Secure Multiparty Computation in the context of Deep Learning

Raj Patil : CS18BTECH11039

November 17, 2021

Contents

Secure Evaluation of Quantized Neural Networks\

1 Abstract

1.1 Focus of the paper

- support from current frameworks for secure evaluation
  - what is secure evaluation?
    - MPC (secure multiparty evaluation being one solution)
- extent the functionality is already present
- trade-offs between different network meta-parameters
  - size, model type, etc
  - overhead of active security over passive attacks

2 Introduction

2.1 Basic overview of Neural Networks

- use of data : of sensitive nature

2.2 Why secure evaluation is needed?

- most data a model is trained on is of sensitive nature
  - medical imaging data, personal profile
  - should remain private.
- Overhead?
  - the training itself is considered to be the bottle-neck and hence enforcing privacy should present itself to be relatively hectic.

2.3 What kind of privacy?

- place infographic for the evaluation phase
- parties being model owner, and image owner
- image-owner should not know about the internals of the model
  - internals of the model covers information about the dataset images as well as the network needing to be treated as a blackbox
- model-owner should not know about the characteristics of the image that was sent for evaluation.

2.4 Secure Multiparty Computation (MPC)

- performing computation on data that needs to be kept secret
- the client-server model
  - model and data owner both share their secrets (model and data) called as secret-share to a set of servers which then run the computation over these inputs.

- on secret sharing
- https://medium.com/dropoutlabs/secret-sharing-explained-acf092660d97
- https://en.wikipedia.org/wiki/Secure_multi-party_computation
  - use to define MPC protocols

3 Towards deploying secure inference

3.1 SOTA

- Garbled circuits
- MPC

3.2 Challenges

3.2.1 ML aspect

- the data-model loop is strenous
- existing solutions introduce constraints:
  - specialized activation functions (CryptoNets)
  - specialized training process (XONN)
- training and secure delivery can only be decoupled to an extent
  - translating to and from specialized approximations results in loss of accuracy
    - floating to fixed point changes, etc
  - limits the expressive capabilities of the securely represented model

1. proposal
   - directly try training nets in generic frameworks to avoid sub-optimal conversion penalties

3.2.2 MPC perspective

- MPC (Multiparty computation) relies on customized sub-protocols and are optimized for particular activation functions.
- need to pursue of model-agnosticism : performance of MPCs is dependent on the model undergoing inference

1. solution
   - test general purpose MPC frameworks to evaluate CNNs
   - better scrutinized by the community
   - more reference implementations

4 Contribution

- MPC friendly models interop with existing frameworks
  - without sacrificing evaluation efficiency
  - without customized conversion
- support for running models as is (out of the box).
  - using general purpose MPC frameworks

4.1 Quantization

- transferability from current frameworks for MPC

4.2 MPC

- towards model agnostic MPC toolchain

4.3 Optimizations

- botteleneck is quantization : bitwise right shifts (truncation)
- propose optimized truncations to improve efficiency

4.4 Experiments

- evaluate efficiency for a large class of quantized models:
  - 16 diff sizes for 16 diff settings

4.5 Peculiarities

1. a quantization scheme being useful for secure evaluation
2. showing compatibility with arithmetic black-box model
   - only secure additions and multiplications are required.
3. optimization over truncation to the ring $\mathbb{Z}_{2^k}$: integers module 2^k
4. Experiment factors:
   - Corruption threshold (honest vs dishonest clients)
   - Corruption Model (passive vs active security)
   - algebraic structure($\mathbb{Z}_{2^k}$ or $\mathbb{Z}_p$)
   - exact or probabilistic truncation

4.6 Conclusions:

- corruption threshold has high impact on efficiency of general-purpose MPC
- corruption model has low impact on efficiency of general-purpose MPC
- k being between 4 and 10 is as efficient as choosing a modulo prime structure
- exact gain in efficiency found by experiments with exact and probabilistic truncation

5 Past/related work

5.1 regarding quantization

- non-intelligent ways: floating point to fixed point conversion without monitoring effect on the model's performance
- relatively little has been explored at the intersection of quantization and multiparty-computation
  - past works generally lie in the FHE domain
  - Fully homomorphic encryption compilers (recent work)
    * allows computation over encrypted data (without decryption)
    * mixing with quantization leads to limitations (can only perform additions and multiplications)
      · the model owner doesn't give up weights though
- frameworks: CrypTFlow :`https://www.microsoft.com/en-us/research/publication/cryptflow-secure-te`
  - custom conversion from floating to fixed point without loss of accuracy (needs extra neurons)

6 Quantization in Deep Learning

- use figure 1 to explain quantization into 8 bit
- do not explain the minor details for each layer : move on to cryptographic applications
- summarize by saying the basic types of opterations needed in this context:
  - basic:
    * multiplication and addition (convolution, dot products,etc)
  - need to be reduced to addition and multiplication
    * truncation during quantization
    * comparison for ReLU and similar min/max operations

7 System and Threat model

- use figure 2 to describe the client server model
  - talk about the degenerated case of no servers and 2 parties being the source and the computer themeselves

7.1 MPC protocols

- well established for addition and multiplication:
  - give a simple example for addition
- give a similar direction for multiplication

7.2 for more abstract operations

- secure comparison
- probabilistic truncation vs deterministic truncation
  - also mention when it is needed (post multiplication of two quantized values)
  - its benefits : avoid usage of expensive binary adders
- list the other computation requirements and the steps one should follow for it to be adapted into a secure MPC protocol: talk meta (condensation)
  - formulate the computation requirement locally in terms of the parties involved, the data they wish to keep private and the data (secret shares) they will need to share.
  - then establish validation protocols to check for the correctness of the output.
  - finally : try fulfilling these abstractions in terms of lower level MPC-secure primitive operations (like addition and multiplication)
    * show example of clamping reduced to comparison reduced to addition and multiplication

8 benchmarks

- discuss protocols: honest majority, dishonest majority

9 Misc (to be arranged)

- fields vs rings
- Homomorphic encryption: `https://en.wikipedia.org/wiki/Homomorphic_encryption`
  - notion of homomorphism from algebra : structure preserving mapping