Improving Availability of Vertical Federated Learning Relaxing Inference on Non-overlapping Data

0. Abstract

limitation

fail to conduct inference on **non-overlapping** samples

(overlapping samples may only take up a small portion of the whole data at each party 重叠的features只是少数)

VFL framework work

- enables federated inference on **non-overlapping data**
- adopt **Oblivious Transfer (OT)** to preserve data ID privacy
- **evaluate** the model on the real-world dataset
- provide a **security analysis**

1. Introduction

现有的解决方案主要是使用**cryptographic tools:**

- homomorphic encryption (HE)
- secure multi-party computation (MPC)

使用over-lapping的samples id的训练过程：

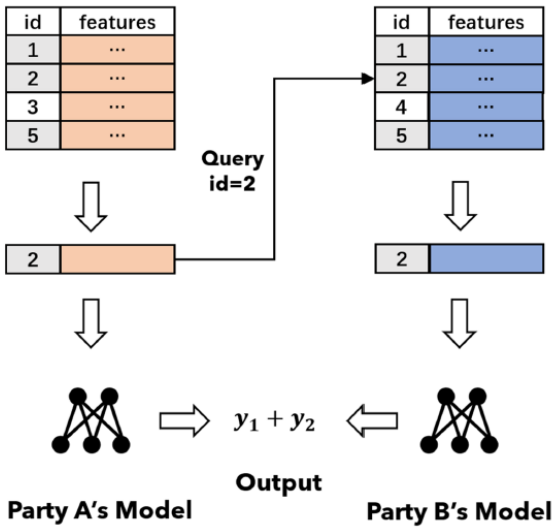

Fig. 1. Inference in VFL model. During inference, party A and party B find the features of the same instance with id = 2. Then the features are entered into models for inference. The results are then aggregated together as the final result.

Problems:

- **Availability**

data sample with the same ID may be absent in one of the parties

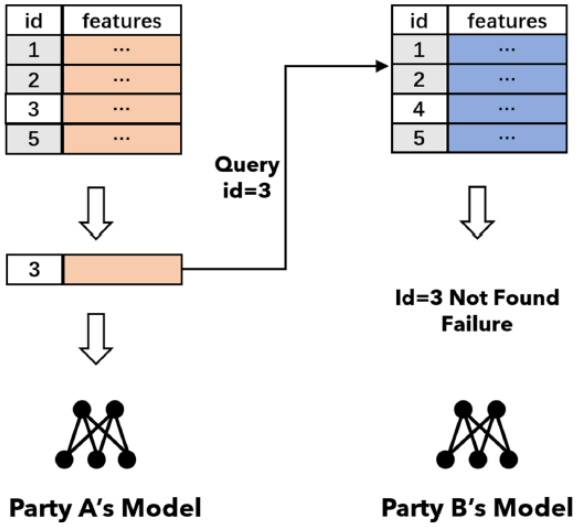

Fig. 2. VFL inference failure. The failure happens because the query for id in B's database is failed.

- **Data ID privacy**

  the inference task will **expose the data ID** to all parties

  heterogeneity of different datasets that distribute

考虑方法：

- **Federated Transfer Learning (FTL)**

  problem: source domain of FTL is limited to one party

- **Oblivious Transfer (OT)**

  problem: leads to tremendous costs in computation and network

challenges:

- The balance between **availability and privacy**

> centralized datasets 可以保证 inference success，但是破坏了privacy
>
> decentralized private solutions 无法保证 inference success

- The balance between **efficiency and privacy**

> centralized solution 可以保证 efficient，但是破坏了 privacy
>
> decentralized private solutions 涉及到 OT，会破坏 efficiency

1.1 Motivation

VFL model 可以 supervise "student model"， “student model” 只需要一个 local datasets 就可以完成 inference

desired solution: 在VFL inference 失败后，使用 student model inference

VFL solution:

- inference success

  使用 local data 和 soft labels train 一个 **student model**

- inference stage’s privacy

  constructing an **oblivious inference protocol** across all parties

- a balance between privacy and efficiency

  constructing **hash tables** in all parties and the preparation of ciphertexts for the **offline** stage

1.2 Contribution

1. propose the training of the **student model** in VFL

   通过从VFL model 到 local model 的 knowledge distillation，确保了inference success

   当 VFL model 因为缺少 data ID fail 时， local model 仍然可以inference

2. propose a **privacy-preserving** inference protocol

   preserves the privacy of data ID

3. **efficiency** in online inference is improved

   constructing a **hash table** and moving encryption operations to the **preparation stage**

Result:

1. 在VFL model 的 supervision下， student model performance increased.
2. The oblivious inference has a **much lower cost.**

2. Background

Federated Learning 首次是Google在2016年提出，主要强调在机器学习模型设备（cell phones）的训练过程中 的 privacy concern

从那以后，其他的许多场景加入到Federated Learning中。例如：在公司的大规模数据集之间的FL，与model devices相比，每个party都有了更大的dataset 和 features

