联邦机器学习

Federated Machine Learning

董天文
East China Normal University
2019.3.21

上海市高可信计算重点实验室
Shanghai Key Laboratory of Trustworthy Computing

联邦机器学习概述

- 定义
- 联邦学习的分类

联邦机器学习过程

- 协议
- 客户端与服务器

联邦机器学习拓展

- 联邦机器学习的应用
- 联邦学习与迁移学习
- TFF框架简介

联邦机器学习概述

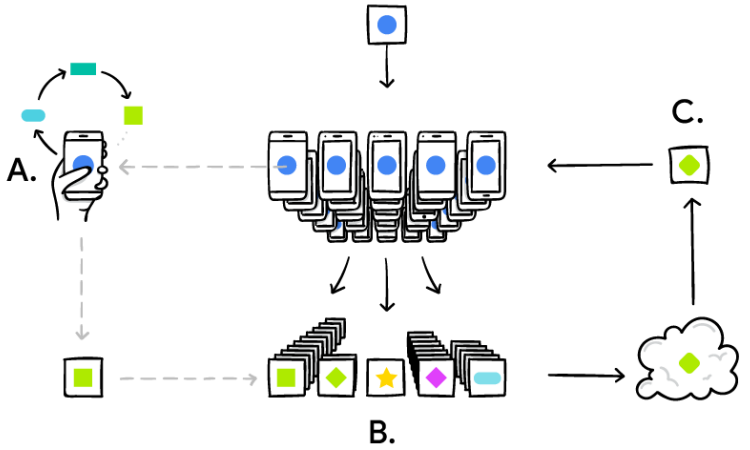
联邦学习的定义

联邦学习的分类

联邦机器学习的定义

Federated Machine Learning

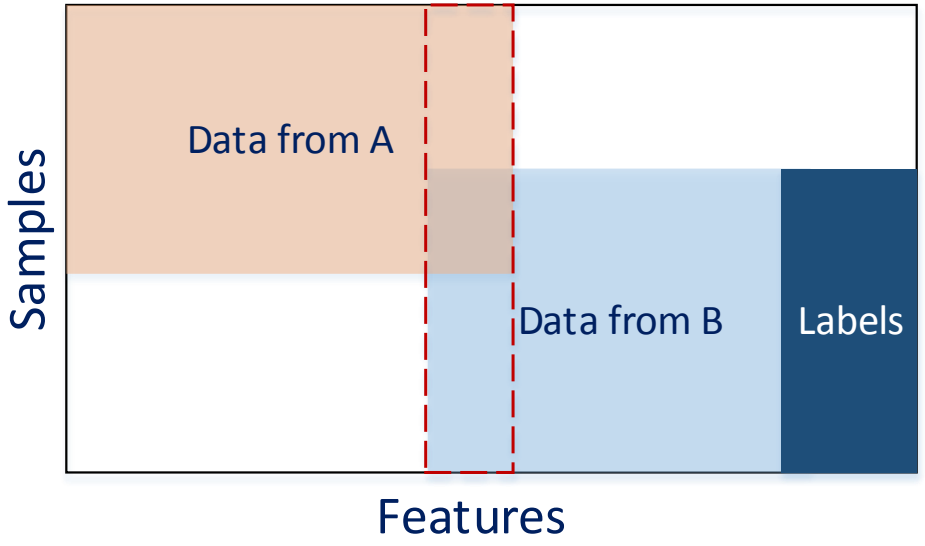
- 分布式机器学习方法
- 对手机等客户端上的大量分散数据进行训练
- 客户端完成训练，只上传权重更新至服务器
- 来自大量客户端的权重由服务器进行聚合，创建改进的全局模型
- 更新下发至各个客户端，进行下一轮训练



