联邦学习在时空领域的应用

· 时空AI研讨会 ·

韩 程 凯

目录

CONTENTS

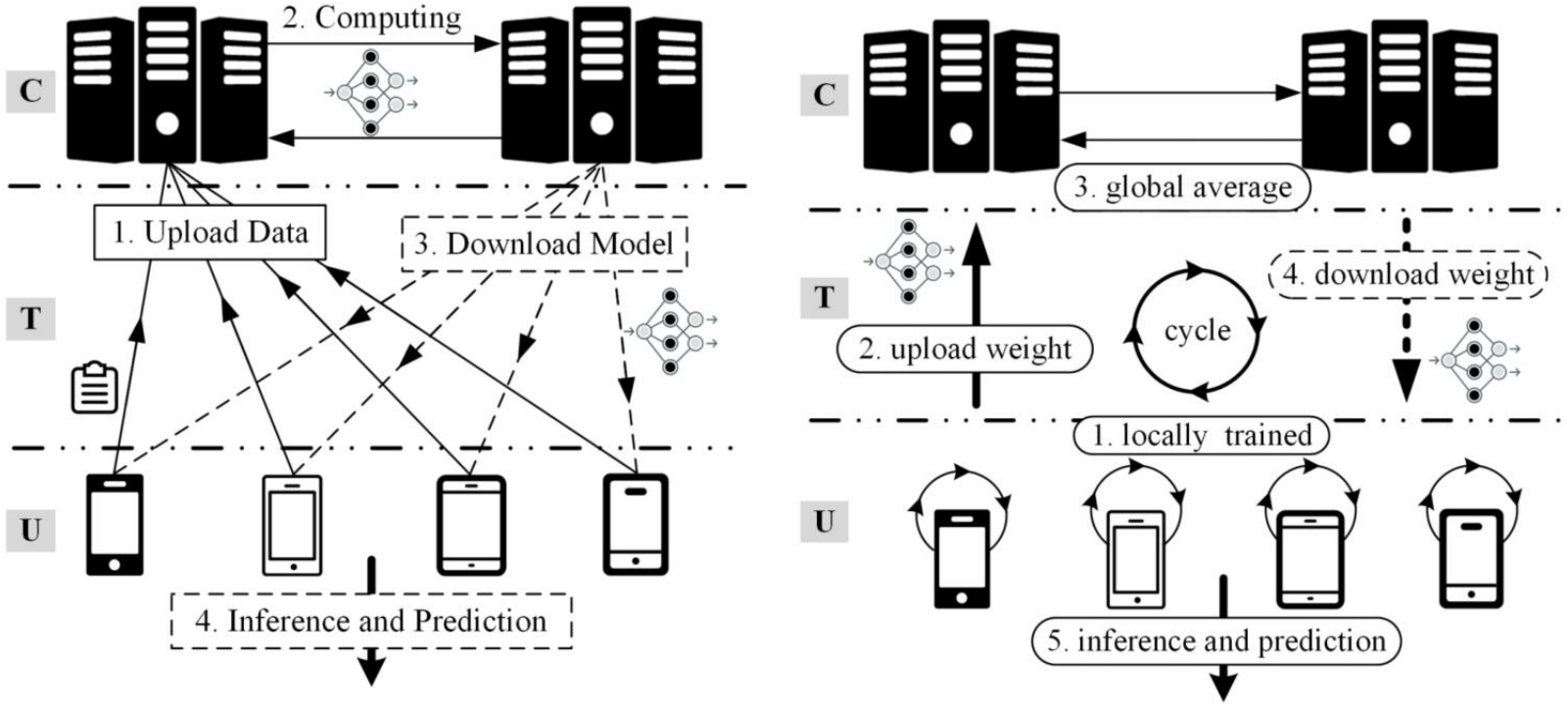
联邦学习背景介绍

数据孤岛问题

- 个体数据量少，数据质量差；但由于隐私保护问题，数据之间不能直接交换，导致存在大量数据孤岛。

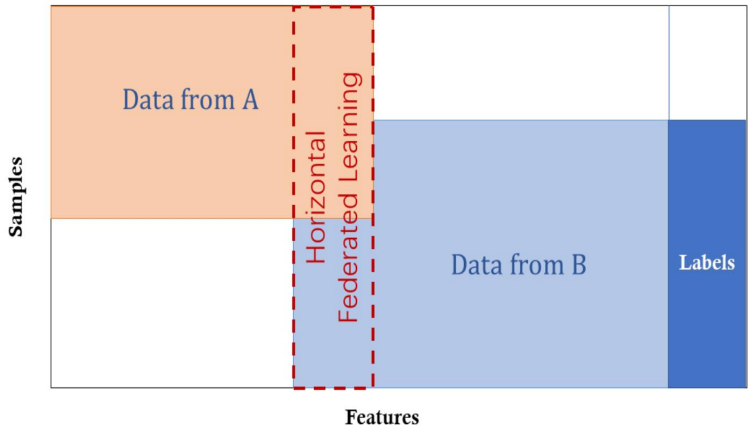
联邦学习（分布式机器学习）

- 目标是在保证数据隐私安全及合法合规的基础上，实现共同建模，提升AI模型的效果；
- 最早在2016年由谷歌提出，用于解决安卓手机终端用户在本地更新模型的问题。