通常 features在不同的parties之间是不同的，同时这些data是不允许在parties之间交互的

因此，VFL 被提出用来训练在跨多方之间的outsourced features

2.1 Vertical Federated Learning

提出的算法基于 ***Paillier homomorphic encryption***

参与方分为 3 类：

（1）**Passive Party**

datasets只有 features

（2）**Active Party**

datasets是包含labels

（3）**Coordinator**

协调 learning process 以及生成 Paillier key pairs

Training steps:

（1）在active party 和 passive party之间的datasets是要对齐aligned，以此来找到 overlapping datasets 组成 training set

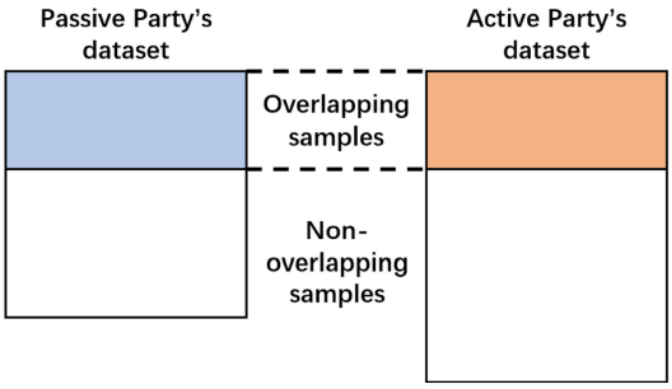

Fig. 3. Data distribution in VFL assumption. The subset marked with orange and blue have overlapping data IDs.

（2）Coordinator (The federated learning platform) 生成 a pair of Paillier keys， send public key 到 active and passive party

（3）active 和 passive party 开始训练，每一轮中， passive party都会loads a batch of data， 以此计算 local model的output， encrypts the output，并发送给 active party.

（4）active party 得到了 passive party output的密文，并加入它自己model 的相同batch下的output，生成 VFL model的 encrypted output

（5）active party 和 passive party 联合计算 encrypted loss 以及 gradient, 由coordinator解密，之后用得到的gradient去更新自己的local models

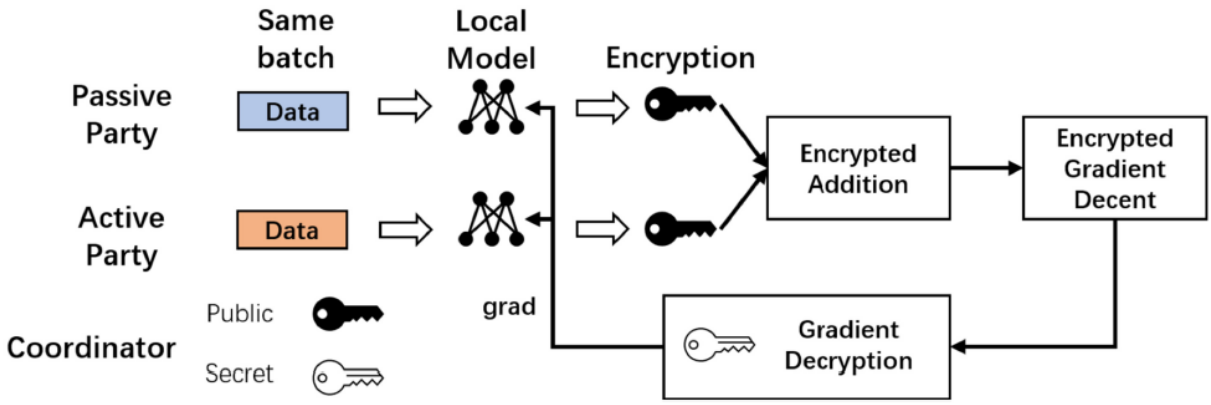

Fig. 4. Vertical federated learning based on homomorphic encrytion.

2.2 Tussle of Privacy and Efficiency

inference task 需要 a series of queries 以及 aggregates the inference results.

但是 parties 不想暴露 non-overlapping 的data ID

主要问题是：**inference** 以及 **privacy**

提高availability的一种方法：

inference success

- *centralize all datasets to one server*

  不会发生 query failure,但是需要可信第三方(impossible in real world)

privacy of data ID

- *perform OT when requesting inference results.*

  但是会损害 efficiency，带来较大的 computation 以及 communication costs

3. Preliminaries

- partially homomorphic encryption (PHE)
- OT

3.1 Partially Homomorphic Encryption

HE 支持一套 cryptosystems， support computing over encrypted data.

PHE 支持 part of the operations on the ciphertext.

