BatchCrypt: Efficient Homomorphic Encryption for Cross-Silo Federated Learning

0. Abstract

Cross-silo federated learning (FL) 使得一些如 financial or medical 可通过 aggregating local gradient updates 训练一个机器学习模型

mask local gradient updates : additively homomorphic encryption (HE)

问题：HE 带来了大量的 computation and communication， dominate the training time

BatchCrypt, a system solution for cross-silo FL

不是去单独 encrypt gradients， 而是 **encode a batch of quantized gradients** 为一个 lone integer, 并一次性加密

使用 gradient clipping technique, 可以允许 gradient-wise aggregation

implementation: BatchCrypt 作为 FATE的一个插件

在几何分布的 datacenters EC2 clients, 训练速度提高了23*93，通宵开销减小了66*101

1. Introduction

数据共享在 data privacy 和 confidentiality情况下是被禁止的

Cross-silo FL 可以打破 “data silos”, 通过 upload local gradient updates到 central server 来协同训练 a global model

对于聚合过程中的隐私问题，$additively\quad homomorphic\quad encryption(HE)$ Paillier crytosystem 是较好的 cross-silo setting, 可以在对学习准确率 loss影响不大的情况下保证强隐私性

在HE 情况下, gradient aggregation 可以直接在 ciphertexts 下操作

Before training: HE key-pair is synchronized across all clients through a secure channel

During training:

1. client encrypts local gradient updates using public key , send ciphertexts to central server.
2. serer aggregates the encrypted gradients, dispatched the result to all clients.
3. client decrypts the aggregated gradients using private key, update local model.

实验表明：

(1) more than 80% of training time is spent on encryption/decryption.

(2) encryption yields larger ciphertexts.

HE in encryption and communication is expensive.

WeBank: cannot afford to use encrypted gradient or limited to less private scenarios.

Solution: a simple batch encryption technique.

- client quantizes gradient values into low-bit integer representations, and encrypts it in one go.
- 与直接加密独立的梯度值相比，batch encryption 显著减少了 encryption overhead and data transfer amount.

batch encryption key challenges:

(1) 直接对两个 batches 的密文求和，解密后的结果，与在明文情况下完成 gradien-wise aggregation on two batches 的效果是一样的。

设计 a customized quantization scheme , quantizes gradient values 到一个对称范围内均匀分布的有符号整数

设计 a new batch encoding scheme, 采用两个 sign bits 的 quantized values compliment representation

使用padding 和 advance scaling to avoid overflow in addition.

(2) gradients values are **unbounded,** be **clipped** before quantization

提出分析模型： dACIQ, 通过 extending ACIQ （clipping technique for ML over centralized data） to cross-silo FL over decentralized data.

dACIQ 可以选择 optimal clipping thresholds.

implementation:
9 个 participating clients geo-distributed in five AWS EC2 datacenters.

协同训练三个模型：

1. 3-layer fully-connected neural network with **FMNIST** dataset
2. AlexNet with **CIFAR** 10 dataset
3. text-generative LSTM model with **Shakespeare** dataset

与FATE 相比，达到23、71、93倍的加速，通信开销减小66、71、101倍

BatchCrypt 精度损失不到1%（可忽略不计）

BatchCrypt提供了第一个在 cross-silo FL framework情况下以 low encryption and communication cost 的高效 HE实现。

2. Background and Related Work

2.1 Cross-Silo Federated Learning

根据应用场景的不同，federated learning 可以分为：

1. cross-device FL

   clients 是 large number of mobile or IoT devices with limited computing power and unreliable communications.

2. cross-silo FL

   clients 是 a small number of organizations (eg. financial and medical) with reliable communications and abundant computing resources.

   与 cross-device FL 相比，cross-silo FL 有显著的对 privacy 和 performance 的要求。

   （1）最终的 model 应该专门发放给参与组织的参与方，任何 external party，包括 central server 都不能直接访问 the trained model.

   （2）强隐私保护强度不应该是以牺牲 learning accuracy 为代价

   （3）作为新的范式， cross-silo FL 在 algorithm 和 systems 都有快速的创新

2.2 Privacy Solutions in Federated Learning

Secure Multi-Party Computation (MPC)

multiple parties collaboratively compute an agreed-upon function，每个参与方除了 自己的 input 和 output外，不会得知任何信息 （zero-knowledge）

MPC 在 clients之间精心设计 computation and synchronous protocols.

协议可以保证 privacy，但是牺牲了 effiency

developers 精心设计 ML algorithms ，划分 parties 之间的 computation，在可能降低 privacy 的情况下，得到 better performance.

Differential Privacy (DP)

injecting noises 来保证 sample 的privacy

recent work: *selective parameter update* 在 data privacy 和 learning accuracy 之间权衡

DP可以很高效地实现，在 aggregation 的时候直接 expose plain gradients to the central server. (recover information from gradients)

Secure Aggregation

server 不从任何的 client 学习 individual updates, 只学习 aggregated updates.

secure aggregation 可以安全用于 cross-device FL，但是在 cross-silo FL失败，reasons:
（1）允许 central server 看到 aggregated gradients, 基于此，外部实体 entity（ 例如，运行 central server 的 public cloud ）可以了解训练过的模型信息

（2）每次迭代，clients 必须同步密钥 和 zero-sum masks， 对于同步训练要求较高

Homomorphic Encryption (HE)

computation directly on ciphertexts

additively HE schemes, notably Paillier：

- 每个 client transfer encrypted local updates to the server for direct aggregation.
- result is sent back to the client for local decryption.

HE meets 3 requirements of cross-silo FL:

（1）保护了训练模型不会被任何的额外的 party learned (server也是)

（2）no learning accuracy loss (no noise)

（3）directly apply to the existing learning systems

cons： HE introduces significant **overhead** to computation and communication.

Summary

- MPC: 提供了强的隐私保证，但是需要重新设计现有的 ML algorithms
- DP: 应用时候 easily and efficiently, 但是 weaker privacy guarantee and potential accuracy loss
- Server aggregation: 对于 cross-device FL is an effective way, 但是导致了 high synchronization cost.