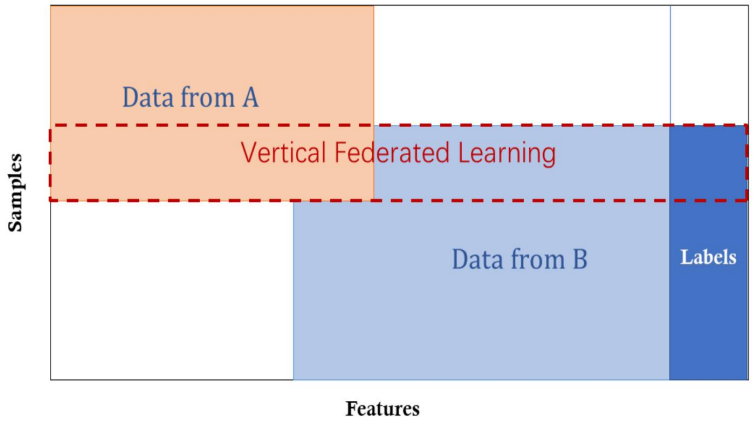
联邦学习分类

横向联邦学习



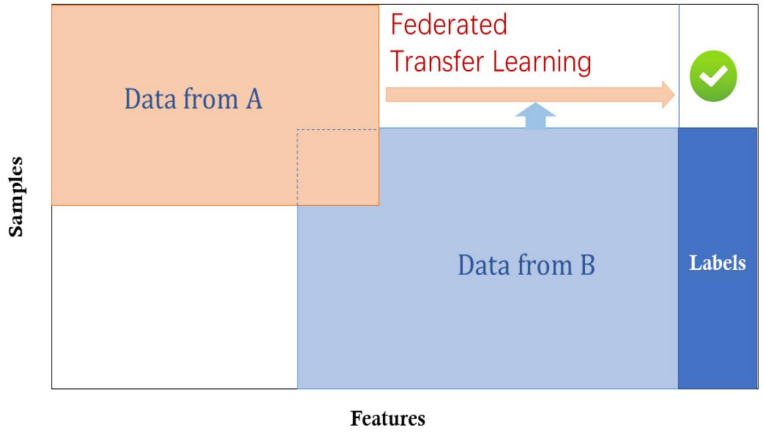
- 业态相同或相似
- 特征重叠多，用户重叠少
- **样本联合**

纵向联邦学习



- 触达用户相同或相似
- 特征重叠少，用户重叠多
- **特征联合**

联邦迁移学习



- 业态、用户均交集较少
- 特征重叠少，用户重叠少
- **迁移学习**

轨迹联邦学习——[UbiComp20]PMF: A Privacy-preserving Human Mobility Prediction Framework via Federated Learning

