Towards Effective, Efficient and Equitable Privacy-Preserving Machine Learning

Nitin Agrawal

Wolfson College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2021

Acknowledgements

This thesis has been an incredible journey. It would not have been possible without the amazing individuals and institutions I have been fortunate to work with. I am grateful to everyone that provided me with the enthusiasm, resources, funding, foundations, support and courage to undertake this research.

First and foremost, I would like to thank my advisor Nigel Shadbolt. He has been a constant motivating force and so supportive in all my endeavours. His thoughts and ideas have been an incredible inspiration in both my professional and personal life. My conversations with him have always been thought provoking and have shaped my research in unique, yet subtle ways. His forward thinking outlook has been instrumental in my evolution as a researcher.

I would also like to take this opportunity to thank Max Van Kleek, Reuben Binns and Jun Zhao for supporting my research. Their inspiring counsel, constructive criticism and mentorship guided my research and added new dimensions to my research plan. Their expertise and insights also helped drive my research into the human centered perspective of privacy preserving computation, enriching any technical contributions in this thesis.

Outside of the Oxford ecosystem, collaborations and internships have significantly contributed towards the pursuit of this thesis. I was fortunate to be mentored by Adrià Gascón and Matthew Kusner at the Alan Turing Institute, helping me build stronger foundations in privacy-preserving machine learning. I am thankful to them for their support and their contributions to this thesis. I would also like to thank Nishanth Chandran and Divya Gupta at Microsoft Research for hosting me, and supporting and shaping my research. I would like to take this opportunity to also thank Kim Laine and James Bell for being great collaborators contributing to my research in a whole variety of ways. My sincere gratitude also to Callsign for providing generous funding support, including Yogesh Patel for being my industrial mentor and Zia Hayat for his continued trust in myself and my research.

Apart from my mentors, I feel fortunate to have collaborated with wonderful peers during my stints at The Alan Turing Institute and Microsoft Research, including Ali Shahin Shamsabadi, Nishant Kumar, Mayank Rathee and David Butler. I would also

like to thank all the great people I was surrounded by, who always supported me with their guidance, constructive criticism and encouragement. I would like to thank Helena Webb, Ulrik Lyngs and others, including the newer members of our research group — Konrad Kollnig, Siddhartha Datta, Anirudh Ekambaranathan, Ge Wang, who supported me in the final stages of preparing this thesis. Thanks as well to Gatis Mikelsons, William Seymour, Claudine Tinsman and Jack Jackson for being fantastic colleagues. I would also like to express my gratitude to the administrative staff for making the Department of Computer Science a fabulous place for research, including Sarah Retz-Jones, Julie Sheppard, Janet Sadler, Lyn Hambridge, Kelly Ryan, Aaron Huish and Michael Green. Special thanks to Helen Gee for all the support she provided.

It has been an amazing ride amidst the rich history and tradition of Oxford. I still feel awed by the city and everything it has to offer. The plethora of avenues and opportunities that the University of Oxford affords, supported my research endeavour. I will always cherish the deep connections I made during my interactions with people from diverse fields and communities. They will continue to have a long lasting impact on my life and world view.

The friendships I made at Oxford, through its diverse social fabric, made the journey almost as enriching and eventful as the destination. I would like to thank Rishi, Sunil, Angira, Aditi, Dongxia, Mayur, Aditya, Natasha, Viveka, Mohita, Varnika, Hemani, Jishnu, Ayush, Arya and others for mostly bearing with me and always cheering me up through difficult times.

Finally, I am thankful to my parents for raising me in a great environment, and always being so very supportive. I would like to thank my sisters and brothers-in-law for their constant guidance and encouragement in this process. A special thanks to Vatika for her support through some of the rough phases in this journey.

Abstract

Machine learning has assumed an increasingly important role in Artificial Intelligence in recent years. Moreover, in spheres such as online shopping, virtual personal assistants, recommendation systems amongst other things, it is quickly becoming part and parcel of our daily lives. The power of data combined with machine learning algorithms has driven widespread uptake of these AI methods. However, there have been sensitivity and privacy concerns over the data to be processed. This is especially salient in areas such as healthcare and finance. Privacy-preserving machine learning mitigates these privacy challenges through private computation over sensitive data. However, the process is not trivial or without trade-offs.

In this thesis, we focus on designing effective and efficient protocols for facilitating end-to-end privacy-preserving machine learning, particularly for neural network training and inference. We primarily focus on multi-party computation alongside non-cryptographic primitives such as federated learning for private computation. We start with designing QUOTIENT, an efficient two-party secure training and prediction framework. QUOTIENT benefits from the holistic adaptation of standard neural network training into a crypto-friendly training process alongside a customized hybrid MPC protocol for secure computation. Next, we introduce federated learning to support highly decentralized training over unlabeled data. We motivate the idea of 'silos' to ensure superior privacy and isolation across sub-groups. To complete the technical contribution, we propose an MPC-friendly covertly secure commitment scheme for enabling certified predictions. More specifically this helps to enforce non-functional constraints such as fairness, interpretability, and safety over trained models at the time of inference, making the process more equitable. We design, implement and benchmark all of these frameworks to show performance gains in computation, communication and accuracy. We conclude the thesis with a user study focused on enhancing usability, efficiency, assisting in design and helping ensure equity in privacy-preserving computation frameworks at large. This study takes the form of semi-structured interviews with various stakeholders in the privacy-preserving computation ecosystem.

We improve upon the state of the art by more than an order of magnitude in speed, for our protocols, alongside significant gains in accuracy and communication. The user

study presents a rich socio-technical perspective to the purely technical contributions. The combination of theory, practice and evaluation, in this thesis, acts as a multi-perspective framework for motivating design, development and further research in effective, efficient and equitable privacy-preserving machine learning.

Contents

List of Figures

List of Abbreviations

DL	Deep Learning
ML	Machine Learning
AI	Artificial Intelligence
MLaaS	Machine Learning as a Service
DNN	Deep Neural Networks
CNN	Convolutional Neural Networks
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent
PETs	Privacy Enhancing Technologies
PPCTs/PPCs	Privacy-Preserving Computation Technologies
SMC/MPC	Secure Multi-Party Computation
GC	Garbled Circuits
OT	Oblivious Transfer
PVC	Publicly Verifiable Covert
PRNG	Pseudo-Random Number Generator
HE	Homomorphic Encryption
DP	Differential Privacy
FL	Federated Learning
LAN	Local Area Network
WAN	Wide Area Network
HCI	Human Centered Computing

1 Introduction

It is increasingly evident that data is an extraordinary asset, possessing great utility when used with modern machine learning and analytic algorithms. For example, in the context of healthcare, machine learning driven models save lives [1, 2]. They have also been shown to help in situations such as Parkinson's disease [3] and in the preemptive detection of potential life threatening conditions such as sepsis [4]. A number of other studies [5, 6] have looked at the potential for machine learning models in domains such as finance and genomics.

The data used in these systems is often sensitive and of a highly personal nature. Sensitive data is susceptible to abuse [7] and can be used to manipulate decisions and preferences of individuals [8]. Moreover, there has been a growing interest among individuals in taking ownership of their data and potentially monetize it while allowing limited access [9]. Finally, legal restrictions, such as the Health Insurance Portability and Accountability Act (HIPAA) in the US and the General Data Protection Regulation (GDPR) in the UK and EU, restrict collection, distribution, and usage of such sensitive data relating to individuals. These factors limit the availability of sensitive data and might impact the ability of critical domains to benefit from modern data science.

Another major challenge with the data-driven frameworks is their tendency to augment the human biases present in the training data, effectively compiling them into the model [10, 11]. Further concerns revolve around the lack of explainability and interpretability of black-box models that can have a direct, real-world impact on the lives and livelihood of individuals [12]. This is particularly true for Deep Learning (DL) systems, which suffer from limited transparency in the resulting models. While a body of research has studied ways to enforce non-functional constraints such as fairness, explainability and safety, this frequently involves sharing of sensitive data and models, compromising privacy.