2.3 Cross-Silo FL Platform with HE

如图，是一个典型的 cross-silo FL system.

aggregator 是 server, coordinates the clients and aggregates encrypted gradients. (honest but curious)

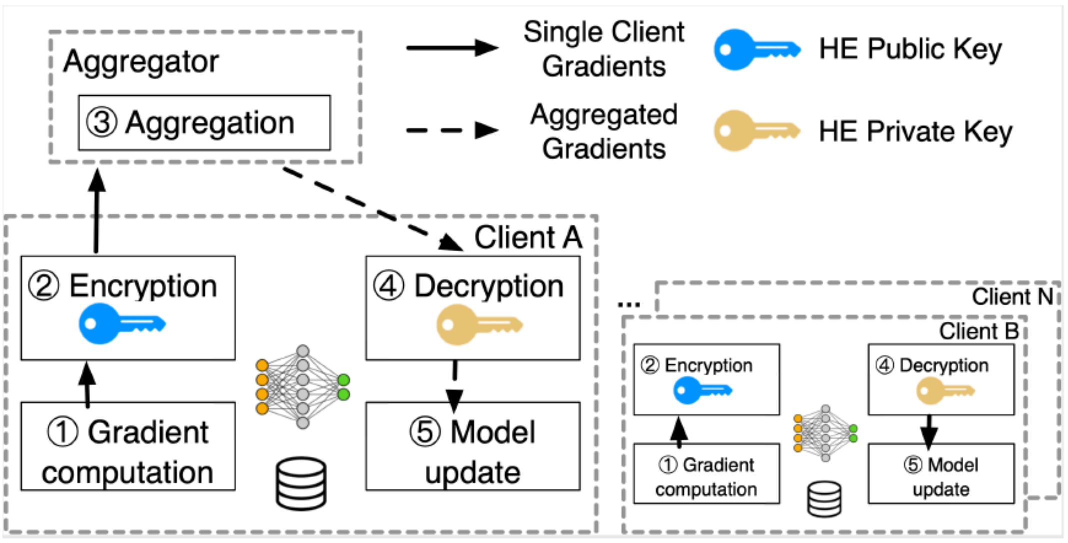


Figure 1: The architecture of cross-silo FL system, where HE is implemented as a pluggable module on the clients.

在 parties 之间的 communication 是由密码协议（SSL / TLS）确保安全

不会有第三方会获取到传输时候的 messages

Overview:

- before training starts, the aggregator 可以随机选择一个 client 作为 leader，来生成 HE key-pair 并且对 all clients 同步。

  并且由 leader 初始化 ML model， sends model weights to others.
- other clients 一收到 HE key-pair and initial weights, clients 就开始训练
- each iteration： each client 计算 local gradient updates, 并用 public key 加密， 发送到 aggregator
- aggregator 直到收到所有 clients 的 updates， 累加后将结果分发给 all clients
- client decrypted the aggregated gradients, update its local model.

该架构 follows the classic distributed SGD pattern. 因此，现有理论和优化算法，包括 synchronization 和 local update SGD 可以自然应用上。

此外，因为模型更新是在 client's side 使用 明文梯度聚合，可以使用最新的 adaptive optimizers, such as Adam 来加快收敛.

3. Characterizing Performance Bottlenecks

介绍 cross-silo FL 在三种真实应用场景下的 deep learning models 的 performance.

encryption 和 **communication** 是两种主要的 bottlenecks 导致 FL 在 organizations 之间无法应用

3.1 Characterization Results

Cross-silo FL 用于 多个 geo-distributed datacenters.

实验：9个EC2 clients 在 five geo-distributed datacenters 协同训练 3 个ML models, FMNIST、CIFAR 、LSTM,

(除非特别强调，否则都是 synchronous training)

实验在FATE基础上开展，使用了内置的 Paillier cryptosystem.

Encryption and Communication Overhead

比较两个FL scenarios, 有没有 HE

HE 会显著增加训练时间和增加了大量的数据传输，与不使用 HE 相比，迭代时间增加了 96倍、135倍、154倍

总的来说，HE 的使用 training time 和网络占用都增加了两个数量级。

（这对于更复杂的models，开销是更大的）

Deep Dive

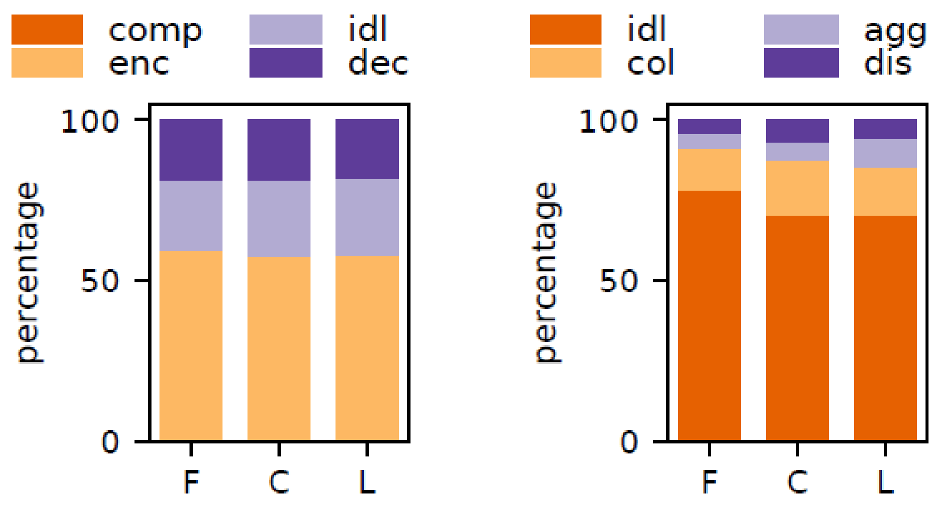
取样测试影响 HE overhead 的方面，

Client: HE-related operations dominate the training time.

> 60% time on gradient encryption.
>
> 20% time on decryption.
>
> 20% time on data transfer and idle waiting for the gradient aggregation to be returned.
>
> < 0.5% for computing the gradients.

Aggregator:

> ＞70% time on idle waiting for a client to send encrypted gradients.
>
> Collecting the gradients from all clients
>
> Dispatch the aggregated results to each party (clients are geo-distributed)
>
> < 10% time on computation time for gradient aggregation.



(a) Client: **comp**ute, **enc**rypt, **idl**e, **dec**rypt

(b) Aggregator: **idl**e, **col**lect, **agg**regate, **dis**patch

Figure 2: Iteration time breakdowns of **F**MNIST, **C**IFAR, and **L**STM for a client and the aggregator.

Why is HE So Expensive

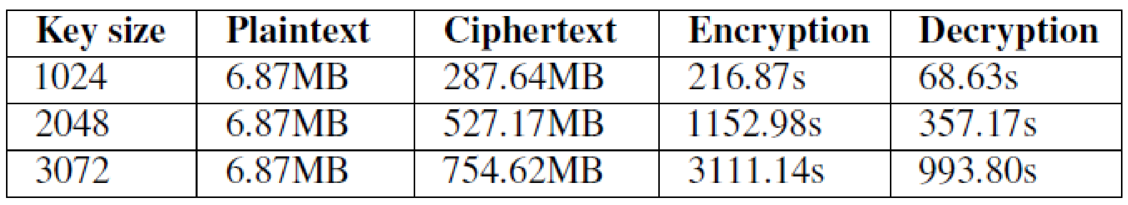
以加法HE 为例，Paillier 在 encryption 和 decryption 上：

1. 引用了大指数和大模数的多重模乘法和取幂运算 （通常 > 512bits）
2. Encryption 后产生了更长的 **ciphertexts** (对于 data transfer 带来了更大了开销)

Benchmark:

computation overhead and **inflated ciphertexts** of Paillier with varying key sizes.

