低成本的道路网络时空预测

Mridul Gupta, Hariprasad Kodamana, Sayan Ranu

摘要

道路网络的时空过程建模是一项日益重要的任务。虽然在开发时空图神经网络 (GNNs) 方面取得了重大进展，但现有的工作是建立在三个假设之上的，这些假设过于理想。首先，他们假设在道路网络的每个节点上都有传感。在现实中，由于预算限制或传感器故障，可能不是所有节点都配备传感器。其次，他们假设所有已安装的传感器都有历史记录。由于传感器故障、通信过程中丢失数据包等原因，这也是不现实的。最后，他们假设路网是静态的。真实路网的连通性会因道路封闭、新道路建设等而改变。该论文开发了 FRIGATE 来解决这些不足。FRIGATE 由一个时空 GNN 提供动力，该 GNN 将位置、拓扑和时间信息集成到丰富的归纳节点表示中。门控 Lipschitz 嵌入与 LSTMs 的新颖组合，使这种不同信息的融合成为可能。本实验证明了所提出的 GNN 架构比最先进算法中使用的消息传递 GNN 更具表现力。本文证明了这种 GNN 架构比现有最先进算法中使用的消息传递 GNN 的表现更好，在交通数据有限的现实世界网络上拥有卓越性能。此外，FRIGATE 对低成本传感器部署、路网连接变化和感知的时间不规则性具有鲁棒性。

关键词： 时空预测；图神经网络；道路网络

1 引言

道路网络可以建模为一个图，其中节点表示区域内的重要路口，边缘对应连接两个路口的街道 [7,15–17,19,27]。近年来，道路网络时空事件演变的建模引起了人们的极大兴趣 [4,12,24,26,28]。具体来说，每个节点 (或边) 都参与一个时间序列。这个时间序列不仅是一天中的时间的函数，也是网络中其他节点中展开的事件的函数。预测任务的目标是为每个节点的时间序列演化建模，并预测近期的值，如下一个小时。例如，在图 1中，实验给出了在北京和成都两个随机选择的十字路口通过的汽车数量。可以看出，每天的交通流量都有很大的变化。此外，时间序列在不同节点之间变化，使得预测问题不并不简单。

人们可以在每个节点独立地学习一个自回归模型来拟合时间序列数据。然而，这一策略受到两个关键因素的限制。首先，这忽略了节点之间由连接性引起的时间序列依赖性。例如，如果一个特定的十字路口 (节点) 正在观察交通拥堵，那么通过传出边连接的相邻节点也可能受到影响。其次，参数的数量随着图中节点的数量线性增长，使得图不可拓展。

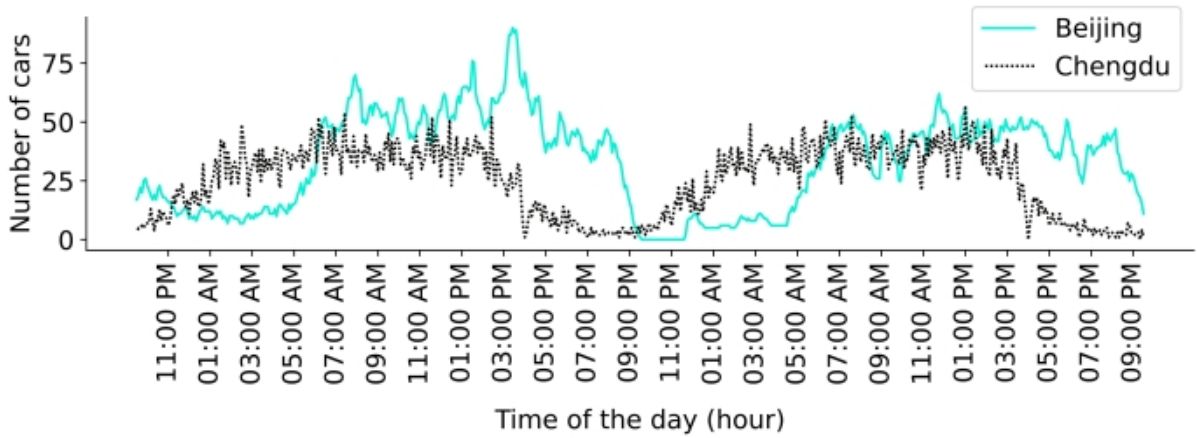


图 1. 北京和成都选定道路的快照，显示交通的潮汐变化

2 相关工作

为了解决这些建模依赖于网络的时空过程的特定需求，已经开发了几种将结构数据模型合并的算法，如 GNNs [8, 9, 28] 或卷积神经网络 [26] 以及自回归架构。这些工作所做的一些假设对于现实世界的道路网络是不现实的。现有工作情况如表 1总结。

表 1. 基线

算法	部分感知	适应路网更新	无时间历史预测
AGCRN [1]	×	×	×
STGODE [4]	×	×	×
DSTAGNN [8]	×	×	×
STFGNN [9]	×	×	×
G2S [30]	×	×	×
GMSDR [14]	×	×	×
Z-GCNETs [2]	×	×	✓
STSGCN [20]	×	×	✓
GraphWavenet [24]	×	×	✓
GMAN [29]	×	×	✓
MTGNN [23]	×	×	✓
TISV [22]	✓	×	✓
STGCN [28]	✓	✓	✓
DCRNN [12]	✓	✓	✓
STNN [26]	✓	✓	✓
LocaleGN [10]	✓	✓	✓
ST-GAN [21]	✓	✓	✓

- **通过部分已知传感器推断未知传感器的能力：**一些现有的算法假设传感器被放置在每个

节点 [1,2,4,8,9,14,20,23,24,29,30]。预测只有在这些节点上才是可行的。实际上，在所有十字路口部署和维护传感器可能过于昂贵和繁琐。因此，在没有明确放置传感器的节点上进行准确预测也很重要。这项工作表明，这确实是可行的，利用节点之间的网络诱导依赖。

- **适应路网更新的能力：**道路网络内的连通性可能会随着时间而变化。边缘的方向性可能会根据一天中的时间而变化，新的道路可能会被建造，增加新的节点和边缘到网络中，现有的道路可能会被移除 (暂时或永久) 以适应紧急需求，如街头节日，建筑活动，洪水等。在这种情况下，重要的是吸收网络拓扑中的小变化，而不需要从头开始重新训练。几个模型 [1,2,8,14,20,22,24,29,30] 无法吸收任何更新，因为它们本质上是可转换的，即其模型中的参数数量是网络大小 (节点或边的数量) 的函数。本工作开发了一个归纳模型，它将模型参数大小与网络参数大小解耦。因此，适应网络结构的变化不需要重新培训。