As a result, we are faced with a range of societal, ethical and legal dichotomies while, at the same time, already utilizing machine learning for solving the most critical problems of our age. This calls for a delicate balance between privacy and utility. This thesis sits at the heart of this trade-off. The frameworks developed as a part of this thesis focus on *privacy-preserving computation*—a mechanism for secure computation over private data.

Privacy-preserving Computation

Privacy-preserving computation is a subset of Privacy-Enhancing Technologies (PETs). PETs are a broad category of approaches which could include everything from a sticker placed over a webcam [13] to advanced cryptographic techniques [14]. Existing and well-established examples include encryption schemes used to secure data at rest, end-to-end encryption protecting data over the network, and anonymous routing protocols to prevent interactions between identities from being revealed. Such technologies are already widespread, embedded in products and are increasingly integrated into the fabric of the global internet infrastructure. While they each have different underlying approaches and motivations, these technologies are primarily concerned with the protection of data at rest and in transit. They generally assume that once data is

safely transferred to a secure endpoint, it can be decrypted and computed on in the clear; that a single entity performs the computation; and that whether or not the result of the computation is 'private' has a binary answer.

A more recent wave of PETs — including homomorphic encryption (HE) [15], secure multi-party computation (MPC) [16], and differential privacy (DP) [17] — allow these assumptions to be relaxed or even abandoned altogether.

All these technologies have their own premises about the threat model, as well as the security and privacy guarantees. A clear idea of these nuances is integral to meaningfully deliver on the growing privacy expectations of the general public. Furthermore, there are a number of trade-offs that these varieties of privacy-preserving computation offer. For example, MPC is based on cryptographic assumptions but adds a communication overhead and needs non-colluding parties to participate. HE is also based on cryptographic assumptions but needs high computation support. On the other hand, DP offers only a statistical guarantee — about limited impact on the result of a query computed in the presence or absence of an individual's record in the database. Each technology has its own strengths and shortcomings. On top of these trade-offs, all these technologies suffer from challenges relating to design and governance.

For the purpose of this thesis, we focus on MPC-based approaches. Notwithstanding, we do consider the whole repertoire of MPC, HE and DP in Chapter 6. Furthermore, while our proposed MPC protocols are practical, they might not always scale well independently, especially when the data is distributed across a large number of entities. For privacy reasons, centralization might not be ideal, as a result we also explore how privacy-preserving approaches can be employed in large scale decentralized settings. We do this by combining approaches such as Federated Learning (FL) with MPC.

Research Question

This thesis aims to answer the following overarching research question.

> How can we design, build and evaluate effective, efficient and equitable privacy-preserving Deep Learning systems?

In order to address this research question and effect provably secure, understandable and deployable privacy aware deep learning, we decompose this question into four sub-questions:

Q1 How can we design crypto-friendly protocols and architectures for secure deep learning?

Q2 Can secure deep learning approaches be decentralised and/or federated?

Q3 How can we ensure the integrity and security of the derived models?

Q4 How do key stakeholders perceive, understand and evaluate cryptographic primitives such as secure multiparty computation, homomorphic encryption and differential privacy?

An Ever Closer Union

As we describe above, privacy-preserving computation technologies each have their own limitations — in particular with respect to compute and communication requirements, inputs and operations. For example, despite recent advances, both MPC and HE-based computations are still significantly slower than computation over plaintext. Similarly, computation over fixed point numbers and operations like addition, subtraction are more efficient than computation over floating point numbers and operations including multiplication, division and square root. However, vanilla machine learning architectures such as Deep Neural Networks (DNNs) tend to involve large matrix-vector multiplications, non-linear operations and dealing with floating point numbers with precision for training over them. While there have been a large number of practical developments in both cryptography and deep learning, most recent works

for model training [18–21] and prediction [22–24] on encrypted data are largely based on optimizations for *either* the ML model or the employed cryptographic techniques in isolation.

To this end, an effective and efficient privacy-preserving deep learning system needs a delicate combination of development on multiple fronts: (a) optimized custom cryptographic protocols, (b) privacy-aware system design, (c) crypto-friendly machine learning pipelines, (d) scalability and decentralization, (e) equitability, and (f) attention to design and governance. Terms like fairness, equality, non-discrimination, and bias have been recently been used to refer to a set of ethical and legal issues around machine learning. A full exploration of the overlapping and differing meanings of these terms is beyond scope. Here, we use 'equitability' as an umbrella term to refer to these issues.

In this thesis, we show that there is a benefit in taking a holistic approach to the problem i.e. optimizing and reinventing on various fronts simultaneously. In Chapters 3 - 5, we address (a)–(d). For instance, in Chapter 3, we propose a customized quantized neural network architecture and crypto-friendly training strategies alongside an efficient mixed MPC protocol for secure training over it. Similarly, scalability and decentralization are key to making a system usable and widely available. Therefore in Chapter 4, we make use of federated learning (FL) [25] — a decentralized ML training paradigm over local data where only processed model updates are revealed to different parties. We use it alongside knowledge distillation and MPC-based secure aggregation strategy to securely train NNs over unlabeled distributed data. In Chapter 5, we design customized commitment schemes to enable secure certified predictions over private models, ensuring integrity and security of the derived models. This enables secure enforcement of model constraints such as fairness, explainability, safety, amongst others. This, in effect also addresses issues around equitability (e), which happens to be a major concern for a machine learning system. The relatively *opaque* nature of privacy-preserving computation compounds the problem and we try to mitigate these issues in Chapter 5.

Finally, in Chapter 6, we motivate how (a)–(d), while necessary, are not sufficient to meet our goals around making our protocols usable. Primitives like HE, MPC and DP can seem mysterious or even magical in isolation. The mathematical and computation complexity around these primitives raise specific challenges to their usability, adoption and governance. Therefore, we identify and address the challenges around (f) to conclude this thesis while setting stage for a more broader perspective pertinent to the material in this thesis. Through this chapter, we also delve deeper into our goals around (e) through greater transparency and understandability.

Contributions

The proposed work is expected to contribute significantly to the existing body of research, enabling an equitable and more inclusive application of machine learning approaches to sensitive application domains, using customized privacy-preserving computation protocols, while shedding light on the acceptability and accessibility of specific privacy-preserving computation primitives. We highlight the contributions as follows. Contributions (A) - (C) are technical advances, while contribution (D) is a step to advance development, deployment and equitability:

(A) **QUOTIENT: Two-Party Secure Neural Network Training and Prediction.** Recently, substantial efforts have been devoted to the design of secure protocols for machine learning tasks. Much of this is aimed at enabling secure *prediction* from highly-accurate DNNs. However, as DNNs are trained on data, a key question is how such models can be also be *trained* securely. The few prior works on secure DNN training have focused either on designing custom protocols for existing training algorithms, or on developing tailored training algorithms and then applying generic secure protocols. In this work, we investigated the advantages of designing training algorithms alongside a novel secure protocol, incorporating optimizations on both fronts. QUOTIENT incorporated key

components of modern DNN training such as layer normalization and adaptive gradient methods. It significantly improved upon the state-of-the-art in DNN training in two-party computation. Compared to prior work, we obtained an improvement of 50× in wireless area network (WAN) time and 6% in absolute accuracy [26].

(B) **Secure federated learning for unlabeled data.** In real life settings, most data is privacy sensitive and unlabeled. There have been efforts devoted to designing highly accurate DNN architectures which ingest large amounts of data [27–29]. Some of these recent efforts such as Federated Learning [30] have focused on user-privacy. However, vanilla federated learning presents little utility for training on unlabeled data. This work proposed multi-model federated generational distillation (MFGD), an approach which combines federated learning with knowledge distillation for secure training on unlabeled data. We investigated the advantages of using multiple federated models and accumulating their combined knowledge into individual models across multiple 'generations'. We also motivated the superior privacy and security of our approach over vanilla FL. Our approach improved model accuracy by up to $\sim 54\%$ on a classification task, exclusively using unlabeled data, over multiple generations.