在不共享数据的前提下，利用多方的数据实现模型增长

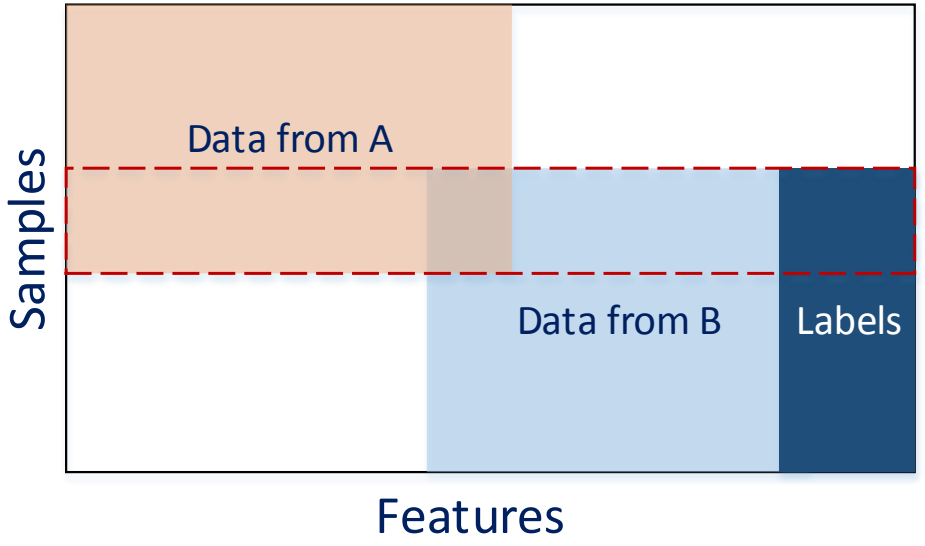
联邦机器学习的分类

横向联邦机器学习



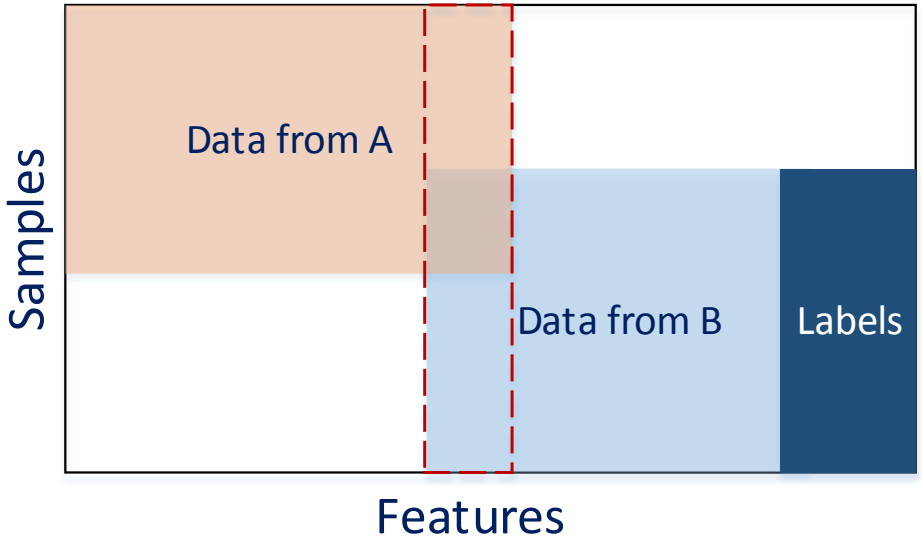
两个数据集的特征大量重叠

纵向联邦机器学习



两个数据集的样本ID（用户）的大量重叠

横向联邦机器学习



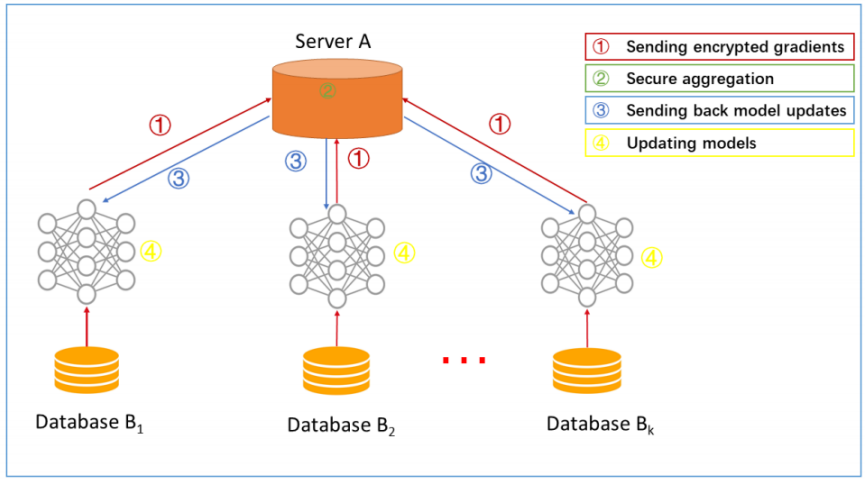
Step 1: 客户端在本地计算训练梯度

- 通过加密、差分隐私等技术隐藏梯度
- 所有客户端将其隐藏后的结果发送给服务器

Step 2: 服务器执行安全聚合（secure aggregation）

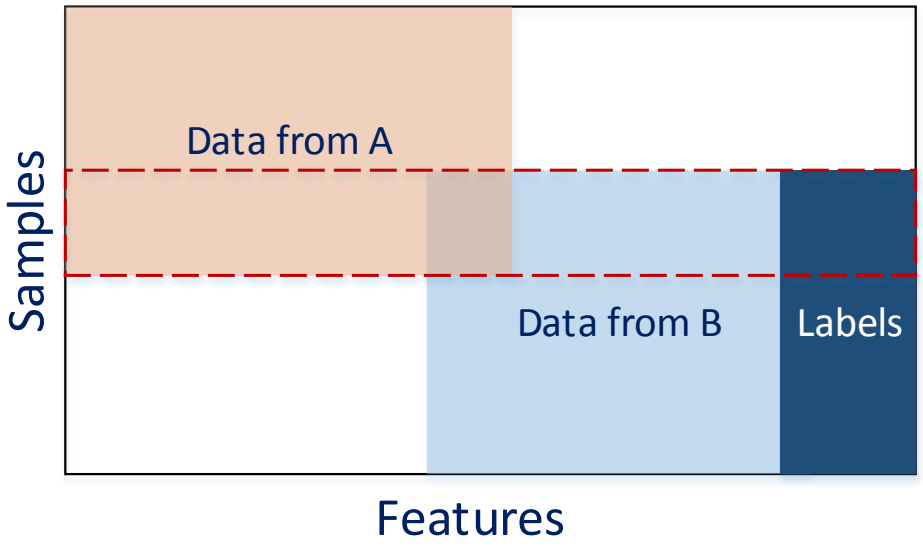
Step 3: 服务器将聚合结果发送回客户端