Table 1: Benchmarking Paillier HE with various key sizes.



Key size	Plaintext	Ciphertext	Encryption	Decryption
1024	6.87MB	287.64MB	216.87s	68.63s
2048	6.87MB	527.17MB	1152.98s	357.17s
3072	6.87MB	754.62MB	3111.14s	993.80s

key size 越大，意味着 higher security.

computation overhead and ciphertexts size 都是线性增长

Summary

（1） 使用高端硬件设备(GPUs and TPUs) 加速已经不再重要，这是对 clients' datacenters 的大规模基础设施的浪费

（2）在geo-distributed datacenters中极大的 network traffics 招致了高昂的网络数据收费，使得 cross-silo FL在经济上不可行

实际上，对于安全性需求不严格的场景，会选择性关闭 HE

3.2 Potential Solutions and Their Inefficiency

Hardware-Accelerated HE

（1）通常的HE 加密方案, eg. Paillier, 有一些限制的独立的运算，因此加速一个单独的HE operation 是非常受限的(FPGA仅能加速 3×)

（2）仅对于 encryption 自身进行加速是无法减少 communication overhead.

Reducing Communication Overhead

data inflation 主要是由于 plaintexts 和 ciphertexts 长度不一致导致的

intuitive idea：

batching gradients together as many as possible to form a long plaintext.

这样可以显著减少加密的次数

challenges: maintain HE's additive property after batching (不影响 ML 算法 和 learning acc)

没有工作系统地介绍 batching, SIMD 可以加速 lattice-based HE schemes.

招致了更多的 computational complexity for lattice-based HE.

现有的加速只能加速 integer cryptographic.

4. BatchCrypt

（1）batching 需要 gradient quantization

通用的 quantization 缺少 flexibility and efficiency

（2） an efficient clipping method to prevent model degradation.

4.1 Why is HE Batching for FL a Problem

batching 技术已经被用于加速 queries 在 Paillier-secured database.

但该技术只适用于 non-negative integers

为了支持浮点数, values 需要被 recorded and grouped by their exponents.

SIMD技术受限于 lattice-based cryptosystems.

目标：a universal batching method for all additively homomorphic cryptosystems.

Why Quantization is Needed

gradients 是有符号浮点数，必须按照模型的权重排列他们，不能简单地按照指数重新排列他们

可行方法：在批处理中使用梯度的整数表示，这需要 quantization.

Existing Quantization Schemes

假设 gradient g 被量化到一个 8-bit 的无符号整数

quantized value of g:

$$Q(g) = [255 * (g - min)/(max - min)]$$

(max = 1, min = -1)

假设 n 个量化的 gradients 被求和，结果表示为 q_n

dequantized value of q_n:

$$Q^{-1}(q_n) = q_n * (max - min)/255 + n * min$$

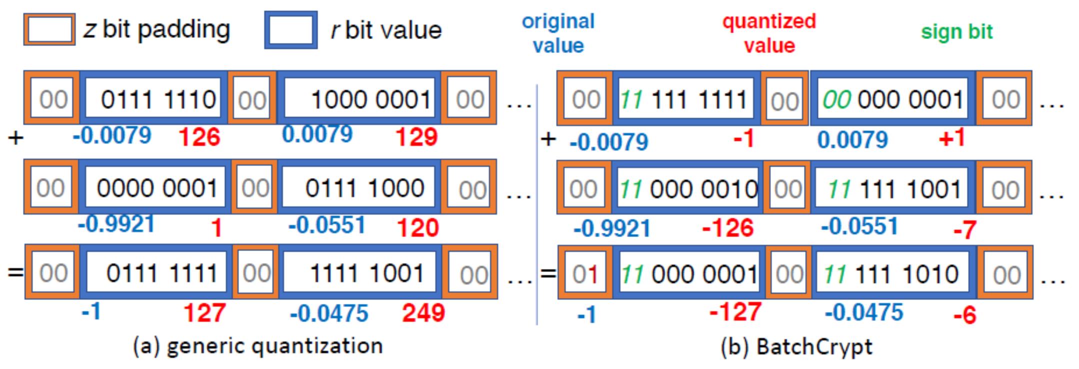


Figure 3: An illustration of a generic quantization scheme and BatchCrypt. The latter preserves additivity during batching, with the sign bits highlighted within values.

overview:

gradients of a client (floating numbers) are first quantized.

batch joined into a large integer.

aggregate the gradients of two clients.

dequantize them to obtain the aggregated results.

limitations:

（1）为了能够 dequantize 成功，必须要知道 aggregated values 的数量，但是这就对灵活性和同步带来了障碍

（2）在 aggregation 时候很容易 overflow, values 都被量化为 正整数

解决方法： batched ciphertexts 必须在添加一些内容后解密，并再次加密

（3）不区分正数和负数的溢出

4.2 HE Batching for Gradients

不满足于现有的量化技术，设计一种新的 batching for gradient aggregation.

The properties are as follows:

（1）保留 HE 的加法特性

（2）可区分 positive overflows from negative ones.

（3）适用于现有的ML 算法和优化技术

（4）足够灵活来 dequantize values，不需要知道额外的信息(eg. 聚合的数量)

Gradient Quantization

现有技术：通过 gradient compression 技术来减少网络流量

（这些方法主要是在传输过程中 compress values 或者加速 inference）

但是这些并不是为了 gradient aggregation 而设计，不能在 compressed gradient 上面进行高效的计算

Quantization requirements:

- Signed Integers

  gradients 要被 quantized into signed integers, 这样在聚合时正负值相互抵消，更不容易 overflow

- Symmetric Range

  量化范围必须是对称的，否则会导致聚合结果的不正确

- Uniform Quantization

  梯度是非均匀分布的，量化也是非均匀分布可以有更好的 compression，但是这样无法利用加法特性

BatchCrypt

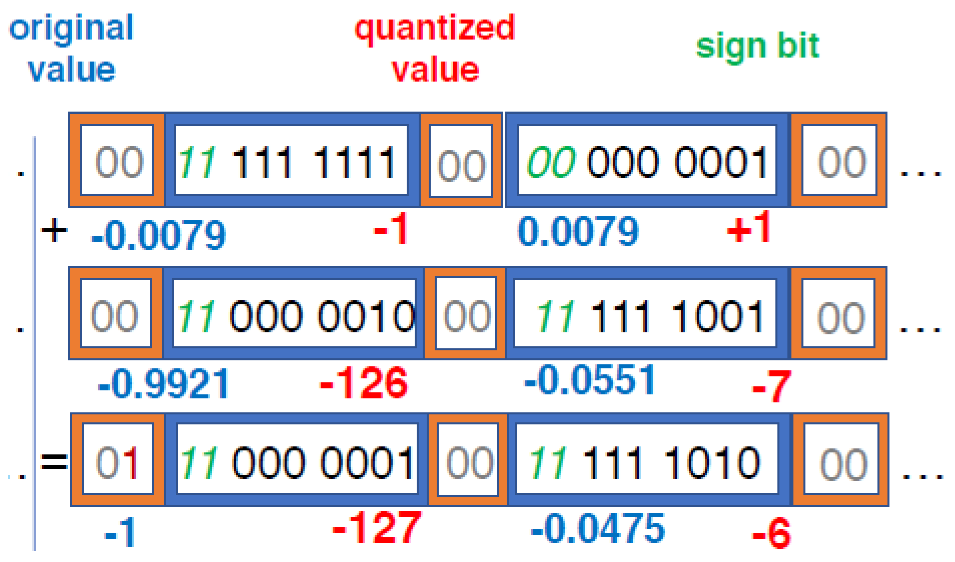
quantize a gradient in $[-\alpha, \alpha]$ into an r-bit integer

uniformly map $[-\alpha, 0]$ and $[0, \alpha]$, to $[-(2^r - 1), 0]$ and $[0, 2^r - 1]$

(0 是以两种codes映射的)

使用 two's complement representation (两位补码)

符号位可以参与加法运算，使用两个符号位来区分 正溢出 和 负溢出



(b) BatchCrypt

4.3 dACIQ: Analytical Clipping for FL

现实中， gradients 可能是 unbounded, 并且需要在 quantization 之前 clipped

gradients 在不同的 layers 有不同的分布

同一层的 gradients 有一个类似 Gaussian 的 bell-shaped distribution.

此特性可以用来高效进行 gradient compression.

找到一个 optimal clipping thresholds unclear.

clipping 裁剪 是将梯度饱和到一个阈值为α的过程

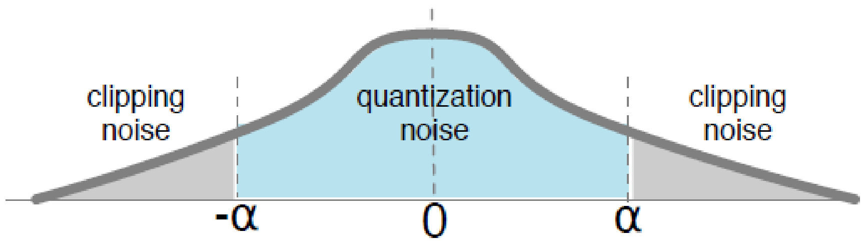


Figure 4: A typical layer gradient distribution. α is the clipping threshold.

两种方法设置 clipping threshold:
（1）profiling-based clipping

选择一个 sample dataset，获得一个样本梯度分布

使用 KL divergence and convergence rate.

但是在FL中不切实际

（2）analytical modeling

- Quantization noise 在 clipping 范围内产生的误差
- Clipping noise 是超过裁剪阈值的饱和范围

为了研究 noise 的影响，最新的 clipping 技术 ACIQ, 假设 quantization 和 clipping 都服从 Gaussian分布

但是 ACIQ 不可直接用于 BatchCrypt:

（1）采用了非对称量化

（2）在 FL 中无法使用明文梯度

accumulated error in BatchCrypt:

前两项是 clipping noise，第三项是 stochastic rounding noise，在知道σ 的情况下就可得到最优的门限值 α

$$\begin{aligned} E[(X-Q(X))^2] &= \int_{-\infty}^{-\alpha} f(x)\cdot(x+\alpha)^2 dx + \int_{\alpha}^{\infty} f(x)\cdot(x-\alpha)^2 dx \\ &+ \sum_{i=0}^{2^r-3}\int_{q_i}^{q_{i+1}} f(x)\ \cdot[\ (x-q_i)^2\cdot(\frac{q_{i+1}-x}{\triangle})\ +\ (x-q_{i+1})^2\cdot(\frac{x-q_i}{\triangle})\ ]dx \\ &\approx \frac{\alpha^2+\sigma^2}{2}\cdot[1-erf(\frac{\alpha}{\sqrt{2}\sigma})]-\frac{\alpha\cdot\sigma\cdot e^{-\frac{\alpha^2}{2\cdot\sigma^2}}}{\sqrt{2\pi}}+\frac{2\alpha^2\cdot(2^r-2)}{3\cdot 2^{3r}}, \end{aligned} \tag{1}$$

Gaussian Fitting

传统方法是使用最大似然估计和贝叶斯推理 （对于 FL 大规模参数不适用）

dACIQ 方法：只需要观测集的大小和 max and min

Advance Scaling

假设 m 个clients

提前设定 quantization range 是 clipping range 的 m 倍，这样 sum of gradients 就不会 overflow

4.4 BatchCrypt：Putting It All Together

- Initialization

  aggregator 随机选择一个 client 作为 leader, 由其生成 HE key-pair，并初始化模型 weights, 之后在其他 client workers 中同步这些 key-pair 和 weights