(C) **MPC-friendly commitment scheme for enabling certified predictions.** We addressed the problem of efficiently verifying a commitment in a two-party computation. This addressed the scenario where a party P1 commits to a value x being used in a subsequent secure computation with another party P2 that wants to receive assurance that P1 did not cheat, i.e. that x was indeed the value input into the secure computation. Our constructions operated in the publicly verifiable covert (PVC) security model, which is a relaxation of the malicious model of MPC appropriate in settings where P1 faces a reputational harm if caught

cheating. We introduced the notion of a PVC commitment scheme and indexed hash functions to build commitment schemes tailored to the PVC framework, and proposed highly effective constructions for both arithmetic and Boolean circuits that result in highly efficient circuits. From a practical standpoint, our constructions for Boolean circuits are 60× faster to evaluate securely, and use 36× less communication than baseline methods based on hashing. Moreover, we showed that our constructions are tight in terms of required non-linear operations, by proving lower bounds on the non-linear gate count of commitment verification circuits. Finally, we presented a technique to amplify the security properties of our constructions that allows us to efficiently recover malicious guarantees with statistical security.

(D) **Exploring design and governance challenges in the uptake of cryptographic primitives for privacy-preserving computation.** Homomorphic encryption, secure multi-party computation, and differential privacy are part of an emerging class of Privacy Enhancing Technologies [31, 32] which share a common promise: preserving privacy whilst also obtaining the benefits of computational analysis. Due to their relative novelty, complexity, and opacity, these technologies provoke a variety of novel questions for design and governance. We interviewed researchers, developers, industry leaders, policymakers, and designers involved in their deployment to explore motivations, expectations, perceived opportunities and barriers to adoption. This provided insights into pertinent challenges in the adoption of these technologies, including how they might make a nebulous concept like privacy computationally tractable and how to make them more usable by developers; and how they could be explained and made accountable to stakeholders and wider society. We concluded with implications for the development, deployment, and responsible governance of these privacy-preserving computation techniques.

Dissemination

To date the work in this thesis has been disseminated via the following papers:

1. Nitin Agrawal, Ali Shahin Shamsabadi, Matt J Kusner, and Adrià Gascón. "QUOTIENT: Two-Party Secure Neural Network Training and Prediction". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 2019, pp. 1231–1247

2. Nitin Agrawal, James Bell, Adrià Gascón, and Matt J Kusner. "MPC-Friendly Commitments for Publicly Verifiable Covert Security". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 2021

3. Nitin Agrawal, Reuben Binns, Max Van Kleek, Kim Laine, and Nigel Shadbolt. "Exploring Design and Governance Challenges in the Development of Privacy-Preserving Computation". In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 2021, pp. 1–13 **(Honorable Mention Award)**

Thesis Outline

The structure of the thesis is as follows:

In Chapter 2, we summarise the background and the motivation for the research. We start with preliminaries, setting out the context, background and foundations for this thesis. Afterwards, we review the relevant pieces of existing research, with a focus on their applicability and limitations.

In Chapter 3, we present, in detail, our proposal for secure and efficient two-party training and inference for DNNs, which we call 'QUOTIENT'. We build on the preliminaries in Chapter 2 to provide other necessary background and prove the security of our method. We conclude with an implementation, benchmarking and comparison of our method with the existing state of the art.

In Chapter 4, we extend the ideas around secure training in Chapter 3 for a more decentralized setting with a large number of parties. We motivate, describe and benchmark a protocol for secure federated learning over unlabeled data.

In Chapter 5, we present a strategy for certified predictions through a novel type of MPC-friendly commitment schemes in a covertly secure model. We define the problem in detail, design a framework, propose a solution and benchmark it against the existing state of the art.

In Chapter 6, we explore a wider user centred context regarding the contributions in this thesis. We interview a variety of stakeholders involved in deployment of privacy preserving computation technologies. We discuss the barriers, opportunities and perceptions associated with the methods proposed in this thesis as well as the privacy-preserving computation technologies at large.

Finally, in Chapter 7, we discuss the contributions of this thesis in their entirety, highlighting the broader challenges and opportunities while setting the stage for future work.

2
Background and Motivation

Contents

2.1 A Brief Overview of Deep Learning

Deep learning is a powerful form of machine learning that has been shown to help solve a variety of complex tasks such as image and speech recognition [27, 28]. DL methods

make use of multiple layers stacked together to identify structure and patterns in large data sets [29]. It does not need the data to be pre-processed and automatically learns to extract relevant features. It is effected through Deep Neural Networks.

2.1.1 Deep Neural Networks

Deep Neural Networks belong to a class of so-called deep architectures, possessing an ability to utilize higher level features. These features are composed from lower level features [35]. More generally, deep architectures, which enable "deep learning" comprise a variety of flavors including graphical models [36] and neural network models with multiple hidden layers [37]. Deep neural networks have been found to be successful at complex supervised learning tasks, where the objective is to learn the relationship between the data and the labels [38]. Around the time of their inception deep neural networks were only moderately realizable and often suffered from optimization issues [39, 40]. However, owing to recent advances in computing hardware, the availability of large datasets and strategies to address over-fitting, they have been enjoying great success [36, 41].

All DNNs are defined by a core set of operations, called a *layer*. These layers are repeatedly applied to an input $\mathbf{a}^0$ to produce a desired output $\mathbf{a}^L$, where the number of repetitions (or layers) L is called the *depth* of the network. Every layer consists of a linear operation and a non-linear operation and, depending on the type of network, each layer takes on different forms. We describe three popular layer types that make up a large portion of state-of-the-art DNNs: *fully-connected*, *convolutional*, and *residual*.

Fully-Connected Layers. To define a layer we simply need to define the linear and non-linear operations they use. In fully-connected layers these operations are as follows: Given an input $\mathbf{a}^{l-1} \in \mathbb{R}^{h_{l-1}}$ of any fully-connected layer $l \in \{1, \ldots, L\}$, the layer performs two operations: (1) a multiplication, $\mathbf{W}^l\mathbf{a}^{l-1}$ with a weight matrix $\mathbf{W}^l \in \mathbb{R}^{h_l \times h_{l-1}}$; and (2) a non-linear operation, most commonly the Rectified Linear Unit: $\textsc{ReLU}(x) = \max(x, 0)$. So the full layer computation is $\mathbf{a}^l = \textsc{ReLU}(\mathbf{W}^l\mathbf{a}^{l-1})$,

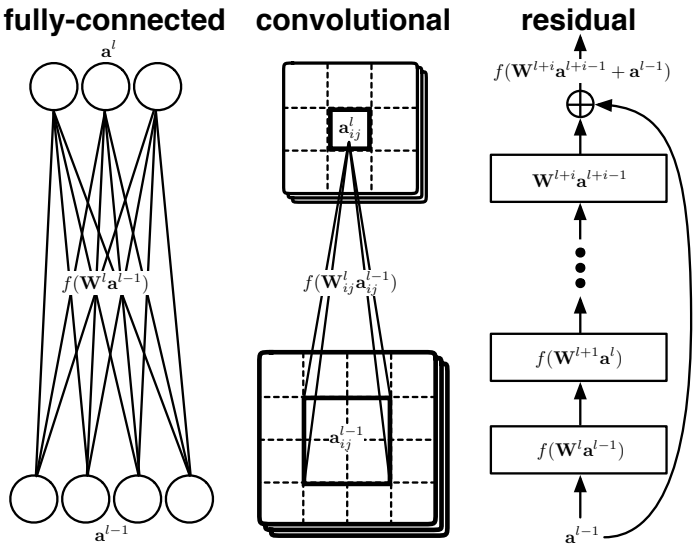

Figure 2.1: Popular deep neural network layers: *fully-connected*, *convolutional*, and *residual*.

where $\mathbf{a}^l$ is called the *activation* of layer l. Note that 'fully-connected' refers to the fact that any entry of $\mathbf{a}^{l-1}$ is 'connected' to the output $\mathbf{a}^l$ via the weight matrix $\mathbf{W}^l$, shown schematically in Figure 2.1 (*left*). Note that, more generic representations involve an intermediate step, adding a bias term b^l to compute $(\mathbf{W}^l\mathbf{a}^{l-1} + b^l)$ before performing the non-linear operation ($\mathbf{a}^l = \textsc{ReLU}(\mathbf{W}^l\mathbf{a}^{l-1} + b^l)$). In practice, this can be handled by suitably modifying $\mathbf{W}^l$ and $\mathbf{a}^{l-1}$ prior to performing operation (1).

