The Title of Your Book

Your Name
Department of Computer Science
Brown University

April 4, 2023

Contents

Preface v

1 Privacy-preserving Technologies for Artificial Intelligence 1

1.1 Overview 1
1.2 Anonymization Techniques 1
1.3 Differential Privacy 4
1.3.1 Mean and Sum 6
1.4 Secure Multi-Party Computations 7
1.4.1 Security in Semi-honest model 8
1.4.2 Yao's General Two Party Secure Function Evaluation 9
1.4.3 Leveled homomorphic encryption (HE) 10
1.5 Secure Algebraic Calculations 11
1.5.1 Secure Sum 11
1.5.2 1-out-of-*N* Oblivious Transfer 12
1.5.3 Federated Learning (FL) 13

Preface

Which labor market institutions worked better in containing job losses during the Great Recession of 2008–2009? Is it good for employment to increase the progressiveness of taxation? Does it make sense to contrast "active" and "passive" labor market policies? Who actually gains and who loses from employment protection legislation? Why are minimum wages generally diversified by age? Is it better to have decentralized or centralized bargaining systems in monetary unions? Should migrants have access to welfare benefits? Should governments regulate working hours? And can equal opportunity legislation reduce discrimination against women or minority groups in the labor market?

Current labor economics textbooks neglect these relevant policy issues. In spite of significant progress in analyzing the costs and benefits of labor market institutions, these textbooks have a setup that relegates institutions to the last paragraph of chapters or to a final institutional chapter. Typically a book begins by characterizing labor supply (including human capital theory), labor demand, and the competitive equilibrium at the intersection of the two curves; it subsequently addresses such topics as wage formation and unions, compensating wage differentials, and unemployment without a proper institutional framework. There is little information concerning labor market institutions and labor market policies. Usually labor market policies are mentioned only every now and then, and labor market institutions are often not treated in a systematic way. When attention is given to these institutions, reference is generally made to the U.S. institutional landscape and to competitive labor markets in which, by definition, any type of policy measure is distortionary.

Acknowledgments

Your acknowledgments are included in the Preface as the final section.

Your Name
March 2014

1 Privacy-preserving Technologies for Artificial Intelligence

Kantarcioglu, Murat

1.1 Overview

We need multiple techniques. Discuss how they can help.

1.2 Anonymization Techniques

In order to protect individual privacy, in many applications, unique identifiers such as social security numbers, names and phone numbers are removed from a data set prior to public release. However, as Sweeney discusses in [Sweeney 2002], such measures may not be sufficient considering the vast amount of additional information that is publicly available (e.g., census rolls). While not unique identifiers by themselves, certain attributes called *quasi-identifier* attributes, can be combined with public directories to accurately identify individuals.

This notion is exemplified in [Sweeney 2002], where an attack against William Weld, the governor of Massachusetts at that time, disclosed his medical records. In fact, Sweeney argues in this study that 87% of US citizens could possibly be uniquely identified using postal code (ZIP), sex and birth date attributes.

Anonymization is one popular solution against such attacks. By generalizing the values of quasi-identifying attributes (i.e., generalization) and/or removing certain records from the data set (i.e., suppression), anonymization methods try to satisfy certain definitions of anonymity.

The most well-known of such definitions is k-anonymity, devised by [Sweeney 2002]. According to k-anonymity, every combination of quasi-identifier values (called an equivalence class) in the anonymized data set should appear at least k times, so that any individual cannot be distinguished within a group of size at least k. Alternatively, any query over the quasi-identifier attributes should not retrieve a result set of size less than k [LeFevre et al. 2006].

Privacy sensitive information related to individuals, such as medical data, is today collected, stored and processed in a large variety of application domains. Such data is typically used to provide better quality services to individuals and its availability is crucial in many contexts. In the case of healthcare, for example, the availability of such data helps prevent

R	A_1	A_2
r_1	Masters	35
r_2	Masters	36
r_3	Masters	36
r_4	11th	28
r_5	11th	22
r_6	12th	33

R'	A_1	A_2
r'_1	Masters	[35-37)
r'_2	Masters	[35-37)
r'_3	Masters	[35-37)
r'_4	Senior Sec.	[1-35)
r'_5	Senior Sec.	[1-35)
r'_6	Senior Sec.	[1-35)

Table 1.1 Data set R and R's 3-anonymous generalization R'

medical errors and enhance patient care. Privacy sensitive data may have many important legitimate uses serving distinct purposes outside the specific domain in which it has initially been collected. For example, drug companies and researchers may be interested in patient records for drug development. Such additional uses of data are important and should certainly be supported. Yet, privacy of the individuals to whom the data is related should be assured as well.

To address the conflicting requirements of assuring privacy while at the same time supporting legitimate use, several different techniques ranging from cryptographic methods [Kantarcıoğlu and Clifton 2004, Lindell and Pinkas 2002] to perturbation methods [Kargupta et al. 2003] have been proposed. Among those methods, anonymization techniques have emerged as one of the most important privacy preservation tools. Recently, several anonymization methods have been proposed based on different privacy definitions (e.g., k-anonymity [Sweeney 2002], l-diversity [Machanavajjhala et al. 2006], t-closeness [Li et al. 2007]), different algorithms, (e.g., DataFly [Sweeney 2002], Top-down specialization, Incognito [LeFevre et al. 2005], Mondarian multi-dimensional k-anonymity [LeFevre et al. 2006]), and different data assumptions (e.g., anonymizing data streams [Tan et al. 2008]). Anonymization techniques such as k-anonymization [Sweeney 2002] usually use privacy metrics that measure the amount of privacy protection and transform the original data in such a way that it will be hard to infer identities of the individuals when released for outside use.

