 ± 0.09	2.82 ± 0.05	2.53 ± 0.09	2.27 ± 0.04	5.82± 0.19	3.38 ± 0.06	3.02 ± 0.09
	3 s	7.51 ± 0.24	6.05 ± 0.17	5.62 ± 0.20	4.42 ± 0.12	3.67 ± 0.12	3.35 ± 0.07	8.96± 0.24	4.89 ± 0.13	4.91 ± 0.18
	4 s	10.57 ± 0.34	8.65 ± 0.26	7.99 ± 0.17	6.75 ± 0.18	5.33 ± 0.15	5.00 ± 0.16	12.44± 0.33	6.81 ± 0.14	6.91 ± 0.18
	5 s	13.00 ± 0.31	12.02 ± 0.43	10.73 ± 0.27	9.97 ± 0.40	7.59 ± 0.29	7.41 ± 0.27	14.86± 0.42	8.77 ± 0.19	8.69 ± 0.31
NLL	1 s	4.11 ± 0.07	4.02 ± 0.04	3.13 ± 0.06	3.35 ± 0.06	3.16 ± 0.06	2.88 ± 0.06	4.75± 0.07	3.84 ± 0.04	3.21 ± 0.08
	2 s	5.36 ± 0.10	4.84 ± 0.03	4.66 ± 0.09	4.22 ± 0.04	3.99 ± 0.07	3.65 ± 0.04	5.82± 0.09	4.60 ± 0.05	4.42 ± 0.09
	3 s	6.55 ± 0.17	5.97 ± 0.07	6.42 ± 0.17	5.49 ± 0.10	4.98 ± 0.11	4.50 ± 0.05	7.32± 0.17	5.55 ± 0.04	5.68 ± 0.09
	4 s	7.71 ± 0.20	7.07 ± 0.14	7.57 ± 0.21	7.20 ± 0.09	6.38 ± 0.19	5.89 ± 0.13	8.65± 0.18	6.79 ± 0.11	6.71 ± 0.08
	5 s	8.63 ± 0.18	8.14 ± 0.18	8.08 ± 0.15	8.54 ± 0.12	7.45 ± 0.21	6.84 ± 0.11	9.11± 0.18	7.54 ± 0.22	7.04 ± 0.14

6 结论

在本研究中，我们提出了 GDSSM，通过图神经网络耦合智能体的潜在动力学，以捕捉多个智能体之间的交互。我们为图神经网络层推导了矩匹配规则，从而实现确定性推断；此外，我们还在初始潜在状态上引入了高斯混合模型先验，以支持多模态预测。二者结合后，形成了一种高效且稳定的算法，能够生成复杂的非线性预测分布。我们在两个具有挑战性的自动驾驶数据集上验证了这一新方法出色的经验性能。最后，考虑到实际车辆控制单元的计算限制，我们提出了协方差矩阵的稀疏近似。具体效果取决于



(a) Kackertstraße（K）　　(b) Thiergarten（T）

图6：我们的模型 GDSSM（2种模式）在域外交通环境中的预测，即训练和测试所使用的交通环境彼此不同。给定每个交通参与者的历史轨迹（虚线），我们将预测的95%置信区间可视化。实线表示真实的未来轨迹。

所需的校准程度；我们的近似方法可以显著缩短运行时间，同时不影响准确性。

未来的工作中，我们希望提高所提出模型在新颖且未见过的交通场景下的稳健性。一种方法是通过对图神经网络的权重设置先验，将认知不确定性纳入我们的模型表述。为实现这一点，我们可以将模型与变分推断领域的最新进展相结合，从而找到难以处理的权重后验分布的近似。另一个值得关注的研究方向是对不规则且部分观测的动力系统进行建模。已有工作在潜在空间中使用连续时间编码器网络以及连续时间转移模型（Rubanova 等，2019；Brouwer 等，2019）。沿着这一研究方向，将我们的模型扩展到连续时间网络似乎是对交互系统进行建模的一个有前景的方向。

参考文献

Daniel Alspach 和 Harold Sorenson。使用高斯和近似进行非线性贝叶斯估计。*IEEE* 自动控制汇刊，17(4)，1972 年。

Shaojie Bai、J Zico Kolter 和 Vladlen Koltun。《用于序列建模的通用卷积网络和循环网络的实证评估》。《*arXiv*》，abs/1803.01271，2018。

Mayank Bansal、Alex Krizhevsky 和 Abhijit S. Ogale。《ChauffeurNet：通过模仿最佳表现并合成最差表现来学习驾驶》。《*arXiv*》，abs/1812.03079，2018 年。

Peter W. Battaglia、Jessica B. Hamrick、Victor Bapst、Alvaro Sanchez-Gonzalez、Vinícius Flores Zambaldi、Mateusz Malinowski、Andrea Tacchetti、David Raposo、Adam Santoro、Ryan Faulkner、Çaglar Gülçehre、H. Francis Song、Andrew J. Ballard、Justin Gilmer、George E. Dahl、Ashish Vaswani、Kelsey R. Allen、Charles Nash、Victoria Langston、Chris Dyer、Nicolas Heess、Daan Wierstra、Pushmeet Kohli、Matthew Botvinick、Oriol Vinyals、Yujia Li 和 Razvan Pascanu。《关系归纳偏置、深度学习与图网络》。*arXiv*，abs/1806.01261，2018。

Justin Bayer、Maximilian Soelch、Atanas Mirchev、Baris Kayalibay 和 Patrick van der Smagt。在 *ICLR*上发表：《在序列潜变量模型中进行摊销推断时，切勿忽视差距》。2021年。

Samy Bengio、Oriol Vinyals、Navdeep Jaitly 和 Noam Shazeer。《用于循环神经网络序列预测的计划采样》。发表于*NeurIPS*，2015年。

David M. Blei、Alp Kucukelbir 和 Jon D. McAuliffe。《变分推断：统计学家综述》。美国统计学会会刊，112(518)，2017年。

Michael W. Brandt 和 Pedro Santa-Clara。《不完全市场中扩散过程的模拟似然估计及其在汇率动态中的应用》。《金融经济学杂志》，63(274)，2002年。