传统轨迹预测方法由于数据上传和集中模型训练的过程，面临着严重的隐私问题

- 分布式训练的联邦学习技术可用于隐私敏感问题的建模。

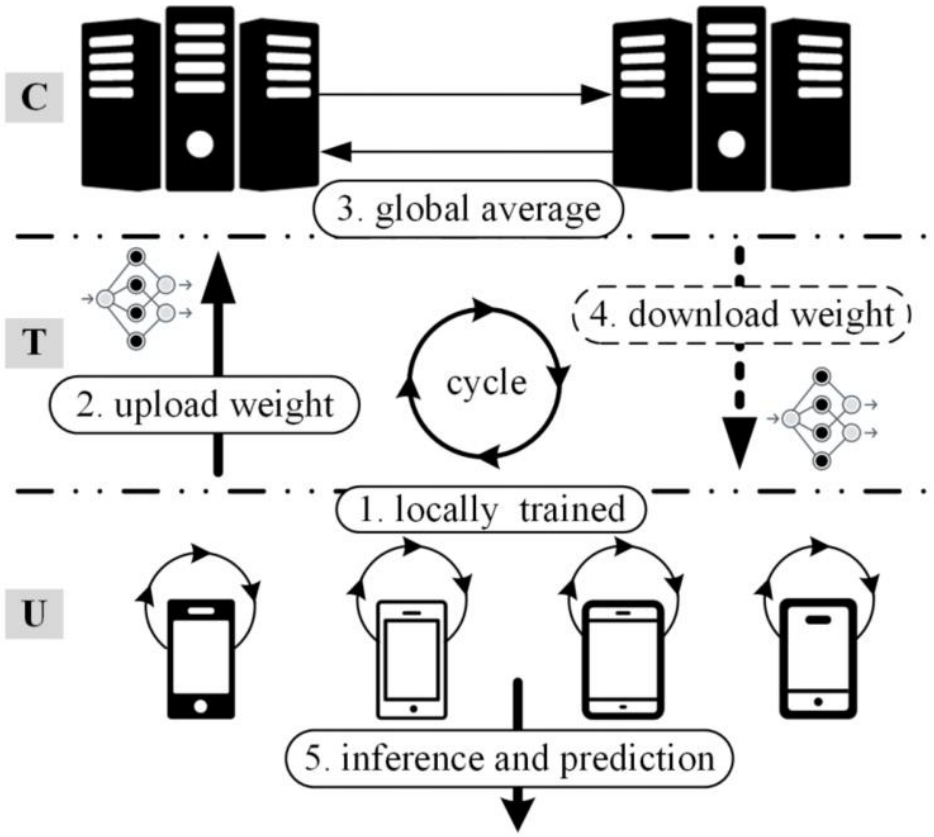
联邦学习直接应用于轨迹预测任务仍可能会泄露个人的隐私信息

- 外部黑客可以通过分析模型权重的变化来攻击上传过程，以获取特定个人的隐私信息；
- 中央训练服务器的所有者也有机会做类似的事情来获取个人信息。

隐私保护方法会导致轨迹预测任务性能显著下降

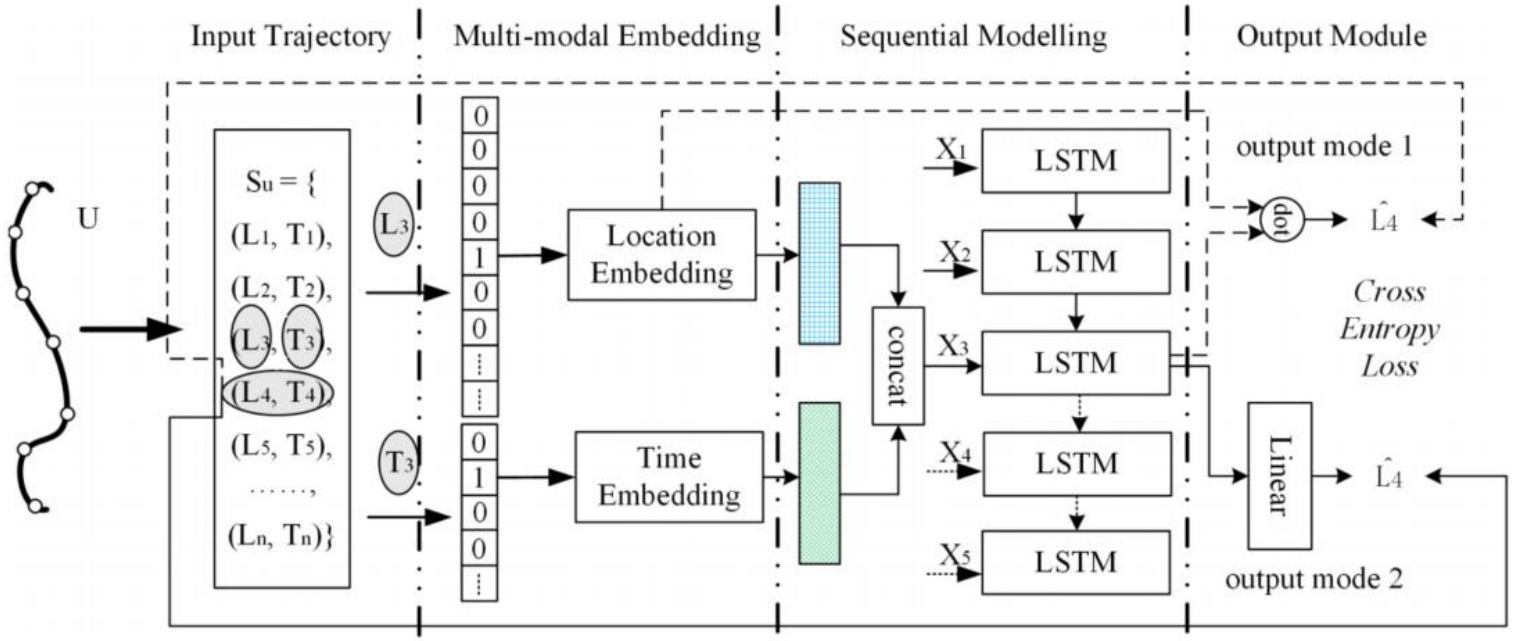
- 一些工作进一步采用差分隐私等方法保护隐私，但是它们会破坏原始数据，影响模型性能。