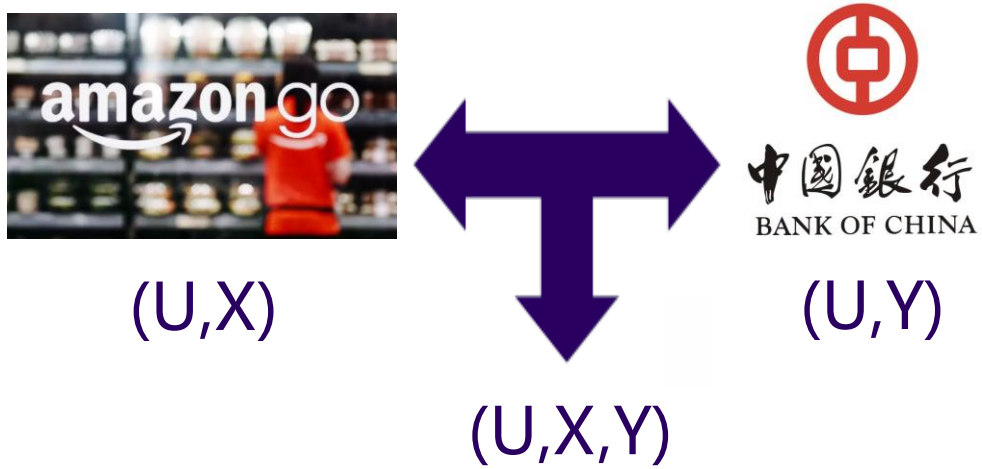
Step 4: 客户端使用解密的梯度更新各自的模型



- 多客户端，单服务器
- 数据在客户端之间水平分割，均匀的features
- 本地训练
- 有选择的客户端

纵向联邦机器学习





目的: 两方或多方共同建立FML模型

假设: 只有一方有label Y；没有一方想公开他们的label

挑战: 只有label X无法构建模型；不能相互交换原始数据

期望: 在保护双方隐私的前提下构建一个Lossless的模型

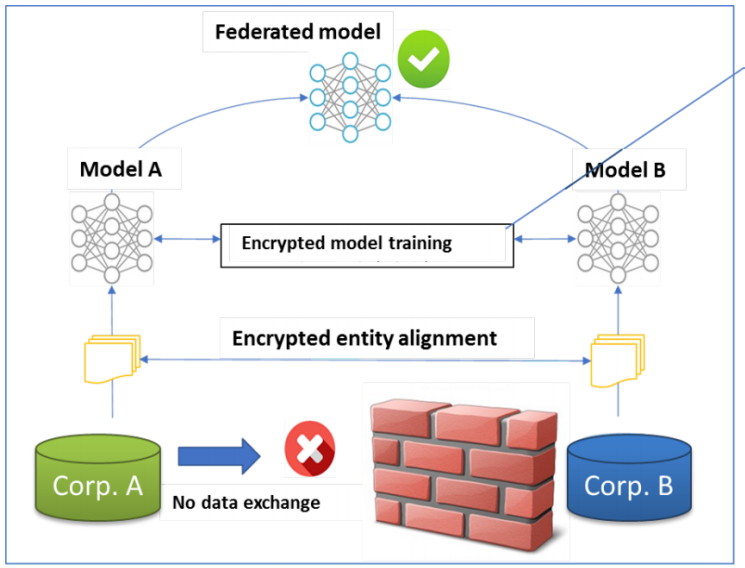
ID	X1	X2	X3
U1	9	80	600
U2	4	50	550
U3	2	35	520
U4	10	100	600
U5	5	75	600
U6	5	75	520
U7	8	80	600

零售数据A