For example, consider data set R in Table 1.1 that stores age and education information of certain individuals. Under the k-anonymity approach, quasi-identifier fields age and education are generalized according to value generalization hierarchies (e.g., see Figure 1.1) such that there are at least k individuals that share the same quasi-identifying information. The data set R' in Table 1.1 is one the many 3-anonymous versions of R.

One important purpose for which anonymized data is used is represented by data mining. Data mining is an important form of knowledge extraction. Data mining harnesses huge amounts of data available in many application domains to extract knowledge crucial to the progress of research and other human activities. A key question is thus whether and how

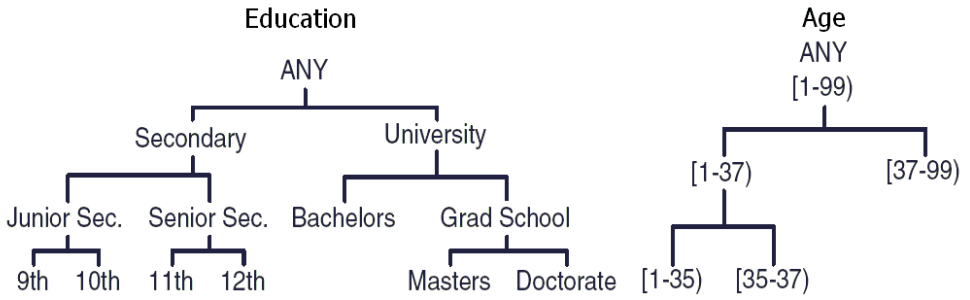

Figure 1.1 Value generalization hierarchies for Education and Age attributes

anonymized data can be effectively used for data mining and which anonymization methods and/or privacy definitions best support different data mining tasks.

For some data mining tasks, there are trivial and effective answers to this question. For example, if each attribute is anonymized to the same level in the data set and we want to build decision trees, we could directly build decision trees on the anonymized data by treating each generalization as a discrete attribute. Consider the anonymized table R' in Table 1.1. Attribute value "Senior Sec." can be considered a discrete value without even considering the fact that it actually represents either "11th" grade, or "12th" grade.

Obviously, given a data set, multiple possible k-anonymizations might exist. In [Meyerson and Williams 2004], Meyerson and Williams prove that finding an optimal k-anonymization that minimally distorts the input data set is NP-hard. The reduction is from k-dimensional perfect matching. Still, various effective heuristic solutions to the problem exist [Fung et al. 2005, LeFevre et al. 2005, 2006, Sweeney 2002].

The concept of k-anonymity as a privacy definition has been challenged by many related works in the area. The first privacy definition to address the limitations of k-anonymity has been the ℓ-diversity privacy definition [Machanavajjhala et al. 2006], where adversaries with certain background knowledge are able to violate individual privacy. Two scenarios are discussed: the homogeneity attack and the background knowledge attack. In the first scenario, lack of diversity over a sensitive attribute (e.g., diagnosis) within equivalence classes allows the attacker to directly infer the value for an individual. The second scenario is rather complex but is also rooted in lack of diversity.

As a solution, ℓ-diversity definition requires each equivalence class to contain ℓ *well-represented* sensitive attribute values. Different instantiations of the privacy definition based on the concept of being well represented are provided in [Machanavajjhala et al. 2006].

One simplified diversity formulation used in [Xiao and Tao 2006] is to bound the fraction of records in an equivalence class that have the same sensitive attribute value. This condition

is easy to enforce, as it amounts to checking that no value occurs with frequency larger than $1/\ell$.

A more complex condition is the recursive (c,ℓ)-diversity of [Machanavajjhala et al. 2006]. Denote by f_i the number of occurrences of the i^{th} sensitive attribute value in an equivalence class, and assume that these values are sorted in decreasing order according to occurrence frequency (i.e., f_1 is the highest value). Then (c,ℓ)-diversity is satisfied if for every equivalence class it holds that

$$f_1 < c(f_\ell + f_{\ell+1} + \ldots + f_m)$$

where m is the cardinality of the sensitive attribute domain.

t-closeness [Li et al. 2007] is another popular privacy definition along the same direction. Here, the privacy parameter t measures the distance between probability distributions over each equivalence class and the entire input data set through the Earth Mover's metric. ℓ-diversity and t-closeness privacy definitions can be satisfied by modifying any method originally devised for k-anonymity.

1.3 Differential Privacy

A statistical database answers aggregate queries such as *Count* or *Sum* queries. In this study, we consider aggregate *range* queries, that can be expressed by d-dimensional hyper-rectangles in the attribute domain space $A_1 \times \ldots \times A_d$. A range query is represented as a Cartesian product $Q = [x_1^Q, y_1^Q] \times \ldots \times [x_d^Q, y_d^Q]$ where $[x_i^Q, y_i^Q]$ is the extent of Q on attribute A_i. For instance, the *Count* query

$$\texttt{SELECT COUNT}(*)\ \texttt{FROM T WHERE}\ (40 \leq Age \leq 60);$$

has extent $[40,60]$ on attribute Age.