Paillier Homomorphic Encryption 支持密文的加法运算，对于不支持的算数操作可以用明文在本地完成

- Key generation

  (pk,sk) = Gen(keylength) 生成指定长度的公私钥对

- Encryption

  Enc(x,pk) 返回的是用公钥对明文加密后的密文

- Decryption

  Dec(c, sk) 返回的是用私钥对密文解密后的明文

明文*x*加密后，记作 *[[x]]*

Paillier支持的其它操作：

- Addition of ciphertext

  Add([[x]], [[y]]) = [[x+y]] 通过 multiplying two ciphertexts

- Multiplication of ciphertext

  Mul(x, [[y]]) = [[xy]] 通过 modular power $[[y]]^x$

因此，可以使用加密后的 loss 和 gradients进行协同训练

vertical linear regression:

$$Loss = \sum_{batch} (y_{pred} - y)^2 + \frac{\Lambda}{2}\theta\theta^T \tag{1}$$

$$grad_{local} = \alpha * \left(\sum_{batch} x_{local}(y_{pred} - y) + \Lambda\theta \right). \tag{2}$$

y_{pred}是加密后的，即使直接计算 y_{pred}^2是不支持的，但是可以通过持有 y_{pred} 的party去在明文下计算，然后加密它。在计算 local model's gradient时候，解是不变的

3.2 Oblivious Transfer

OT 是一个重要的密码学原语，被引用与 MPC 系统以及协同训练过程中。

1-out-of-2 OT protocol:

> Sender S
>
> Receiver: R
>
> Messages: m_0, m_1
>
> Choice bit: $r \in 0, 1$
>
> Result: S 不知道 r， R 收到了 m_r，但是不知道 m_{1-r}

OT 也扩展到从一组 elements中进行选择

1-out-of-n OT:

> Sender: S
>
> Message Set: X
>
> Message length: n
>
> Receiver bit string: t （t的每一位都是0或者1）
>
> Receiver bit string length: $m = \lceil logn \rceil$
>
> Result: S remains Oblivious， R 收到了 X[t]

Send (X, n)

Recv(choice, n)

- 一个naive的extension是从 1-out-of-2 OT 扩展到 1-out-of-n OT的方法是执行 $\lceil logn \rceil$次OT operations.
- 一些优化算法：执行 k basic OTs, k 是安全参数（128或者256）

transfer分为两个阶段：

（1）sender 和 receiver 完成 backward OT，为之后的OT共享信息

（2）完成transfer，receiver收到想要的message

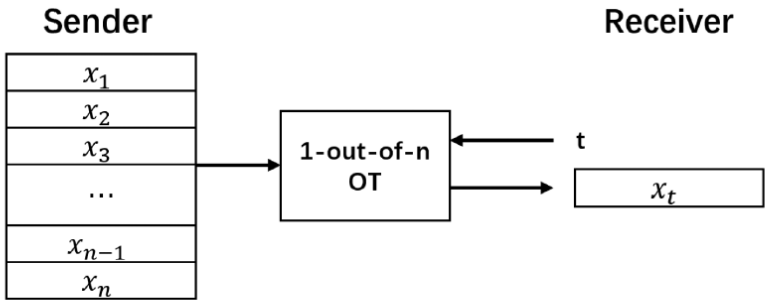

Fig. 5. 1-out-of-n OT. The sender has dataset $X = x_1, x_2 \ldots x_n$ while the receiver holds choice index t. At the end of protocol, the receiver gets x_t and the sender learns nothing about t.

使用 1-out-of-n OT 作为基本的工具来在 inference 过程中保护 data ID

使得对于 data owners 来说， data ID remains oblivious

进一步优化：将密文生成部分，移到准备阶段完成

4. Methodology

在training过程中用到的一些notations

Table 1. List of Notations used in Training Section

Notation	Description
λ	The weight for soft label from VFL model
L_s, L_d	Loss function from real labels and soft labels
f_s, f_t	Output of student model and teacher model
X_B^o	Overlapping data in the active party
X^o	Overlapping data across two parties
W_s, W_t	Parameters for the teacher model and student model

training也被分为两个阶段：

（1）VFL model 使用 VFL method训练，并给出 encrypted soft labels, 用于student model的监督学习

（2）student model 从real labels 和 soft labels的 loss 中训练

除了训练 VFL model，还采用了 knowledge distillation的方法，将federated model's knowledge tranfer 到 party's local model.

passive party 有 dataset D_A

active party 有 dataset D_B

D_0 是overlapping samples

$\overline{D_A^0} = D_A - D^0$

$\overline{D_B^0} = D_B - D^0$

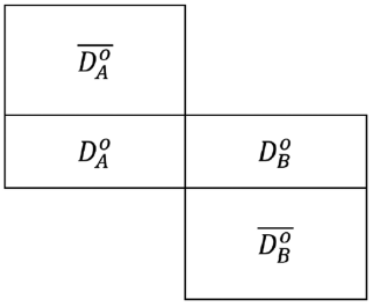

Fig. 6. Distribution of 2 parties' datasets. $D^o = D_A^o \cup D_B^o$ is the training set for VFL model.

之前的 knowledge distillation 被用来从一个 large model 到一个 smaller model transform knowledge, 因此 inference 可以在不损失ACC的情况下得到加速

在 knowledge distillation 过程中，训练两个models:

- a teacher model
- a student model

  在 the second stage，使用 teacher model输出的 soft label 和 training data 的 labels 进行训练

在 VFL中，从 VFL model transfers the active party's student model(用来处理active party的 local features)

目标函数如下：

$$\min_{W_s}(1-\lambda) * L_s(y, f_s(X_B^o, W_s)) + \lambda * L_d(f_t(X^o, W_t), f_s(X_B^o, W_s)). \quad (3)$$

student model 的输出是用X_B^0来计算的，也就是 student model 只需要处理 active party的 local features.