- **没有时间历史或规律性的预测能力：**现有的几种算法只有在过去的时间瞬间的数据可用时才能对节点的时间序列进行预测 [4,9]。这种限制通常源于一种方法，其中所有节点之间的依赖关系是通过计算它们过去时间序列信息之间的相似性来学习的。如果没有过去的数据，那么这种相似性就不能计算出来。此外，一些算法将时空道路网络建模为三维张量，隐式假设时间数据以规则粒度 (例如每五分钟) 收集 [4,26]。在现实中，传感器可能会失效，因此可能以不规则的间隔提供数据，或者在过去的几个时间瞬间没有时间历史。对于要在实际工作负载中部署的模型，对传感器故障的鲁棒性是相关的。

  受上述限制的启发，本工作提出了以下问题：是否有可能通过一小部分节点，基于部分感知，设计一个跨图中所有节点的准确、归纳的预测模型？此外，该模型是否可以预测具有不规则时间序列可见性的节点，或者在最坏的情况下，根本没有可见性？本实验表明，这确实是可能的，通过 FRIGATE：低成本和归纳 lipschitz 门控时空 GNN。

3 本文方法

3.1 本文方法概述

在单个节点级别，提出的问题是一个序列到序列的回归任务，其中本实验通过过去的快照提供时间序列，并预测目标节点上的未来时间序列。为了模拟这个问题，本实验使用如图 2所示的编码器-解码器架构。为了共同捕获节点间的时空依赖关系，编码器由一堆连体 GNNs 和 LSTMs 组成，即每个堆栈的权重和架构相同。堆栈的数量对应于希望训练的历史快照的数量。每个堆栈被分配一个索引，这取决于它从当前时间向后的距离 t。由于 siamese 架构，堆栈的数量不会影响参数的数量。此外，siamese 设计允许在推理时根据可用数据量以特定于查询的方式动态指定堆栈大小。为简单起见，实验假设历史快照的数量与预测范围相同，用 Δ 表示。

在 k^{th} 堆栈中 $(0 \leq k \leq \Delta)$，通过 GNN 传递图形快照 $G_{t-\Delta+k}$ 来学习节点表示。节点表示不仅表征其自身的状态，还表征其“邻居”的状态。影响目标节点时间序列的“邻域”是通过 Lipschitz-gating 自动学习的。编码器中的每个 siamese LSTMs 堆栈接收两个输入：与时间戳对应的目标节点的节点嵌入和来自前一个堆栈的 LSTMs 输出。

解码器也有一堆 LSTMs。但是，解码器 LSTMs 比编码器 LSTMs 具有不同的权重。堆栈大小与预测范围 Δ 相同。解码器 LSTMs 的输出被目标节点附近传感器值分布的矩增强，这给模型注入了强先验，提高了模型的性能。最后，将该增广表示通过 MLP 来预测时间序列值。

如图 2中所示, 生成预测输出通过 k^{th} LSTM 解码器。下一节将详细介绍这些单独的组件。

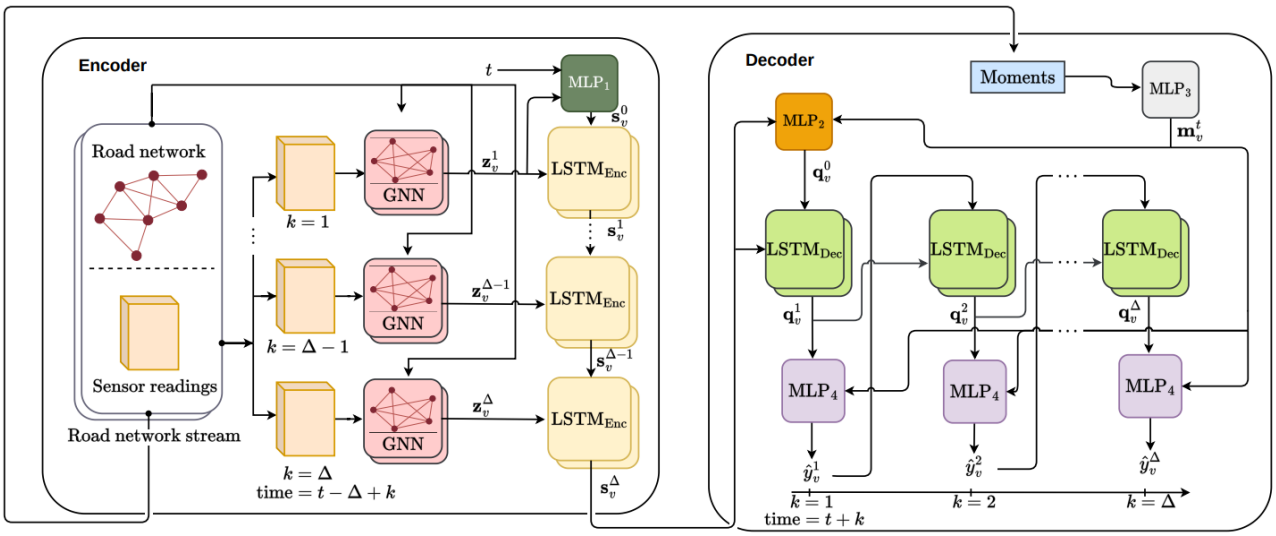


图 2. FRIGATE 的结构

3.2 GNN 模块

本节中将讨论单个 GNN 堆栈的体系结构。本实验使用 L-layered 消息传递 GNN 来建模节点依赖关系。在每一层中，每个节点 v 接收来自其邻居的消息。这些消息被聚合并通过神经网络 (通常是 MLP) 传递，以在下一层构建 V 的表示。

最简单的聚合框架是 MeanPool 层，其中来自相邻节点的聚合消息 v 只是它们表示的平均值。然而，对于所提出的问题，这种方法过于简单。具体来说，道路网络是定向的，因此仅通过传入 (或传出) 边获得消息是不够的。另一方面，与方向无关的消息传递方案通过统一从所有事件边获取消息并无法捕获流量语义。此外，并不是所有的边都同样重要。与主干道相对应的边可能比来自小型住宅区的道路对十字路口 (节点) 产生更大的影响。因此，必须从数据中学习到道路 (边) 对路口 (节点) 的重要性，并相应地对其消息进行加权。为了获取这些语义，实验将消息传递模式表述如下。

设 h_v^ℓ 表示节点 v 在 k^{th}GNN 的第 $\ell \in [0, L]$ 层中的表示。此外, 传出和传入的邻居 v 被定义为 $\mathcal{N}_v^{out} = \{u \mid (v,u) \in \varepsilon\}$ 和 $\mathcal{N}_v^{in} = \{u \mid (u,v) \in \varepsilon\}$。现在，$h_v^{\ell+1}$ 的构造如下:

$$\mathbf{h}_v^{\ell+1} = \sigma_1(\mathbf{W}_\ell^1(h_v^\ell \parallel \mathbf{m}_v^{\ell,out} \parallel \mathbf{m}_v^{\ell,in}), where \tag{1}$$

$$\mathbf{m}_v^{\ell,out} = AGGR^{out}(\{\mathbf{h}_v^\ell \mid u \in N_v^{out}\}) \tag{2}$$

$$\mathbf{m}_v^{\ell,in} = AGGR^{in}(\{\mathbf{h}_v^\ell \mid u \in N_v^{in}\}) \tag{3}$$