Privacy-preserving Mobility prediction framework via Federated learning

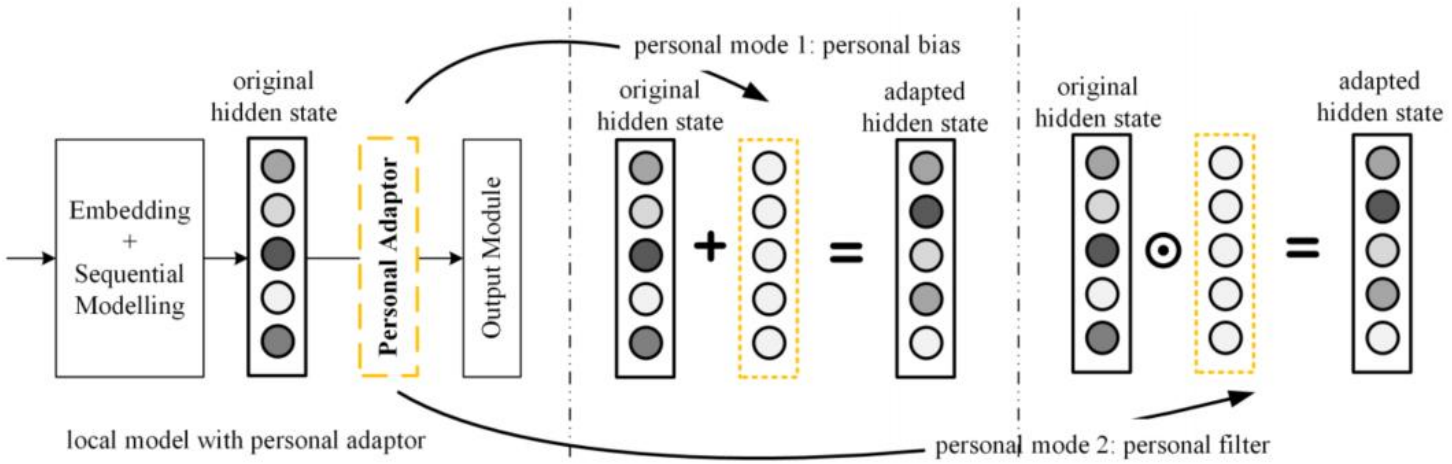


1. 每个设备仅用自身数据本地训练移动模型
2. 候选设备将训练的移动模型上传到云服务器
3. 云服务器通过聚合接收到的各种局部模型来生成全局模型
4. 云服务器选择候选移动设备来分发更新的全局模型，以重复上述4个步骤，直到满足停止标准
5. 移动设备下载最佳模型以推断和预测移动行为