Convolutional Layers. Convolutional layers are essentially fully-connected layers with a very particular connectivity structure. Whereas the input and output of a fully-connected layer are vectors, the input and output of a convolutional layer are 3^{rd}-order tensors (i.e., an array of matrices). Specifically, the input $\mathbf{a}^{l-1} \in \mathbb{R}^{h_{l-1} \times w_{l-1} \times c_{l-1}}$ can be thought of as an image with height h_{l-1}, width w_{l-1}, and channels c_{l-1} (e.g., $c_{l-1} = 3$ for RGB images).

To map this to an output $\mathbf{a}^l \in \mathbb{R}^{h_l \times w_l \times c_l}$, a convolutional layer repeatedly looks at small square regions of the input, and moves this region from left-to-right, from top-to-bottom, until the entire image has been passed over. Let $\mathbf{a}^{l-1}_{ij} \in \mathbb{R}^{k_{l-1} \times k_{l-1} \times c_{l-1}}$ be the $k_{l-1} \times k_{l-1}$ region starting at entry (i, j). This is then element-wise multiplied

by a set of weights $\mathbf{W}^l \in \mathbb{R}^{k_l \times k_l \times c_{l-1} \times c_l}$ and summed across the first three dimensions of the tensor. This is followed by a non-linear operation, also (most often) ReLU to produce an entry of the output $\mathbf{a}_{ij}^l \in \mathbb{R}^{1\times1\times1\times c_l}$. Note that $\mathbf{a}_{ij}^{l-1}, \mathbf{a}_{ij}^l$ and $\mathbf{W}^l$ can all be vectorized such that $\mathbf{a}_{ij}^l = f(\mathbf{W}^l \mathbf{a}_{ij}^{l-1})$. This operation is shown in Figure 2.1 (*center*). Apart from the dimensionality of the layers, other hyperparameters of these networks include: the size of square region, the number of 'pixels' that square regions jump in successive iterations (called the *stride*), and whether additional pixels with value 0 are added around the image to adjust the output image size (called *zero-padding*).

A common variant of a convolutional layer is a *pooling layer*. In these cases, the weights are always fixed to 1 and the non-linear function is simply $f(\cdot) \triangleq \max\{\cdot\}$ (other less popular $f(\cdot)$ include simple averaging or the Euclidean norm).

Residual Layers. The final layer type we consider are residual layers [42]. Residual layers take an activation from a prior layer $\mathbf{a}^{l-1}$ and add it to a later layer $l+i$ before the non-linear function $f(\cdot)$ (usually ReLU) is applied: $f(\mathbf{W}^{l+i}\mathbf{a}^{l+i-1} + \mathbf{a}^{l-1})$. The intermediate layers are usually convolutional but may be any layer in principle. Figure 2.1 (*right*) shows an example of the residual layer. DNNs with residual layers were the first to be successfully trained with more than 50 layers, achieving state-of-the-art accuracy on many computer vision tasks, and are now building blocks in many DNNs.

2.1.2 Federated Learning

Federated learning was proposed by Google in 2015 [30, 43, 44]. The main idea centered around training a machine learning model over distributed datasets in a decentralized fashion while preventing data leakage. As a result, the local data of a user never leaves the device and only the intermediate model updates does. More formally, it starts with N clients $\{C_1 \dots C_N\}$ willing to collaboratively train a model over the consolidation of their individual respective datasets $\{DS_1 \dots DS_N\}$. In a conventional setting, one would centralize all the data as $DS = DS_1 \cup \dots DS_N$ to train a model. In a federated learning process, the data owners jointly train a model such that any client C_i does

not need to share its data D_i with any other client. In fact, the data never leaves any of the clients' premises. Broadly the process involve the following steps:

1. Clients maintain a local copy of their model $\{C_1 \dots C_N\}$. They locally compute training gradients $\{G_1 \dots G_N\}$ over their local data and send the results to the server S.

2. The server performs aggregation over the individual gradients to compute $G = G_1 \oplus \dot{G}_N$, where $\oplus$ is the aggregation function.

3. The server sends the aggregated results back to the clients.

4. Clients update their model with the gradients communicated by the server.

Recently, there have been further developments addressing the statistical challenges [45, 46] and enhancing security [47, 48] of the federated learning process. We build upon this approach in Chapter 4.

2.1.3 Deep Learning as a Service

Another recent development is the provision by big cloud service providers of machine learning as a service (MLaaS). This is an extremely important innovation in the case of deep learning which is both compute and data intensive. Some of these cloud platforms include Amazon ML[1], Microsoft Azure ML Studio[2], Google Prediction API[3], BigML[4], IBM Watson ML Builder[5] and several other startups[6,7,8]. However, such services expose both the users as well as the service providers to privacy risks [49]. We discuss some these risks in the next few subsections.

[1] https://aws.amazon.com/machine-learning
[2] https://studio.azureml.net
[3] https://cloud.google.com/prediction
[4] https://bigml.com
[5] https://dataplatform.cloud.ibm.com/docs/content/analyze-data/ml-model-builder.html
[6] https://www.havenondemand.com
[7] https://mljar.com
[8] https://www.nexosis.com

2.1.4 Privacy Threats for Deep Learning Models

A machine learning task often comprises three parties— the data owners, the entity undertaking the computation and an end user [50]. While these roles might not always be exclusive, involvement of multiple parties introduces a number of potential privacy risks. One of the immediate threats is to data privacy when the computation over sensitive data is outsourced to an external party such as cloud service providers. This potentially exposes data to breaches [51] and data abuse [52]. Strategies to mitigate these types of attacks involve storage of data in an encrypted format and limiting the data storage to just the extracted features i.e. discarding the raw data once the desired features are curated or determined. Similar challenges also crop up when performing collaborative learning over distributed data. Quite often, in domains such as healthcare, it is useful to combine data from multiple entities and perform model training. However, this generally would need data to be shared, which is a privacy threat. Solutions like federated learning [44] and multi-party computation [53] provide mechanisms for safeguarding data privacy while computing on it. This thesis focuses particularly on these attacks and mitigation strategies. We discuss specific applications of these technologies such as secure neural network training (Chapter 3) and certified predictions for equitability (Chapter 5) in the next few chapters.

White-box access attacks Quite often the features extracted from a dataset are stored and retained by the party carrying out the computation under unsecure settings. This is particularly true for machine learning models such as Support Vector Machines (SVM) [54] where the features are stored as a part of the model itself. Under such settings, it might be possible to reconstruct the original data using the extracted features. For example, there have been attempts at reconstructing fingerprint images using minutiae-based features [55] and reconstruction of touch gestures using features such as direction and velocity [56]. Nonetheless, such attacks are contingent upon white-box access to the model i.e. the model's intrinsic parameters must be

accessible by the adversary. Strategies to limit such attacks involve limiting insecure storage of intermediate features [57].

Black-box access attacks While white-box access, naturally, opens a large number of opportunities for an adversary, restricting such access might not be sufficient to entirely limit an adversary. In recent times, a number of attack strategies such as model inversion attacks [58] and membership inference attacks [59] have been shown to be feasible even when an adversary is only limited to black-box access to the model i.e. the adversary is only able to query the model and infer the output.