$\|$ 表示拼接操作，$\mathbf{W}_1^\ell \in \mathbb{R}^{3d \times d}$ 为可学习权矩阵；d 是节点表示的维度。更简单地说，本实验对从传入和传出邻居接收到的消息执行两个单独的聚合。将聚合向量连接起来，然后通过线性变换，然后通过激活函数 σ_1 进行非线性。在本实验的实现中，σ_1 是 ReLU。通过在传入边和传出边上执行两个单独的聚合，然后将它们连接起来，本实验捕获了方向性和拓扑。

$$AGGR^{out}(\{\mathbf{h}_v^\ell \mid u \in N_v^{out}\}) = \sum_{\forall u \in N_v^{out}} \beta_{v,u} \mathbf{h}_u^\ell \tag{4}$$

$$AGGR^{in}(\{\mathbf{h}_v^\ell \mid u \in N_v^{in}\}) = \sum_{\forall u \in N_v^{in}} \beta_{u,v} \mathbf{h}_u^\ell \tag{5}$$

$$\beta_{vi,vj} = \frac{1}{1 + \mathbf{e}^{-\omega_{vi,vj}}}, where \tag{6}$$

$$\omega_{vi,vj} = \mathbf{w}^\ell \cdot \mathbf{L}_{vi,vj} + \mathbf{b}, where \tag{7}$$

$$\mathbf{L}_{vi,vj} = \sigma_2(\mathbf{W}_L^\ell((\mathbf{w}_\delta^T \cdot \delta(vi, vj)) \parallel \mathbf{L}_{vi} \parallel \mathbf{L}_{vj})) \tag{8}$$

$\mathbf{L}_v$ 表示节点 v 的位置嵌入。直观地说，位置嵌入表示一个节点的位置，如果两个节点在最短路径距离上接近，那么它们的位置嵌入也应该是相似的。由于 $\mathbf{L}_{vi,vj}$ 是其两个端点的位置嵌入和它们之间的空间距离的函数，因此 $\mathbf{L}_{vi,vj}$ 可以解释为边 (v_i, v_j) 的位置嵌入。通过 MLP 传递其位置表示来计算边的重要性，随后将其转换为标量。最后，标量通过一个 Sigmoid 门得到它的权重。此处，$\mathbf{w}_\delta \in \mathbb{R}^{d_\delta}$，$\mathbf{W}_L^\ell \in \mathbb{R}^{2d_L + d_\delta \times d_{L_e}}$ 和 $\mathbf{w}^\ell \in \mathbb{R}^{d_{L_e}}$ 是可学习权重参数，d_L 和 d_{L_e} 分别为节点和边位置嵌入的维度。d_δ 是在边距离为 $\delta_{vi,vj}$ 时创建的投影的维度，公式 8中的 σ_2 是应用非线性的激活函数。公式 7中的 b 为可学习偏差项。

现在讨论这个设计背后的基本原理。如图 3所示，在道路网络中，边缘上的交通分布类似于幂律。虽然主干道承载了大量的交通，因此是下游道路状态的一个强有力的决定因素，但构成大多数的局部道路影响较小。因此，需要从数据中了解道路的重要性。此外，本实验希望 GNN 在没有过度平滑或过度压缩的情况下是深度的 [3]。Sigmoid 门控满足这些双重要求。首先，它为每条边分配一个权重来确定它的重要性。其次，它使本实验能够深入，因为随着层数的增加，只能从具有高权重的边接收消息，从而避免过度挤压。语义上，权重高的道路应与主干道相对应。请注意，与图注意网络不同，图注意网络中的边在重要性上相互竞争 (由于 SoftMax 中的归一化)，在 Sigmoid 门控中没有这种效果。这对于道路网络来说更自然，因为繁忙十字路口的表示幅度应该高于没有进入主干道的十字路口。最后，值得注意的是，注意权值是定向的，因为它是边的位置嵌入函数，其中 $L_{vi,vj} \neq L_{vj,vi}$。

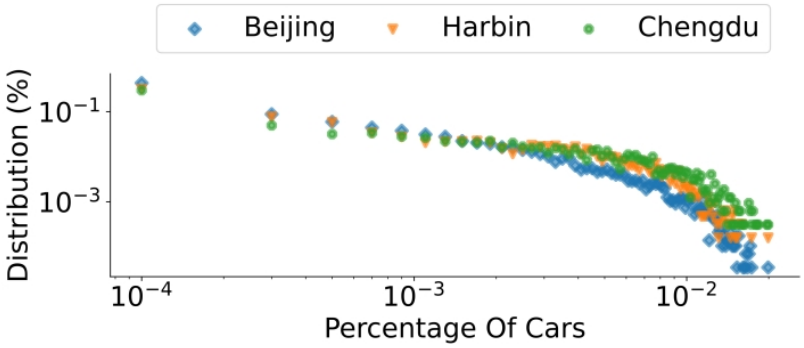


图 3. 北京、哈尔滨、成都道路汽车流量分布

节点的初始嵌入: 设 t 为当前时间，Δ 为实验正在训练的历史快照的数量。因此, 实验将 Δ 层的 GNN, k^{th}GNN 对应的快照时间 $t-\Delta+k, k \in (0, \Delta)$。所有节点的初始嵌入 $v \in V_{t-\Delta+k}$, 在 k^{th} GNN 被定义为:

$$\mathbf{h}_v^{0,k} = (\mathbf{w}_\tau^T \cdot \tau_{t-\Delta+k}^v) \parallel \mathbf{L}_v \tag{9}$$

$\mathbf{w}_\tau$ 是一种可习得的矢量传感器的值映射到一个高维空间。$\mathbf{L}_v$ 是 v 的位置嵌入。注意，本实验没有使用公式 1中 $\mathbf{h}_v^\ell$ 表示的堆栈索引，因为除了公式 9中与时间相关的输入外，所有计算都是相同的。在 k^{th} GNN 中 L^{th} 层之后的最终输出记为 $\mathbf{z}_v^k = \mathbf{h}_v^{L,k}$。