Statistical databases allow users to retrieve coarse-grained aggregate information. On the other hand, it is important to preserve the privacy of individual records, by not allowing fine-grained aggregate queries. One solution to preserve privacy is to disallow queries that have extents smaller than a certain threshold, e.g., 10 years of age, and $10k$ salary. However, this may not be sufficient. For instance, consider malicious user Mallory who knows that his colleague Alice is the only one in the company who is 51 years old. Mallory can issue the following two queries:

$$\begin{aligned}
&\texttt{Q1}: \quad \texttt{SELECT COUNT}(*)\ \texttt{FROM T} \\
&\qquad\quad \texttt{WHERE}\ (40 \leq Age \leq 50)\ \texttt{AND}\ (30k \leq Salary \leq 40k); \\
&\texttt{Q2}: \quad \texttt{SELECT COUNT}(*)\ \texttt{FROM T} \\
&\qquad\quad \texttt{WHERE}\ (40 \leq Age \leq 51)\ \texttt{AND}\ (30k \leq Salary \leq 40k).
\end{aligned}$$

Assume that the response to Q_1 is 10, and the response to Q_2 is 11. Then, Mallory can infer that Alice must be included in the result of query Q_2, and hence her salary is in the range $[30k, 40k]$. Note that, the attack was successful because Mallory was able to access the query results for the two views T_1 and T_2 of table T that differ in a single record (that is, the record of Alice).

Differential privacy [Dwork et al. 2006] aims to prevent this type of inferences by adding random noise to each query result. The interaction between a user and the database is expressed as a *transcript*. Formally,

Definition 1 (Transcript)**.** *Let $\mathcal{Q} = \{Q_1, \ldots, Q_q\}$ be a set of aggregate queries, and denote by $Q_i^T{}_{(1 \le i \le q)}$ the result of answering query Q_i on table T. A* transcript

$$\mathcal{TR}_{\mathcal{Q}}^T = \{(Q_1, a_1), \ldots, (Q_q, a_q)\}$$

is the response of a statistical database to the query set $\mathcal{Q}$ on database table T, where $a_i = Q_i^T$.

A statistical database satisfies ε-differential privacy if the ε-indistinguishability condition [Dwork et al. 2006] is fulfilled:

Definition 2 (ε-indistinguishability)**.** *Consider a statistical database that produces the transcript $\mathcal{U}$ on the set of queries $\mathcal{Q} = \{Q_1, \ldots, Q_q\}$, and let $\varepsilon > 0$ be an arbitrarily-small real constant. Transcript $\mathcal{U}$ satisfies* ε-indistinguishability *if for every pair of views T_1, T_2 such that $|T_1| = |T_2|$ and T_1, T_2 differ in only one record, it holds that*

$$\left| \ln \frac{Pr[\mathcal{TR}_{\mathcal{Q}}^{T_1} = \mathcal{U}]}{Pr[\mathcal{TR}_{\mathcal{Q}}^{T_2} = \mathcal{U}]} \right| \le \varepsilon \tag{1.1}$$

In other words, an attacker is not able to learn whether the transcript was obtained by answering the query set $\mathcal{Q}$ on view T_1, or on view T_2. To achieve ε-indistinguishability, a statistical database injects noise into each query result Q_i^T. The amount of noise required is proportional to the *sensitivity* of the query set $\mathcal{Q}$, which is defined as follows:

Definition 3 (L_1-sensitivity)**.** *Over any two views T_1, T_2 such that $|T_1| = |T_2|$ and T_1, T_2 differ in only one record, the L_1-*sensitivity *of query set $\mathcal{Q} = \{Q_1, \ldots, Q_q\}$ is measured as*

$$S_{L_1}(\mathcal{Q}) = \max_{\forall T_1, T_2} \sum_{i=1}^{q} |Q_i^{T_1} - Q_i^{T_2}|$$

The following theorem from [Dwork et al. 2006] gives a sufficient condition for a statistical database to satisfy ε-differential privacy:

Theorem 1. *Let $\mathcal{Q}$ be a set of queries answered by a statistical database, and denote by $S_{L_1}(\mathcal{Q})$ the L_1-sensitivity of $\mathcal{Q}$. Then, differential privacy with parameter ε can be achieved by adding to each query result random noise X, i.e., $Q_i^T \leftarrow Q_i^T + X$, where X is a random, i.i.d. variable drawn from a Laplace distribution with mean 0 and magnitude $\lambda \ge S_{L_1}(\mathcal{Q})/\varepsilon$.*

An essential operation in enforcing differential privacy is determining the sensitivity $S_{L_1}(Q)$. Interestingly, it is shown in [Dwork et al. 2006] that $S_{L_1}(Q)$ is independent of the data set T, and can be determined based on the query set Q alone. However, for arbitrary query sets, it is shown in [Xiao and Tao 2008] that computing sensitivity is NP-hard.