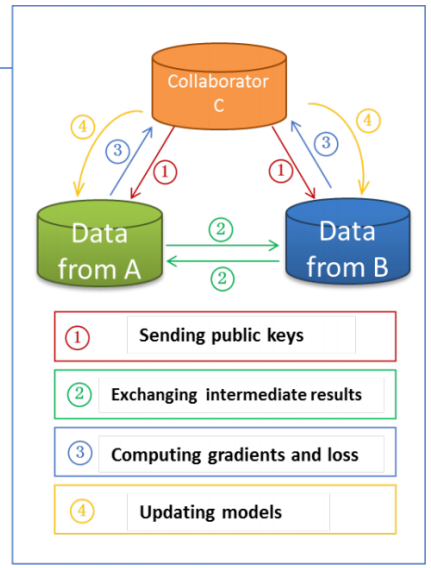
ID	X4	X5	Y
U1	6000	600	No
U2	5500	500	Yes
U3	7200	500	Yes
U4	6000	600	No
U8	6000	600	No
U9	4520	500	Yes
U10	6000	600	No

银行数据B

纵向联邦机器学习



加密实体对齐



加密模型训练

Step 1: 协作者C创建加密对，将公钥发送给A和B

Step 2: A和B加密并交换中间结果以进行梯度和loss计算

Step 3: A和B分别计算加密梯度并添加额外的掩码（mask）; A和B将加密值发送给C

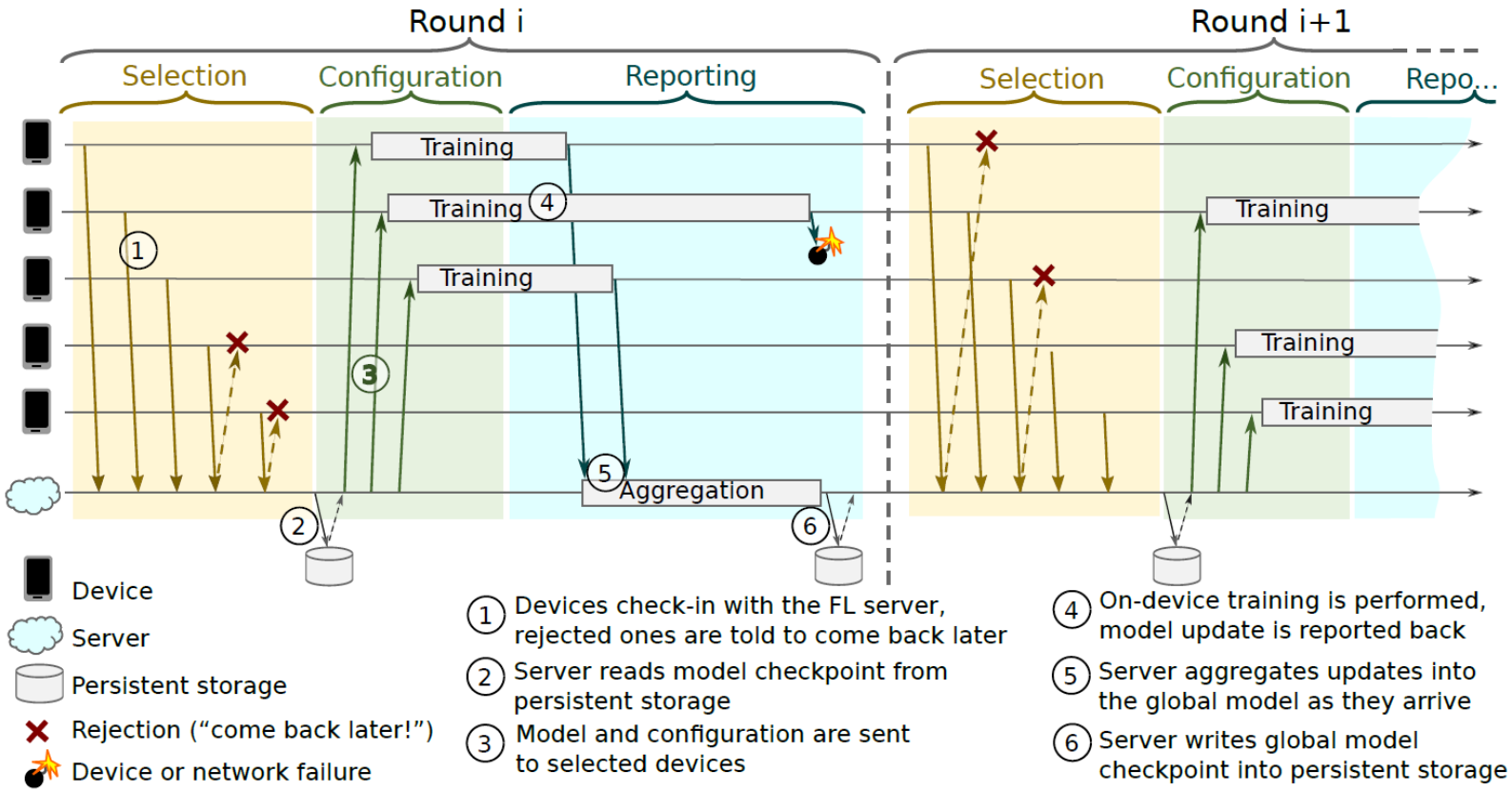
Step 4: C解密并将解密的梯度和loss发送回A和B; A和B取消掩码梯度，相应地更新模型参数