3.3 编码时间依赖性

为了对节点表示上的长期时间依赖性进行编码，本实验使用 LSTM 编码器。具体来说，k^{th} GNN 的最后一层输出，表示为 $\mathbf{z}_v^k$，被馈送到 k^{th} LSTM 堆栈。此外，k^{th} LSTM 还接收到 $(k-1)^{th}$ LSTM 的隐藏状态 (参见图 2)。在数学上，v 节点上的 k^{th} LSTM 的输出 (记为 $\mathbf{s}_v^k \in \mathbb{R}^{d_{Lenc}}$) 计算如下:

$$\mathbf{s}_v^k = \begin{cases} LSTM_{Enc}(\mathbf{s}_v^{k-1}, \mathbf{z}_v^k) & \text{if } k > 1 \\ LSTM_{Enc}(MLP_1(t, \mathbf{z}_v^k), \mathbf{z}_v^k) & \text{if } k = 1 \end{cases} \tag{10}$$

由于 s_v^0 在第一个 LSTM，也就是 k=1，是未定义的，本实验通过 Mlp_1 使它成为可学习的向量。最终 LSTM 堆栈的输出对应于当前时间 t(见图 2)，输入到解码器中。d_{Lenc} 是 LSTM 表示的维度。

Algorithm 1 FRIGATE forward pass

Require: FRIGATE, stream $\overrightarrow{\mathcal{G}}$, target node v, current timestamp t, prediction horizon Δ
Output: Predicted sensor values $\hat{y}_{t+1}^{v} \dots \hat{y}_{t+\Delta}^{v}$

$\mathbf{s}_v^0 \leftarrow \text{MLP}_1(t, \mathbf{z}_v^1)$
$k \leftarrow 1$
for all $\mathcal{G} \in \overrightarrow{\mathcal{G}}_{[t-\Delta,t]}$ **do**
    $\mathbf{z}_v^k \leftarrow \text{GNN}(v)$
    $\mathbf{s}_v^k \leftarrow \text{LSTM}_{\text{Enc}}(\mathbf{s}_v^{k-1}, \mathbf{z}_v^k)$
    $k \leftarrow k+1$
$\mathbf{m}_v^t \leftarrow \text{MLP}_3\left(moments\left(\left\{\tau_u^{t'} \neq \varnothing | t' \in [t-\Delta, t], (u,v) \text{ or } (v,u) \in \mathcal{E}^{t'}\right\}\right)\right)$
$\hat{y}_0 \leftarrow \text{MLP}_2(\mathbf{s}_v^{\Delta}, \mathbf{m}_v^t)$
$\mathbf{q}_v^0 \leftarrow \mathbf{s}_v^{\Delta}$
for all $k \in [1 \dots \Delta]$ **do**
    $\mathbf{q}_v^k \leftarrow \text{LSTM}_{\text{Dec}}(\mathbf{q}_v^{k-1}, \hat{y}_v^{k-1})$
    $\hat{y}_v^k \leftarrow \text{MLP}_4(\mathbf{q}_v^k, \mathbf{m}_v^t)$
Re-index $\hat{y}_k^v \mapsto \hat{y}_{t+k}^v$
return $\hat{y}_{t+1}^v, \dots, \hat{y}_{t+\Delta}^v$

代码 1 中的第 1 - 6 行总结了编码器组件。给定流 $\vec{\mathcal{G}}$ 、当前时间 t 和目标节点 v，提取子集 $\vec{\mathcal{G}}_{[t-\Delta,t]}$，其中 k 为 GNN-LSTM 堆栈数。过程从 $\vec{\mathcal{G}}_{t-\Delta}$ 开始，当 k^{th} GNN 堆栈将 v 嵌入 $\mathbf{z}_v^k$ 中。接下来，$\mathbf{z}_v^k$ 被传递给 k^{th} LSTM。这样就完成了一个计算。迭代地，本实验移动到 $(k+1)^{th}$ 堆栈，重复相同的操作直到 k=Δ，之后将公式 10中的 $\mathbf{s}_v^0$ 馈送到解码器。

3.4 解码器

解码器由一堆 Δ(预测范围)siamese LSTMs 组成。假设 t 为当前时间，则解码器中 k^{th} Lstm 的输出对应于时间 t+k 的预测传感器读数。每个 LSTM 接收两个输入:(1) 前一个时间戳中预测的传感器值，表示为 $\hat{y}_v^{k-1} \in \mathbb{R}$ ；(2) 前一个 LSTM 的隐藏状态，表示为 $\mathbf{q}_v^{k-1} \in \mathbb{R}^{d_{dec}}$ 。因此，k^{th} LSTM 的输出表示为:

$$\mathbf{q}_v^k = \begin{cases} LSTM_{Dec}(\hat{y}_v^{k-1}, \mathbf{q}_v^{k-1}) & \text{if } k > 1 \\ LSTM_{Dec}(MLP_2(\mathbf{s}_v^{\Delta}, \mathbf{m}_v^t), \mathbf{s}_v^{\Delta}) & \text{if } k = 1 \end{cases} \tag{11}$$

k=1 是一个特殊的情况，因为 $\hat{y}_v^0$ 和 $\mathbf{q}_v^0$ 都是未定义的。由于本实验假设该组节点仅存在部分感知节点，因此可能无法得到 τ_v^t，因此不能使用它来替换 $\hat{y}_v^0$。为了缓解这种情况，将 $\mathbf{q}_v^{k-1}$ 替换为编码器中最后一个 LSTM 的输出 $\mathbf{s}_v^{\Delta}$，并通过 Mlp_2 估计出 $\hat{y}_v^0$。该 MLP 以 $\mathbf{s}_v^{\Delta}$，以及观测到的传感器值在 v 邻域内的矩表示 m_v^t 作为输入。即,

$$\mathbf{m}_v^t = MLP_3(moments(\left\{\tau_u^{t'} \neq \emptyset \mid t' \in [t-\Delta, t], (u,v)\, or\, (v,u) \in \varepsilon^{t'}\right\})) \tag{12}$$

最后, 预测传感器值 $\forall k \in (1, \Delta)$，$\hat{y}_v^k$ 的计算公式为:

$$\hat{y}_v^k = MLP_4(\mathbf{q}_v^k, \mathbf{m}_v^t) \tag{13}$$

注意不是直接从 LSTM 隐藏状态预测传感器值 $\mathbf{q}_v^k$，本实验还增加了节点 v 时间序列的统计信息 $\mathbf{m}_v^t$。这为模型提供了强烈的归纳偏置并提高了其性能，之后将在消融实验中实证地证实这一点。代码 1 中的第 7-12 行总结了解码器中的计算。

4 复现细节

4.1 与已有开源代码对比

复现过程中参考了文章作者公开发表的源代码（见 https://github.com/idea-iitd/frigate），我在复现源码的基础上调整了一下网络结构，并加入了正则化，控制模型复杂度并防止过拟合。

