一个隐私保护深度学习的通用框架

Abstract

我们详细介绍了一个新的隐私框架，以保护深度学习并讨论其优点。该框架重视所有权和数据的安全处理，并基于命令链和张量引入了有价值的表示。这种抽象**允许人们实现复杂的隐私保护结构，如联合学习，安全多方计算和差异隐私，同时仍然向最终用户公开熟悉的深度学习API**。我们报告基于Boston Housing和Pima Indian Diabetes数据集的早期结果。虽然除差异隐私之外的隐私功能不会影响预测准确性，框架的当前实现在性能方面引入了显着的开销，将在开发的后期阶段解决。里程碑，第一个通用框架基于隐私保护的深度学习框架。

1 Introduction

安全多方计算（SMPC）作为在不信任环境中执行操作而不泄露数据的方式正变得越来越流行。在机器学习模型的情况下，SMPC将保护模型权重，同时允许多个工作节点使用自己的数据集参与训练阶段，比如联合学习。但是，已经证明，安全训练的模型仍然容易受到逆向工程攻击，这些攻击可以直接从模型中提取有关数据集的敏感信息。标记为差分专用（DP）方法的另一组方法解决了这个问题，并且可以有效地保护数据。我们为每个PyTorch用户提供透明的隐私保护框架，保护深度学习，从直观的界面中使用FL，MPC和DP。我们展示了框架支持MPC和DP解决方案的各种实现的能力，并报告了在联合学习环境中分别为MPC和DP实例化SPDZ和时刻会计方法时获得的结果。

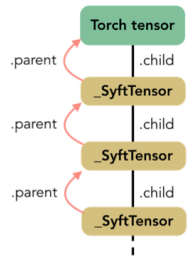


Figure 1: General structure of a tensor chain

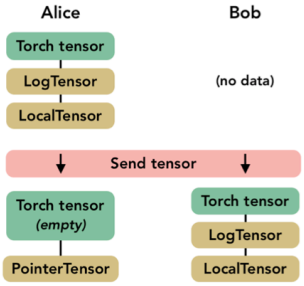


Figure 2: Impact of sending a tensor on the local and remote chains

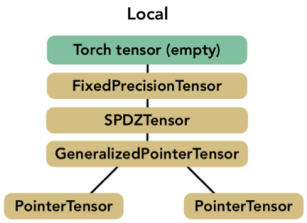


Figure 3: Chain structure of a SPDZ tensor

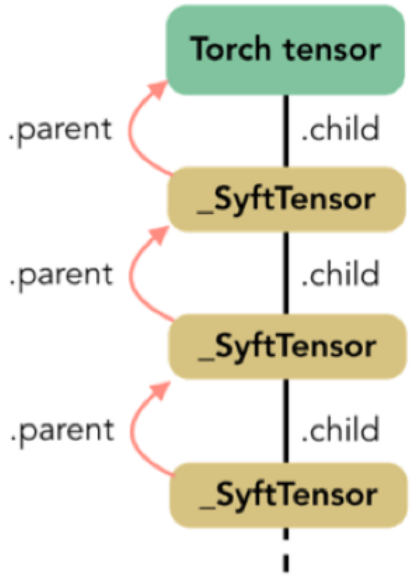
我们的主要贡献如下：

- 建立了一个**标准化协议**在worker之间的通信，协调联合学习
- 开发一个基于Tensor的链式抽象模型，可以重写操作 例如发送和共享一个Tensor 在两个worker之间。
- 运用框架实现 最近提出的dp和mpc算法。