轨迹预测模型

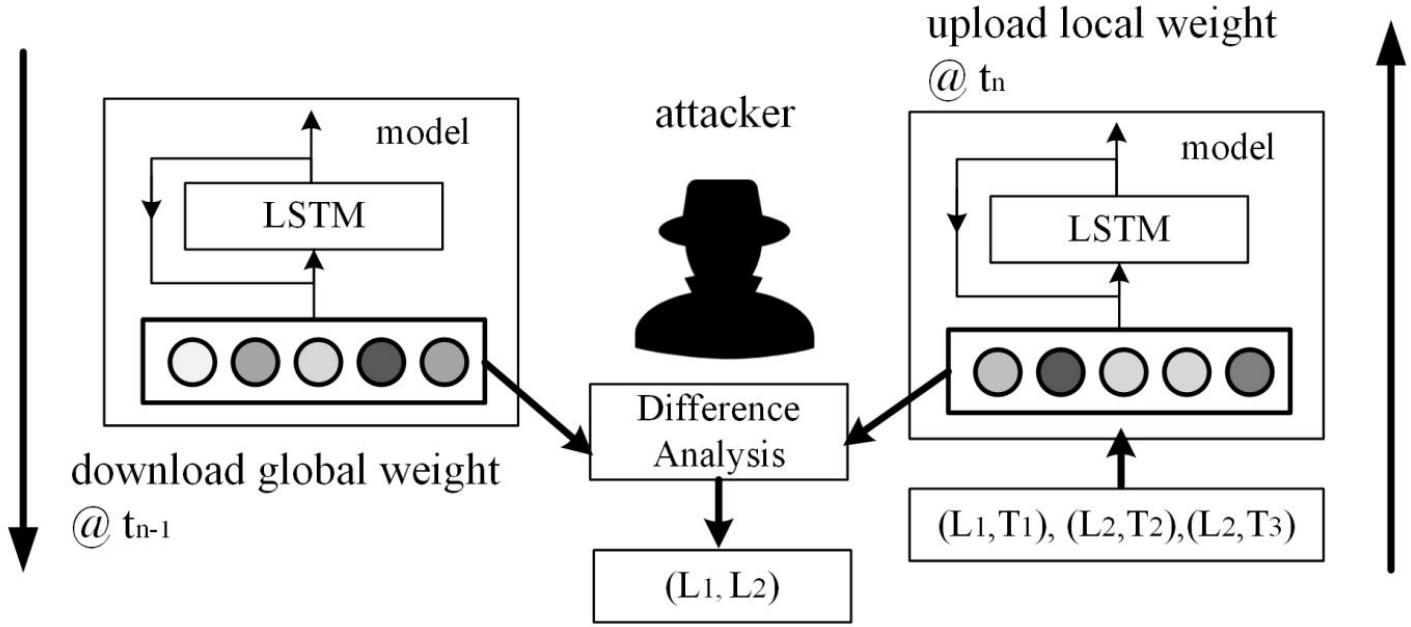


Personal Adaptor



隐私保护优化机制

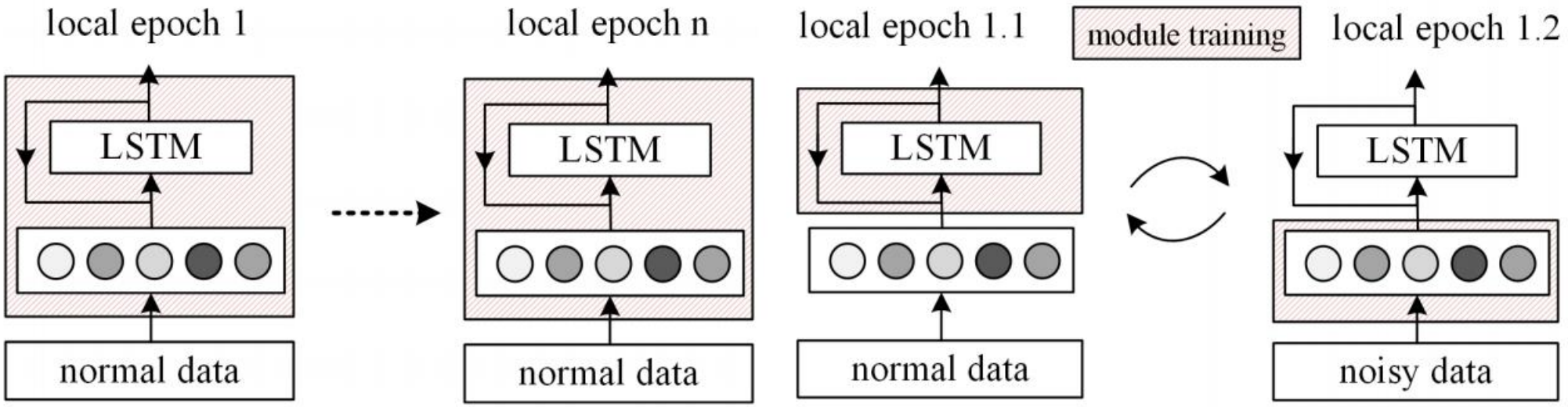
一个简单的攻击案例



- 用户在t_{n-1}时刻从云服务器接收到全局模型A；
- 以模型A作为出发点，用户在其自己私人移动轨迹上训练后，在t_n时刻得到一个局部模型B；
- 通过比较全局模型A和局部模型B之间位置嵌入表之间的差别，攻击者可以轻易地推断出目标用户访问过哪些地点。

隐私保护优化机制

Group Optimization on the Local Devices with Differential Privacy



(a) Conventional local optimization.　　(b) Iteratively group optimization.

(1) 固定location embedding模块，用**正常数据**训练上面模块；

(2) 固定上面模块的同时，用**噪声数据**训练location embedding模块；

(3) 重复以上两个步骤。

$$D_\epsilon(l,p) = \frac{\epsilon^2}{2\pi} e^{-\epsilon d_S(l,p)}, d_S(l,p) = d_{euclidean}(l,p)$$

将每个真实位置按照概率密度函数映射到随机生成的位置

隐私保护优化机制

Global Optimization on the Centralized Server

Algorithm 1: Training procedure of the proposed system

Global parameters: M^t is the global model at t step; K the number of selected clients; α is the learning rate.
Local parameters: m^k is the local model of client k; p^k is the personal adaptor for client k; *epoch* is the number of local epoch; $\mathcal{D}$ is the private data on the local device; ϵ is the parameter of differential privacy.

Server:
initialize M^0
for *round* $t \in \{1, 2, ...\}$ **do**
  construct/update client candidates pool
  select clients set from candidates pool by polling scheme
  for *client* $k \in \{1, 2, ..., K\}$ ***in parallel*** **do**
    $(m^k, \beta^k) \leftarrow$ **Client**$(M^t, \text{epoch}, \alpha)$
  normalize $\beta^k \in \{1, 2, ..., K\}$
  $M^t \leftarrow \sum_{k=1}^{K} \beta^k m^k$

Client:
// construct noisy data with differential privacy ϵ
$\mathcal{D}_\epsilon \xleftarrow{+noise} \mathcal{D}$
for $i \in \{1, 2, ..., epoch\}$ **do**
  // train the risky module group r of model m with protected data
  $m_r \leftarrow m_r - \alpha \nabla l(m; \mathcal{D}_\epsilon)$
  // train the normal module group n of model m with normal data
  $m_n \leftarrow m_n - \alpha \nabla l(m; \mathcal{D})$
return (m, l)
Fine-tune personal adaptor p **on clients before inference.**

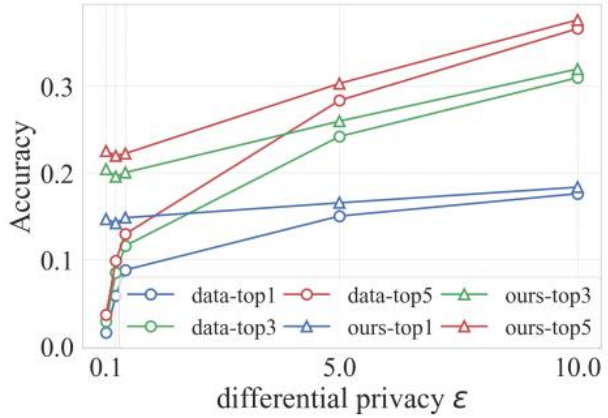
性能对比实验

Personal Model：Personal Model直接在移动设备中执行，**仅使用本地私人数据。**在不共享私人数据的情况下，个人模式**保护了个人隐私**，但**无法提供具有竞争力的性能。**