Nevertheless, if certain restrictions are imposed on Q, sensitivity (or its approximation) for sets of *Count* queries can be efficiently computed. Specifically:

1. If all queries in Q have disjoint ranges, $S_{L_1}(Q) = 2$.
2. If queries in Q have overlapping ranges, then a 2-approximation of $S_{L_1}(Q)$ is given by the maximum number of queries that overlap the same point in the data space. Formally, for any data space point $p \in A_1 \times \ldots \times A_d$, define

$$Q^p = \{Q \in Q | p \in Q\}$$

i.e., the set of queries that cover point p. Then, we have

$$S_{L_1}(Q) \leq 2 \cdot \max_{p \in A_1 \times \ldots \times A_d} |Q^p|.$$

1.3.1 Mean and Sum

Assume two sibling databases D_1 and D_2 have n records each, and they differ by one record. Next we establish the sensivity of queries given sample size n and d attributes.

Theorem 2. *Two sibling database $|D_1| = |D_2| = n$, where $n \geq 1$. Let $Q = \{Mean_1, ..., Mean_d\}$, where $d \geq 1$. $S(Q) = d/n$.*

Proof 1. *Let $Mean_i^{(n-1)}$ be the mean of A_i of the common $n-1$ records shared by D_1 and D_2. Let the unique record in D_1 be x_1 and the unique record in D_2 be x_2. Then the mean values of A_i in D_1 and D_2 are*

$$Mean_i^{(n),1} = \frac{(n-1) \times Mean_i^{(n-1)} + x_1[A_i]}{n},$$

$$Mean_i^{(n),2} = \frac{(n-1) \times Mean_i^{(n-1)} + x_2[A_i]}{n}.$$

We have

$$|Mean_i^{(n),1} - Mean_i^{(n),2}| = \frac{|x_1[A_i] - x_2[A_i]|}{n}.$$

Then we have

$$max_{\{D_1,D_2\}} \sum_{1}^{d} |Mean_i^{(n),1} - Mean_i^{(n),2}| = \frac{1}{n} max_{\{D_1,D_2\}} \sum_{i=1}^{d} |x_1[A_i] - x_2[A_i]| \leq \frac{d}{n} = S(Q)$$

When all the d attributes in the x_1 and x_2 differ by 1, we reach the maximum, which determines the sensitivity.

All the Theorems so far do not depend on the distribution of A_i over the interval $[0,1]$. The sensitivity of Q depends on both the sample size n and the number of attributes d. It requires $n >> d$ to reduce the sensitivity.

Theorem 3. *Two sibling database $|D_1| = |D_2| = n$, where $n \geq 1$. Let $Q = \{Sum_1, ..., Sum_d\}$, where $d \geq 1$. $S(Q) = d$.*

Proof 2. *Let $Sum_i^{(n-1)}$ be the sum of attribute A_i of the common $n-1$ records shared by D_1 and D_2. Again let the unique record in D_1 be x_1 and the unique record in D_2 be x_2. Then the sum of A_i in D_1 and D_2 are*

$$Sum_i^{(n),1} = Sum_i^{(n-1)} + x_1[A_i],$$

$$Sum_i^{(n),2} = Sum_i^{(n-1)} + x_2[A_i].$$

When all the d attributes in the x_1 and x_2 differ by 1, we have

$$max_{\{D_1,D_2\}} \sum_{1}^{d} |Sum_i^{(n),1} - Sum_i^{(n),2}| = max_{\{D_1,D_2\}} \sum_{i=1}^{d} |x_1[A_i] - x_2[A_i]| = d = S(Q)$$

All the Theorems so far do not depend on the distribution of A_i over the interval $[0,1]$. The sensitivity of $Q = \{Mean_1, ..., Mean_d\}$ improves linearly as the sample size n increases given a fixed d. On the other hand increasing the sample size n will not improve the sensivity of $Q = \{Sum_1, ..., Sum_d\}$, which is determinded solely by dimensionality.

1.4 Secure Multi-Party Computations

Secure Multi-party Computation protocols provide provably secure, cryptographic solutions to the more general problem of privacy preserving data analytics. As explained in [Yao 1982] through the famous Yao's Millionaire problem, any function whose secret inputs are distributed across multiple parties can be computed securely without revealing nothing but the result and what can be inferred from the result. In fact, such secure protocols can be generated automatically as a circuit of logic gates.

Unfortunately, secure circuit evaluation typically takes forbiddingly long time to execute compared to custom-tailored solutions. [Clifton et al. 2003] discuss such solutions for constructs commonly used in privacy preserving data mining.

what kind of functions can be evaluated without revealing anything other than the function result? The answer shown by Yao[Yao 1986] is: any function that can be represented as a polynomial-size circuit. This result indicates that almost any data mining task can be done without violating privacy. Although this generic result is too inefficient to be used for data mining, it provides a framework and a general methodology to prove security of the proposed solutions. In this dissertation, we use the above framework to provide provably secure (i.e., nothing is revealed other than the result itself) and efficient solutions.

Strong theoretical foundations for secure multi-party computation and many cryptographic protocols already exist. To enhance the understanding of this dissertation, we review these concepts in this chapter.

Imagine the case, where two millionaires want to learn who is richer. (Yao's Millionaire problem [Yao 1986]) Also, due to privacy reasons, they do not want to disclose their net worth to each other. We can easily solve this problem using a trusted third party. Each millionaire can send his or her input to the trusted party. Later, the trusted party can send the final result back to the millionaires. Assuming the communication between the trusted party and the millionaires is secure, millionaires only learn who is richer. The obvious question is what happens if the millionaires do not trust any one. The goal of secure multi-party computation is to come up with solutions where we can reach the privacy level of having a third trusted party without actually having one.