- Training

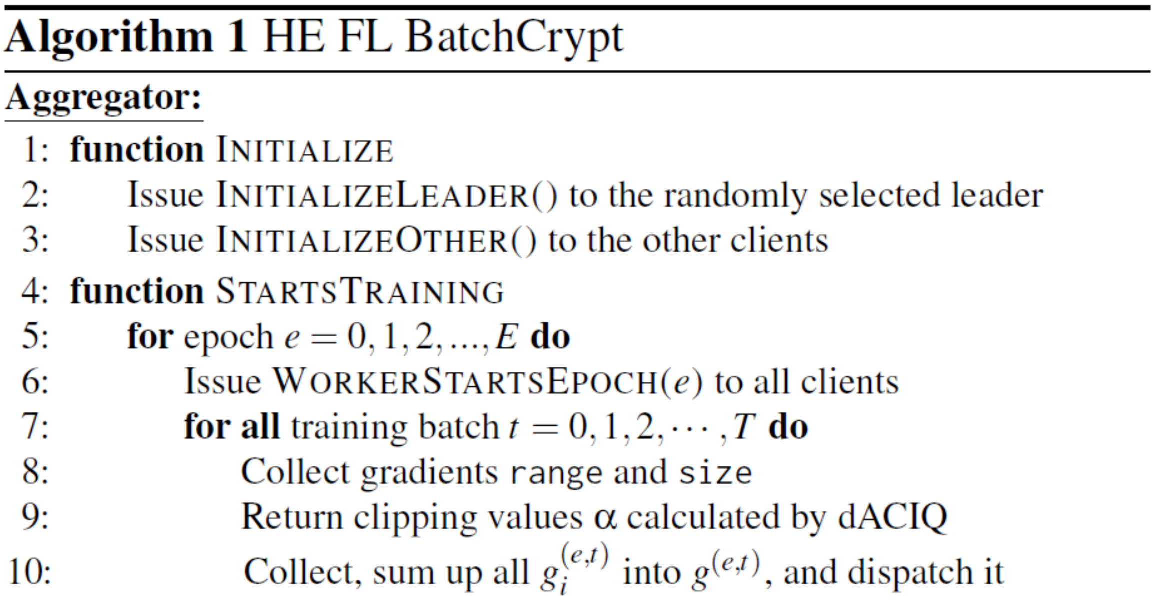
  此处，leader与其他 workers 没有区别

  clients 计算 gradient，并发送到aggregator

  aggregator 估计 Gaussian parameters, 并计算每层的 clipping thresholds.

  clients quantize the gradients (range scaled by number of clients), 并使用 BatchCrypt 加密 quantized values.

  aggregator summed up the encrypted gradients, 并返回给 clients.



Algorithm 1 HE FL BatchCrypt

Aggregator:

```
1: function INITIALIZE
2:     Issue INITIALIZELEADER() to the randomly selected leader
3:     Issue INITIALIZEOTHER() to the other clients
4: function STARTSTRAINING
5:     for epoch $e = 0, 1, 2, ..., E$ do
6:         Issue WORKERSTARTSEPOCH($e$) to all clients
7:         for all training batch $t = 0, 1, 2, \cdots, T$ do
8:             Collect gradients range and size
9:             Return clipping values $\alpha$ calculated by dACIQ
10:            Collect, sum up all $g_i^{(e,t)}$ into $g^{(e,t)}$, and dispatch it
```

```
Client Worker: $i = 1, 2, \ldots, m$
    – $r$: quantization bit width, $bs$: BatchCrypt batch size
 1: function INITIALIZELEADER
 2:     Generate HE key-pair pub_key and pri_key
 3:     Initialize the model to train w
 4:     Send pub_key, pri_key, and w to other clients
 5: function INITIALIZEOTHER
 6:     Receive HE key-pair pub_key and pri_key
 7:     Receive the initial model weights w
 8: function WORKERSTARTSEPOCH($e$)
 9:     for all training batch $t = 0, 1, 2, \cdots, T$ do
10:         Compute gradients $g_i^{(e,t)}$ based on $w$
11:         Send per-layer range and size of $g_i^{(e,t)}$ to aggregator
12:         Receive the layer-wise clipping values $\alpha$'s
13:         Clip $g_i^{(e,t)}$ with corresponding $\alpha$, quantize $g_i^{(e,t)}$ into $r$ bits, with
    quantization range setting to $m\alpha$                      ▷ Advance scaling
14:         Batch $g_i^{(e,t)}$ with $bs$ layer by layer
15:         Encrypt batched $g_i^{(e,t)}$ with pri_key
16:         Send encrypted $g_i^{(e,t)}$ to aggregator
17:         Collect $g^{(e,t)}$ from aggregator, and decrypt with pub_key
18:         Apply decrypted $g^{(e,t)}$ to $w$
```

5. Implementation

Overview

most of efforts are made on client side.

clients 的架构如下：

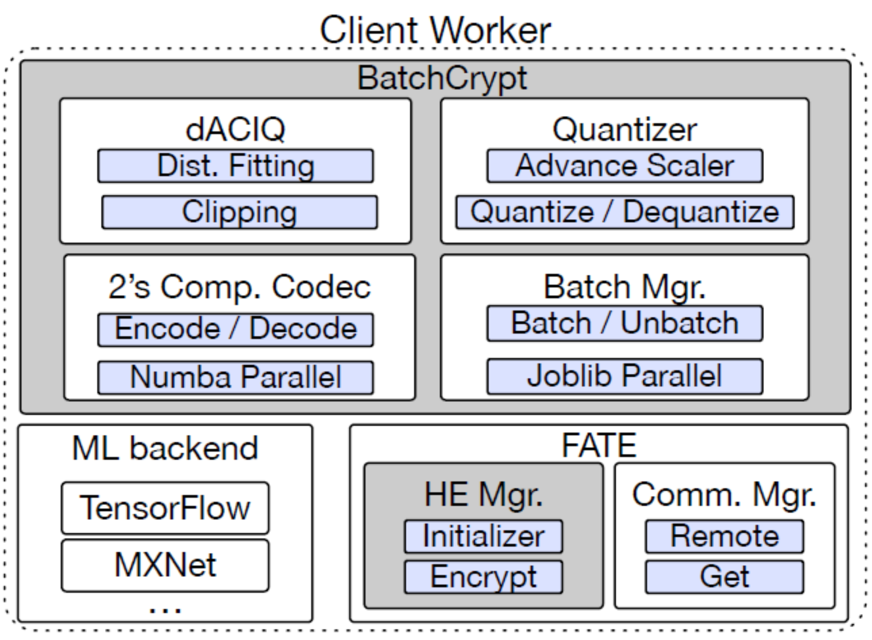


Figure 5: The architecture of a client worker in BatchCrypt.

consists of dACIQ, Quantizer, two's Compliments Codec, and Batch Manager.

dACIQ: Gaussian fitting and clipping threshold calculation.

Quantizer: takes threshold to Advance Scaler and dequantization.

Two's Compliments: 量化值的转换 encode， Numba enables Parallel

BatchManager：管理 batch/ unbatch 的 gradients， 利用 joblib 通过多进程执行计算

Model Placement

传统 PS 架构，model wights 保存在 server side.

BatchCrypt: 将 weights 放在worker side

传统方法：

clients 使用HE加密初始化的weights，并发送给 aggregator

aggregator将收到的加密后的gradients应用到用相同 HE key加密的weights

Drawbacks:

(1) 将weights放在 aggregator 需要 re-encryption

(2) 应用加密的梯度，无法使用复杂 ML optimizers

6. Evaluation

6.1 Methodology

Setting

a geo-distributed FL scenario

9 clients in five AWS EC2 datacenters.

不适用GPU，因为计算不是 bottleneck.

Benchmarking Models

- 3-layer fully-connected neural network on FMNIST dataset.