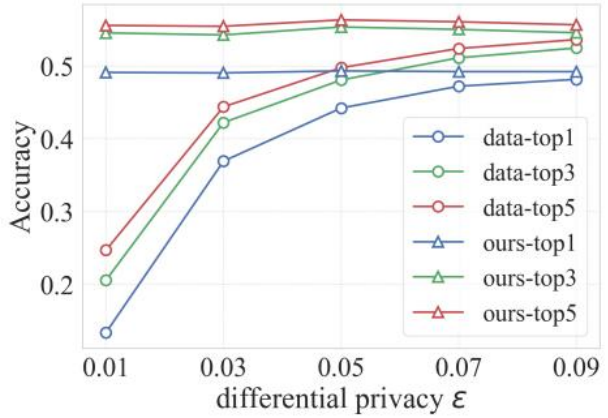
Joint Model：通过**共享来自不同用户的数据**，Joint Model成功地捕获了通用的移动性模式，并**实现了更好的建模性能。**但Joint Model**没考虑隐私问题。**

Privacy-Level	Methods	Top-1@Foursquare	Improv.	Top-1@Twitter	Improv.	Top-1@DenseGPS	Improv.
Personal Model (privacy-preserving)	P-Markov	0.151±0.000	-7.93%	0.345±0.000	-10.62%	0.618±0.000	-0.80%
	HMM	0.164±0.023	0	0.386±0.003	0	0.623±0.005	0
	LSTM	0.135±0.002	-17.68%	0.371±0.005	-3.89%	0.617±0.002	-0.96%
Joint Model (privacy-leakage)	J-Markov	0.186±0.000	+13.41%	0.441±0.000	+14.25%	0.694±0.000	+11.40%
	FPMC	0.195±0.001	+18.90%	0.460±0.003	+19.17%	0.570±0.000	-8.51%
	LSTM	0.217±0.002	+32.32%	0.497±0.001	+28.76%	0.718±0.001	+15.25%
	DeepMove	0.232±0.000	+41.46%	0.498±0.001	+29.01%	0.714±0.001	+14.07%
Our Model (privacy-preserving)	Our Basic Model	0.209±0.001	+27.44%	0.489±0.000	+26.68%	0.694±0.000	+11.40%
	+Personal Adaptor	0.213±0.001	+29.88%	0.495±0.001	+28.24%	0.715±0.000	+14.77%

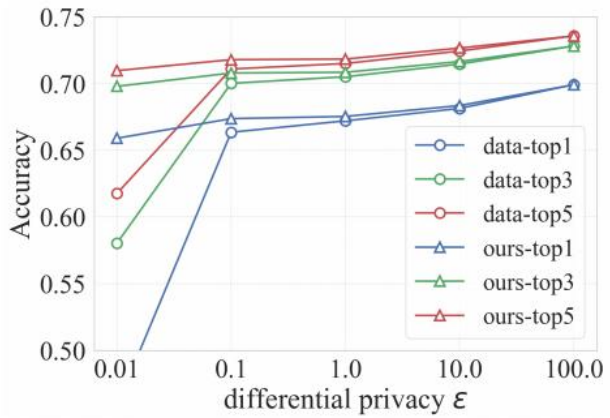
隐私风险分析



(a) Results on Foursquare dataset.



(b) Results on Twitter dataset.



(c) Results on DenseGPS dataset.

和**直接使用差分隐私机制保护数据进行训练的传统方法**进行对比：

- 当差分隐私参数ε变小时，传统方法的性能**显著下降**；
- 当通过使用较小的ε来严格差分隐私要求时，作者方法的性能**缓慢下降。**

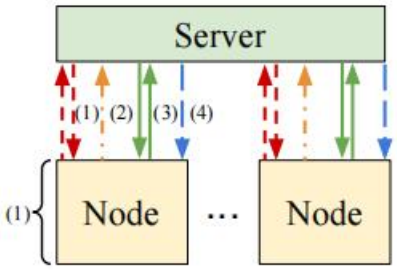
交通联邦学习——[KDD21]Cross-Node Federated Graph Neural Network for Spatio-Temporal Data Modeling

- 尽管联邦学习最近的工作提供了一种用于在多个设备上训练具有分散数据的模型的解决方案，它们**没有考虑内在的时空依赖性**，或**仅通过将正则化中的图结构强加在模型权重上来对其进行隐式建模。**
  - 基于正则化的方法由于**图仅对节点的相似性进行编码**的假设受到限制，并且不能在训练期间仅观察到一部分设备这种设定下使用。
  - 需要一种时空数据建模架构，该架构能够在边缘进行可靠的计算，同时保持数据分散。

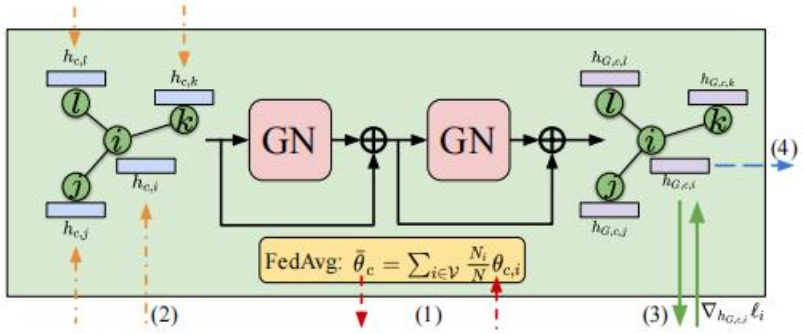
问题定义

- 给定图G，特征张量X和标签张量Y的数据集，X为输入，Y为预测目标。考虑在**跨节点联邦学习约束**下学习模型：**节点特征x_i，节点标签y_i和模型输出$\hat{y}_i$仅对节点i可见。**