4.2 实验环境搭建

- **计算引擎：**使用搭载 Intel Core i7-10700K CPU 以及 NVIDIA GeForce RTX 3090 GPU 的计算机，在 Windows 10 系统下进行实验。
- **参数设置：**所有的实验从一个小时的历史交通信息中预测一个小时的交通信息。具体来说，由于在数据集中两个快照之间的持续时间为 5 分钟，设置 $\Delta = 12$。之后还会模拟快照间隔时间不规律的情况，但即使这样，模型也有过去一小时的数据，并对未来一小时的交通状况进行预测。同时，实验使用 16 个锚节点来计算所有图的 Lipschitz 嵌入，且在 FRIGATE 中使用了 10 层 GNN。
- **评价指标：**采用平均绝对误差 (MAE) 作为评价指标。跨所有方法的多个度量 (越低越好)。本实验还对表 3 中的关键结果使用了 sMAPE 和 RMSE。此外，实验通过在不可见节点的度量分布上使用自举来报告平均值周围的 95% 置信区间。
- **训练设置：**本实验使用 70%-20%-10% 的训练-验证-测试分割进行训练。这种分割是在时间维度上的。因此，明确地说，在训练过程中，使用了所有已知节点上总时间序列的 70%。从节点和时间步的角度来看，验证和测试都是针对数据的不同部分。如果该模型的验证损失超过 15 个 epoch 没有改善，训练将停止。
- **基线：**将 DCRNN [12]、STGCN [28]、LocaleGN [10] 和 STNN [26] 作为实验的基线。通过移除节点嵌入矩阵或基于动态时间翘曲的图计算，使 GraphWavenet [24] 和 STGODE [4] 适应实验的设置，以便它们支持表 1中概述的三重目标。所有六个基线均获得了作者发布的官方代码库。

4.3 源码结构分析

model 文件夹下主要包含了 model、trainer、tester 三个文件，model 定义了 Frigate 模型的架构，包括 GNN（Graph Neural Network）、LSTM（Long Short-Term Memory）等组

件。trainer 包含了模型的训练逻辑，包括损失函数的定义和优化器的设置。tester 负责模型的测试逻辑，包括模型加载和性能评估。

utils 文件夹包含了一些通用的工具模块，如日志配置、邻居采样器的定义等。这些工具模块为整个工程提供了辅助功能。data_utils 和 test_data_utils 是数据处理工具模块。data_utils 主要包含了获取数据加载器和邻接矩阵的函数，而 test_data_utils 包含了生成测试数据集的相关函数。这两个模块为整个 Frigate 模型提供了数据基础。

train 文件是 Frigate 模型的训练脚本。在这个文件中，首先导入了必要的库和模块，包括处理命令行参数、PyTorch 模块、路径管理等。接着，定义了一个用于计算 MAE 损失的函数'masked_mae_loss'，该函数考虑了特定的遮罩，以适应模型的训练需求。随后，通过'get_run_num' 函数确定当前运行的编号，以便在记录日志和保存模型时使用。同时，通过'config_logging' 函数设置了日志记录，包括记录在文件中并处理标准错误流，以便后续分析和调试。主函数'main' 负责解析命令行参数，设置模型参数和数据加载器，并调用'model_train' 函数开始模型训练。在训练过程中，还会记录训练日志和相关信息。总体而言，train 文件是 Frigate 模型训练的入口，负责设置训练环境、加载数据、定义损失函数，并启动训练过程。

test 是 Frigate 模型的测试脚本，主要负责评估已训练模型的性能。该脚本通过解析命令行参数，配置测试所需的各项参数。其中，'–pred_file' 参数指定了用于测试的预测结果文件路径，通常是模型在验证集上的预测结果。在脚本开始执行时，首先加载命令行参数，包括测试所需的数据路径、模型名称、运行编号等。接着，调用'model_test' 函数，该函数负责加载事先训练好的模型，并对测试数据进行预测。这一过程涵盖了模型的加载、数据准备、预测和性能评估。

metric_calculation 文件用于计算模型测试结果的指标。通过解析命令行参数，读取预测结果文件（通常为 pred_true.npz），提取其中的真实值和预测值，然后计算 MAE（Mean Absolute Error）指标。计算完成后，输出 MAE 指标的数值，用于评估模型性能。

4.4 创新点

结合原工作，本工作现有的创新点如下：

- FRIGATE 是一个基于时空图神经网络的预测模型，能够在实际的道路网络中处理传感器部署不足、传感器历史数据不完整以及道路网络的动态变化等问题。
- FRIGATE 采用了一种新颖的组合方式，将位置、拓扑和时间信息融合到节点表示中，通过门控 Lipschitz 嵌入和 LSTMs 实现了多样化信息的联合融合。
- 与现有算法相比，FRIGATE 的图神经网络架构更具表达能力，能够在真实世界的网络约束交通数据上取得更好的实验性能。
- FRIGATE 对于传感器部署的节约、道路网络连接性的变化以及传感器数据的时间不规则性具有鲁棒性。
- 本工作改变源码中 GNN 的输入层数，尝试最佳的组合。
- 本工作在模型的优化器中添加正则化项，以控制模型复杂度并防止过拟合。

5 实验结果分析

在本节中，对本实验改进的 FRIGATE-PRO 和 FRIGATE 进行基准测试，并得出：

- **预测精度：**在道路网络的时空预测任务中，与其他 baseline 相比，FRIGATE-PRO 和 FRIGATE 更具优势。
- **消融实验：**通过广泛的消融研究，本实验阐明了 FRIGATE 每个部件的重要性，并提供了它们在整体性能中所起关键作用的见解。
- **鲁棒性：**本实验测试了本模型在节俭感知、对图结构修改和对快照的非均匀时间粒度的弹性方面的限制。

5.1 数据集

本实验使用了三个真实世界的数据集，这些数据集是通过跟踪出租车的 GPS 轨迹收集的。如表 1所示，这三个数据集分别对应北京、成都和哈尔滨 [6]。实验将原始轨迹与 Openstreetmap [18] 获得的相应道路网络进行地图匹配 [25]。将流量聚合为 5 分钟为一个间隔，传感器值对应于每 5 分钟间隔内通过每个节点的车辆数量。从整个节点集中，本实验均匀随机地选择一个节点子集作为安装了传感器的可见节点。推断和损失计算只在不可见的节点上进行。保留的默认节点百分比如表 2所示。

表 2. 数据集细节

数据集	节点数	边数	地理面积 (km2)	平均值	标准差	默认可见节点比例
Beijing [13]	28465	64478	16,411	3.22	9.09	5%
Chengdu	3193	7269	14,378	4.54	7.19	50%
Harbin [11]	6235	15205	53,068	77.85	124.11	30%