Edward De Brouwer、Jaak Simm、Adam Arany 和 Yves Moreau。GRU-ODE-Bayes：对间歇观测的时间序列进行连续建模。载于 *NeurIPS*，2019年。Yuri Burda、Roger Baker Grosse 和 Ruslan Salakhutdinov。重要性加权自编码器。载于 *ICLR*，2016年。Sergio Casas、Cole Gulino、Simon Suo、Katie Luo、Renjie Liao 和 Raquel Urtasun。用于场景一致性运动预测的隐式潜变量模型。载于 *ECCV*，2020年。Yuning Chai、Benjamin Sapp、Mayank Bansal 和 Dragomir Anguelov。MultiPath：用于行为预测的多种概率锚点轨迹假设。*arXiv*，abs/1910.05449，2019年。Guangxi Chen、Ling Hu、Qieshi Zhang、Ziliang Ren、Xiangyang Gao 和 Jun Cheng。ST-LSTM：基于时空图的长短期记忆网络，用于车辆轨迹预测。载于 *IEEEICIP*，2020年。Junyoung Chung、Caglar Gulcehre、KyungHyun Cho 和 Yoshua Bengio。在序列建模中对门控循环神经网络进行实证评估。*arXiv*，abs/1412.3555，2014年。Junyoung Chung、Kyle Kastner、Laurent Dinh、Kratarth Goel、Aaron C Courville 和 Yoshua Bengio。用于序列数据的循环潜变量模型。载于 *NeurIPS*，2015年。Henggang Cui、Vladan Radosavljevic、Fang-Chieh Chou、Tsung-Han Lin、Thi Nguyen、Tzu-Kuo Huang、Jeff Schneider 和 Nemanja Djuric。利用深度卷积网络进行自动驾驶多模态轨迹预测。载于 *ICRA*，2019年。Nachiket Deo 和 M. Trivedi。用于车辆轨迹预测的卷积社会池化。载于 ***CVPR***研讨会，2018。Frederik Diehl、Thomas Brunner、Michael Le 和 Alois Knoll。用于建模交通参与者交互的图神经网络。载于 ***IEEE***智能车辆研讨会，2019年。Nemanja Djuric、Vladan Radosavljevic、Henggang Cui、Thi Nguyen、Fang-Chieh Chou、Tsung-Han Lin、Nitin Singh 和 Jeff Schneider。面向自动驾驶交通参与者的不确定性感知短期运动预测。载于*IEEE WACV*，2020年。Arnaud Doucet、Simon Godsill 和 Christophe Andrieu。关于贝叶斯滤波的序贯蒙特卡洛采样方法。《统计与计算》，10(3)，2000年。Ola Elerian、Siddhartha Chib 和 Neil Shephard。对离散观测非线性扩散过程的似然推断。*Econometrica*，69(4)，2001年。Marco Fraccaro、Søren Kaae Sønderby、Ulrich Paquet 和 Ole Winther。具有随机层的序贯神经模型。载于*NeurIPS*，2016年。Yarin Gal 和 Zoubin Ghahramani。将 Dropout 作为贝叶斯近似：在深度学习中表示模型不确定性。载于*ICML*，2016年。Pascal Germain、Francis Bach、Alexandre Lacoste 和 Simon Lacoste-Julien。PAC-贝叶斯理论与贝叶斯推断相遇。载于*NeurIPS*，2016年。Soumya Ghosh、Francesco Delle Fave 和 Jonathan Yedidia。用于学习贝叶斯神经网络的假定密度滤波方法。载于*AAAI*，2016年。Justin Gilmer、Samuel S. Schoenholz、Patrick F. Riley、Oriol Vinyals 和 George E. Dahl。用于量子化学的神经消息传递。载于 *ICML*，2017年。Tilmann Gneiting 和 Adrian E Raftery。严格适当评分规则、预测与估计。美国统计学会会刊，102(477):359–378，2007年。

Danijar Hafner、Timothy P. Lillicrap、Ian Fischer、Ruben Villegas、David Ha、Honglak Lee 和 James Davidson。《从像素学习用于规划的潜在动力学》。发表于 *ICML*，2019年。John Halkias 和 James Colyar。《美国101号公路数据集》。技术报告，联邦公路管理局，2007年。URL `https://rosap.ntl.bts.gov/view/dot/38724/Print`。William L. Hamilton、Rex Ying 和 Jure Leskovec。《大规模图上的归纳表示学习》。发表于 *NeurIPS*，2017年。Trevor Hastie、Robert Tibshirani、Jerome H Friedman 和 Jerome H Friedman。《统计学习要素：数据挖掘、推断与预测》，第2卷。Springer，2009年。Manuel Haussmann、Sebastian Gerwinn 和 Melih Kandemir。《采用 PAC 正则化的贝叶斯证据深度学习》。*arXiv*，abs/1906.00816，2020年。Michael Herman、Jörg Wagner、Vishnu Prabhakaran、Nicolas Möser、Hanna Ziesche、Waleed Ahmed、Lutz Bürkle、Ernst Kloppenburg 和 Claudius Gläser。《自动驾驶中的行人行为预测：需求、指标与相关特征》。*IEEE*智能交通系统汇刊，23(9)，2022年。Jose Miguel Hernandez-Lobato 和 Ryan Adams。《用于可扩展贝叶斯神经网络学习的概率反向传播》。发表于 *ICML*，2015年。Sepp Hochreiter。《动态神经网络研究》。慕尼黑工业大学学位论文，91(1)，1991年。Ajay Jain、Sergio Casas、Renjie Liao、Yuwen Xiong、Song Feng、Sean Segal 和 Raquel Urtasun。《用于概率行人行为预测的离散残差流》。发表于 *CoRL*，2019年。Ashesh Jain、Amir Roshan Zamir、Silvio Savarese 和 Ashutosh Saxena。《Structural-RNN：时空图上的深度学习》。发表于 *CVPR*，2016年。Robert Krajewski、Tobias Moers、Julian Bock、Lennart Vater 和 Lutz Eckstein。《rounD数据集：德国环岛道路使用者轨迹的无人机数据集》。发表于*ITSC*，2020年。Rahul G. Krishnan、Uri Shalit 和 David Sontag。《非线性状态空间模型的结构化推断网络》。发表于 *AAAI*，2017年。Tuan Anh Le、Maximilian Igl、Tom Rainforth、Tom Jin 和 Frank Wood。《自动编码序贯蒙特卡洛》。发表于 *ICLR*，2018年。David Lenz、Frederik Diehl、Michael Truong-Le 和 Alois C. Knoll。《高速公路场景中用于马尔可夫交互场景预测的深度神经网络》。发表于 *IEEE*智能车辆研讨会，2017年。Shiyang Li、Xiaoyong Jin、Yao Xuan、Xiyou Zhou、Wenhu Chen、Yu-Xiang Wang 和 Xifeng Yan。《增强 Transformer 在时间序列预测中的局部性并突破其记忆瓶颈》。发表于 *NeurIPS*，2019年。Yaguang Li、Rose Yu、Cyrus Shahabi 和 Yan Liu。《扩散卷积循环神经网络：数据驱动的交通预测》。发表于 *ICLR*，2018年。Yunzhu Li、Hao He、Jiajun Wu、Dina Katabi 和 Antonio Torralba。《学习用于基于模型控制的组合 Koopman 算子》。发表于 *ICLR*，2020年。Andreas Look、Melih Kandemir、Barbara Rakitsch 和 Jan Peters。《神经随机微分方程的确定性近似》。*IEEE TPAMI*，2022年。Chris J Maddison、John Lawson、George Tucker、Nicolas Heess、Mohammad Norouzi、Andriy Mnih、Arnaud Doucet 和 Yee Teh。《滤波变分目标》。发表于 *NeurIPS*，2017年。