  (batch size = 128, adopt Adam optimizer)

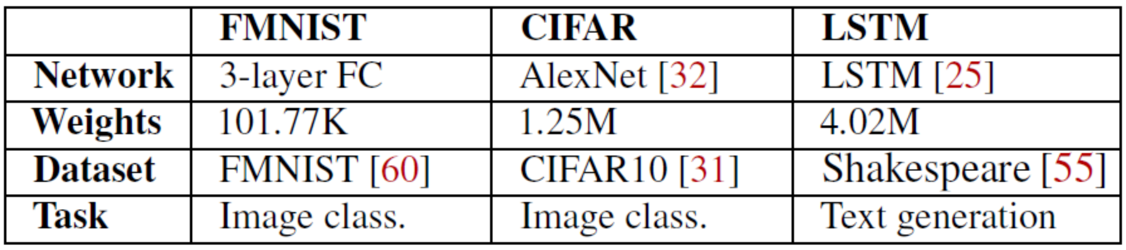
- AlexNet on CIFAR10 dataset

  (batch size = 128, RMSprop optimizer with 10^{-6} dacay.)

- LSTM model on Shakespeare dataset

  (batch size = 64, adopt Adam optimizer)

Table 3: Summary of models used in characterizations.



	FMNIST	CIFAR	LSTM
Network	3-layer FC	AlexNet [32]	LSTM [25]
Weights	101.77K	1.25M	4.02M
Dataset	FMNIST [60]	CIFAR10 [31]	Shakespeare [55]
Task	Image class.	Image class.	Text generation

6.2 Impact of BatchCrypt’s Quantization

查看 quantization bit 是多少会影响 model quality.

对 9 个 clients 使用 BatchCrypt's scheme including dACIQ clipping.

以 plain training 作为 baseline，quantization bit width 8,16,32.

FMNIST：

> plain baseline: acc 88.62% 40epochs
>
> 8-bit: acc 88.67% 122epochs

> 16-bit: acc 88.37% 68epochs
>
> 32-bit: acc 88.58% 32epochs

CIFAR:

> plain baseline: acc 73.97% 285epochs
>
> 8-bit: acc 71.47% 234epochs
>
> 16-bit: acc74.04% 279epochs
>
> 32-bit: acc73.91% 280epochs

LSTM:

> plain baseline: loss0.0357 20epochs
>
> 8-bit: loss0.1359 29epochs
>
> 16-bit: loss0.0335 23epochs
>
> 32-bit: loss0.0386 22epochs

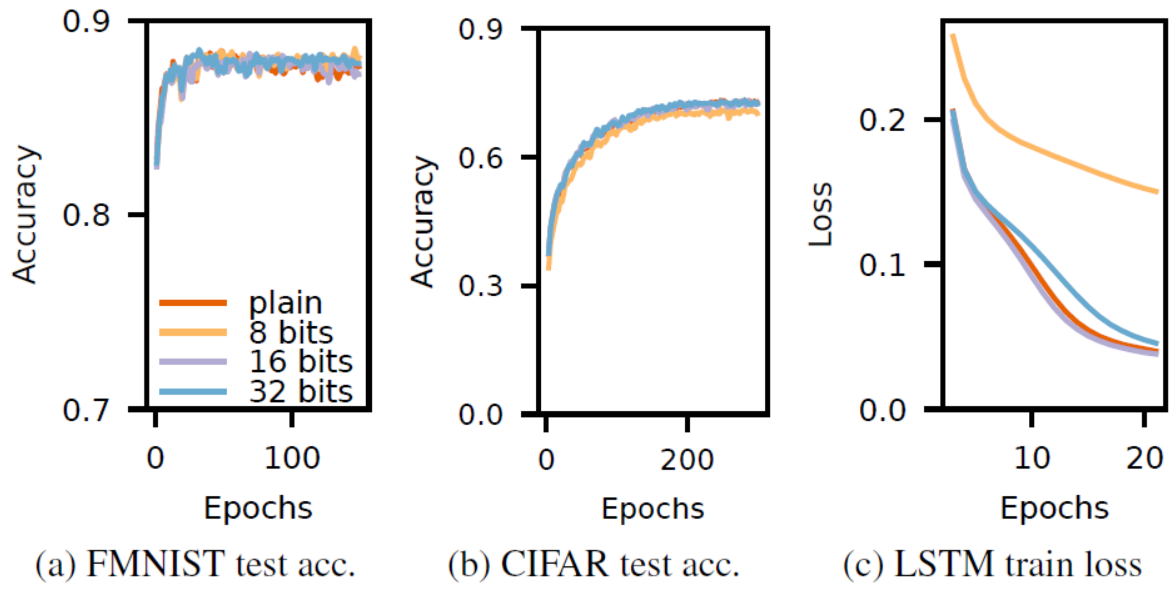


Figure 6: The quality of trained model with different quantization bit widths in BatchCrypt.

Conclusion:

在适当的 quantization bit 下，**BatchCrypt**的量化对模型的负面影响可忽略不计（有的地方epochs较大，这可以通过BatchCrypt加速弥补）

注意：longer bit width 并不一定意味着更高模型质量

量化训练可以看作 regularizer，reduce overfitting, 类似于 a dropout layer.

Summary:

- 在合适的位宽条件下，梯度量化方案不会对训练后的模型质量产生不利影响
- 采用BatchCrypt时不需要考虑量化引起的错误

6.3 Effectiveness of BatchCrypt

BatchCrypt vs. FATE

> quantization bit width to 16
>
> batch size to 100
>
> iterations to 50

communication time （idle in worker and transfer in aggregator）显著减少

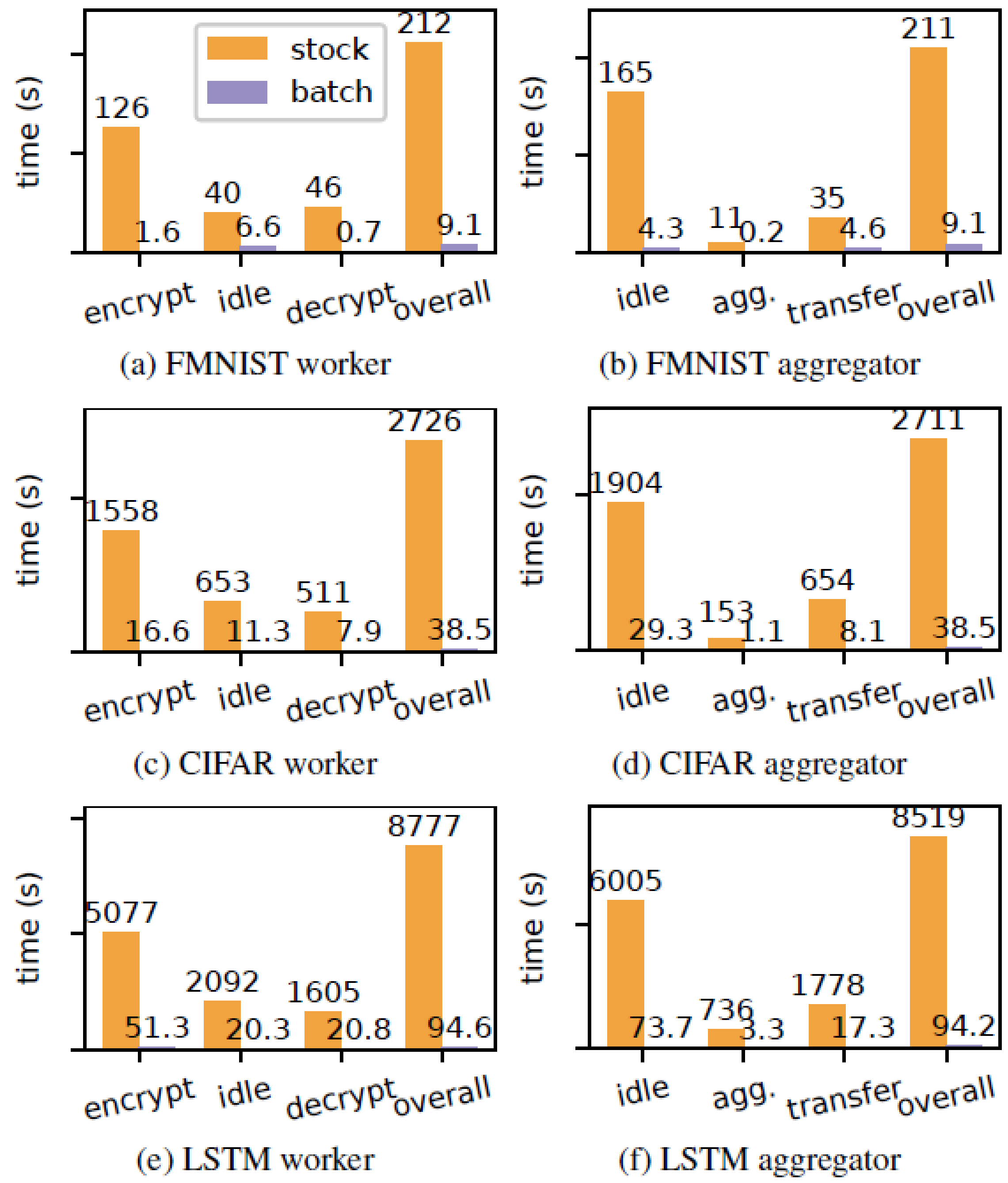


Figure 7: Breakdown of training iteration time under stock FATE and BatchCrypt, where "idle" measures the idle waiting time of a worker and "agg." measures the gradient aggregation time on the aggregator. Note that model computation is left out here as it contributes little to the iteration time.

BatchCrypt 大大减小了网络占用 66、70.5、101.2倍

BatchCrypt 在 larger model 上效果更好：

（1）more encryption-related operations for BatchCrypt

（2）higher chances of forming long batches.

BatchCrypt 加速达到了两个数量级，这完全可以抵消量化带来的开销。

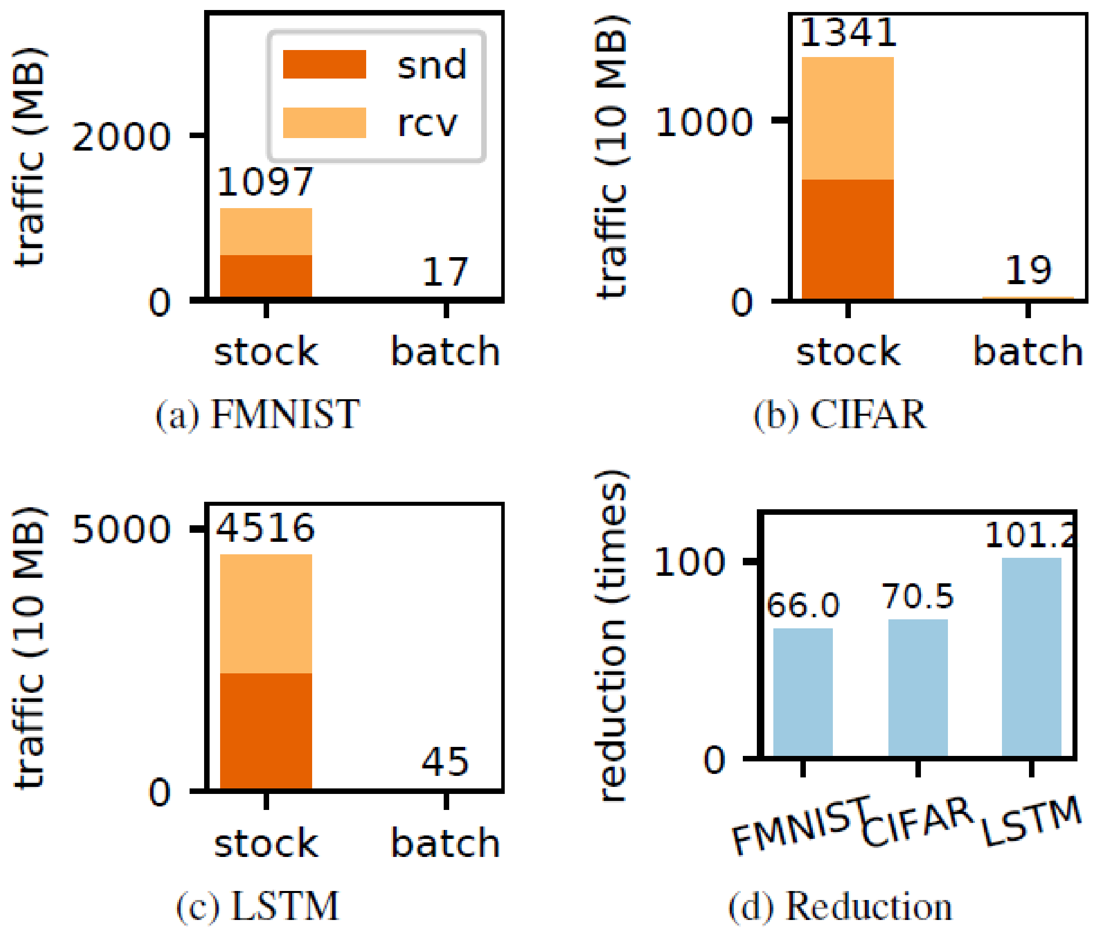
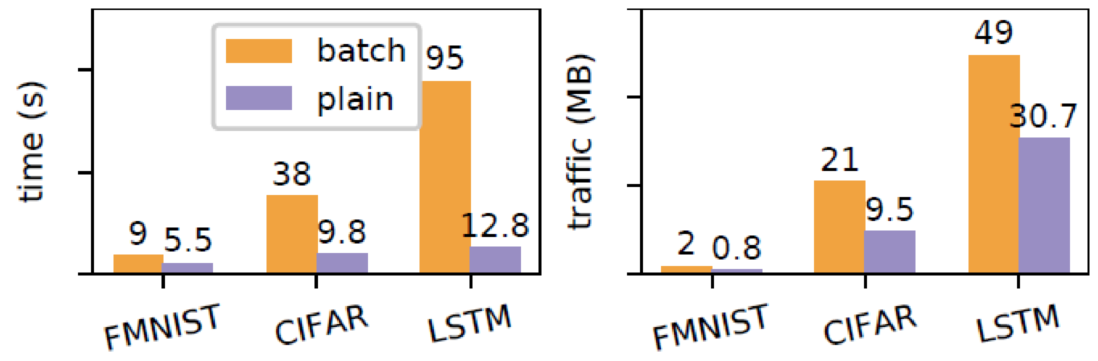