联邦机器学习过程

协议

客户端与服务器

联邦机器学习协议

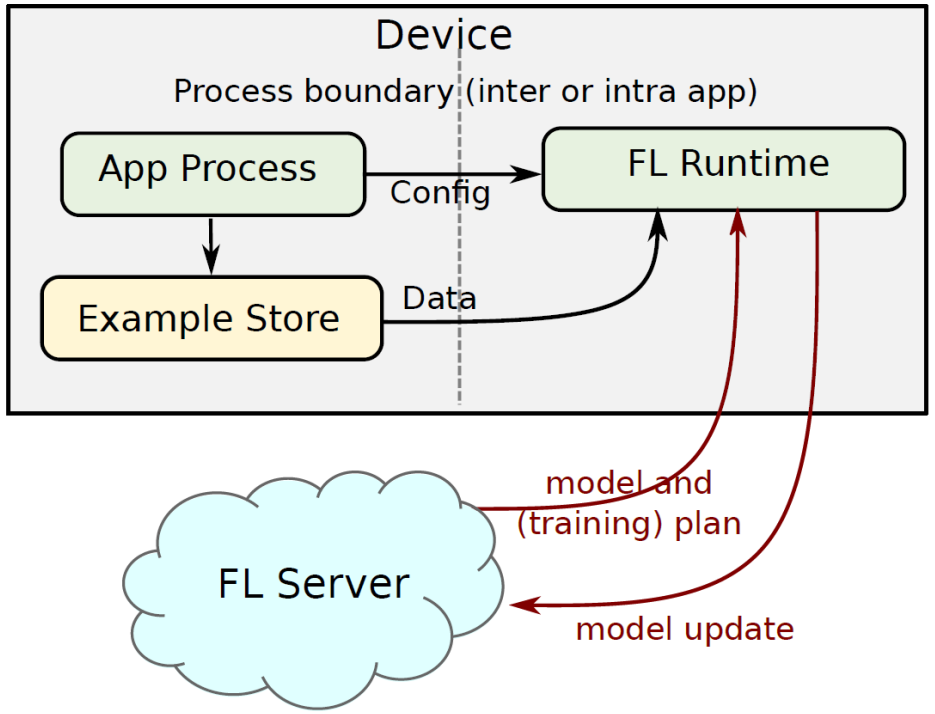


Pace steering 流量控制机制

- 向客户端建议重新连接的最佳时间窗口
- 对于小FL群体，使用该机制来确保足够数量的客户端同时连接到服务器
- 对于大型FL群体，该机制用于随机化设备check-in的时间，避免瞬时接入的客户端过多的问题
- 该机制还考虑了有效客户端数量的昼夜振荡，并且能够相应地调整时间窗口