Jean Mercat、Nicole Zoghby、Guillaume Sandou、Dominique Beauvois 和 Guillermo Gil。《基于 NGSIM 数据集的惯性单车轨迹预测基线方法及应用》。*arXiv*，abs/1908.11472，2019年。Christian Naesseth、Scott Linderman、Rajesh Ranganath 和 David Blei。《变分序贯蒙特卡洛》。发表于 *AISTATS*，2018年。Aaron van den Oord、Sander Dieleman、Heiga Zen、Karen Simonyan、Oriol Vinyals、Alex Graves、Nal Kalchbrenner、Andrew Senior 和 Koray Kavukcuoglu。《WaveNet：原始音频生成模型》。*arXiv*，abs/1609.03499，2016年。Soumyasundar Pal、Liheng Ma、Yingxue Zhang 和 Mark Coates。《结合粒子流的 RNN：概率时空预测》。发表于*ICML*，2021年。Asger Roer Pedersen。《一种基于离散观测的随机微分方程最大似然估计新方法》。《斯堪的纳维亚统计学杂志》，22(1)，1995年。Andreas Philipp 和 Daniel Goehring。《自主车辆导航的解析碰撞风险计算》。发表于 *ICRA*，2019年。Yulia Rubanova、Ricky T. Q. Chen 和 David K Duvenaud。《不规则采样时间序列的潜在常微分方程》。发表于 *NeurIPS*，2019年。Alvaro Sanchez-Gonzalez、Jonathan Godwin、Tobias Pfaff、Rex Ying、Jure Leskovec 和 Peter Battaglia。《利用图网络学习模拟复杂物理》。发表于 *ICML*，2020年。Simo Särkkä。《贝叶斯滤波与平滑》。剑桥大学出版社，2013年。Simo Särkkä 和 Juha Sarmavuori。《连续-离散动态系统的高斯滤波与平滑》。《信号处理》，93(2)，2013年。Simo Särkkä 和 Arnold Solin。《应用随机微分方程》。剑桥大学出版社，2019年。Simo Särkkä、Jouni Hartikainen、Isambi Sailon Mbalawata 和 Heikki Haario。《通过非线性高斯滤波与自适应马尔可夫链蒙特卡洛对随机微分方程参数进行后验推断》。《统计与计算》，25(2)，2015年。Thomas B Schön、Adrian Wills 和 Brett Ninness。《非线性状态空间模型的系统辨识》。《自动化》，47(1)，2011年。Thomas B Schön、Fredrik Lindsten、Johan Dahlin、Johan Wågberg、Christian A Naesseth、Andreas Svensson 和 Liang Dai。《用于系统辨识的序贯蒙特卡洛方法》。*IFAC*，48(28)，2015年。Arno Solin、Ella Tamir 和 Prakhar Verma。《通过直接匹配 Fokker-Planck-Kolmogorov 方程实现随机微分方程的可扩展推断》。发表于 *NeurIPS*，2021年。Charlie Tang 和 Russ R Salakhutdinov。《多未来预测》。发表于 *NeurIPS*，2019年。Belinda Tzen 和 Maxim Raginsky。《神经随机微分方程：扩散极限下的深度潜在高斯模型》。*arXiv*，abs/1905.09883，2019年。Benjamin Ummenhofer、Lukas Prantl、Nils Thuerey 和 Vladlen Koltun。《利用连续卷积进行拉格朗日流体模拟》。发表于 *ICLR*，2019年。Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser 和 Illia Polosukhin。《注意力就是一切》。发表于 *NeurIPS*，2017年。Tim A. Wheeler 和 Mykel J. Kochenderfer。《用于汽车安全分析的因子图场景分布》。发表于 *ITSC*，2016年。

Anqi Wu、Sebastian Nowozin、Edward Meeds、Richard E. Turner、Jose Miguel Hernandez-Lobato 和 Alexander L. Gaunt。《面向鲁棒贝叶斯神经网络的确定性变分推断》。载于*ICLR*，2019。

Fan Yang、Ling Chen、Fan Zhou、Yusong Gao 和 Wei Cao。《随机多对象系统的关系状态空间模型》。载于*ICLR*，2020。

Bing Yu、Haoteng Yin 和 Zhanxing Zhu。《时空图卷积网络：一种用于交通预测的深度学习框架》。发表于*IJCAI*，2018 年。

定理 1 的证明

定理 1. 边缘分布 $p(y_t|\mathcal{I})$ 可通过解析计算得出

$$p(y_t|\mathcal{I}) = \sum_{v=1}^{V} \pi_v(\mathcal{I}) \mathcal{N}(y_t|a_{t,v}(\mathcal{I}), B_{t,v}(\mathcal{I})),$$