VFL model 的输出是用X^0来计算的，包含了 active 和 passive party的输出

4.1 VFL Model Training

Stage one:

parties 用他们选择的protocol协同训练一个VFL model

VFL model 被用来预测每个 training set.的 data sample

在 training 的过程中，predictions 是用相同的 public key 被加密的

这些 predictions 记作 labels , 并且用 $\tilde{y}$ 代表。

4.2 Student Model Training

student model 的 loss 是原自 real labels 以及 soft labels (supervision of the VFL model)

soft labels 是从VFL model 的 D^0中得到的，因此可能包含 passive party 的隐私信息，使用 PHE 在 soft labels

Stage two:

1. **passive party**: 使用 public key 生成encrypted soft labels，并发送给active party
2. **active party**: 在iteration中，计算加密后的 loss 和 gradient，并给 gradient 加noise（mask），并发送给 coordinator
3. **coordinator**: 解密 loss 和 gradient，并发送 gradient 给active party
4. **active party** : eliminates（unmask）the noise，并更新 student model

(y 是 本地， [[y]] 全局)

ALGORITHM 2: Student Model Training in Active Party

Result: Student model W_s
Input: VFL model f_t with parameter W_t, hard label y, encrypted soft label $[[\tilde{y}]]$, soft label squares $[[\tilde{y}^2]]$, X_B^o, max iterations t, balance factor λ;
Initialize student model parameter W_s;
i = 0;
while $i < t$ **do**
  $[[Loss]] = \lambda * L_s(y, f_s(W_s, X_B^o) + (1-\lambda) * L_d([[\tilde{y}]], f_s(W_s, X_B^o))$;
  send $[[Loss]]$ to Coordinator;
  $[[grad]] = \frac{\partial Loss}{\partial W_s}$;
  generate rand_mask;
  $[[mask\_grad]] = [[mask\_grad]] + rand\_mask$;
  send $[[mask\_grad]]$ to Coordinator;
  Recv mask_grad from Coordinator;
  $grad = mask\_grad - rand\_mask$;
  $W_s = W_s - \alpha * grad$;
  Recv converged from Coordinator;
  if *converged* **then**
    Return W_s;
  else
    i = i + 1;
  end
end

active party 主要是在每一轮计算加密后的 loss 和 gradient，并发送到 coordinator

因为 gradient是被masked，因

此不会泄露信息到 coordinator

所有的参与方都是 honest but curious (半诚实)

Coordinator 每一轮都要解密 loss 和 mask_grad，并检查model 是否收敛

coordinator algorithm：

ALGORITHM 1: Student Model Training in the Coordinator

Result: Student model W_s
Input: max iterations t, converge threshold c;
Initialize student model parameter W_s;
i = 0, pred_loss = 0;
while $i < t$ **do**
  Recv [[*Loss*]] from active party;
  Loss = Decrypt([[*Loss*]]);
  Recv [[*mask_grad*]] from active party;
  mask_grad = Decrypt([[*Loss*]]);
  send mask_grad to active party;
  if *abs(Loss - pred_loss)* < *c* **then**
    converged = true;
    send converged to active party;
  else
    converged = false;
    send converged to active party;
    i = i + 1;
  end
end

4.3 ID Oblivious Inference

这里主要是在 inference 过程中保护 data ID 的方法

方法分为两个阶段：

(1) OT preparation phase

(2) online phase

overview:

(1) map parties' data samples into several buckets (bucket size N)

a hash function, unique address (bucket_id, offset) to all data ID

(2) passive party 在它的 data samples上面进行 inference, 将 inference 的结果放入与 data samples相同位置的 hash tables

(3) active party: 在inference时，首先计算 bucket_id 和 of f set，把 bucket_id 发送给 passive party

(4) passive party: 执行 1-out-of-n OT 与 active party

active party 作为 receiver, 持有 of f set

passive party 作为 sender，持有bucket的 inference result

(5) active party 从 passive party 中收到了 inference result

passive party 只知道可能存有 id 的bucket

4.4 Security Definition and Notation

assumption: **semi-honest** adversary model

parties 会在协议执行的过程中尝试提取信息

与 stronger malicious adversary model相比，这个假设是 weaker 但是highly-efficient

Table 2. List of Notations in During Inference

Notation	Description
$\binom{n}{1} - OT$	1-out-of-n OT
N	Bucket size
m	Inference result or an indicator for failure
key	Encryption/Decryption key
$Enc(x, key)$	Encryption with plaintext x and key
$Dec(c, key)$	Decryption with ciphertext c and key
$M_k = \{m_0, m_1, \ldots, m_{N-1}\}$	Inference result in bucket k
l	Size of ciphertext