客户端与服务器

客户端



Step1: 程序化配置

应用程序通过提供FL任务名称并注册其存储来配置FL runtime。

Step2: 任务调度

当任务调度程序在单独的进程中调用时，FL runtime联系FL服务器以宣布它已准备好运行给定的FL任务。

Step3: 任务执行

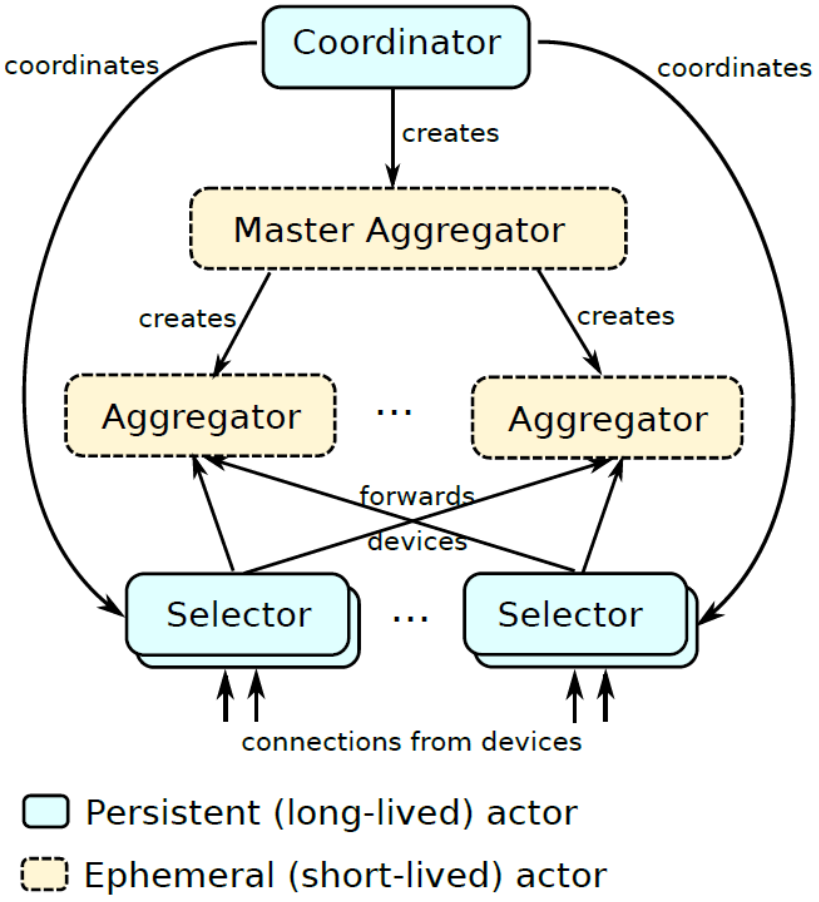
如果已选择设备，FL runtime将接收FL计划，在应用程序的存储中查询计划请求的数据，并计算计划确定的模型更新和度量标准。

Step4: 报告

FL计划执行后，FL runtime将计算的更新和指标报告给服务器并清除所有临时资源。

客户端与服务器

服务器



Coordinator

实现全局同步，推进训练轮次，每个协调器负责一组FL客户端。协调器接收每个选择器连接了多少客户端的信息，并根据计划的FL任务指示他们接受参与的设备数量。协调器产生主聚合器来管理每个FL任务的轮次。

Selector

接收和转发客户端连接。定期从协调器接收有关每个FL群体需要多少设备的信息。在生成主聚合器和聚合器集之后，协调器指示选择器将其连接的设备的子集转发到聚合器。

Master Aggregator

管理每个FL任务的轮次。根据客户端数量和更新大小进行扩展，会做出动态决策，生成一个或多个委派工作的Aggregators

客户端与服务器

联合平均算法

- 在服务器上用于组合客户端更新并生成新的全局模型
- 每个客户端采用当前模型 w_t 使用随机梯度下降（SGD）计算其本地数据上的平均梯度 g_k。 对于客户端学习速率e，本地客户端的更新 w_{t+1}^k 由下式给出：

$$w_t - eg_k \rightarrow w_{t+1}^k$$

- 服务器对客户端模型进行加权聚合以获得新的全局模型 w_{t+1}，其中 $N = \sum_k n_k$ ：

$$\sum_{k=1}^{K} \frac{n_k}{N} w_{t+1}^k \rightarrow w_{t+1}$$

- 客户端在本地计算SGD更新，这些更新被传送到服务器并进行聚合。超参数包括客户端batch size，客户端epochs和每轮客户端数（全局batch size）