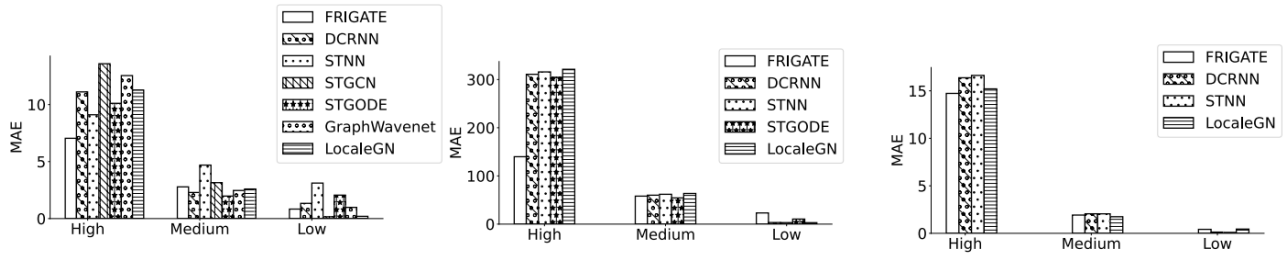
5.2 预测精度

在表 3中，本实验给出了由本实验改进的 FRIGATE-PRO、FRIGATE 和所有其他基线在所有三个数据集上获得的 MAE。实验观察到，FRIGATE-PRO 比 FRIGATE 表现稍好一点，而 FRIGATE 始终优于所有其他基线。平均而言，FRIGATE 的 MAE 比最接近的基线低 25% 以上。在其他基线中，LocaleGN 和 STGODE 表现最好，其次是 DCRNN，其次是 GraphWavenet，其次是 STNN，最后是 STGCN。进一步注意到，即使在 48 小时后，STGCN [28] 也未能在哈尔滨和北京进行完成训练 (STGCN 中评估的最大数据集有 1026 个节点，并且只考虑了工作日的流量)。同样，GraphWavenet [24] 也未能在哈尔滨和北京完成训练 (GraphWavenet 中评估的最大数据集有 325 个节点)。将结果的 MAE 与表 2中传感器值的标准差进行比较后，可以观察到 MAE 与标准差有明显的相关性，这解释了为什么哈尔滨的所有算法表现都相对较差。由于 STGCN 的性能明显较差，无法在大数据集上扩展，后续实验只考虑 DCRNN、STNN、STGODE 和 LocaleGN 作为基线。

为了进一步诊断基准测试技术之间的性能模式，本实验根据通过节点的汽车数量将节点分为三个区间，并绘制这三个区间中的错误分布。结果如图 4所示。这里，“高”、“中” 和 “低” 分别对应于前 33 个百分位数、33-66 个百分位数和后 33 个百分位数。从这个实验中可以得出了三个关键的见解。首先，在所有算法中，当处理较低流量的节点时，MAE 较小。这是很自然的，因为这些节点在流量方面变化很小。相比之下，高流量节点的波动更大，这取决于一天中的时间，以及工作日与周末等。其次，可以注意到 FRIGATE 在高频节点上的表现优于所有其他模型，在中频节点上表现良好，在低频节点上有时表现不如其他模型。总的来说，在高频和中频节点中做得更好更重要，因为它们处理了大部分的流量，而 FRIGATE 在这方面的表现大多优于其他技术。最后，DCRNN、STNN 和 STGODE 中性能不佳的一个关键原因是，由于大多数节点都是低频节点（如图 3所示），它们对高频节点进行了过度平滑，从而推断出较低的值。位置嵌入和通过门通进行的深层分层使 FRIGATE 避免了过度平滑。

表 3. 不同交通量预测方法的性能比较

模型	数据集	MAE	sMAPE	RMSE
FRIGATE-PRO	Chengdu	3.464 ± 0.20	130.25 ± 3.39	5.92 ± 0.31
FRIGATE		3.547 ± 0.20	131.17 ± 3.39	5.93 ± 0.31
STNN		5.633 ± 0.18	199.86 ± 0.00	9.50 ± 0.50
DCRNN		4.893 ± 0.26	138.17 ± 2.07	8.10 ± 0.48
LocaleGN		4.597 ± 0.22	192.80 ± 0.19	8.57 ± 0.33
STGODE		4.693 ± 0.19	147.12 ± 2.25	7.72 ± 0.39
STGCN		5.712 ± 0.33	198.38 ± 0.00	9.47 ± 0.50
GraphWavenet		5.308 ± 0.30	147.77 ± 1.81	8.89 ± 0.43
FRIGATE-PRO	Harbin	73.045 ± 2.04	92.47 ± 1.89	103.89 ± 3.51
FRIGATE		73.559 ± 2.04	93.00 ± 1.89	105.01 ± 3.51
STNN		126.305 ± 4.54	199.68 ± 0.02	206.91 ± 6.93
DCRNN		124.109 ± 5.06	185.73 ± 0.68	204.75 ± 6.36
LocaleGN		128.674 ± 5.10	169.54 ± 1.43	207.36 ± 6.58
STGODE		122.493 ± 4.97	152.28 ± 1.27	198.81 ± 6.74
STGCN		OOT	OOT	OOT
GraphWavenet		OOM	OOM	OOM
FRIGATE-PRO	Beijing	5.588 ± 0.11	169.54 ± 0.45	11.84 ± 0.30
FRIGATE		5.651 ± 0.11	169.73 ± 0.45	11.87 ± 0.30
STNN		6.215 ± 0.12	199.99 ± 0.01	12.82 ± 0.27
DCRNN		6.122 ± 0.14	195.56 ± 0.15	12.63 ± 0.24
LocaleGN		5.759 ± 0.13	172.73 ± 0.33	12.10 ± 0.03
STGODE		OOM	OOM	OOM
STGCN		OOT	OOT	OOT
GraphWavenet		OOM	OOM	OOM