4.5 OT Preparation

两个操作组成：

（1）hash table

（2）ciphertext generation

Building the hash table

两方计算data ID 的 hash value

eg. $H(x) = \lfloor x/N \rfloor$ (N 是由双方协定的)

collision 是通过将冲突的元素放入 index of f set = x mode N 的数组中

（1）party 之间协调 N

（2）用 random numbers 填充bucket 中的 empty entries

（3）每个 element 被放到 第 n 个 bucket 的第 m 个 entry

$m = xmodN, n = \lfloor x/N \rfloor$

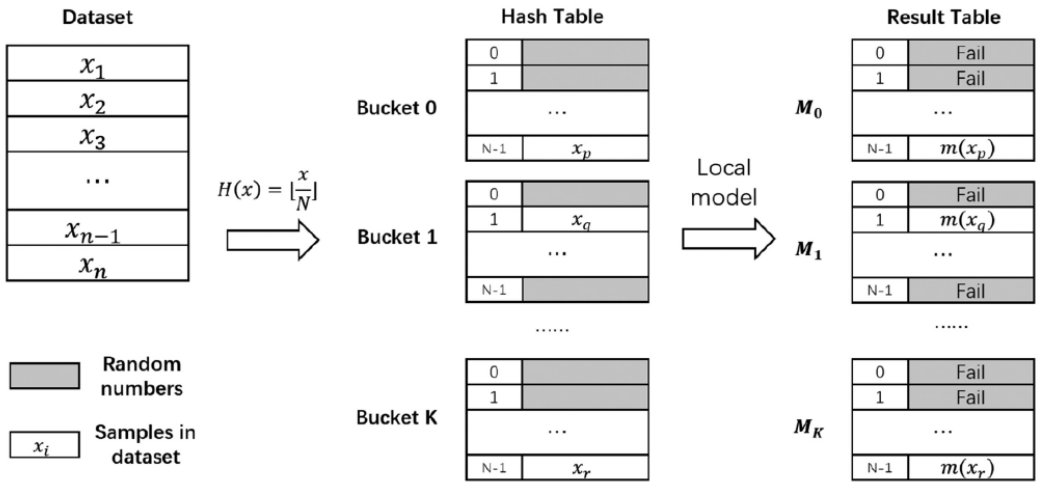

Fig. 7. OT Preparation—Hash table setup.

mapping 完成后，passive party 在real samples 的 elements 上执行 inferences 在hash table 中，对于 random masks 直接给出 FALL。

Result Hash Table 是有 K 个 buckets ， 记作 $M_0, M_1, \ldots, M_K$

$Ciphertext$ Generation

（1）passive party generates N * B * 2 keys, 并使用AES 或者 RSA 或者其他协议作为密码学机制

$B = \lceil logN \rceil$, keys 对应的 index 是
$(i, j, k), i \in \{0, \ldots, N-1\}, j \in \{0, .., B-1\}, k \in \{0, 1\}$

（2）passive party 执行 recursive encryption

eg. 某个element的二进制表示是$\{b_0, b_1, \ldots, b_{\lceil logN \rceil - 1}\}$, 那么就使用对应的 (i, j, b_j) 进行加密

（依次使用N套keys，j对应的element的 index，b_j对应的是对应位置是0或者1）

每个element在加密后都有N个对应的密文

ALGORITHM 3: Generate Encrypted elements for x_i

```
Result: Encrypted array of elements C = {c_0, c_1, ..., c_{N-1}}
Input: Element x_i, keys and bucket size N;
i = 0;
set b as the binary representation of i;
while i < N do
    tmp = x_i j = 0;
    while j < ⌈logN⌉ do
        tmp = Enc(keys(i, j, b_j), tmp);
        j += 1;
    end
    c_i = tmp;
    i += 1;
end
```

因此，对于一个 N 个elements 的bucket，每个element对应 N 个密文，那么对这个bucket加密后，是 a set of encrypted buckets

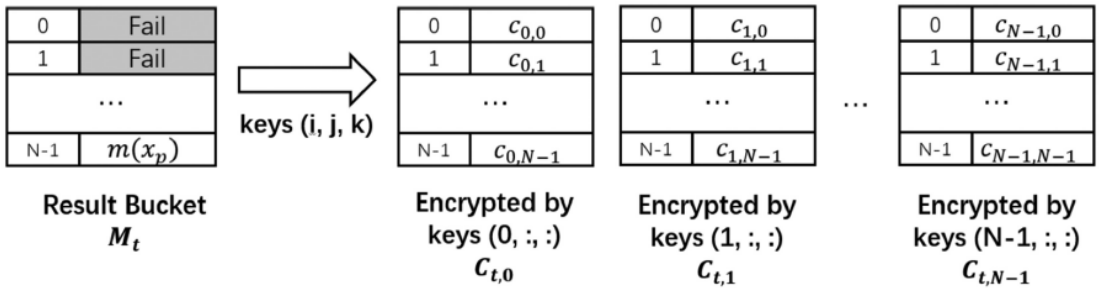

Fig. 8. Encrypted matrix for result bucket M_t.