对于具有如下生成模型的 *GDSSM*

$$\begin{aligned}
v &\sim Cat([\pi_1(\mathcal{I}), \ldots, \pi_V(\mathcal{I})]), \\
x_0 &\sim \mathcal{N}(\mu_{0,v}(\mathcal{I}), diag(\Sigma_{0,v}(\mathcal{I}))), \\
x_t &\sim \mathcal{N}\left(x_t|x_{t-1} + f(t,v,\mathcal{I})x_{t-1}, diag\left(L(t,v,\mathcal{I})\right)\right), && t = 1, \ldots, T \\
y_t &\sim \mathcal{N}\left(y_t|g(t,v,\mathcal{I})x_t, diag\left(\Gamma(t,v,\mathcal{I})\right)\right), && t = 1, \ldots, T
\end{aligned}$$

其中 $f(t, v、\mathcal{I})$ $L(t, v、\mathcal{I})$ $g(t, v、\mathcal{I})$ $\Gamma(t, v、\mathcal{I})$ 分别是取决于时间 t、分量 v 和上下文 $\mathcal{I}$ 的矩阵，其维度适当。

证明。 证明很直接，因为转移函数和发射函数的输出矩可以通过解析方式获得

$$\begin{aligned}
p(y_t|\mathcal{I}) &= \int p(y_t|v, x_t, \mathcal{I}) p(x_t|v, x_0, \mathcal{I}) p(v, x_0|\mathcal{I}) dv dx_0 dx_t \\
&= \sum_{v=1}^{V} \pi_v(\mathcal{I}) \int p(y_t|v, x_t, \mathcal{I}) p(x_t|v, x_0, \mathcal{I}) \mathcal{N}(x_0|\mu_{0,v}(\mathcal{I}), \mathrm{diag}(\Sigma_{0,v}(\mathcal{I}))) dx_0 dx_t \\
&= \sum_{v=1}^{V} \int p(y_t|v, x_t, \mathcal{I}) \mathcal{N}(x_t|\mu_{t,v}(\mathcal{I}), \Sigma_{t,v}(\mathcal{I})) dx_t \\
&= \sum_{v=1}^{V} \pi_v(\mathcal{I}) \mathcal{N}(y_t|a_{t,v}(\mathcal{I}), B_{t,v}(\mathcal{I})).
\end{aligned}$$

线性时变动力系统在时间步 t 处的矩可表示为（Särkkä，2013）

$$\begin{aligned}
\mu_{t,v}(\mathcal{I}) &= \prod_{t'=1}^{t} \left(I + f(t', v, \mathcal{I})\right) \mu_{0,v}(\mathcal{I}), \\
\Sigma_{t,v}(\mathcal{I}) &= \left[\prod_{t'=1}^{t} \left(I + f(t', v, \mathcal{I})\right)\right] \mathrm{diag}(\Sigma_{0,v}(\mathcal{I})) \left[\prod_{t'=1}^{t} \left(I + f(t', v, \mathcal{I})\right)^T\right] \\
&\quad + \sum_{t'=1}^{t-1} \left[\prod_{t''=t'+1}^{t} \left(I + f(t'', v, \mathcal{I})\right)\right] \mathrm{diag}(L(t', v, \mathcal{I})) \left[\prod_{t''=t'+1}^{t} \left(I + f(t'', v, \mathcal{I})\right)^T\right] \\
&\quad + \mathrm{diag}(L(t, v, \mathcal{I}))
\end{aligned}$$

通过 BMM 算法也能得到相同的表达式，这一点只需将局部线性系统代入式（6）即可轻松证明。最后，时间步 t 的输出均值 $a_{t,\ v}(\mathcal{I})$ 和协方差 $B_{t,\ v}(\mathcal{I})$ 可表示为（Särkkä，2013）

$$\begin{aligned}
a_{t,v}(\mathcal{I}) &= \mathbb{E}[g(t, v, \mathcal{I})x_t] = g(t, v, \mathcal{I})\mu_{t,v}(\mathcal{I}), \\
B_{t,v}(\mathcal{I}) &= \mathrm{Cov}[g(t, v, \mathcal{I})x_t] + \mathrm{diag}\left(\mathbb{E}[\Gamma(t)]\right) = g(t, v, \mathcal{I})\Sigma_{t,v}(\mathcal{I})g(t, v, \mathcal{I})^T + \mathrm{diag}\left(\Gamma(t, v, \mathcal{I})\right).
\end{aligned}$$

□

B ReLU 激活函数的输出矩

假设我们在层 l 和时间步 t 对输入 x_t^l应用 ReLU 激活函数

$$x_t^{l+1} = \max(0, x_t^l).$$

输出矩 我们可以将 x_t^{l+1}的输出矩近似表示为输入矩的函数（Wu 等，2019）

$$\mathbb{E}[\max(0, x_t^l)] \approx \sqrt{\text{diag}(\text{Cov}[x_t^l])}\ \text{SR}\left(\mathbb{E}[x_t^l] / \sqrt{\text{diag}(\text{Cov}[x_t^l])}\right),$$
$$\text{Cov}[\max(0, x_t^l)] \approx \sqrt{\text{diag}(\text{Cov}[x_t^l])\text{diag}(\text{Cov}[x_t^l])^T}\ F\left(\mathbb{E}[x_t^l], \text{Cov}[x_t^l]\right),$$

其中 $\text{SR}(x) = \phi(x) + x\Phi(x)$，$\phi$和 Φ 分别表示标准高斯变量的概率密度函数和累积分布函数。函数 $F(\mathbb{E}[x_t^l], \text{Cov}[x_t^l])$ 是以下表达式的简写：

$$F(\mathbb{E}[x_t^l], \text{Cov}[x_t^l]) = A\left(\mathbb{E}[x_t^l], \text{Cov}[x_t^l]\right) + \exp\left[-Q\left(\mathbb{E}[x_t^l], \text{Cov}[x_t^l]\right)\right].$$

函数 $A(\mathbb{E}[x_t^l], \text{Cov}[x_t^l])$ 可估计为

$$A(\mathbb{E}[x_t^l], \text{Cov}[x_t^l])) = \text{SR}(\epsilon_k^l)\text{SR}(\epsilon_k^l)^T + \rho_k^l \Phi(\epsilon_k^l)\Phi(\epsilon_k^l)^T,$$