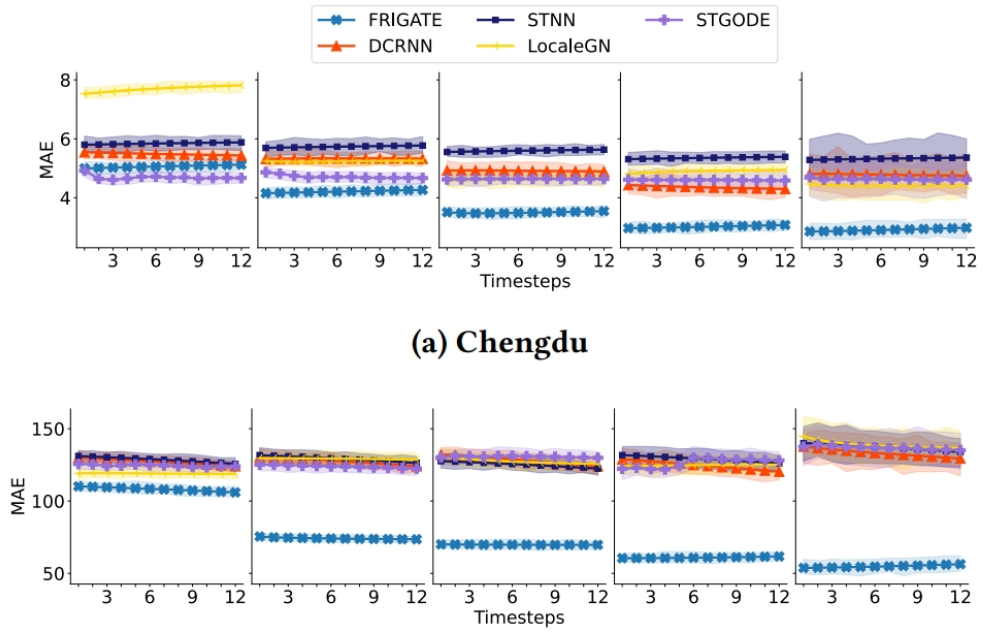


(a) Chengdu 50%　　(b) Harbin 30%　　(c) Beijing 5%

图 4. 比较通过车辆频率分组的节点上的误差分布

5.3 可见节点数量的影响

与任何机器学习任务一样，本实验希望看到的节点数量越多，准确率越高。这增加了训练数据，也增加了不可见节点接近可见节点的可能性。为了实现这一目标，实验将模型可见节点数量在各自道路网络中节点总数的比例从 10% 改变至 90%，并根据预测范围测量 MAE。结果如图 5所示。可以观察到，FRIGATE 表现明显更好，因为更多的信息先验。此外，其他基线需要更多的数据才能达到相同的性能。对于成都，在 50% 可见节点的图上训练的 DCRNN 打败了本工作只在 10% 可见节点的图上训练的模型，而在哈尔滨，没有一个模型甚至超过了本工作只在 10% 可见节点的图上训练的模型。



(a) Chengdu

(b) Harbin

图 5. 从左到右逐渐增加看到的节点的比例

随着数据的增加，本模型和基线之间的差距也在增加。这种数据的经济使用而不失去归纳能力是 FRIGATE 所包含的理想属性。可以注意到的另一个有趣的趋势是，从 70% 到 90% 的基线性能略有恶化。很难找出确切的原因。可以假设有两个可能的原因。

首先，随着训练数据量的增加，过度拟合的可能性更大。此外，可以注意到，由于测试节点的样本量减少，置信区间随着看到节点的百分比的增加而扩大。这种趋势与统计理论是一致的，这意味着在低容量的测试集上，结果具有更高的可变性。FRIGATE 不受这种趋势的影响，这意味着它不会过拟合。FRIGATE 利用矩作为一个强大的归纳偏差，减少了过拟合风险。

5.4 消融实验

在消融研究中，本实验系统地关闭了 FRIGATE 的各个组件，测量了其对 MAE 的影响。图 6(a) 为成都和哈尔滨的结果。

- **门控：**在这里，实验用 GraphSAGE 层 [5] 代替门控卷积，保留架构的其余部分不变。可以看到 MAE 明显增加，表明门控卷积层的重要性。
- **位置嵌入：**为了理解其对性能的效用，本实验将 Lipschitz 嵌入作为节点的特征，从而将其作为门控计算的输入删除。可以看到了结果的显著下降，成都 Lipschitz 的降幅最大。这一结果从实践上证明了位置嵌入在道路网络时间序列预测中的价值。
- **In-out 聚合：**在 FRIGATE 中分别聚合来自传入和传出邻居的消息。在这里，仅衡量聚合成单个向量的影响。可以观察到 MAE 的增加，显示了对基于方向的聚合的需求。
- **力矩：**本实验从模型中删除矩，并保持架构中的其他所有内容不变。可以在两个数据集上都看到性能的大幅下降，其中哈尔滨的表现更为明显。这就确立了拥有信息性先验的重要性。
- **GNN 层数：**为了了解 GNN 层数对 FRIGATE 性能的影响，本实验在成都 50% 可见节点的数据集上训练时改变了这个参数。如图 6(b) 所示，性能在 10 处达到饱和，本实验将其作为默认值。

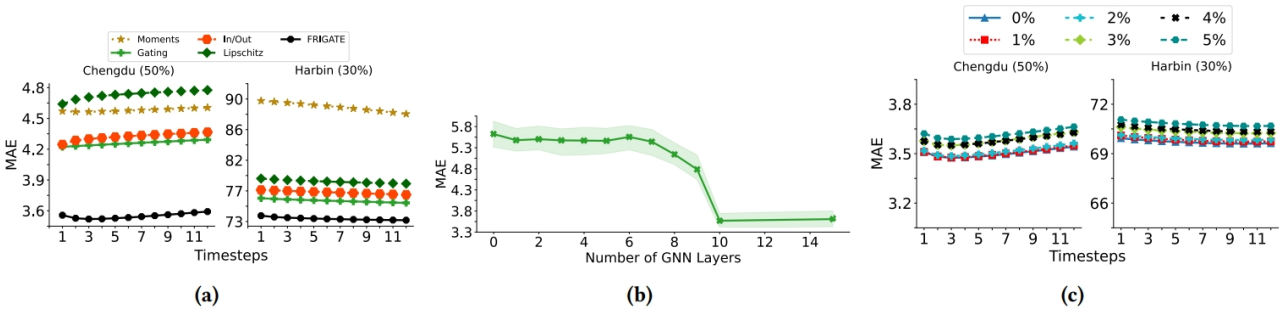


图 6. (a) 消融研究 (b)GNN 层变化对成都 50% 可见节点的影响 (c) 模型对道路网络拓扑推理时间变化的弹性

5.5 鲁棒性

在本节中，本实验分析了 FRIGATE 对网络拓扑变化和不规则拓扑感知的鲁棒性和弹性。在这些实验中，实验在原始数据集上训练 FRIGATE，然后稍微变化拓扑以及时间粒度。本工作在这个扰动数据集上评估推理性能。注意，本实验不会在扰动后重新训练模型。

- **拓扑变化：**为了模拟现实世界中道路网络可能因路障或新道路的开通而发生变化的情况，首先假设要引入 $X\%$ 的扰动量，先通过随机删除最大 $\frac{X}{2}\%$ 的边来改变网络。此外，实验在距离在阈值内的未连接节点之间创建相等数量的边。这些边缘的距离从原始边缘距离分布中采样，并进行替换。然后本实验改变 X 并测量对 MAE 的影响。结果如图 6(c) 所示，FRIGATE 对道路网络变化是有适应性的，准确性下降的幅度很小。

- **不规则时间感知：**在这个实验中，实验试图通过在测试时改变时间步长粒度来测试本模型。本实验在 $\vec{\mathcal{G}}_{[t-\Delta,t]}$ 中随机丢弃了 33.33% 的快照，有效地减少了时间序列长度，并改变了捕获数据的粒度。结果列在表 4中，可以观察到 MAE 的增加可以忽略不计。结果表明，该模型对快照粒度的变化具有弹性。