对于 active party:

(1) 生成一个长度为 N 的随机排列，即 $R = \{r_0, r_1, \ldots, r_{N-1}\}$

(2) 对于每个element，执行 $\lceil logN \rceil$次$1 - out - of - 2 \quad OT$

(3) active party 是 receiver，passive party 是sender

receiver使用 $r_i, i \in \{0, \ldots, N-1\}$ 作为 choice string

passive party 使用 the i-th key matrix key (i, :, :)作为 message

summarize

（1）passive party 生成keys，为每个 inference result 生成 N 个加密后的 buckets

（2）active party 生成 a permutation of {0,1,...N-1} as R

（3）conduct OT， 对 每个R 中的element，得到 an array of keys （用于 online 阶段）

4.6 Online Phase

starts: active party 想要对于 data id = x 执行 inference task

(1) active party 计算 hash function 的 bucket index 和 offset

$$bkt\_id = \lceil x/N \rceil, offset = xmodN$$

(2) active party 找到 random permutation R 的 index t， 使得 R[t] = offset

(3) active party 发送 bkt_id 以及 t 给 passive party， passive party 返回$C_{bkt_id,t}$到 active party

(4) active party 最终收到了一个 encrypted bucket

decryption:

此时 active party 持有 R[t] 和它对应的keys array$\{k_0, k_1, \ldots, k_{\lceil logN \rceil}\}$

只有一个 elemetn x_{offset}可以成功解密，如下图

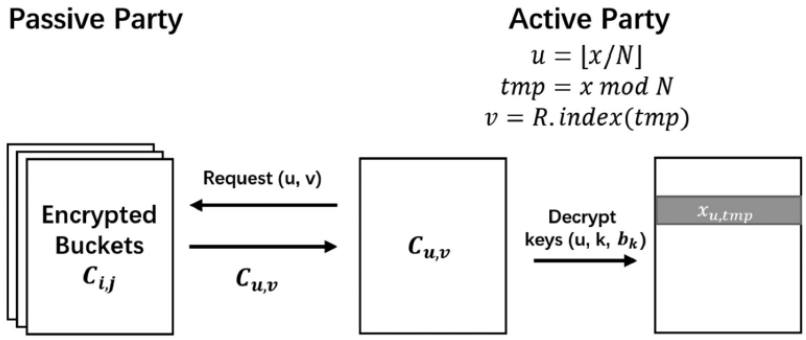

Fig. 9. The passive party sends the encrypted bucket with index u,v to the active party for decryption.

> 给出证明： 只有 x_{offset}是可以成功解密，passive party不能知道 offset 的任何信息
>
> Proof:
>
> (1) x_{offset}在 $C_{bkt\_id,t}$ 使用keys unique combination (t, j, b_j)，其中 b_j 是offset的二进制表示的第 j 个bit。因此， x_{offset}可以被 active party 正确解密
>
> (2) OT 阶段，passive party 不知道 active party chosen 的 keys 信息，因此也就不知道offset 的信息

最后一个step，active party 会 check the result of inference

若成功，则 passive party 的inference 会被聚合到 VFL model's inference。

否则，active party 必须在 student model上进行一次 inference

5. Evaluation

通过实验展示，student model 的 **efficiency** 和 **performance**

click-through **dataset** from Criteo and Taobao

5.1 Experiment Setup

> Criteo: 是否点击广告
>
> Taobao：在web网页的product 是否被点击

这两个 datasets 是在VFL applications 中很有应用的，因为 advertisements 的记录是跨公司或者跨网页的，可能包括不同的 features, 而且因为 Privacy 的原因不能直接共享

eg. Taobao dataset可能包含三种 data sheets:

(1) features of the products

(2) features of users

(3) click-through records

因为隐私的限制，可以得到 (1) (3) 但是不能得到 (2)，features of users

experimental platform introduction:
on FATE on 2 parties

例如：Equation (3) 中，$\lambda = 0$时候，model 就等价于一个 local model

training set 被垂直划分为两个 datasets:

the click-through dataset 和 the user dataset

logistic regression model 使用这两个 datasets训练。

（也考虑两方的不同的overlapping份额）

implement:

（1）dataset loaded into FATE data table on 2 parties

（2）FATE 会使用安全交集来计算交叉的数据集

（3）intersection dataset 会被 fed into VFL training module

（4）student model 使用 active party's local dataset 和 VFL模型的监督来训练

5.2 Student Model Performance

使用AUC(area under the ROC curve)，广泛应用于分类问题中，AUC 越接近 1，则 model performance better.