Substantial work has been done on secure multi-party computation (SMC). The key result is that a wide class of computations can be computed securely under reasonable assumptions. We give a brief overview of this work, concentrating on material that is used later in the thesis. The definitions given here are from Goldreich[Goldreich 2004]. For simplicity, we concentrate on the two party case. Extending the definitions to the multi-party case is straightforward.

1.4.1 Security in Semi-honest model

A semi-honest party follows the rules of the protocol using its correct input, but is free to later use what it sees during execution of the protocol to compromise security. This is somewhat realistic in the real world because parties who want to mine data for their mutual benefit will follow the protocol to get correct results. In some cases, parties may only want to learn the results but not the data itself. For example, recently one credit card transaction processing company was hacked and the credit card transactions that involve 40 million credit card numbers were stolen. Apparently, the transaction data was stored only to do data mining. On the other hand, the loss of the data made the company vulnerable to potential law suits [Bray and Talcott 2005]. Similar examples can be also given for health care data. Clearly such companies may follow the protocol to just learn the data mining results and nothing else. Also a protocol that is buried in a complex software may be difficult to alter.

A formal definition of private two party computation in the semi-honest model is given below. Computing a function privately is equivalent to computing it in the ideal model where we can use a third trusted party [Goldreich 2004].

Definition 4. *(privacy w.r.t. semi-honest behavior):[Goldreich 2004]*

Let $f : \{0,1\}^* \times \{0,1\}^* \longmapsto \{0,1\}^* \times \{0,1\}^*$ *be probabilistic, polynomial-time functionality, where* $f_1(x,y)$*(resp.,* $f_2(x,y)$*) denotes the first (resp., second) element of* $f(x,y)$*) and let* Π *be two-party protocol for computing* f.

Let the view of the first (resp. second) party during an execution of Π *on* (x,y)*, denoted* $view_1^{\Pi}(x,y)$ *(resp.,* $view_2^{\Pi}(x,y)$*) is* $(x,r_1,m_1,\ldots,m_t)$ *(resp.,* $(y,r_2,m_1,\ldots,m_t)$*) where* r_1 *represents the outcome of the first (resp.,* r_2 *second) party's internal coin tosses, and* m_i *represents the* i^{th} *message it has received.*

The output of the first (resp., second) party during an execution of Π *on* (x,y) *is denoted* $output_1^{\Pi}(x,y)$ *(resp.,* $output_2^{\Pi}(x,y)$*) and is implicit in the party's view of the execution.*

Π *privately computes* f *if there exist probabilistic polynomial time algorithms, denoted* S_1, S_2 *such that*

$$\{(S_1(x,f_1(x,y)),f_2(x,y))\}_{x,y\in\{0,1\}^*} \equiv^C \left\{\left(view_1^{\Pi}(x,y),output_2^{\Pi}(x,y)\right)\right\}_{x,y\in\{0,1\}^*} \quad (1.2)$$

$$\{(f_1(x,y),S_2(x,f_1(x,y)))\}_{x,y\in\{0,1\}^*} \equiv^C \left\{\left(output_1^{\Pi}(x,y),view_2^{\Pi}(x,y)\right)\right\}_{x,y\in\{0,1\}^*} \quad (1.3)$$

where $\equiv^C$ *denotes computational indistinguishability.*

The above definition says that a computation is secure if the view of each party during the execution of the protocol can be effectively simulated by the input and the output of the party. This is not quite the same as saying that private information is protected. For example, if two parties use a secure protocol to mine distributed association rules, a secure protocol still reveals that if a particular rule is not supported by particular site and that rule appears in the globally supported rule set then it must be supported by the other site. A site can deduce this information by solely looking at its locally supported rules and the globally supported rules. On the other hand, there is no way to deduce the exact support count of some itemset by looking at the globally supported rules. With three or more parties, knowing a rule holds globally reveals that at least one site supports it, but no site knows which site (other than, obviously, itself). In summary, secure multi-party protocol will not reveal more information to a particular party than the information that can be induced by looking at that party's input and the output.

We also use a key result from the Secure Multi-party Computation literature, the composition theorem. We state it for the semi-honest model. A detailed discussion of this theorem, as well as the proof, can be found in [Goldreich 2004].

Theorem 4. *(Composition Theorem for the semi-honest model)[Goldreich 2004]: Suppose that* g *is privately reducible to* f *and that there exists a protocol for privately computing* f*. Then there exists a protocol for privately computing* g*.*

This allows us to use existing secure protocols as components in a black-box fashion.

1.4.2 Yao's General Two Party Secure Function Evaluation

Yao's general secure two party evaluation is based on expressing the function $f(x,y)$ as a circuit and encrypting the gates for secure evaluation[Yao 1986]. With this protocol any two

party function can be evaluated securely in semi-honest model but the functions that can be efficiently evaluated must have small circuit representation. We will not give details of this generic method here. In some of our protocols, we will use this generic result for finding out whether $a \geq b$ for arbitrary a, b (Yao's millionaire problem). For comparing any two integers securely, Yao's generic method is one of the most efficient methods known, although other asymptotically equivalent but practically more efficient algorithms could be used as well[Ioannidis and Grama 2003].