In particular, a model inversion attack as proposed in [58] enables an adversary with black-box access to a model $f(x_1, x_2...x_n)$ to infer a feature x_k given the error statistics, marginal priors for variables and knowledge about the dependent variable and other features. However, the attack is brute force driven and can only be realized when x_k assumes a limited number of values. This attack is a black-box access analogue of the previously discussed reconstruction attack. Proposed methods to limit this type of attack include restricting black-box access to the model i.e. supplying the end users with approximate confidence values [58] or only granting access to the specific output labels rather than the probability values. In contrast to model inversion attacks, membership inference attacks are more specific in nature. Given a model f and a sample x, they aim to infer whether the sample x was part of the training dataset for model f [59]. Such attacks are a consequence of shadow training techniques. Shokri et al. [59] proposed coarsening the precision of the prediction vector and limiting the prediction vector to only k classes as ways to mitigate such attack. Furthermore, since overfitting is one of the leading characteristics that facilitates such attacks, regularization while training also helps in limiting them [60].

2.1.5 Potential of Deep Learning for Sensitive Data

Deep learning has also been evaluated to be effective in a variety of domains where privacy is critical. Within medical diagnosis, it has been shown to be promising for

early diagnosis of Alzheimer's disease [61], diagnosis of breast nodules and lesion [62] and classification of skin cancer [63]. It has also been shown to be effective in healthcare delivery and automated patient treatment planning [64, 65]. Similarly, learning over genomic data has been used for classifying cancers [66] and predicting chromatin marks in DNA sequences [67]. In finance and banking, DL has proven to be an asset for a variety of use cases such as portfolio management [68], risk and credit assessment [69], and regulatory compliance [70]. Besides these, DL has been explored to be useful for a variety of other privacy-centric domains such as predicting crimes [71] and identifying fraud [72].

2.2 Privacy-Preserving Computation Technologies

We introduced privacy-preserving computation technologies in Chapter 1. Here, we focus on a subset of these technologies in detail.

2.2.1 Homomorphic Encryption

Homomorphic encryption enables blindfolded evaluation of a function f over encrypted data. Early attempts at homomorphic encryption were limited in either the number or the type of operations over the data which could be encrypted [73–81]. These include partially homomorphic encryption schemes, where unlimited applications of limited operation types are supported. An early example of such a scheme is RSA [74], the first realizable public key cryptosystem. RSA is based on the difficulty of factoring a product of two large prime numbers [82]. In 1978, RSA was shown to be multiplicatively homomorphic [83] i.e. the underlying plaintexts could be multiplied by operating upon the corresponding ciphertexts (by multiplying the ciphertexts, in this case). In 1985, another public key cryptosystem El-Gamal [76] was proposed, which satisfied the multiplicatively homomorphic property. In 1982, the first additively homomorphic encryption system Goldwasser-Micali (GM) [80] was proposed i.e. two ciphertexts could be operated upon in a manner to in effect add the corresponding underlying

plaintexts. However, the homomorphic property of GM was only limited to binary numbers. The scheme is based on the hardness of the quadratic residuosity problem [84]. GM was extended into a more inclusive additively homomorphic cryptosystem, Benaloh [77]. In the Benaloh scheme, addition on plaintext could be achieved through multiplication of corresponding ciphertexts. In 1999, another additively homomorphic encryption system, Paillier [81], was proposed. The system is based on the composite residuosity problem [85]. Similar to the Benaoh scheme, multiplication of Paillier ciphertexts enables addition of corresponding plaintext messages. Additionally, Paillier supports ciphertext-plaintext multiplication upon exponentiation of a ciphertext to a plaintext message i.e. raising an encrypted message to the power of a plaintext message will result in product of the two plaintexts upon decryption. This makes Paillier apt for evaluating an encrypted query over a plaintext machine learning model [86]. Other examples of additively homomorphic encryption schemes involve lattice-based schemes Kawachi-Tanaka-Xagawa (KTX) [79], Okayamoto-Uchiyama [78], Naccache-stern [75], a generalization of Benaloh, Damagard-Jurik [73] and Galbraith [87]. The latter two were generalizations of Paillier.

The schemes discussed thus far are partially homomorphic i.e., they are homomorphic with respect to a limited number of operations, predominantly either addition or multiplication. In 2005, the Boneh-Goh-Nissim (BGN) [88] method was proposed. BGN supports an unrestricted number of additions and one multiplication while keeping the size of the ciphertext constant. Such schemes are better referred to as Somewhat Homomorphic Encryption Schemes (SWHE). Before BGN, a few other SWHE schemes were proposed [89–91], however, they were later proven to be either impractical or insecure. Another class of SWHE includes Sander, Young and Yung (SYY) [92] and Melchor [93] which support evaluation of AND, OR & NOT gates over constant depth circuits. These schemes were further extended for branching problems [94].

While PHE and SWHE schemes are limited in their application, a Fully Homomorphic Encryption (FHE) scheme admits an unrestricted type and number of operations.

In 2009, Gentry proposed the first realizable FHE scheme as a part of his doctoral research [15]. This scheme was based on ideal lattices and proposed the key methods of bootstrapping and squashing. These methods enable a reduction in intermittent noise, facilitating decryption of ciphertexts after being operated upon. This was not feasible in the case of underlying CGH-type SWHE schemes [95]. Although the scheme on its own was too compute-intensive to be practical, it motivated a number of faster and optimized FHE schemes. Initial improvements included integer-based FHE schemes [96–100]. Following this, a FHE scheme based on the hardness of Ring-Learning with Errors (RLWE or Ring-LWE) was proposed in 2011 [101]. Subsequently, a LWE based scheme using relinearization technique for regularizing the size of ciphertext was proposed in 2014 [102]. The same scheme was later optimized as the FV scheme [103]. More recently, a class of FHE schemes based on NTRUEncrypt has been explored. NTRUEncrypt is an encryption scheme proposed in 1998 [104]. The scheme is a lattice-based public key cryptosystem and relies on the hardness of approximating the shortest vector problem in lattices [105]. The encryption and decryption process only involves polynomial multiplication, which makes the scheme faster than other public key cryptosystems. NTRUEncrypt scheme was shown to be adaptable for FHE in 2012 [106]. Some modern examples of NTRU encryption schemes include [107–111].

Implementations of FHE Schemes

While most of the PHE, SWHE and FHE schemes have been implemented in some form or other, in this section we review the most scalable FHE frameworks. IBM's HElib[9] is one the most elaborate FHE implementations. The framework has been implemented in C++ and uses the BGV [101] encryption scheme. It employs a variety of optimizations including SV ciphertext packing [112]. Some of the algorithms and optimization used in HElib are described in [113]. The latest version of HElib supports bootstrapping and multi-threading [114]. libScarab[10] [98] is another FHE encryption

[9] `https://github.com/shaih/HElib`

[10] `https://github.com/hcrypt-project/libScarab`

library, implemented in C. It is based on the SV cryptosystem [112] but does not support some of the recent FHE optimizations. One of the more recent and elaborate FHE libraries include Microsoft's Simple Encrypted Arithmetic Library (SEAL)[11] [115]. The library is implemented in C++ and is built on the FV cryptosystem [103].

The Fastest Homomorphic Encryption in the West (FHEW)[12] [116] system is a C++-based library which offers fast bootstrapping and hence a fast evaluation of NAND gates. The library is based on LWE over Torus. Another variant of the same approach is the C/C++-based library TFHE which claims to offer fast unrestricted gate-by-gate bootstrapping. It implements the ring variant of the GSW cryptosystem [117] and employs the optimizations proposed in [116, 118, 119]. Other prominent homomorphic encryption libraries incorporate the lattice-based library PALISADE[13], the GPU accelerated library cuHE [120] based on the DHS Scheme [106, 121] and the Homomorphic Encryption for Arithmetic of Approximate Numbers (HEAAN) framework [122], which supports approximate operations between rational numbers.