其中 $\rho_k^l = \text{Cov}[x_t^l] / \left(\sqrt{\text{diag}(\text{Cov}[x_t^l])}\sqrt{\text{diag}(\text{Cov}[x_t^l])^T}\right)$是一个无量纲矩阵，而 $\epsilon_k^l = \mathbb{E}[x_t^l] / \sqrt{\text{diag}(\text{Cov}[x_t^l])}$是一个无量纲向量。

i, j阶元素 $Q(\mathbb{E}[x_t^l], \text{Cov}[x_t^l])$ 可估计为：

$$Q\left(\mathbb{E}[x_t^l], \text{Cov}[x_t^l])\right)_{i,j} = \frac{\rho_{k_{i,j}}^l}{2g_{k_{i,j}}^l(1+\bar{\rho}_{k_{i,j}})}\left((\epsilon_{k_i}^l)^2 + (\epsilon_{k_j}^l)^2\right) - \frac{\arcsin(\rho_{k_{i,j}}^l) - \rho_{k_{i,j}}^l}{\rho_{k_{i,j}} g_{k_{i,j}}^l}\epsilon_{k_i}^l \epsilon_{k_j}^l - \log\left(\frac{g_{k_{i,j}}^l}{2\pi}\right),$$

其中，$g_k^l = \arcsin(\rho_k^l) + \rho_k^l \oslash \left(1 + \sqrt{1 - \rho_k^l \odot \rho_k^l}\right)$。算子 $\oslash$ 表示逐元素除法，$\odot$ 表示逐元素乘法。

期望雅可比矩阵由于激活函数是逐元素应用的，期望梯度的非对角元素均为零。ReLU 函数的雅可比矩阵的对角元素是期望 Heaviside 阶跃函数 Wu 等人（2019）

$$\text{diag}\left(\mathbb{E}\left[\nabla_{x_t^l} \max(0, x_t^l)\right]\right) \approx \Phi\left(\mathbb{E}[x_t^l] / \sqrt{\text{diag}(\text{Cov}[x_t^l]))}\right).$$

关于更详细的推导，请参阅 Wu 等人（2019）的研究。

C 训练细节

我们使用 ADAM 优化器和随机小批次训练所有模型。批量大小设为 4，学习率设为 0.0001。为了加速多模态 GDSSM 的训练，我们使用单模态 GDSSM 的预训练版本初始化转移神经网络和观测神经网络。

C.1 rounD

我们对确定性模型（GDSSM Det. 和 Non Recur. GNN）进行 50k 次权重更新训练。对于基于 MC 的模型（GDSSM MC），我们使用 100k 次权重更新。该数据集包含 4,314 个训练片段和 1,091 个测试片段。由于未观察到过拟合，我们不使用单独的验证数据集。

C.2 NGSIM

我们在 NGSIM 数据集上对所有模型进行 1000k 次更新训练。该数据集包含 5,922k/860k/1,505k 个训练/验证/测试片段。每进行 10k 次权重更新后，我们使用来自验证数据集的 1k 个随机小批次对模型进行验证。

C.3 分布外测试

我们对所有模型进行了 50k 次权重更新训练。该数据集由三个子数据集组成，分别包含 254/251/137 个片段；每个子数据集的 80% 用于训练，其余 20% 用于测试。由于子数据集规模较小，我们观察到了过拟合。为此，我们将每个训练子数据集最后 20% 的数据用于验证。每进行 1k 次权重更新后，我们都会验证一次模型。

D 评估指标

下面我们简要介绍不同的评估指标。论文正文中给出了负对数似然（NLL）和均方根误差（RMSE），附录 F 中给出了最小均方根误差（ minRMSE ）。

Ro均方误差（MSE）时间点 t 处的（均方根）误差定义为

$$\mathrm{MSE}_{\mathrm{Base}}(t)=\frac{1}{N}\sum_{n}^{N}\sum_{m}^{M(n)}\frac{(y_{t,n}^{m}-\hat{y}_{t,n}^{m})^{T}(y_{t,n}^{m}-\hat{y}_{t,n}^{m})}{M(n)},\quad \mathrm{RMSE}_{\mathrm{Base}}(t)=\sqrt{\mathrm{MSE}_{\mathrm{Base}}(t)}, \tag{25}$$

其中， N 是数据集中的片段数， $M(n)$是第 n 个片段中的智能体数量，$y_{t,\,n}^{m}$ 是第 n 个片段中第 m 个智能体在时间点 t 的真实位置，而 $\hat{y}_{t,\,n}^{m}$ 是相应的预测位置。

遗憾的是，将 $\mathrm{RMSE}_{\mathrm{base}}$ 扩展到概率预测并非易事。据我们所知，现有变体中没有一种是严格适当评分规则（Gneiting 和 Raftery，2007）；只有当预测分布与真实分布相匹配时，后者才是最优的。本文还给出了一种严格适当评分规则的示例，即预测负对数似然:

$$\mathrm{NLL}(t)=-\frac{1}{N}\sum_{n}^{N}\sum_{m}^{M(n)}\frac{\log p(y_{t,n}^{m}|\mathcal{I})}{M(n)}. \tag{26}$$

对于确定性 GDSSM，我们使用第 4.2 节中介绍的方法直接近似 $p(y_{t,\,n}^{m}|\mathcal{I})$。对于不采用假定密度方法的 GDSSM，我们通过蒙特卡洛积分近似 $p(y_{t,\,n}^{m}|\mathcal{I})$。

下面我们将更详细地讨论不同的均方根误差变体。

贝叶斯预测器：机器学习社区常用的一种评估指标（例如 Gal & Ghahramani（2016年））是计算

$$\mathrm{RMSE}(t)=\sqrt{\frac{1}{N}\sum_{n}^{N}\sum_{m}^{M(n)}\frac{(y_{t,n}^{m}-\mathbb{E}[\hat{y}_{t,n}^{m}])^{T}(y_{t,n}^{m}-\mathbb{E}[\hat{y}_{t,n}^{m}])}{M(n)}}, \tag{27}$$

这里我们假设概率预测器会在给出最终预测之前进行模型平均。该分数只评估一阶矩的优劣，不考虑高阶矩的任何信息。我们在论文正文中给出了它的数值。

吉布斯预测器：将确定性损失函数扩展到概率情形的另一种方法，可以在 PAC-贝叶斯框架中找到（例如 Germain 等人（2016年）），其做法是考虑平均损失。

风险函数。在我们的案例中，这将导致

$$\mathrm{RMSE}_{\mathrm{Gibbs}}(t) = \sqrt{\mathbb{E}\left[\frac{1}{N}\sum_{n}^{N}\sum_{m}^{M(n)}\frac{(y_{t,n}^{m}-\hat{y}_{t,n}^{m})^{T}(y_{t,n}^{m}-\hat{y}_{t,n}^{m})}{M(n)}\right]}, \tag{28}$$