Garbled circuits [Yao 1986] allow two parties to evaluate an arbitrary function represented as a Boolean circuit on private inputs from both parties, leaking nothing beyond the output of the function. A garbling scheme is a tuple of algorithms $\mathsf{GC} = (\mathsf{Garble}, \mathsf{Eval}, \mathsf{Decode})$. During Garble, the *garbler* party creates a "garbled" representation of a Boolean circuit in which each wire is associated with two cryptographic keys, called *labels*, one for 0 and one for 1. In the garbled representation, the gates of the circuit are tables of ciphertexts encrypted using the wire labels (e.g., via AES). The garbled circuit consisting of all garbled gates is sent to the *evaluator* party. Then, the garbler provides the input garbled values to the evaluator. For its own input bits, the garbler directly sends the corresponding garbled labels. For the evaluator's input bits, the two parties run OT protocols, enabling the evaluator to obtain the garbled labels without revealing its inputs. During Eval, the evaluator evaluates the garbled circuit on the received garbled labels. Finally, the garbled output is decoded using the corresponding output wire labels in Decode.

Formally, the GC algorithms are:

- $\mathsf{Garble}(f, x) \rightarrow (\mathcal{C}, X, d)$: on input function f and input bits $x \in \{0,1\}^n$, outputs garbled representation $\mathcal{C}$, garbled input labels pairs $X = \{\mathsf{lab}_i^0, \mathsf{lab}_i^1\}_{i \in [n]}$, decoding map d.
- $\mathsf{Eval}(\mathcal{C}, \bar{X}) \rightarrow Y$: on input garbled representation $\mathcal{C}$ and a set of garbled input labels $\bar{X} = \{\mathsf{lab}_i^{x_i}\}_{i \in [n]}$ associated to input $x \in \{0,1\}^n$, output garbled output labels Y.
- $\mathsf{Decode}(d, Y) \rightarrow y$: on input decoding map d and garbled output labels Y, output the circuit's output y.

A garbling scheme is *correct* if Decode outputs $y = f(x)$ for the desired function f and input x. A garbling scheme is *private* if, given $\mathcal{C}$, the garbled input labels $\bar{X}$ of the circuit wires and the decoding map d, an adversary learns nothing apart from the number of inputs, the size of the circuit and y.

1.4.3 Leveled homomorphic encryption (HE)

A *leveled homomorphic encryption scheme* supports the encrypted evaluation of bounded-degree polynomials or bounded-depth arithmetic circuits. In HE schemes based on the Learning with Errors hardness problem, each operation evaluated on ciphertexts introduces some

noise, which can cause incorrect decryption if it overflows. Multiplications introduce the most noise, therefore we want to evaluate low-depth circuits, i.e., few sequential multiplications.

Plaintexts and ciphertexts can encode a vector of scalars; an operation applied on a ciphertext is applied component-wise on the encrypted vector. This allows us to perform component-wise *additions* and *multiplications* between ciphertexts or between a plaintext and a ciphertext, and vector *rotations* (cyclic permutations of the encrypted vector).

Formally, a leveled homomorphic encryption scheme is a tuple of algorithms $\mathsf{LHE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$:

- $\mathsf{KeyGen} \rightarrow (\mathsf{pk}, \mathsf{sk}, \mathsf{evk})$: outputs a public key pk, a secret key sk, and an evaluation key evk.
- $\mathsf{Enc}_{\mathsf{pk}}(m) \rightarrow c$: on input public key pk and a message m, output a ciphertext c.
- $\mathsf{Dec}_{\mathsf{sk}}(c) \rightarrow m$: on input secret key sk and a ciphertext c, output a message m.
- $\mathsf{Eval}_{\mathsf{evk}}(f, c_1, c_2, m_3) \rightarrow c$: on input evaluation key evk, ciphertexts c_1, c_2 encrypting messages m_1, m_2, a message m_3 and a depth-d arithmetic circuit f, output a new ciphertext c. The inputs c_2, m_3 are optional.

Briefly, LHE is *correct* if $\mathsf{Dec}_{sk}(\mathsf{Enc}_{\mathsf{pk}}(m)) = m$ and $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Eval}_{\mathsf{evk}}(f, \mathsf{Enc}_{\mathsf{pk}}(m), \mathsf{Enc}_{\mathsf{pk}}(m'))) = f(m, m')$. LHE is *IND-CPA secure* if the encryptions of any two messages are computationally indistinguishable, and *function private* if a ciphertext obtained from the homomorphic evaluation of a function, together with the private key, reveals only the result of the evaluation and nothing else about the function.

1.5 Secure Algebraic Calculations

Some of the privacy-preserving algorithms developed later are based on the several secure two-party computation protocols. While most have either been previously published, or are straightforward given previously published work, we summarize them here for completeness.

1.5.1 Secure Sum

One building block frequently required is a way to securely calculate the sum of values from individual sites. Assuming three or more parties and no collusion, the following method from [Kantarcıoğlu and Clifton 2002] securely computes such a sum.

Assume that the value $v = \sum_{i=1}^{k} v_i$ to be computed is known to lie in the range $[0..n-1]$ where v_i denotes the share of the i^{th}.

One site is designated the *master* site, numbered 1. The remaining sites are numbered $2..k$. Site 1 generates a random number R, uniformly chosen from $[0..n-1]$. Site 1 adds this to its local value v_1, and sends the sum $R + v_1 \mod n$ to site 2. Since the value R is chosen uniformly from $[0..n-1]$, the number $R + v_1 \mod n$ is also distributed uniformly across this region, so site 2 learns nothing about the actual value of v_1.