Figure 8: Comparison of the network traffic incurred in one training iteration using the stock FATE implementation and BatchCrypt.

BatchCrypt vs. Plaintext Learning

encryption remains the major bottleneck

迭代时间和网络占用：



(a) Time (b) Communication

Figure 9: Time and communication comparisons of one iteration on workers between BatchCrypt and plain distributed learning without encryption.

Training to Convergence

Table 4: Projected total training time and network traffic usage until convergence for the three models. The converged test accuracy for FMNIST, CIFAR as well as loss for LSTM and their corresponding epoch numbers are listed in the table.

Model	**Mode**	**Epochs**	**Acc./Loss**	**Time** (h)	**Traffic** (GB)
FMNIST	stock	40	88.62%	122.5	2228.3
	batch	68	88.37%	8.9	58.7
	plain	40	88.62%	3.2	11.17
CIFAR	stock	285	73.97%	9495.6	16422.0
	batch	279	74.04%	131.3	227.8
	plain	285	73.97%	34.2	11.39
LSTM	stock	20	0.0357	8484.4	15347.3
	batch	23	0.0335	105.2	175.9
	plain	20	0.0357	12.3	10.4

- 与 FATE stock 相比，BatchCrypt减少训练时间13.76、72.32、80.65倍

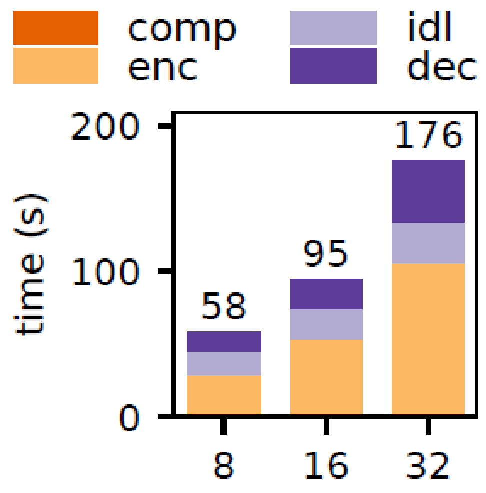
  network footprints shrink by 37.96、72.01、87.23

- 与 Plain learning 相比，BatchCrypt 仅慢了1.78、2.84、7.55倍（plain 是不需要加密的）

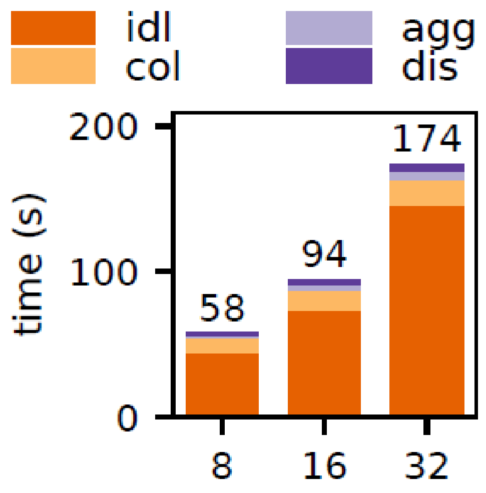
6.4 Batching Efficiency

BatchCrypt 可以有效处理量化值，不考虑 quantization bit width.

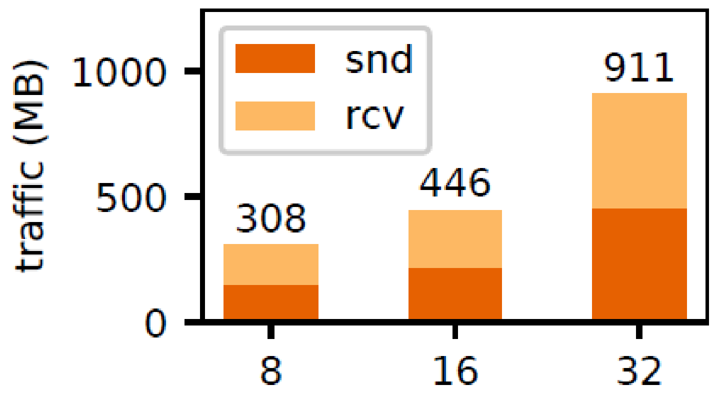
a shorter quantization bit width enables a larger batch size, leading to a shorter training time.



(a) Worker: **comp**ute, **enc**rypt, **idl**e, **dec**rypt



(b) Aggregator: **idl**e, **col**lect, **agg**regate, **dis**patch



(c) Aggregator traffic

Figure 10: Breakdown of iteration time and communication traffic of BatchCrypt with LSTM model with various quantization bit widths in one iteration. The corresponding batch sizes for bit width 8, 16, and 32 are 200, 100, and 50, respectively.

8-bit到16-bit 的提升没有 16-bit 到32-bit的提升大，因为HE操作在较大batch size 的情况下影响变小

BatchCrypt的批处理方案随着批处理大小的增加而线性地降低计算和通信成本。

6.5 Cost Benifits

BatchCrypt降低了97.4%，98.6%、98.8%的网络成本

7. Discussion

Local-update SGD & Model Averaging

only additional operations involved, BatchCrypt can be easily adopted.

Split Model Inference

BatchCrypt可用于加速中间推理结果的加密和传输。

Flexible Synchronization

our design allows it to take advantage of the flexible synchronization schemes

Potential on Large Models

某些模型即使使用1位或2位量化也能收敛

Applicability in Vertical FL

批处理这种计算超出了BatchCrypt当前的能力

8. Concluding Remark

- 系统地利用HE实现安全的 cross-silo FL
- BatchCrypt:一批梯度编码为长整数，并加密，大幅减少加密开销和密文总量
- 与 sotck FATE 相比，加速81倍，减少通信开销达到101倍，部署到云环境时候节省99%的开销