其中，我们对预测值 \(\hat{y}_t^m\) 求期望. $_n$。然而，我们不建议将这一表述作为评估指标，因为它会惩罚预测器中的任何形式的方差，并倾向于在真实分布的期望值附近形成狄拉克分布。值得注意的是，这一结果与统计学习理论中的偏差-方差权衡密切相关（例如，Hastie 等人，2009 年）。

最小预测器：交通预测文献中一种流行的替代方法（例如 Tang & Salakhutdinov (2019)）是计算一组 S 潜在预测的最小误差

$$\mathrm{minRMSE}(t) = \sqrt{\frac{1}{N}\sum_{n}^{N}\sum_{m}^{M(n)}\min_{s=1,\ldots,S}\frac{(y_{t,n}^{m}-\hat{y}_{t,n,s}^{m})^{T}(y_{t,n}^{m}-\hat{y}_{t,n,s}^{m})}{M(n)}}, \tag{29}$$

其中 $(\hat{y}_t^m{}_{,n,s})_{s=1}^S$ 是 S 个可能预测的集合。此评分倾向于选择能生成多样化预测集合的模型，其中至少有一个候选预测接近真实结果，但忽略其余 $S-1$ 个预测器的表现。虽然它能提供一些关于模型容量的有用信息，但当最佳模式未知时，它无法完整反映测试时性能。其数值见附录 F。

E 替代参数推断方法

机器学习领域中一种常用的推断方法不再最大化对数似然，而是最大化证据下界 (ELBO)

$$\mathrm{ELBO}(y_1,\ldots,y_T|\mathcal{I}) = \mathbb{E}_{q(x_0,\ldots,x_T)}\left[\log\frac{p(y_1,\ldots,y_T,x_0,\ldots,x_T|\mathcal{I})}{q(x_0,\ldots,x_T)}\right], \tag{30}$$

该方法涉及学习一个近似分布 $q(x_0,\ldots,x_T)$，以逼近难以处理的平滑分布$p(x_0,\ldots,x_T|y_1,\ldots,y_T,\mathcal{I})$（Krishnan 等人，2017）。

通过计算重要性加权对数似然，可以得到更紧的对数似然 $\log\ p(y_1,\ldots,y_T|\mathcal{I})$ 下界；我们将其称为蒙特卡洛目标 (MCO)（Maddison 等人，2017；Burda 等人，2016）

$$\mathrm{MCO}(y_1,\ldots,y_T|\mathcal{I}) = \mathbb{E}_{q(x_0,\ldots,x_T)}\left[\log\frac{1}{K}\sum_{k=1}^{K}\frac{p(y_1,\ldots,y_T,x_{0,k},\ldots,x_{T,}|\mathcal{I})}{q(x_{0,k}\ldots,x_{T,k})}\right], \tag{31}$$

其中，K 是蒙特卡洛样本的数量，$x_{t,k}$ 是时间步 t上的第 k 个样本。对于状态空间模型，近期研究将 MCO 与粒子滤波器结合起来（Naesseth 等人，2018；Maddison 等人，2017；Le 等人，2018）。

F 扩展结果

我们提供了 rounD 和 NGSIM 数据集上的最小均方根误差（minRMSE）值。我们将提议预测设置为每种模式的均值，并通过最小值运算符，为每个智能体选择在整个预测时域内每个片段上产生最低误差的提议。只要至少有一个候选预测接近真实结果，该评分指标就会倾向于提议预测具有多样性的模型。

如表 5 所示，随着模式数量的增加，最小均方根误差逐渐降低，这表明不同模式对应于一组多样的合理轨迹。

表 5：rounD 和 NGSIM 数据集上最小均方根误差随模式数量变化的数值。我们给出了 10 次运行的平均值和标准误差。关于均方根误差和负对数似然的结果，请参见表 1 和表 2。

		rounD				NGSIM			
		1种模式	2种模式	3 模态	4种模式	1种模式	2种模式	3种模式	4种模式
minRMSE	1 s	0.79 ± 0.02	0.75 ± 0.04	0.76 ± 0.03	0.69 ± 0.02	0.53 ± 0.01	0.46 ± 0.01	0.35 ± 0.01	0.32 ± 0.01
	2 s	1.87 ± 0.02	1.85 ± 0.06	1.83 ± 0.07	1.68 ± 0.05	1.18 ± 0.01	1.05 ± 0.01	0.80 ± 0.01	0.71 ± 0.01
	3 s	3.36 ± 0.03	3.46 ± 0.11	3.05 ± 0.16	2.75 ± 0.12	1.98 ± 0.02	1.73 ± 0.02	1.33 ± 0.02	1.18 ± 0.03
	4 s	5.08 ± 0.04	5.07 ± 0.13	4.55 ± 0.28	4.04 ± 0.45	2.99 ± 0.03	2.60 ± 0.04	2.02 ± 0.04	1.75 ± 0.04
	5 s	7.24 ± 0.05	6.29 ± 0.23	5.95 ± 0.47	4.43 ± 0.53	4.29 ± 0.04	3.69 ± 0.06	2.88 ± 0.05	2.46 ± 0.05

