Machine Learning on Encrypted Data

Kim Laine
Microsoft Research, Redmond WA

January 5, 2017

Joint Mathematics Meetings 2017, Atlanta GA
AMS-MAA Special Session on Mathematics of Cryptography

Two Tracks

- Evaluate already existing model on private data
  - Privacy of data
  - Privacy of model
  - Who gets the result?
  - Many data owners?

- Train new model on private data
  - Who gets the resulting model?
  - Privacy of training data
  - What does model leak?

Many data owners with little data

Few data owners with lots of data

Two Tracks

- Evaluate already existing model on private data
  - Privacy of data
  - Privacy of model
  - Who gets the result?
  - Many data owners?
- Train new model on private data
  - Who gets the resulting model?
  - Privacy of training data
  - What does model leak?

Many data owners with little data

Few data owners with lots of data

Example: Prediction as a Service

Medical

Genomic

Financial

Neural Networks, Decision Trees, Linear Mixed Models, etc.

Fully Homomorphic Encryption Using Ideal Lattices

Craig Gentry
Stanford University and IBM Watson
cgentry@cs.stanford.edu

ABSTRACT

We propose a fully homomorphic encryption scheme – i.e., a scheme that allows one to evaluate circuits over encrypted data without being able to decrypt. Our solution comes in three steps. First, we provide a general result – that, to construct an encryption scheme that permits evaluation of *arbitrary circuits*, it suffices to construct an encryption scheme that can evaluate (slightly augmented versions of) its *own decryption circuit*; we call a scheme that can evaluate its (augmented) decryption circuit *bootstrappable*.

Next, we describe a public key encryption scheme using *ideal lattices* that is *almost* bootstrappable. Lattice-based cryptosystems typically have decryption algorithms with low

duced by Rivest, Adleman and Dertouzos [54] shortly a ter the invention of RSA by Rivest, Adleman and Sham [55]. Basic RSA is a multiplicatively homomorphic encry tion scheme – i.e., given RSA public key pk $= (N, e)$ an ciphertexts $\{\psi_i \leftarrow \pi_i^e \bmod N\}$, one can efficiently compu $\prod_i \psi_i = (\prod_i \pi_i)^e \bmod N$, a ciphertext that encrypts t product of the original plaintexts. Rivest et al. [54] ask a natural question: What can one do with an encrypti scheme that is *fully* homomorphic: a scheme $\mathcal{E}$ with an ef cient algorithm $\mathsf{Evaluate}_{\mathcal{E}}$ that, for any valid public key p *any* circuit C (not just a circuit consisting of multiplicati gates), and any ciphertexts $\psi_i \leftarrow \mathsf{Encrypt}_{\mathcal{E}}(\mathrm{pk}, \pi_i)$, outpu

$$\psi \leftarrow \mathsf{Evaluate}_{\mathcal{E}}(\mathrm{pk}, C, \psi_1, \ldots, \psi_t) ,$$

Fast for certain types of models

Low-degree polynomial models (integral features) needed for HE

For performance want to use “CRT batching” with RLWE-based schemes

$$\frac{\mathbb{Z}_t[x]}{(x^n+1)} \cong \mathbb{Z}_t^n$$

Fast for certain types of models

Linear regression, linear mixed models

- Useful in genomics
- Super fast to evaluate on homomorphically encrypted data
- Demonstrated on realistic size examples
- Easy to utilize batching even for single prediction

Other intelligible polynomial models

- Demonstrated for realistic medical risk prediction
- Batching may or may not be easy to utilize

Less fast for other types of models

Neural networks

- Very powerful, high accuracy for difficult problems
- Homomorphic evaluation challenging due to size
- Depth limited with leveled fully homomorphic encryption
- Activation functions need to be low-degree polynomials
- Demonstrated (C)NN on MNIST data (hand-written digit recognition)
    - CryptoNets
    - O(10 seconds) for 4096 simultaneous predictions on encrypted data
    - Great *amortized* performance
    - Problem: How to use batching more efficiently?
    - Problem: How to extend to bigger NNs?

Not currently feasible for some models

Decision Trees

- Very powerful
- Need to always evaluate entire tree homomorphically
- Conditional statements based on integer comparison hard
  - Bit-wise encoding: comparisons easy, arithmetic hard
  - Integer-wise encoding: comparisons hard, arithmetic easy
  - Trees require both comparisons and arithmetic
- Could some pre-computation help with evaluating tree?
- Simplify by assuming structure of tree public but thresholds not?

Homomorphic Encryption

Pros

- Protects the privacy of the data owner
- Protects the privacy of the model owner (evaluator)
- Multiple data owners possible
- In best case can use CRT batching for good performance

Cons

- Currently needs to be applied and tuned on case-by-case basis
- Performance depends hugely on model
- Difficulty of integer comparisons is a major issue

Two Tracks

- ➢ Evaluate already existing model on private data
  - Privacy of data
  - Privacy of model
  - Who gets the result?
  - Many data owners?
- ➢ Train new model on private data
  - Who gets the resulting model?
  - Privacy of training data
  - What does model leak?

Many data owners with little data

Few data owners with lots of data

Two Tracks

- Evaluate already existing model on private data
  - Privacy of data
  - Privacy of model
  - Who gets the result?
  - Many data owners?
- Train new model on private data
  - Who gets the resulting model?
  - Privacy of training data
  - What does model leak?

Many data owners with little data

Few data owners with lots of data

Training on Private Data

- Much more difficult than evaluating existing models
- Requires extremely high degree polynomials (iteration)
  - HE possibly not practical

- Secure Multi-Party Computation better tool
  - Two or more parties compute a function on private inputs
  - One or more of parties gets output
  - Few data owners with lots of data, local computation possible?
- Many data owners with little data hardest scenario?

- Often not enough to keep training data private during training
- Need differential privacy to prevent model from leaking information

Lots of interesting questions

- Algorithmic problems that the ML community has not had to deal with
- New types of ML models needed?
- How to use batching in HE better?
- Formalize and classify scenarios in privacy-preserving training
- How to utilize local computation better?
- Can we do anything realistic with many data owners with little data?

Thank You!

Questions?

Contact: kim.laine@microsoft.com