2.2.2 Secure Multi-Party Computation Protocols

Secure Multi-Party Computation (MPC) is a class of cryptographic protocols which rely on secure evaluation of data shared across multiple parties for private processing over sensitive data. In the most trivial case, party P_i possesses an input x_i and gets an output y_i upon computation of function f over the combined set of $x_i s$ ($i \in \{1...n\}$, where n is the number of parties). Most MPC protocols could be defined by the choice of the circuit for computing a particular function and the type of secret sharing scheme. We review some of the prominent protocols and secret sharing schemes in this section.

[11] `https://sealcrypto.codeplex.com/`
[12] `https://github.com/lducas/FHEW`
[13] `https://git.njit.edu/palisade/PALISADE`

Oblivious Transfer

Oblivious transfer (OT) [123] is a key cryptographic primitive and a crucial building block of a number of secure multi-party computation protocols. In the case of 1-out-of-2 OT [124], party P_1 (Sender) holds two messages, m_0, m_1 and party P_2 (Receiver) chooses one of the two messages based on its Boolean choice bit b. Here b indicates the index of the message i.e. P_2 must end up with message m_b. In this process, P_2 learns nothing about the other message m_{1-b} and P_1 learns nothing about b. The OT protocol could potentially be used to compute any function [125]. One of the ways to achieve this functionality is to have two rounds of communication where the party P_1 sends the encryption of b to P_2 while P_2 constructs encryption of messages m_0 and m_1 such that P_1 is only able to decrypt the message corresponding to b as modelled in [126]. The protocol relies on the hardness of the Decisional Diffie Hellman (DDH) assumption [127]. However, the original protocol is compute-intensive in terms of the public key cryptography operations. A remarkable advancement in the practicality of OT protocols was the discovery of OT extension [128]. This protocol enabled computing a small number of OTs, say 128, and then bootstrap them to execute many fast OTs. Since then, optimizations in both the base OTs and the OT extension procedure [129] have led to implementations that can perform over ten million OT executions in under one second [130]. More recently, optimizations such as Correlated OT (C-OT) and Random OT (R-OT) [129] have been proposed to further optimize the protocol.

Yao's Garbled Circuits

Yao's garbled circuit (GC) [131] is a protocol for two-party secure computation. GC extensions for more than two parties have also been explored [132]. The goal is to be able to compute a function over shared inputs privately. Under two party settings P_i holds a bitstring input x_i and the function f is represented as a Boolean circuit ($i \in \{1, 2\}$). In the initial construction [131], P_1 acts as the garbler and encrypts the circuit. For each wire in the circuit, P_1 chooses k-bit long random labels corresponding

to each possible inputs ($\{0,1\}$). Here k is the security parameter and is usually set to 128 [133]. P_1 then encrypts the output wire label for gate g using the wire labels for each possible combination of the inputs ($b_i \in \{0,1\}$). This computation table is then permuted and sent over to P_2. P_2 employs strategies such as 1-out-of-2 Oblivious transfer or homomorphic encryption protocols along with its input bits, in order to obtain the encrypted value corresponding to $P_1's$ input. The process is repeated for all the gates g in the circuit f. The method is described in detail by Lindell and Pinkas [134]. A variety of optimizations have been proposed to make GCs efficient. These include a row reduction strategy [16] for reducing the size of the garbled tables to 3 rows instead of 4 and the point-and-permute approach [132] where a selection bit is used to determine the order of rows in the table instead of a random permutation of the garbled table, reducing the computation load by about 4 times. Another key optimization involves the construction described in [135], which enables almost free computation of XOR gates over shared inputs. Optimized implementations of GCs have been an area of recent focus [136–139].

Secret Sharing

Secret sharing is another key building block for secure multi-party computation. This determines the manner in which a secret is shared among a set of parties [140, 141]. The underlying aim is to be able to reconstruct the secret only when a predetermined subset of parties agrees to do so. This approach is referred to as k-out-of-n secret sharing, where the secret is distributed across n parties and at least k shares are required to reconstruct the secret. The first such scheme was proposed by Shamir in 1979 [140]. The secret shared values can be used for secure computation protocols over the shares [142, 143]. The most used secret sharing schemes for SMC protocols rely on n-out-of-n secret sharing. Additive secret sharing [144] is one such scheme, where operations are performed in ring $\mathbb{Z}_{2^k}$. Here secrets are represented with k-bit integers. In order to share a secret x between two parties, one party is supplied with

a random value $r \in$ ring $\mathbb{Z}_{2^k}$ while the other is supplied with the share $x - r \bmod 2^k$. Under this scheme, operations such as addition and scalar multiplication could be computed by merely performing the operation on the shares locally, while protocols such as OT and GC could enable other non trivial operations on the shares. The analogous sharing schemes for Boolean values uses xor operations (instead of addition) and is commonly referred to as Boolean sharing [145].

Note that cryptographic primitives discussed thus far are not mutually exclusive. A number of secure multi-party computation protocols employ homomorphic encryption, especially where a precomputation or an offline phase is involved. One such example is the SPDZ protocol [146] which uses SWHE for offline computation phase. The protocol was later optimized into SPDZ2 protocol [147].

Implementations of SMC protocols

Similar to FHE approaches, there are a number of publicly available SMC frameworks. Obliv-C [148] is a C-based framework for two-party computation (2PC) with GCs, offering security against semi-honest adversaries i.e. assuming that the corrupted parties will not deviate from the protocol. ABY [145] is a C++-driven 2PC implementation which combines SMC computation schemes based on Yao's garbled circuits, Arithmetic sharing, and Boolean sharing. The implementation assumes the semi-honest adversary model. BatchDualEx [149] offers security against malicious adversaries for general-purpose 2PC supported by an offline computation phase. EMP-toolkit [150, 151] offers an highly customizable 2PC and MPC with garbled circuits, secure against semi-honest adversaries (emp-sh2pc) as well as malicious adversaries (emp-ag2pc, emp-agmpc). It furthermore offers OT, OT extensions and some of the other advanced variations of OT (emp-ot). SPDZ [152], MP-SPDZ[14], and FRESCO[15] offer implementations of the SPDZ protocol in addition to other protocols for security

[14] `https://github.com/n1analytics/MP-SPDZ`
[15] `https://github.com/aicis/fresco`

against semi-honest as well as malicious parties i.e. security against corrupted parties which could potentially deviate from the protocol. Tf-encrypted[16] is an up-and-coming implementation of 3PC with secret sharing tailored for machine learning application. The framework is based upon Tensorflow [153]. Other prominent SMC frameworks involve TinyLego[17] and MpyC [154].

2.2.3 Differential Privacy

Broadly, differential privacy pertains to a setting, where addition or removal (or replacement, under some interpretations) of an entry from a database D does not significantly change the result of a query on the database. In 1979, [155] proposed an adversary Tracker, which could use multiple queries in order to infer private contents of a database. This was extended upon in [156], where it was shown that astonishingly few number of queries could be used to reconstruct a private database. This led to the development of (ϵ, δ)-differential privacy [17, 157], which can be formally defined as follows:

A randomized function f gives (ϵ, δ)-differential privacy for all databases D and D', non-negative values ϵ, δ and $\forall S \subseteq$ range of f, where D and D' differs by at most one record iff,

$$Pr[f(D) \in S] \leq \delta + e^{\epsilon} Pr[f(D') \in S]$$

Here δ represents the confidence level or the probability by which the result upon application of f on D varies in excess of a factor of e^{ϵ} upon application of f onto D'. ϵ is commonly referred to as the privacy budget. The smaller the values of ϵ and δ, the tighter are the privacy guarantees. This definition reduces to ϵ-differential privacy [158] when $\delta = 0$. It delivers comparatively stricter and non-probabilistic privacy guarantees.

[16] `https://github.com/mortendahl/tf-encrypted/`
[17] `https://github.com/AarhusCrypto/TinyLEGO`

Differential privacy is employed by addition of noise. In particular, ϵ-differential privacy could be realized using Laplacian mechanism [158] and Exponential mechanism [159] while (ϵ, δ)-differential privacy could be realized using Smooth sensitivity framework [160].