G 分布外测试的实验设置

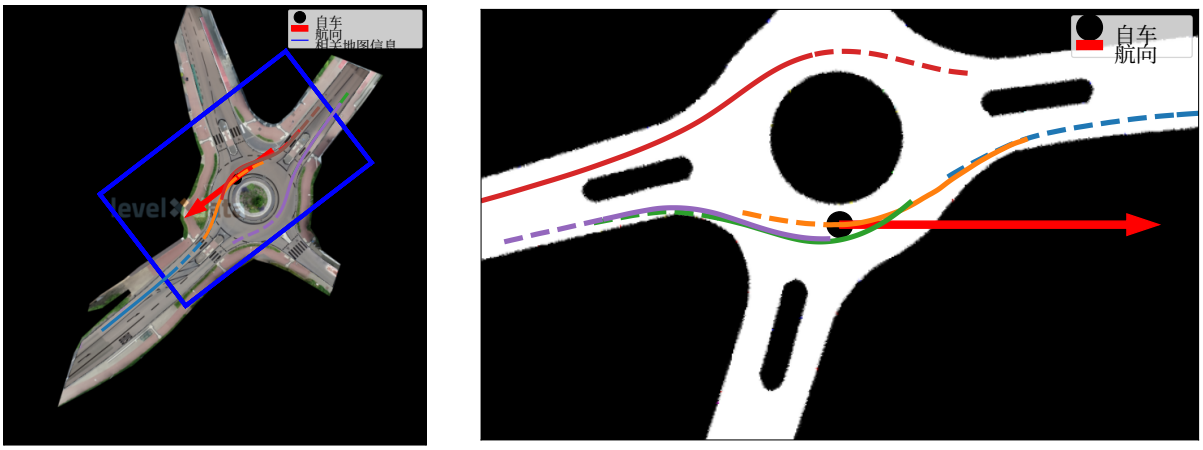
G.1 数据集构建

我们构建了一个数据集，该数据集由三种不同的交通环境组成。我们分别选取了亚琛 Kackertstraße（K）环岛的一段记录、Alsdorf Thiergarten（T）环岛的一段记录，以及亚琛附近 Neuweiler（N）环岛的一段记录。我们采用与第 5.1 节相同的预处理流程。我们从每个交通环境中移除行人、自行车和停放车辆。在交通环境 K/T/N 中，分别剩下 319/264/389 个跟踪目标，时间跨度为 0.3/0.3/0.15 小时。我们将记录下采样 5 倍，然后为每个交通环境构建一个数据集，该数据集由长度为 8 秒、重叠率为 50% 的片段组成。前 3 秒用作轨迹历史，后 5 秒用作预测时域。我们在交通环境 K/T/N 中分别获得 254/251/137 个片段。对于每个交通环境，我们使用前 80% 的片段进行训练，剩余 20% 的片段进行测试。

G.2 地图处理

地图处理遵循交通预测领域的现有研究 Bansal 等人（2018）；Herman 等人（2022年）。给定一辆自车及其历史记录，我们首先计算车辆的航向。航向是根据最近 0.2 秒观测到的航向方向取平均值计算得出的。位置和航向方向共同定义地图上的感兴趣区域 （ROI）。ROI 是一个长 74 米、宽 44 米的矩形，以自车为中心，并根据自车的航向定向。我们进一步将 RGB 图像转换为二值道路图像。我们将其可视化

