Data-Mining on GBytes of Encrypted Data

Valeria Nikolaenko

In collaboration with
Dan Boneh (Stanford); Udi Weinsberg, Stratis Ioannidis, Marc Joye, Nina Taft (Technicolor).
Outline

• **Motivation**
  • Background on cryptographic tools
  • Linear regression
  • Our solution
  • Experiments and performance
Motivation

Users

Data mining engine

Data model

Privacy concern!

Engine learns *nothing* more than the model!
Data Mining

• Classification
• Regression: linear regression
• Clustering
• Summarization: matrix factorization
• Dependency modeling

Main challenge: make these algorithms privacy preserving and efficient.
Contribution

- Design of a practical system for privacy preserving linear regression
- Implementation
- Experiments on real datasets

Comparison to state of the art:
- Hall et al.'11: 2 days vs 3 min
- Graepel et al.'12: 10 min vs 2 sec
Outline

- Motivation
- **Background on cryptographic tools**
  - Linear regression
  - Our solution
  - Experiments and performance
Computations on Encrypted Data

- 2009, **C. Gentry** – FHE
  (slow for our problems)
- 1979, **A. Shamir**
  1988, **BGW**
  Secret sharing
  (huge communication overhead)
- 1982, **A.C. Yao** – Garbled circuits

- Our approach: hybrid of Yao and hom. encryption
Yao’s Garbled Circuits

Circuit

$\begin{array}{cccc}
0 & 0 & 0 & \ldots & 0 \\
1 & 1 & 1 & \ldots & 1 \\
\end{array}$

$x=010\ldots1$

Nothing is leaked about $x$, other than $C(x)$!
Data Mining System Architecture

User $i$

$X_i$

Evaluator

$\{G^{x_1}, G^{x_2}, \ldots, G^{x_n}\}$

Compute $C(x_1, \ldots, x_n)$

$C(x_1, \ldots, x_n)$

Crypto Service Provider (CSP)

Create “garbled circuit” $\hat{C}$

$\{G^0_1, G^0_2, \ldots, G^0_n\}$

$\{G^1_1, G^1_2, \ldots, G^1_n\}$

Valeria Nikolaenko
System Properties

✔ Evaluator learns the model, not the inputs

Problems:
- Not scalable with the number of users
- Users need to be online
Outline

• Motivation
• Background on cryptographic tools
• Linear regression
  • Our solution
  • Experiments and performance
Linear Regression I

- \((f,r)\) – from users
- solve for \(\beta\)

\[
A\beta = b
\]

\[r = \langle f, \beta \rangle\]
Linear Regression II

Users

\[ f_1, r_1, \ldots, f_n, r_n \]

Training engine

\[ \beta \]

Training engine learns **nothing** about \( f \)'s and \( r \)'s, other than \( \beta \)!
Outline

• Motivation
• Background on cryptographic tools
• Linear regression
• Our solution
  • Experiments and performance
Construct Matrices

Phase 1: compute

\[ A = \sum f_i^T f_i \quad \text{and} \quad b = \sum f_i^T r_i \]

**Users**

\[ f_i^T f_i \]

\[ f_n^T f_n \]

**Evaluator**

computes \( A = \sum f_i^T f_i \) in the circuit

For additions can use **homomorphic encryption**:

\[ [HE(A_1), HE(A_2)] \rightarrow HE(A_1 + A_2) \]
Construct Matrices

Phase 1: compute  \[ A = \sum f_i^T f_i \quad \text{and} \quad b = \sum f_i^T r_i \]

Users

Evaluator

• computes \( HE(A) = HE(\sum f_i^T f_i) \)
• decrypts in the circuit

For additions can use homomorphic encryption:

\[ [HE(A_1), HE(A_2)] \rightarrow HE(A_1 + A_2) \]

Circuit – independent of the number of users
Users – offline
Solve Linear System

Phase 2: solve for $\beta$  \[ A\beta = b \]

Input: HE(A), HE(b)

Decrypt A and b

Cholesky: compute L, s.t. \[ A = LL^T \]

Solve for $\beta$: \[ L(L^T \beta) = b \]
Privacy Preserving Regression System

Evaluator

Crypto Service Provider (CSP)

\begin{align*}
(pk, sk) & \leftarrow \text{HE.KeyGen} \\
\text{Create “garbled circuit” } & \hat{C} \\
\text{Send } & \hat{C}
\end{align*}

\begin{enumerate}
\item \text{Phase 1:}
\begin{align*}
\text{Compute } & \text{HE}_{pk}(A = \sum f_i^T f_i), \\
& \text{HE}_{pk}(b = \sum f_i^T r_i)
\end{align*}

\item \text{Garbled inputs}

\item \text{Garbled inputs}
\begin{align*}
\text{Compute } & \text{C(HE}_{pk}(A),\text{HE}_{pk}(b))
\end{align*}
\end{enumerate}
System Properties

- Evaluator learns the model, not the inputs
- Scalable with the number of users
- Users can be offline

Extensions:
- Masking instead of decryption in circuit
- Protection against malicious Evaluator and CSP
Outline

• Motivation
• Background on cryptographic tools
• Linear regression
• Our solution
• Experiments and performance
Performance

• Hybrid vs. Pure-Yao: 100 times improvement in time!

• For up to 20 features, 1000’s users
  • time < 3 min
  • communication < 1GB

• For 100 million of users, 20 features: 8.75 hours

• Tested on real datasets
Conclusion

• Privacy-preserving data-mining is efficient
• Our approach can be used in practice

• Current work: matrix factorization

• Future work:
  – implement other data mining algorithms
  – improving implementation to support high parallelization
Thank you!

Questions? valerini@stanford.edu