2.3 Privacy-Preserving and Equitable Deep Learning

We now review how the various cryptographic methods, discussed earlier, have been used in frameworks that support modern machine learning. In particular, we describe how they have been adapted for building privacy-preserving and equitable deep learning systems. Privacy-preserving deep learning can be divided into an inference problem i.e. when a pre-trained network is used for an inference/classification task, and a training problem, where a neural network is trained securely on private data. In this section, we review state-of-the-art approaches for neural network inference and training employing homomorphic encryption, secure multi-party computation and differential privacy. For the sake of brevity we do not discuss approaches using hardware-protected enclaves [49] such as Intel SGX [161]. While relevant, these approaches are orthogonal to approach in this thesis.

2.3.1 Privacy-preserving Inference

CryptoNets [23] was one of the first efforts at secure inference over deep neural networks. It used homomorphically encrypted input for prediction over a pre-trained neural network, predominantly using square activation function. Square activation function makes the architecture more feasible for HE. Conventional activation functions such as sigmoid and rectified linear units [162] would need a high-degree polynomial approximation for compatibility with HE, making them highly compute-intensive. The authors report a running time of 297.5s per image classification task on MNIST dataset [163, 164] using a convolutional neural network (CNN) model. CryptoNets rely on PHE

i.e. it can only support a finite number of operations in order to keep the noise tractable. Therefore, they could only support limited depth networks. More recently, Bourse et al. [22] proposed a FHE-driven secure inference strategy for Discretized Neural Networks (DiNNs). The approach uses fast bootstrapping in order to realize high depth networks. Similarly, Sanyal et al. [24] proposed a FHE-driven approach for prediction over binary neural networks [165]. The approach uses the ternary nature of weights and activation in conjunction with machine learning heuristics. However, both these approaches are limited by the architecture of the network i.e. they can only be adapted for discretized networks. While retraining a network for secure inference is not ideal, secure multi-party computation driven models have shown promise in terms of being more generic. DeepSecure [166] offers a garbled circuit driven framework for secure inference over floating point deep neural networks. It proposes lower dimensional prepossessing in order to accelerate the secure computation process. DeepSecure takes 1.08s and 9.67s for classification of MNIST images using a CryptoNets-like CNN with and without the preprocessing step respectively. SecureML [18] uses Yao's garbled circuits for secure two party computation (2PC) inference over fully-connected neural networks. It uses Pallier's additive homomorphic encryption [81] for accelerating some of the intermediate processes. SecureML-driven approach takes about 4.7s for predictions over the MNIST dataset using a 2 hidden layer fully connected network. However, its online phase only takes about 0.18s. MiniONN [167] is built upon SecureML's approach and proposes a further optimized protocol for an offline matrix multiplication phase. In particular, it uses lattice-based additively homomorphic encryption for generation of multiplication triples. It is about 5x faster than SecureML for its offline computation phase. Chamaleon [168], another MPC-driven framework uses a three-party setup employing the third party as the dealer and eliminating the need for communication and compute-intensive oblivious transfer for matrix multiplication. It claims to be 4.3x faster than the DeepSecure framework. More recently, Gazelle [86], a low latency hybrid SMC and HE framework has been proposed. It proposes a packed

additively homomorphic encryption (PAHE) strategy for optimized computation and communication over fully connected and convolutional layers. It uses this approach in conjunction with the garbled circuit driven secure computation of non-linear operations (activation functions). It furthermore uses the linear algebraic optimization proposed in [113]. Gazelle has been reported to be about 13x, 25x, 48x, 160x and 9900x faster than Chameleon, MiniONN, DeepSecure, SecureML, and CryptoNets respectively.

2.3.2 Privacy-preserving Training

While there have been a number of studies for secure inference over deep neural networks, only a limited body of research has focused on secure training. This could be attributed to the relative complexity of the training task. In 2015, Shokri and Shmatikov [169] proposed a distributed learning environment employing differential privacy. It enables multiple parties to train a model locally on their own dataset and share a fraction of their parameters, obtaining a globally accurate model. They, furthermore, explored differential privacy to limit indirect privacy leakage through these parameters. The approach has more recently been extended into federated learning driven strategies [30]. However, these approaches could leak information about local data to an honest-but-curious server. Phong et al. [170] addressed this by employing additively homomorphic encryption for sharing local parameters (gradients) with a curious server. Beyond the realms of differential privacy, Aono et al. [171, 172] proposed a purely homomorphic encryption driven logistic regression. This is enabled through polynomial approximation of non-linear operations. This was extended upon for neural networks in [21, 173], where polynomial approximation of the activation function was proposed. However, these approaches produce inferior accuracy values. Secure distributed approaches employing differential privacy have also been explored in [47], however under quite restricted settings. Most of the implemented systems rely on SMC protocols for training. SecureML [18] is one of the most successful attempts at training neural networks privately using secure 2PC. SecureML employs

garbled circuit driven protocols for activation functions and an offline phase using multiplication triples generation for fully connected layers. It takes about 80 hours for training a 2 hidden layer neural network on MNIST. While promising, this approach achieves low accuracy levels of 93.4% as opposed to the state-of-the-art accuracy of 99.8% on MNIST [174] for training using plaintext. ABY^3 [175] extends SecureML to a 3PC setting. It has been reported to be about 80× faster than SecureML. The approach also claims to be potentially extendable for CNNs. Finally, SecureNN[176] presents a 3PC and 4PC secure neural network training framework. The approach uses Beaver multiplication triplets [177] for secure matrix multiplication over shared secrets and proposes customized protocols for computing activation functions. SecureNN claims to be 93× faster than SecureML and is able to support CNNs, achieving accuracy levels close to 99% on MNIST.

2.3.3 Equitable Machine Learning

There have been increasing concerns about the fairness of machine learning models owing to their tendency to encode human biases [10, 11]. This could result in unfair discrimination against a sub-population [178–180]. Avoiding sensitive attributes such as age, race or gender could potentially mitigate issues associated with direct discrimination. However, it has long been established that non-sensitive features could potentially be used to reconstruct sensitive features [181], giving rise to indirect discrimination or disparate impact. A number of measures of fairness have been proposed in the literature [10, 182–185]. Although one might ideally wish to satisfy all possible measures for fairness, this is likely to be impractical in real life applications [186, 187]. Therefore it is still important that we seek to ensure satisfiability of one or more measures of fairness for a data-driven model. A number of studies [188–190] have proposed mechanism to ensure such measures. However, most of the approaches so far expose sensitive attributes to a data controller or a third party which in itself poses a privacy risk [191]. Furthermore, in Section 5.7, we show how manipulating even a single

parameter, among thousands of parameters, in a neural network model can overhaul the fairness of the model. This directly affects the equitability of the model. In this thesis, we propose methods to address such concerns in a privacy-preserving manner.

2.4 Summary

In this chapter, we have discussed background research related to the contributions in this thesis. We started with an introduction to deep learning, going on to a review of primitives such as federated learning. We discuss potential attacks on deep learning systems and the rich potential of deep learning for sensitive data domains. This in turn motivates the research question *"How can we design, build and evaluate effective, efficient and equitable privacy-preserving Deep Learning systems?"* and the constituent sub-questions presented in Chapter 1. More broadly this motivates the use-case for combining deep learning with privacy-preserving computation technologies. Next, we review the most relevant privacy preserving computation technologies, for this thesis — homomorphic encryption, secure multi-party computation and differential privacy. Afterwards, we discussed literature at the intersection of privacy preserving computation and deep learning for secure training and inference. Finally, we motivate the issue of equity in machine learning and review related research. In essence, this chapter serves as a bridge between the introduction in Chapter 1, the research questions and the Chapters 3 - 6. In the next chapter, we describe our approach to two party secure training, addressing the very first sub-question *"How can we design crypto-friendly protocols and architectures for secure deep learning?"*.

3

Two-Party Secure Neural Network Training and Prediction

Contents

3.1 Introduction

The field of secure computation, and in particular Multi-Party Computation (MPC) techniques such as garbled circuits and lower level primitives like Oblivious Transfer (OT), have undergone impressive developments in the last decade. This has been due to a sequence of engineering and theoretical breakthroughs, among which OT Extension [128] is of special relevance.