图 7b 中处理后的地图信息



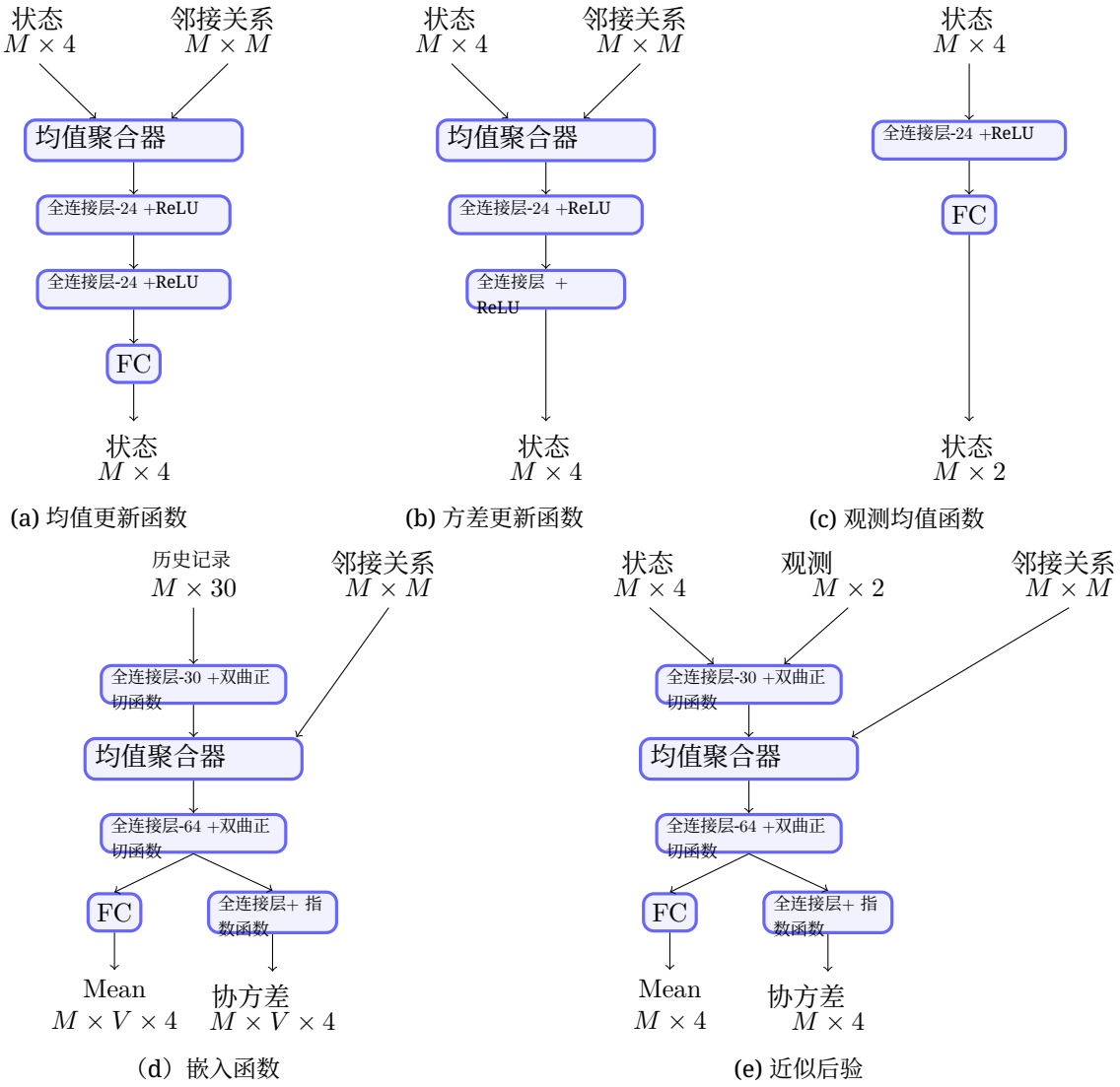
（a）包含自车的完整地图。　　（b）旋转、裁剪并掩码处理后的地图信息。

图 7：地图信息的处理。

H 网络架构

H.1 无世界模型的网络架构

下面给出第 5.1、5.2 和 5.3 节实验所用的神经网络架构。



(a) 均值更新函数　(b) 方差更新函数　(c) 观测均值函数

（d）嵌入函数　(e) 近似后验

图 8：不含地图信息的架构。对于全连接层，我们给出了输出神经元的数量。

嵌入函数接收所有 M 智能体的历史记录。该历史记录时长为 3 秒，时间步为 0.2 秒，由二维坐标组成。展平后，输入为一个大小为 30 的向量。嵌入函数的输出是一个包含 V 个混合分量的高斯混合模型。我们在嵌入、均值和方差更新函数中使用均值聚合器。均值聚合器根据公式 22 计算发送给智能体 m 的消息。计算出消息 $x_t^{\mathcal{N}_m}$后，将其与状态 x_t^m拼接。均值和方差更新函数均为神经网络，它们在每个预测步骤中执行一轮消息传递，然后计算输出。我们的发射模型使用神经网络作为均值函数 $g(x_t)$，并使用一个常数向量作为 $Q(x_t)$。发射模型将每个智能体的状态映射回观测空间，且不依赖于交互。近似后验用于

对于在第 5.1 节中基于证据下界或蒙特卡洛目标训练的基线方法。相比之下，最大化 PLL 不需要近似后验。近似后验网络接收前一个状态和当前观测，以近似当前时间步的潜在分布。

H.2 带地图信息的网络架构

这些神经网络架构已用于第 5.4 节的实验。其结构与 H.1 节中的架构非常相似。我们额外加入了一个神经网络，将大小为 500x300 的掩码映射编码为一个展平的 256 维向量。该地图嵌入用作嵌入函数的额外输入。

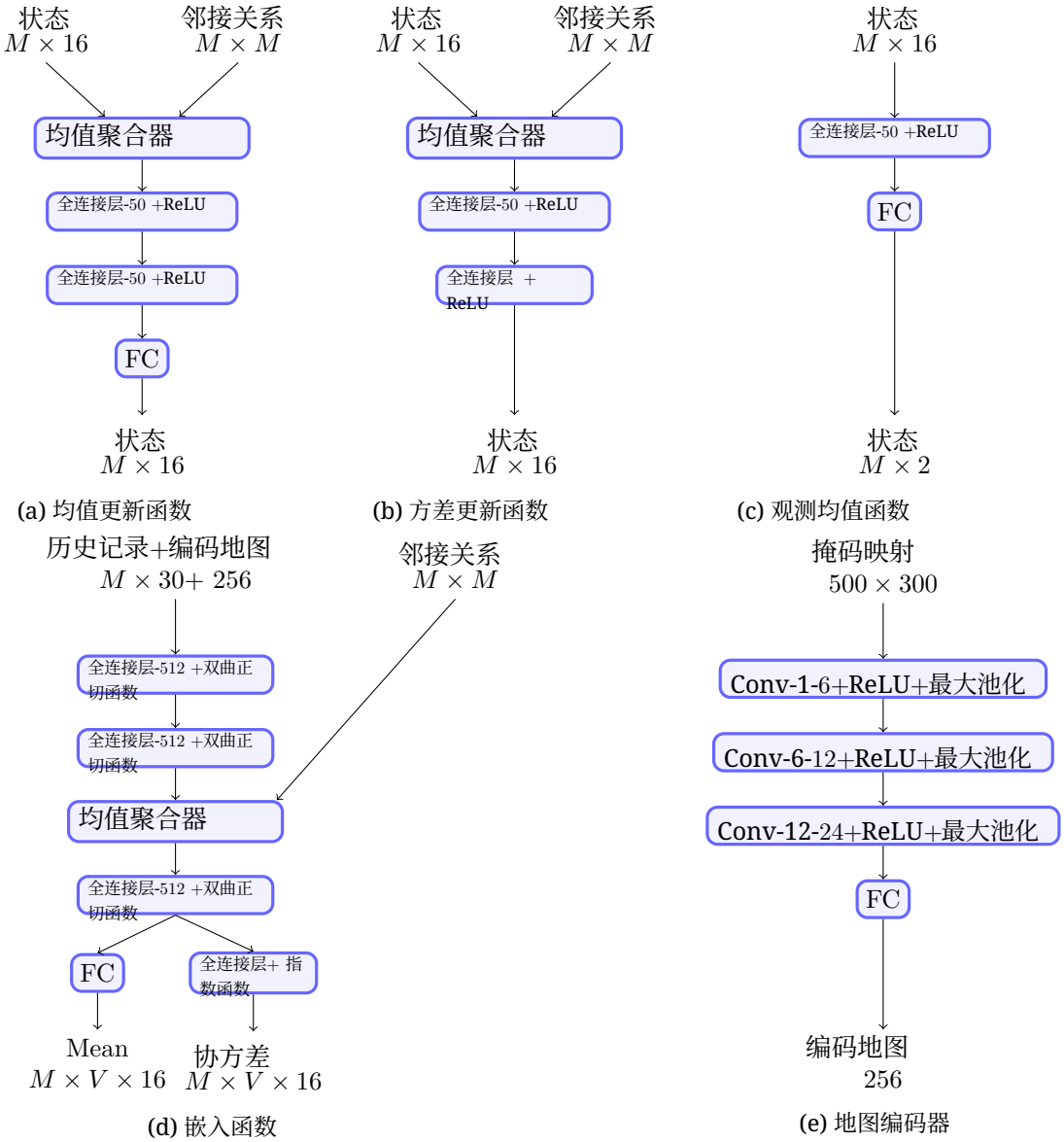


图9：包含地图信息的网络架构。对于全连接层，我们给出输出神经元的数量。对于卷积层，我们给出输入通道和输出通道的数量。我们使用大小为3的卷积核，步幅为2。对于最大池化层，我们使用大小为2的卷积核，步幅为2。