α 代表在 training set 中的 overlapping samples的 portion

(1) evaluate and compare the AUC ---> VFL model、local model、non-federated model (local model)

> VFL model 是使用两方的协同的 datasets
>
> local model 是只使用 local dataset 训练
>
> student model 是使用 local dataset 和 soft labels 训练

实验测试了不同的 λ 和 α 值的组合实验，来测量VFL model 的 supervision 的效果

使用 dataset上的 Logistic Regression model 来预测 “clicked” label

如图，student model's performance better than local model

Table 3. Student Model's AUC under $\lambda = 0.5$ and Different Portions of Overlapping Samples

α	0.1	0.25	0.5	0.75	0.9
Criteo local model	0.6642	0.6642	0.6642	0.6642	0.6642
Criteo student model	**0.7234**	**0.7304**	**0.7341**	**0.7351**	**0.7349**
Criteo VFL model	0.7289	0.7476	0.7534	0.7536	0.7603
Taobao local model	0.5109	0.5109	0.5109	0.5109	0.5109
Taobao student model	**0.5104**	**0.5136**	**0.5274**	**0.5448**	**0.5439**
Taobao VFL model	0.5744	0.5965	0.6308	0.6405	0.6509

The bolded entries in the table show the performance of student model which surpasses local model in most cases.

在 Criteo dataset上表现较好，在 Taobao dataset上提升较少

主要因为，在 Taobao 上的 click-through dataset 几乎没有与 labels有关的特征

VFL model 的 supervision 在性能上没有保证，因为 local dataset 上的 features 仍然 dominate model's performance，这种情况下，仍然需要 collaborative inference.

λ的选择依赖于 **features** 的分布

在实验中是按照同样的 possibility来划分，因此 features在两方中是同等重要的

Table （3）中设置 $\lambda = 0.5$，loss from real labels 和 loss from VFL models 是同等重要的

如果在另一方 dataset 中有更重要的特征，则 λ应该设置的大一点

如图，在不同的α 和λ中，Criteo student model's AUC

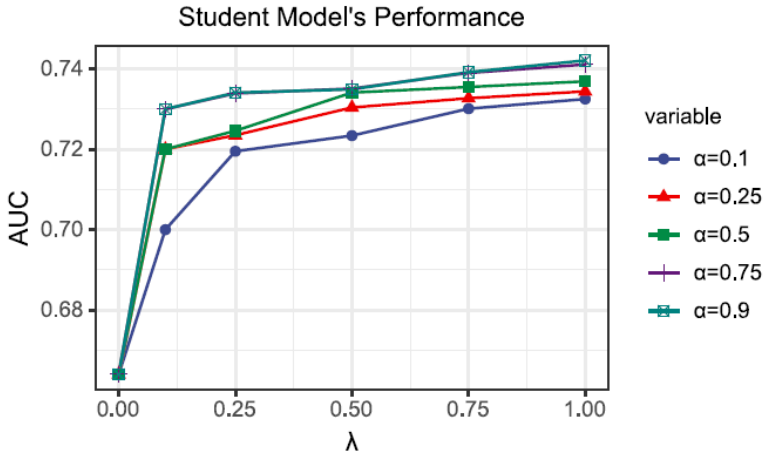

Fig. 10. Criteo student model's AUC under different overlapping portions and λ.

- overlapping samples 的 portion 越大，student model AUC increases

  (1) 更多的 overlapping data, VFL model 生成更多的 soft labels，来supervise the student model's training.

  (2) 更多的overlapping data，VFL model 可以得到更多的 training data，因此得到 better performance

- λ越大， student model AUC increases (λ 是用 VFL model 标记的loss 的weight）

  VFL model 产生的 soft labels，是samples 被预测为1 的概率

  与real labels 只取 0 和 1 相比，soft labels 给出了连续的目标

  当 $\lambda = 1$，targets 只有 soft labels，$\lambda = 0$，模型退化为 local dataset 上训练。

  （λ的增加是有度的，soft labels 并不总是相同的 real labels）

5.3 Oblivious Inference Cost

与 naive $1 - out - of - n$ OT相比，假设 OT 是通过 Bellare-Micali OT 实现的，如下表：

Table 4. Comparison of the Cost between Oblivious Inference and Naive OT

	OT Preparation		Online Phase		
Method	Comp.	Comm.	Comp.	Comm.	Rounds
$\binom{MN}{1} - OT$	–	–	$2\lceil logMN \rceil + MN$	$2l\lceil logMN \rceil + lMN$	2
$\binom{N}{1} - OT$	–	–	$2\lceil logN \rceil + N$	$2l\lceil logN \rceil + lN$	2
Oblivious Inference	$N^2M\lceil logN \rceil$	$lN\lceil logN \rceil$	$\lceil logN \rceil$	$N * l$	1

> bucket 数量是 M，每个 bucket 有 N 个 elements
>
> l 是 ciphertext 的长度
>
> 与$\binom{MN}{1}-OT$ 和 $\binom{N}{1}-OT$ 两种方法对比

在$\binom{MN}{1}-OT$中:

- activity party holds the data ID
- passive party hols $M \times N$ inference results

通过执行$\lceil logMN \rceil$ 次 $\binom{2}{1}-OT$

优点：提供了 strongest privacy for data ID

缺点：在每轮的 inference 中传输大量的 data

在$\binom{N}{1}-OT$，由 $\lceil logN \rceil$次 $\binom{2}{1}-OT$组成：

通信的开销取决于 ciphertexts 的数量，即 keys 的总长度 + encrypted bucket 的数量，因为每个bucket中有 N 个 element，总的开销是：$2l\lceil logN \rceil + lN$