联邦机器学习拓展

联邦机器学习的应用

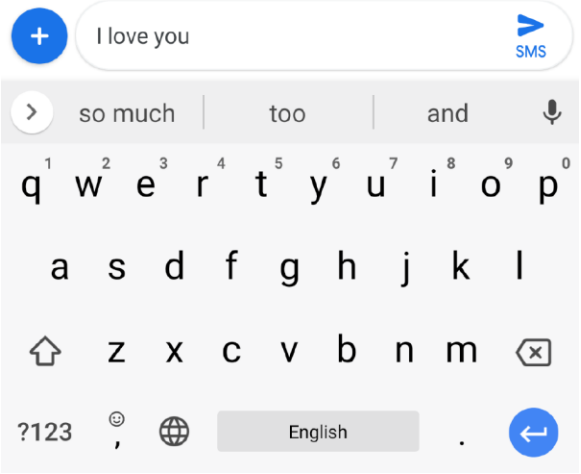
联邦学习与迁移学习

TFF框架简介

联邦机器学习的应用

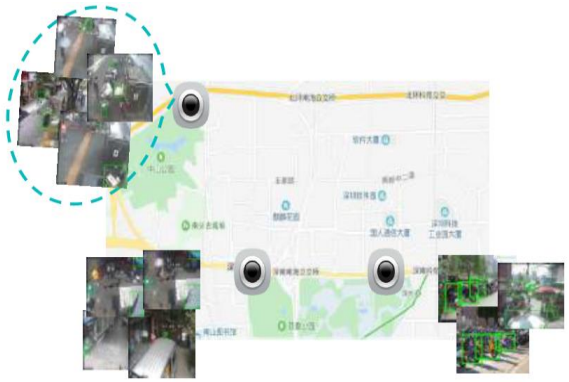
通常用于监督学习任务，如从用户活动推断标签。

- 设备上的app排名
- 设备键盘上的内容与下一词的预测（谷歌的Gboard团队）



用于对象检测的FML网络

- 停车和街头小贩违规



车辆通信领域与医疗领域

联邦学习与迁移学习

联邦学习相比于迁移学习的优点在于“无损失”。

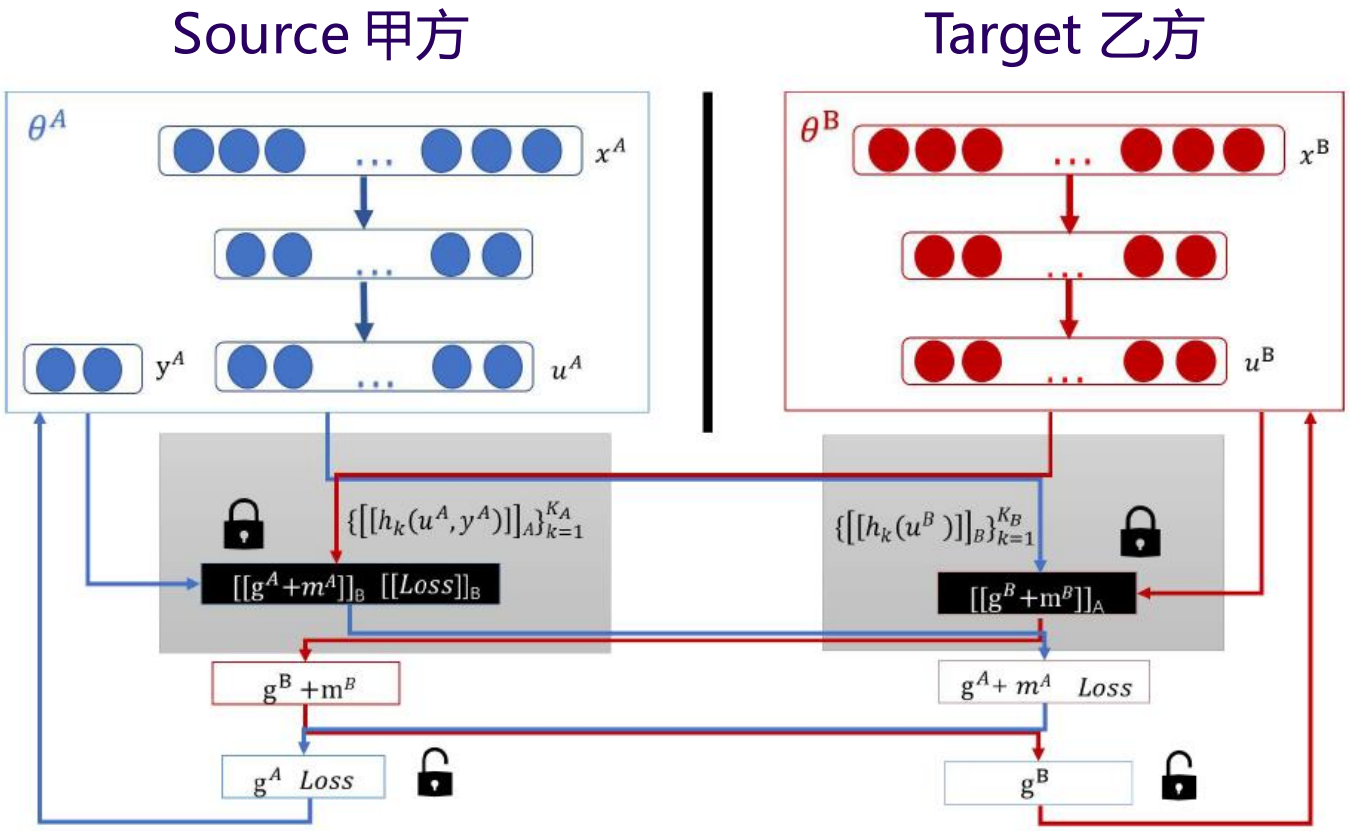
之前的迁移学习都是存在性能损失的，当模型从领域 A 迁移到领域 B，从模型中学到的一大部分关于 A 的知识全丢了，只有和 B 共享的一小部分保留下来，甚至会出现负迁移。