2. abstract operations on Tensors

2.1 the chain structure

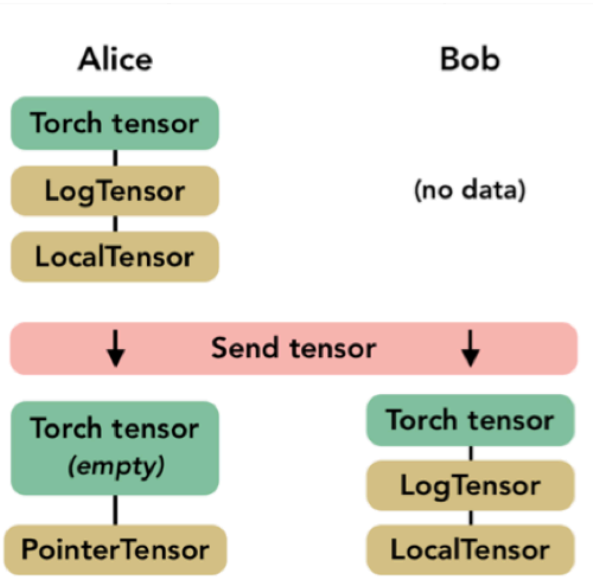
执行转换或向其他worker发送Tensor可以表示为一系列操作，每个操作都由一个特殊的类来体现。为此，我们创建了一个名为SyftTensor的抽象类。



SyftTensor有两个重要的子类。

首先，LocalTensor是在实例化Torch张量时自动创建的。它的作用是在Torch张量上执行与重载操作相对应的本机操作。

Second, the PointerTensor which is created when a tensor is sent to a remote worker.



the whole chain is sent to the worker and replaced by a two-node chain

2.2 From virtual to real context execution of federated learning

[local worker <---> 1..* remote workers]

Virtual Workers -- test

Network worker:1. Web Socket workers; 2.plain network sockets

3 Towards a Secure MPC framework

3.1 Building an [MPCTensor]

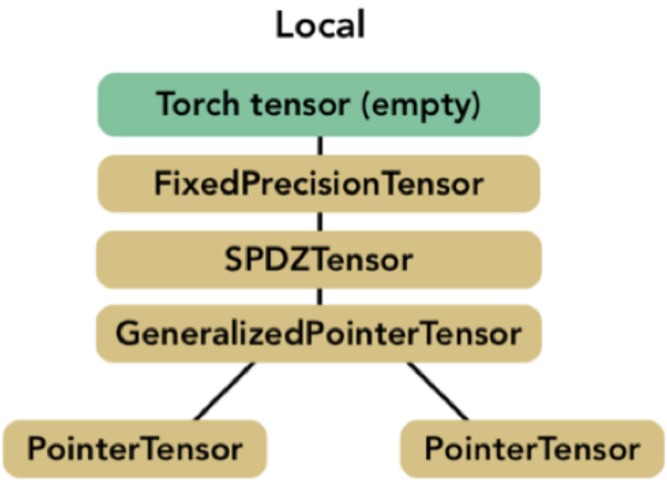
第2节中介绍的元素构成了创建MPCTensor所需的构建块。可以使用一系列的PointerTensors完成拆分和发送共享Tensor。

The MPC toolbox(proposed in our framework implements the SPDZ protocol from [3, 2])： **[2]** 即同态加密的多方计算. **[3]实用秘密安全的mpc 对不信任网络**