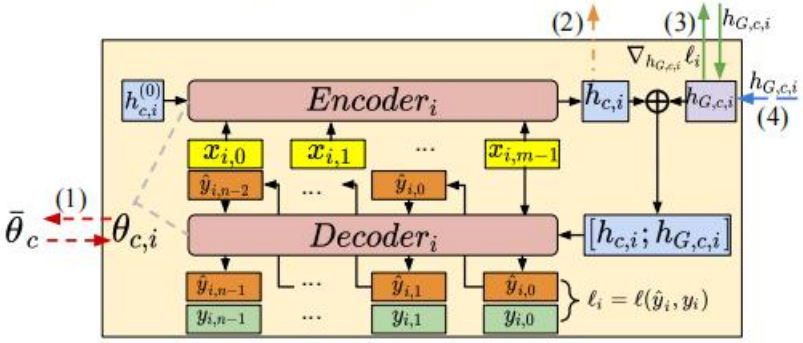
Cross-Node Federated Graph Neural Network



(a) Overview of the training procedure.



(b) Server-side Graph Network (GN).



(c) Encoder-decoder on the i-th node.

(a) 将节点级时间动态建模和服务器级空间动态建模分离开来。

(b) 在中央服务器上，图网络传播提取的节点时间特征并输出节点嵌入，其中包含节点之间的关系信息。只需要上传和下载特征和梯度，而不是节点上的原始数据。

(c) 在每个节点上，编码器-解码器模型从节点上的数据中提取时间特征并进行预测。可以访问不可共享的节点数据，并在本地每个节点上执行。

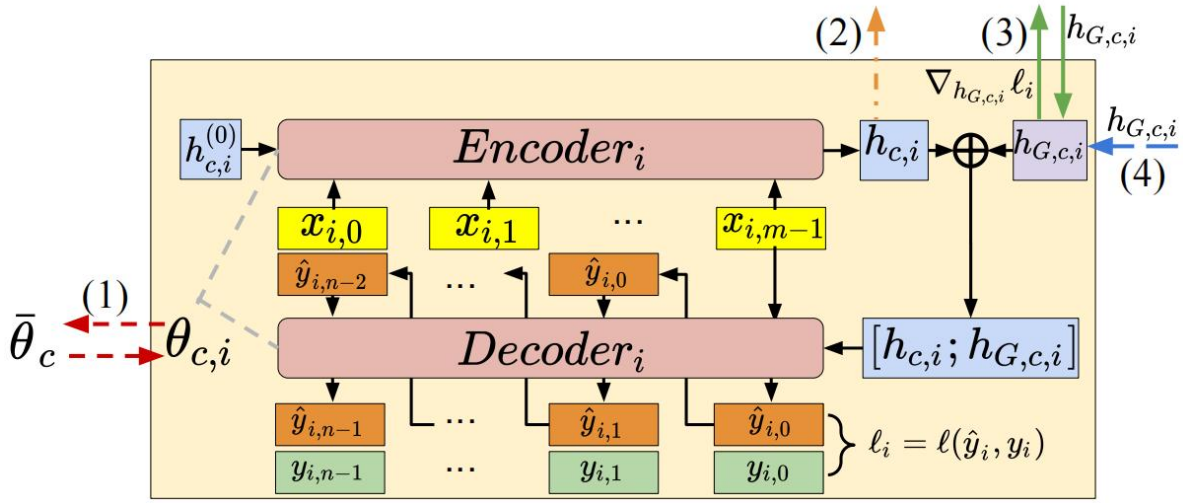
节点级时间动态性建模

- 使用基于GRU的编码器-解码器架构对每个节点上的节点级时间动态性进行建模。给定第i个节点上的输入序列x_i，编码器顺序读取整个序列并输出隐藏状态$h_{c,i}$作为输入序列的总结。

$$\boldsymbol{h}_{c,i} = Encoder_i(\boldsymbol{x}_i, \boldsymbol{h}_{c,i}^{(0)})$$

- 将$h_{c,i}$和中央服务器生成的节点嵌入$h_{G,c,i}$（包含空间信息）拼接起来作为解码器的初始状态向量。解码器从输入序列的最后一帧$x_{i,m}$开始，使用拼接的隐藏状态向量，以自回归的方式生成预测$\hat{y}_i$。

$$\hat{\boldsymbol{y}}_i = Decoder_i(x_{i,m}, [\boldsymbol{h}_{c,i}; \boldsymbol{h}_{G,c,i}])$$