OT Preparation: keys generation, encryption, OT operations on keys 都是可以在 inference 开始前完成的

Data ID对于 Passive party 来说是不可知的

6. Related work

6.1 Feature Distributed Machine Learning

Feature Distributed Machine Learning (FDML) 定义了 features 在许多 clients之间分布的机器学习模型

之前的 FDML 中，parties 传输的是明文，在训练过程中导致隐私泄露

data-parallel distributed machine learning ， 通过在parameter server聚合 gradients

FDML 是去 learn embedding on the client，在central server 上训练a global model，以 embedding 作为输入

FDML 和 VFL 的主要区别在： privacy protection

> VFL: 数据集对客户是私有的，因此数据的交换不会泄露隐私信息
>
> FDML: 数据ID 是知道的，而且 embedding 也是不加密的
>
> （VFL 是更严格的FDML）

其他的研究考虑在 training 过程中 architecture 的设计

VAFL 讨论了在 VFL 训练过程中的 asynchronous pattern

VAFL 采用 parameter server architecture，更关注如何减少通信开销

6.2 Federated Transfer Learning

FTL： 定义隐私地训练一个model，使用一个相对小的 dataset 或者一个小数量的 labels

FTL 通过将 knowledge transfer 到一个不同的，但是相关的 model 上进行训练

在FTL中训练的 model 的性能与这两个 domains 的 relevant 密切相关

FTL 以及本模型主要关注在少量 intersection labels 的情况下，协同训练 machine learning model

区别：

> FTL 是将knowledge 从 source domain 迁移到 target domain，这两个 domain 在不同的 parties
>
> 本模型：是将 knowledge 从 VFL model transfer 到 active party's local model.

实验表明：从VFL model transfer可以有效提高 active party's local model 的 performance

6.3 Privileged Feature Distillation

Knowledge distillation: Transfer knowledge 从一个 large model （teacher model）到一个 smaller one (student model)

该方法在 teacher model 的supervision 下，student model 的learning process 效果较好

privileged information： information 只在 training stage 是 available，但是在 test stage 或者 inference stage 下是 unavailable

6.4 OT in Machine Learning

OT 是一种重要的密码学工具，对于构建 privacy-preserving machine learning protocols 是高效的

OT extension 被提出来，执行一定数量轮数的 base OTs，可以使得在计算和通信过程中的 cost reduced

其余情况：correlated OT (C-OT), random OT (R-OT)

OT 被用来构建 privacy-preserving machine learning protocols.

ABY 和 ABY3 支持在任意划分的数据集下训练机器学习模型，使用 secret sharing 和 OT

7. Summary

当前的 VFL models 不能保证在 online stage inference 的成功

因为，一些parties 的 datasets 可能没有具体的 data ID 的 instance

在send data ID 的时候也可能会损害隐私

contribution: 提出了一个新的 training 和 inference的algorithm 来提高 availability 和 privacy

inspired by: knowledge distillation，从一个 large model 到一个 smaller model

区别：

> knowledge distillation 依赖于 student model 的 inference
>
> 本模型： inference 首先在 VFL model (teacher model)上完成，因为缺乏某个party 的 data ID 失败后，会在 student model 上执行 inference

此外，在inference 过程中，通过在多方之间执行 OT 协议来保护信息

通过构建 hash table 和将加密操作移到 preparation stage 提高了效率

实验表明：从 federated model transfer knowledge 到 student model 可以提高在 testing datasets上的性能

分析：federated model 在training 时候有更多的来自其他 parties 的额外的 features knowledge，在federated model 的 supervision下，local model 可以给出接近于 VFL model的性能，可以比只在 local datasets 上的表现更好

8. Discussion and Future Work

8.1 Additional Overhead

在确保 inference success 的时候使用了一个额外的步骤，这可能导致在inference 时候额外的延迟

这种开销可以通过 paralleling the inference on VFL model and student model来缓解

实际上，VFL model 和 student model 上的inference 可以同时开始

因为VFL model需要额外的 communication and computation 会带来更多的 latency, 总的 inference lantency几乎与只进行 VFL inference 相同

8.2 Other VFL Solutions

使用了 Paillier homomorphic encryption 来为密文提供加法操作

但是也有其他的协议可以被应用于 federated learning

such as secure MPC, fully homomorphic encryption (FHE), and Trusted Execution Environment (TEE)

本模型适用于所有以上 cryptographic tools:

(1) framework 不依赖于某个密码学工具，仅用到了在其他VFL solutions 中也支持的加法同态

MPC 在算数秘密共享情况下支持密文加法

TEE在 hardware enclave情况下支持加法

FHE 支持密文加法

(2) 唯一的假设是：feature distribution of dataset，假设了 active party至少有一个feature来在本地执行 inference，这个假设在加密后端为 MPC、FHE 或者 TEE情况下都是成立的