1. 基本操作，例如加法和乘法，还包括预处理工具，用于生成例如用于乘法的三元组，
2. 以及包括矩阵乘法在内的神经网络的更具体操作

对卷积网络的传统元素进行了一些调整：1. 使用average pooling而不是max pooling；2. 近似高度sigmoid而不是relu作为激活函数

SPDZ协议假设数据以整数形式给出，我们在链中添加了一个名为FixedPrecisionTensor的节点，该节点将浮点数转换为固定精度数。



1. Unlike the MPC protocol proposed by[2]; players are not equal in our framework

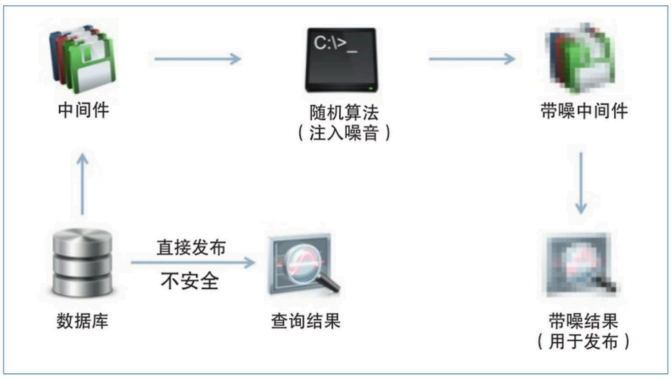
the local worker（模型的所有者）是一个leader controlling the training procedure on all the other players (the remote workers)。

localworker可以创建远程共享张量，这个张量基于的数据他是无法看到和不拥有。

So far, the current implementation does not come with a mechanism to ensure that every player behaves honestly. An interesting improvement would be to implement MAC authentication of the secret shared value.

3.2 Applying Differential Privacy

基本思想



其提供了适度（“单个数字”）隐私预算内的深度神经网络的训练方法。为了实现这一目标，本文提供了用于仔细调整所需噪声的隐私损失的新估计，以及提高private training 效率的新算法.

1. 随机梯度下降（SGD）而不是以相同的方式在数据集和epoch上迭代，训练由阶段组成，每个阶段包括从数据集的N个项目中抽样L个项目并使用它们来升级模型.

2. 直接重用了[1]提供的privacy accountant，但实施了我们自己的sanitizer，它可以clips gradients and adds Gaussian noise

实际应用中：1. 首先，当抽样时，我们随机选择一个worker并在自己的数据中进行抽样。2. 其次，对remote worker进行sanitizer，以便有效地确保数据隐私。这样，local worker将获得用于更新模型的安全梯度，该模型不能公开关于数据集的信息。

另一种方法，通过使用预训练和未发表模型（教师）的嘈杂和聚合投票训练最终模型（称为学生模型）来确保差异隐私。它目前正在实施，并将作为我们框架中的另一个DP Tensor进行整合。

4 Results and discussion

Training mode	Training time (s)
PySyft (Virtual)	10.1
PySyft (Socket)	14.6
PySyft (Virtual) + DP*	15.3
Pure PyTorch	0.22

Table 1: Training time using different training settings on the Boston Housing dataset (10 epochs)
**Equivalent time for the same number of batches processed for DP*

(ϵ, δ)-privacy	Boston MSE	Pima Acc.
$(0.5, 10^{-5})$	29.4	60.6%
$(1, 10^{-5})$	29.2	64.2%
$(2, 10^{-5})$	28.5	66.1%
$(4, 10^{-5})$	28.6	67.1%
no privacy	23.7	70.3%

Table 2: Accuracy of differentially private federated learning on the Boston Housing and Pima Diabetes datasets

1. 表1： 耗时高

2. 表2： 对比； 增加e以牺牲数据隐私为代价改进模型

最后一个观察是，启用DP后收敛速度要慢得多。在50次采样的第一阶段，MSE的值保持在500的范围内。然后MSE开始下降并稳定地达到10-50 MSE值。有两个原因可以解释这种行为：首先，梯度削波会降低最后一层更新的效率，其次，高斯噪声会干扰梯度建议的更新，因此梯度更低，因此效率更低。注意，提高梯度限幅也会增加高斯噪声的方差。

5 Conclusions

一个基于PyTorch构建的隐私保护联合学习框架。该设计依赖于在local和remote worker之间交换的张量链。我们的张量实现支持PyTorch API的命令，并在同一框架内结合MPC和DP功能。

最重要的是减少训练时间。效率尚未得到解决，但目前的开销表明，纯粹的Python框架存在改进的空间，而高级Python API则依赖于优化的低级库。另一个问题与SMPC有关，以确保检测并防止恶意企图破坏数据或模型。

hook.py; tutorial federating learning