如何在不共享数据的前提下进行迁移学习？

Federated Transfer Learning

联邦学习与迁移学习

Federated Transfer Learning



Step 1: 甲乙双方互发公钥

Step 2: 双方进行计算，加密和交换中间结果

Step 3: 双方计算加密的梯度，添加掩码并相互发送

Step 4: 双方解密梯度并在本地进行交换，取消掩码和更新模型

TFF框架简介

TensorFlow Federated（TFF）是一个开源框架，用于机器学习和分散数据的计算。使开发人员能够在其模型和数据上模拟所包含的联合学习算法，并尝试新颖的算法。TFF提供的构建块还可用于实现非学习计算，例如对分散数据的聚合分析。

界面主要分为两层：

- Federated Learning (FL) API

该层提供了一组高级接口，允许开发人员将所包含的联合训练和评估实现应用于其现有的TensorFlow模型。

- Federated Core (FC) API

该层的核心是一组低级接口，通过在强类型函数编程环境中将TensorFlow与分布式通信运算符相结合，简洁地表达新的联合算法。这一层也是建立联邦学习的基础。

TFF框架简介

MNIST数据集举例

创建MNIST的原始NIST数据集包含从3,600名志愿者收集的810,000个手写数字的图像，我们的任务是建立一个识别数字的ML模型。

我们采用的传统方法是立即将ML算法应用于整个数据集。
但是，如果我们无法将所有数据合并在一起，例如，志愿者不同意将原始数据上传到中央服务器，该怎么办？

载入数据

```
emnist_train, emnist_test = tff.simulation.datasets.emnist.load_data()
```

选择客户端

在模拟时，只需对客户的随机子集进行抽样。

```
NUM_CLIENTS = 3
sample_clients = emnist_train.client_ids[0:NUM_CLIENTS]
federated_train_data = make_federated_data(emnist_train, sample_clients)
len(federated_train_data), federated_train_data[0]
```

```
# 加载模拟数据
source, _ = tff.simulation.datasets.emnist.load_data()
def client_data(n):
  dataset = source.create_tf_dataset_for_client(source.client_ids[n])
  return mnist.keras_dataset_from_emnist(dataset).repeat(10).batch(20)

# 封装Keras模型与TFF一起使用
def model_fn():
  return tff.learning.from_compiled_keras_model(
      mnist.create_simple_keras_model(), sample_batch)

# 使用选定的客户端设备模拟几轮培训
trainer = tff.learning.build_federated_averaging_process(model_fn)
state = trainer.initialize()
for _ in range(5):
  state, metrics = trainer.next(state, train_data)
  print (metrics.loss)
```

Thanks

董天文
East China Normal University
2019.3.21

上海市高可信计算重点实验室
Shanghai Key Laboratory of Trustworthy Computing