For the remaining sites $i = 2..k-1$, the algorithm is as follows. Site i receives

$$V = R + \sum_{j=1}^{i-1} v_j \bmod n.$$

Since this value is uniformly distributed across $[0..n-1]$, i learns nothing. Site i then computes

$$R + \sum_{j=1}^{i} v_i \bmod n = (v_i + V) \bmod n$$

and passes it to site $i+1$.

Site k performs the above step, and sends the result to site 1. Site 1, knowing R, can subtract R to get the actual result. Note that site 1 can also determine $\sum_{i=2}^{k} v_i$ by subtracting v_1. This is possible from the global result *regardless of how it is computed*, so site 1 has not learned anything from the computation.

This method faces an obvious problem if sites collude. Sites $i-1$ and $i+1$ can compare the values they send/receive to determine the exact value for v_i. The method can be extended to work for an honest majority. Each site divides v_i into shares. The sum for each share is computed individually. However, the path used is permuted for each share, such that no site has the same neighbor twice. To compute v_i, the neighbors of i from each iteration would have to collude. Varying the number of shares varies the number of dishonest (colluding) parties required to violate security.

1.5.2 1-out-of-N Oblivious Transfer

The 1-out-of-N Oblivious Transfer protocol involves two parties, Alice and Bob. Alice has an input $\sigma, 1 \leq \sigma \leq N$, while Bob has N inputs $X_1, \ldots, X_n$. At the end of the protocol, Alice learns only X_σ and nothing else while Bob learns nothing at all. The 1-out-of-2 Oblivious Transfer (OT_1^2) was suggested by Even, Goldreich and Lempel [Even et al. 1985] as a generalization of Rabin's "oblivious transfer" [Rabin 1981]. Naor and Pinkas [Naor and Pinkas 2001] provide efficient protocols for 1-out-of-N Oblivious Transfer. For completeness, we now describe a very simple (though inefficient) method for doing Oblivious Transfer for semi-honest parties.

Bob generates N public key pairs $E_1, D_1, \ldots, E_N, D_N$

Bob sends $E_1, \ldots, E_N$ to Alice.

Alice generates an asymmetric key K.

Alice forms the vector $\vec{V}$: if $i = \sigma$, $V_i = E_i(K)$, otherwise $V_i =$ (a random) R_j.

Alice sends the N-dimensional vector $\vec{V}$ to Bob

Bob decrypts $\vec{V}$ to form the vector $\vec{K}$ where $K_i = D_i(V_i)$

Bob encrypts his data items with the keys in $\vec{K}$ ands sends them to Alice (i.e. Bob sends $K_i(X_i), i = 1 \ldots N$ to Alice)

Since $K_\sigma = D_\sigma(E_\sigma(K)) = K$, Alice decrypts the σ row with K to get X_σ

Clearly this protocol reveals nothing to Bob[Goldreich 2004]. In the semi-honest model, as long as Alice acts exactly according to the protocol, she too does not learn anything since all the other values are encrypted with random keys unknown to her. Though it is easy to break this protocol when parties are allowed to be malicious, better protocols (more secure and efficient) can easily be found in the literature.

1.5.3 Federated Learning (FL)

At a high level, FL is multi-round protocol between an aggregation server and a set of agents in which agents jointly train a model. Formally, participating agents try to minimize the average of their loss functions,

$$_{w\in R^d} f(w) = \frac{1}{K}\sum_{k=1}^{K} f_k(w),$$

where f_k is the loss function of kth agent. For example, for neural networks, f_k is typically empirical risk minimization under a loss function L such as cross-entropy, ,

$$f_k(w) = \frac{1}{n_k}\sum_{j=1}^{n_k} L(x_j, y_j; w),$$

with n_k being the total number of samples in agent's dataset and (x_j, y_j) being the jth sample.

Concretely, FL protocol is executed as follows: at round t, server samples a subset of agents S_t, and sends them w_t, the model weights for the current round. Upon receiving w_t, kth agent initializes his model with the received weight, and trains for some number of iterations, , via stochastic gradient descent (SGD), and ends up with weights w_t^k. The agent then computes his update as $\Delta_t^k = w_t^k - w_t$, and sends it back to the server. Upon receiving the update of every agent in S_t, server computes the weights for the next round by aggregating the updates with an aggregation function $\mathbf{g}$: $R^{|S_t|\times d} \to R^d$ and adding the result to w_t. That is, $w_{t+1} = w_t + \eta \cdot g(\{\Delta_t\})$ where $\{\Delta_t\} = \cup_{k\in S_t}\Delta_t^k$, and η is the server's learning rate. For example, original FL paper [McMahan et al. 2016] and many subsequent papers on FL [Bagdasaryan et al. 2020, Bhagoji et al. 2019, Bonawitz et al. 2017, Geyer et al. 2017, Sun et al. 2019] consider weighted averaging to aggregate updates. In this context, this aggregation is referred as Federated Averaging (FedAvg), and yields the following update rule,

$$w_{t+1} = w_t + \eta \frac{\sum_{k\in S_t} n_k \cdot \Delta_t^k}{\sum_{k\in S_t} n_k}.$$

In practice, rounds can go on indefinitely, as new agents can keep joining the protocol, or until the model reaches some desired performance metric (, accuracy) on a validation dataset maintained by the server.