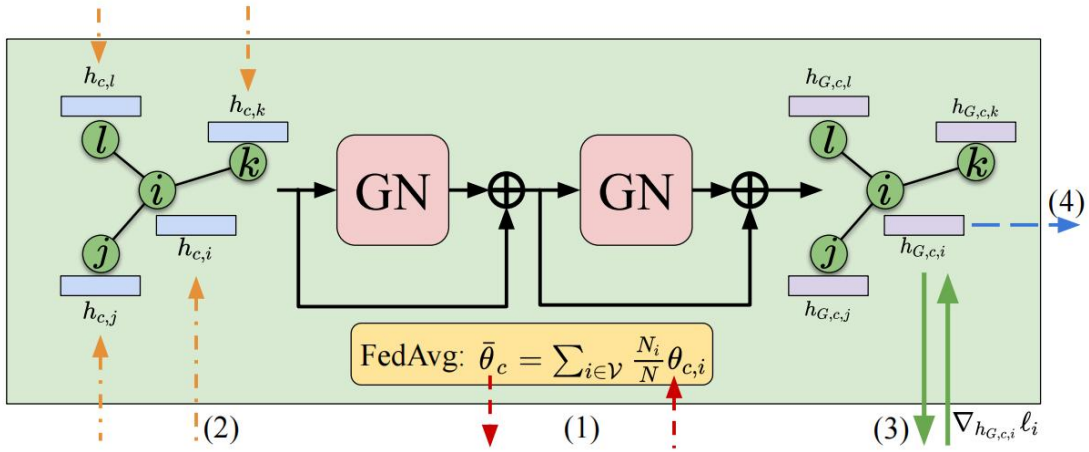
(c) Encoder-decoder on the i-th node.

空间动态性建模

- 采用图网络来生成包含所有节点关系信息的节点嵌入。中央服务器从所有节点采集隐藏状态$\{h_{c,i}|i \in V\}$作为图网络的输入。图网络的每一层都如下更新输入特征：

$$\begin{aligned} \mathbf{e}'_k &= \phi^e\left(\mathbf{e}_k, \mathbf{v}_{r_k}, \mathbf{v}_{s_k}, \mathbf{u}\right) & \bar{\mathbf{e}}'_i &= \rho^{e\to v}\left(E'_i\right) \\ \mathbf{v}'_i &= \phi^v\left(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u}\right) & \bar{\mathbf{e}}' &= \rho^{e\to u}\left(E'\right) \\ \mathbf{u}' &= \phi^u\left(\bar{\mathbf{e}}', \bar{\mathbf{v}}', \mathbf{u}\right) & \bar{\mathbf{v}}' &= \rho^{v\to u}\left(V'\right) \end{aligned}$$

- 使用了一个带残差的两层图网络用于实验。服务器端图网络输出所有节点的嵌入$\{h_{G,c,i}|i \in V\}$，并发送给每个节点相应的节点嵌入。



(b) Server-side Graph Network (GN).

节点级模型和空间模型的交替训练

训练阶段的高通信成本

中央服务器：

- **在前向传播中，需要从所有节点接收隐藏状态，并向所有节点发送节点嵌入；**
- **在反向传播中，需要从所有节点接收节点嵌入的梯度，并向所有节点发送隐藏状态的梯度；**
- **假设所有隐藏状态和节点嵌入的大小均为S，在图网络训练的每一轮数据传输总量是$4|V|S$。**

交替训练

- **固定节点嵌入，优化编码器-解码器模型R_c轮；**
- **固定节点上的所有模型，优化图网络模型R_S轮；**
- **对于GN模型的R_S轮训练，每轮中传输的平均数据量减少到$\frac{2+2R_S}{R_S}|V|S$。**

性能对比实验

GRU(centralized)：用集中传感器数据训练的GRU模型。

GRU+GN(centralized)：直接组合GRU和图网络的模型，用集中数据训练，其架构类似于CNFGNN，但节点上的所有GRU模块始终共享相同的权重。（**CNFGNN性能上界**）

GRU(local)：每个节点训练一个仅用本地数据的GRU模型。

GRU+FedAvg：用FedAvg算法训练的GRU模型。

GRU+FMTL：每个节点使用联邦多任务学习训练的GRU模型，并使用邻接矩阵给出的簇正则化。

Method	PEMS-BAY	METR-LA
GRU (centralized, 63K)	4.124	11.730
GRU (centralized, 727K)	4.128	11.787
GRU + GN (centralized, 64K + 1M)	**3.816**	**11.471**
GRU (local, 63K)	4.010	11.801
GRU (local, 727K)	4.152	12.224
GRU (63K) + FedAvg	4.512	12.132
GRU (727K) + FedAvg	4.432	12.058
GRU (63K) + FMTL	3.961	11.548
GRU (727K) + FMTL	3.955	11.570
CNFGNN (64K + 1M)	**3.822**	**11.487**

Method	Comp Cost On Device (GFLOPS)	PEMS-BAY		METR-LA	
		RMSE	Train Comm Cost (GB)	RMSE	Train Comm Cost (GB)
GRU (63K) + FMTL	0.159	3.961	57.823	11.548	99.201
GRU (727K) + FMTL	1.821	3.955	359.292	11.570	722.137
CNFGNN (64K + 1M)	0.162	**3.822**	237.654	**11.487**	222.246

感　谢　观　看

· 时空AI研讨会 ·

2023年3月3日