表 4. 不规则时间粒度对 MAE 的影响

时间序列	数据集	MAE
Regular (original)	Chengdu (50%)	3.547 ± 0.23
Irregular (perturbed)	Chengdu (50%)	3.582 ± 0.17
Regular (original)	Harbin (50%)	69.700 ± 3.21
Irregular (perturbed)	Harbin (50%)	69.834 ± 2.78

6 总结与展望

本文提出了一种新的时空 GNN，称为 FRIGATE，用于预测道路网络上的网络约束时间序列过程。本工作通过对真实世界交通数据集的实验表明，FRIGATE 的性能明显优于现有的基线。此外，FRIGATE 对一些实际挑战具有鲁棒性，例如跨节点的部分感知，适应道路网络拓扑的微小更新以及对不规则时间感知的弹性。FRIGATE 的核心竞争力源于其新颖的设计，该设计结合了 Lipschitz 嵌入来编码位置，用 Sigmoid 门控来学习信息重要性，确保了节点间的远程通信，以及信息的定向聚合，和时间序列分布矩形式的强先验。总的来说，FRIGATE-PRO 和 FRIGATE 是使我们更接近实际工作的可部署技术。

在本次复现过程中，实现了文章中提出的方法，并在 FRIGATE 的基础上改变了网络层数，又加入了正则化。但是训练时间有些长，时间成本比较高，未来可以通过改进减短训练时长。

未来的工作中，可以在此文的基础上进行到达时间预测，开发更多应用领域。也可以把对未来的路网流量预测与一些地图应用用户处收集到的实时交通状况结合，由此对未来短时间的交通流量进行更精确的预测。

参考文献

[1] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems*, 33:17804–17815, 2020.

[2] Yuzhou Chen, Ignacio Segovia, and Yulia R Gel. Z-gcnets: Time zigzags at graph convolutional networks for time series forecasting. In *International Conference on Machine Learning*, pages 1684–1694. PMLR, 2021.

[3] Vijay Prakash Dwivedi, Ladislav Rampášek, Michael Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long range graph benchmark. *Advances in Neural Information Processing Systems*, 35:22326–22340, 2022.

[4] Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. Spatial-temporal graph ode networks for traffic flow forecasting. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 364–373, 2021.

[5] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[6] Jayant Jain, Vrittika Bagadia, Sahil Manchanda, and Sayan Ranu. Neuromlr: Robust & reliable route recommendation on road networks. *Advances in Neural Information Processing Systems*, 34:22070–22082, 2021.

[7] Manas Joshi, Arshdeep Singh, Sayan Ranu, Amitabha Bagchi, Priyank Karia, and Puneet Kala. Batching and matching for food delivery in dynamic road networks. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 2099–2104. IEEE, 2021.

[8] Shiyong Lan, Yitong Ma, Weikang Huang, Wenwu Wang, Hongyu Yang, and Pyang Li. Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In *International conference on machine learning*, pages 11906–11917. PMLR, 2022.

[9] Mengzhang Li and Zhanxing Zhu. Spatial-temporal fusion graph neural networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4189–4196, 2021.

[10] Mingxi Li, Yihong Tang, and Wei Ma. Few-sample traffic prediction with graph networks using locale as relational inductive biases. *IEEE Transactions on Intelligent Transportation Systems*, 24(2):1894–1908, 2022.

[11] Xiucheng Li, Gao Cong, Aixin Sun, and Yun Cheng. Learning travel time distributions with deep generative model. In *The World Wide Web Conference*, pages 1017–1027, 2019.

[12] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

[13] Jing Lian and Lin Zhang. One-month beijing taxi gps trajectory dataset with taxi ids and vehicle status. In *Proceedings of the First Workshop on Data Acquisition To Analysis*, pages 3–4, 2018.

[14] Dachuan Liu, Jin Wang, Shuo Shang, and Peng Han. Msdr: Multi-step dependency relation networks for spatial temporal forecasting. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1042–1050, 2022.

[15] Sourav Medya, Sayan Ranu, Jithin Vachery, and Ambuj Singh. Noticeable network delay minimization via node upgrades. *Proceedings of the VLDB Endowment*, 11(9):988–1001, 2018.

[16] Shubhadip Mitra, Sayan Ranu, Vinay Kolar, Aditya Telang, Arnab Bhattacharya, Ravi Kokku, and Sriram Raghavan. Trajectory aware macro-cell planning for mobile users. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 792–800. IEEE, 2015.

[17] Shubhadip Mitra, Priya Saraf, Richa Sharma, Arnab Bhattacharya, Sayan Ranu, and Harsh Bhandari. Netclus: A scalable framework for locating top-k sites for placement of trajectory-aware services. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pages 87–90. IEEE, 2017.

[18] Contributers OpenStreetMap. Planet dump retrieved from https://planet. osm. org, 2017.

[19] Shiladitya Pande, Sayan Ranu, and Arnab Bhattacharya. Skygraph: Retrieving regions of interest using skyline subgraph queries. *Proceedings of the VLDB Endowment*, 10(11):1382–1393, 2017.

[20] Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 914–921, 2020.

[21] Pengkun Wang, Chaochao Zhu, Xu Wang, Zhengyang Zhou, Guang Wang, and Yang Wang. Inferring intersection traffic patterns with sparse video surveillance information: An st-gan method. *IEEE Transactions on Vehicular Technology*, 71(9):9840–9852, 2022.

[22] Yang Wang, Yiwei Xiao, Xike Xie, Ruoyu Chen, and Hengchang Liu. Real-time traffic pattern analysis and inference with sparse video surveillance information. In *IJCAI*, pages 3571–3577, 2018.

[23] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 753–763, 2020.

[24] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*, 2019.

[25] Can Yang and Gyozo Gidofalvi. Fast map matching, an algorithm integrating hidden markov model with precomputation. *International Journal of Geographical Information Science*, 32(3):547–570, 2018.

[26] Song Yang, Jiamou Liu, and Kaiqi Zhao. Space meets time: Local spacetime neural network for traffic flow forecasting. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 817–826. IEEE, 2021.

[27] Pranali Yawalkar and Sayan Ranu. Route recommendations on road networks for arbitrary user preference functions. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 602–613. IEEE, 2019.

[28] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

[29] Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. Gman: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 1234–1241, 2020.

[30] Zhengyang Zhou, Kuo Yang, Wei Sun, Binwu Wang, Min Zhou, Yunan Zong, and Yang Wang. Towards learning in grey spatiotemporal systems: A prophet to non-consecutive spatiotemporal dynamics. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pages 190–198. SIAM, 2023.