It has been shown that models trained via FL can perform better than locally trained models at agents' side in various settings [Hard et al. 2018, McMahan et al. 2016].

Bibliography

E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov. 2020. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 2938–2948.

A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo. 2019. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pp. 634–643.

K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth. 2017. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191.

H. Bray and S. Talcott. June 18 2005. Firm says up to 40m credit card files stolen. *Boston Globe*.

C. Clifton, M. Kantarcıoğlu, X. Lin, J. Vaidya, and M. Zhu. Jan. 2003. Tools for privacy preserving distributed data mining. *SIGKDD Explorations*, 4(2): 28–34.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *TCC*, pp. 265–284. Springer.

S. Even, O. Goldreich, and A. Lempel. 1985. A randomized protocol for signing contracts. *Communications of the ACM*, 28(6): 637–647.

B. C. M. Fung, K. Wang, and P. S. Yu. 2005. Top-down specialization for information and privacy preservation. In *ICDE '05*, pp. 205–216. Washington, DC, USA.

R. C. Geyer, T. Klein, and M. Nabi. 2017. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*.

O. Goldreich. 2004. *The Foundations of Cryptography*, volume 2, chapter General Cryptographic Protocols. Cambridge University Press. http://www.wisdom.weizmann.ac.il/~oded/PSBookFrag/prot.ps.

A. Hard, C. M. Kiddon, D. Ramage, F. Beaufays, H. Eichner, K. Rao, R. Mathews, and S. Augenstein, 2018. Federated learning for mobile keyboard prediction. https://arxiv.org/abs/1811.03604.

I. Ioannidis and A. Grama. Jan. 6-9 2003. An efficient protocol for Yao's millionaires' problem. In *Hawaii International Conference on System Sciences (HICSS-36)*, pp. 205–210. Waikoloa Village, Hawaii.

M. Kantarcıoğlu and C. Clifton. June 2 2002. Privacy-preserving distributed mining of association rules on horizontally partitioned data. In *The ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'02)*, pp. 24–31. Madison, Wisconsin. http://www.bell-labs.com/user/minos/DMKD02/Papers/kantarcioglu.pdf.

M. Kantarcıoğlu and C. Clifton. Sept. 2004. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE TKDE*, 16(9): 1026–1037. http://ieeexplore.ieee.org/iel5/69/29187/01316832.pdf?isnumber=29187&prod=JNL&arnumber=1316832&arnumber=1316832&arSt=+1026&ared=+1037&arAuthor=Kantarcioglu%2C+M.%3B+Clifton%2C+C.

H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. 2003. On the privacy preserving properties of random data perturbation techniques. In *ICDM '03*, pp. 96–106. Melbourne, FL, USA.

K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. 2005. Incognito: efficient full-domain k-anonymity. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pp. 49–60. ACM, New York, NY, USA. ISBN 1-59593-060-4. DOI: http://doi.acm.org/10.1145/1066157.1066164.

K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. 2006. Mondrian multidimensional k-anonymity. In *ICDE '06: Proceedings of the 22nd International Conference on Data Engineering*, p. 25. IEEE Computer Society, Washington, DC, USA. ISBN 0-7695-2570-9. DOI: http://dx.doi.org/10.1109/ICDE.2006.101.

N. Li, T. Li, and S. Venkatasubramanian. 2007. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE '07*, pp. 106–115. IEEE, Istanbul, Turkey.

Y. Lindell and B. Pinkas. 2002. Privacy preserving data mining. *Journal of Cryptology*, 15(3): 177–206. http://www.research.ibm.com/people/l/lindell//id3_abs.html.

A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. 2006. l-diversity: Privacy beyond k-anonymity. In *ICDE '06*, p. 24. IEEE Computer Society, Atlanta, GA, USA.

H. B. McMahan, E. Moore, D. Ramage, S. Hampson, et al. 2016. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*.

A. Meyerson and R. Williams. 2004. On the complexity of optimal k-anonymity. In *Proceedings of the 23rd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2004)*. ACM Press, Paris, France.

M. Naor and B. Pinkas. Jan. 7-9 2001. Efficient oblivious transfer protocols. In *Proceedings of SODA 2001 (SIAM Symposium on Discrete Algorithms)*. Washington, D.C.

M. Rabin. 1981. How to exchange secrets by oblivious transfer. Technical Report TR-81, Aiken Computation Laboratory, Harvard University.

Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan. 2019. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*.

L. Sweeney. 2002. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5): 557–570.

K.-L. Tan, B. Carminati, E. Ferrari, and C. Jianneng. 2008. Castle: A delta-constrained scheme for k-anonymizing data streams. In *ICDE '08*, pp. 1376–1378. Cancún, México.

X. Xiao and Y. Tao. 2006. Anatomy: simple and effective privacy preservation. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pp. 139–150. VLDB Endowment, Seoul, Korea.

X. Xiao and Y. Tao. 2008. Output perturbation with query relaxation. *PVLDB*, 1(1): 857–869.

A. C. Yao. 1982. Protocols for secure computation. In *Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science*, pp. 160–164. IEEE.

A. C. Yao. 1986. How to generate and exchange secrets. In *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science*, pp. 162–167. IEEE.

Author's Biography

Your Name

Your Name began life as a small child ...