However, classical generic secure computation protocols do not scale to real-world Machine Learning (ML) applications. To overcome this, recent work has combined different secure computation techniques to design custom protocols for specific ML tasks. These include optimization of linear/logistic regressors and neural networks [18, 19, 192, 193], matrix factorization [194], constrained optimization [195], and k-nearest neighbor classification [196, 197]. For example, Nikolaenko et al. [192] propose a protocol for secure distributed ridge regression that combines additive homomorphic encryption and garbled circuits, while previous works [18, 193] rely in part on OT for the same functionality. However, these approaches are based largely on isolated optimization of the ML model or the cryptographic protocol. In this chapter, we show that there is a benefit in taking a holistic approach to the problem. Specifically, our goal is to design an optimization algorithm alongside a secure computation protocol customized for it. In particular, in this chapter, we address the sub-question *"How can we design crypto-friendly protocols and architectures for secure deep learning?"* related to our main research question. We do this by designing QUOTIENT, an efficient two-party secure training and prediction framework.

Secure Distributed Deep Neural Network Training. Until recently there has been little work on training Deep Neural Networks (DNNs) on encrypted data. The significant work that we are aware of are ABY3 [19] and SecureML [18]. Different from our work, ABY3 [19] designs techniques for encrypted training of DNNs in the 3-party case, and a majority of honest parties. The work most similar to ours is

SecureML [18]. They propose techniques based on secret-sharing to implement a stochastic gradient descent procedure for training linear/logistic regressors and DNNs in two-party computation. While the presented techniques are practical and general, there are three notable downsides:

1. They require an "offline" phase, that while being data-independent, takes up most of the time (more than 80 hours for a 3-layer DNN on the MNIST dataset in the 2-Party Computation (2PC) setting).

2. Their techniques are not practical over WAN (more than 4277 hours for a 3-layer DNN on the MNIST dataset), restricting their protocols to the LAN setting.

3. The accuracy of the obtained models are lower than state-of-the-art deep learning methods.

More recently, secure training using homomorphic encryption has been proposed [21]. While the approach limits the communication overhead, it's estimated to require more than a year to train a 3-layer network on the MNIST dataset, making it practically unrealizable. Furthermore, the approaches outlined above omit techniques necessary for modern DNN training such as normalization & adaptive gradient methods, instead relying on vanilla SGD with constant step size. In this chapter, we argue that significant changes are needed in the way ML models are trained in order for them to be suitable for practical evaluation in MPC. Crucially, our results show that securely-trained DNNs do not need to take a big accuracy hit if secure protocols and ML models are customized *jointly*.

3.2 Overview and Problem Description

Here, we introduce concepts in DNN training and inference, fixed-point encodings, and MPC applied to ML.

Training Deep Neural Networks. The goal of any machine learning classifier is to map an input $\mathbf{a}^0$ to a desired output y (often called a *label*). For training DNNs to learn this mapping, we would like to adjust the weights $\{\mathbf{W}^l\}_{l=1}^L$ so that the output $\mathbf{a}^L$ after L layers is $\mathbf{a}^L = y$. To do so, the most popular method for training DNN weights is via Stochastic Gradient Descent (SGD). Specifically, given a training dataset of input-output pairs $\{(\mathbf{a}_i^0, y_i)\}_{i=1}^n$, and a loss function $\ell(\mathbf{a}^L, y)$ that measures the difference between prediction $\mathbf{a}^L$ and output y (e.g., the squared loss: $(\mathbf{a}^L - y)^2$), SGD consists of the following sequence of steps:

1. The *randomization step*: sample a random single input $\mathbf{a}_i^0$.
2. The *forward pass*: pass $\mathbf{a}_i^0$ through the network to produce prediction $\mathbf{a}_i^L$.
3. The *backward pass*: compute the gradients $\mathbf{G}^l$ and $\mathbf{e}^l$ of the loss $\ell(\mathbf{a}_i^L, y_i)$ with respect to each weight $\mathbf{W}^l$ and layer activation $\mathbf{a}^l$ in the network, respectively: $\mathbf{G}^l = \frac{\partial \ell(\mathbf{a}_i^L, y_i)}{\partial \mathbf{W}^l}, \mathbf{e}^l = \frac{\partial \ell(\mathbf{a}_i^L, y_i)}{\partial \mathbf{a}^l}, \forall l \in [L]$.
4. The *update step*: update each weight by this gradient: $\mathbf{W}_l = \mathbf{W}_l - \eta \mathbf{G}^l$, where η is a constant called the *learning rate*.

These four steps, called an *iteration* are repeated for each input-output pair in the training set. A pass over the whole training set is called an *epoch*. The first step is often generalized to sample a set of inputs, called a *batch*, as this exploits the parallel nature of modern GPU hardware and has benefits in terms of convergence and generalization.

State-Of-The-Art Training: Normalization & Adaptive Step-Sizes.

While the above "vanilla" gradient descent can produce reasonably accurate classifiers, alone they do not produce state-of-the-art results. Two critical changes are necessary: (1) normalization and (2) adaptive step-sizes. We describe each of these in detail.

Normalization. The first normalization technique introduced for modern DNNs was *batch normalization* [198]. It works by normalizing the activations $\mathbf{a}^l$ of a given layer l so

that across a given batch of inputs $\mathbf{a}^l$ has roughly zero mean and unit standard deviation. There is now a general consensus in the machine learning community that normalization is a key ingredient to accurate DNNs [199]. Since batch normalization, there have been a number of other successful normalization schemes including *weight normalization* [200] and *layer normalization* [201]. The intuition behind why normalization helps is because it prevents the activations $\mathbf{a}^l$ from growing too large to destabilize the optimization, especially for DNNs with many layers that have many nested multiplications. This means that one can increase the size of the learning rate η (see the *update step* above for how the learning rate is used in SGD), which speeds up optimization [199]. Normalization has been shown to yield speedups of an order of magnitude over non-normalized networks. Further, without normalization, not only is convergence slower, in some cases one cannot even reach lower minima [201].

Adaptive Step-Sizes. While normalization allows one to use larger learning rates η, it is still unclear how to choose the correct learning rate for efficient training: too small and the network takes an impractical amount of time to converge, too large and the training diverges. To address this there has been a very significant research effort into designing optimization procedures that adaptively adjust the learning rate during training [202–204]. So-called *adaptive gradient methods* scale the learning rate by the magnitude of gradients found in previous iterations. This effectively creates a per-dimension step-size, adjusting it for every entry of the gradient $\mathbf{G}^l$ for all layers l. Without adaptive gradient methods, finding the right step-size for efficient convergence is extremely difficult. A prominent state-of-the-art adaptive gradient method is AMSgrad [204]. It was developed to address pitfalls with prior adaptive gradient methods. The authors demonstrate that it is able to converge on particularly difficult optimization problems that prior adaptive methods cannot. Even on datasets where prior adaptive methods converge, AMSgrad can cut the time to converge in half.

Algorithm 1 describes the steps for training deep neural networks using the standard AMSgrad optimizer. This can be summarised as:

Algorithm 1: Standard AMSgrad Optim. (Δ_{ams})

Input: Weights $\{\mathbf{W}^l\}_{l=1}^L$,
Data $\mathcal{D} = \{(\mathbf{a}_i^0, y_i\}_{i=1}^n$, learning rate η
Output: Updated weights $\{\mathbf{W}^l\}_{l=1}^L$

1: Initialize: $\{\mathbf{M}^l, \mathbf{V}^l\}_{l=1}^L = 0$
2: **for** $t = 1, \ldots, T$ **do**
3: $(\mathbf{a}_{(p_a)}^0, y_{(p_a)}) \sim \mathcal{D}$
4: $\{\mathbf{a}^l\}_{l=1}^L = \texttt{Forward}(\{\mathbf{W}^l\}_{l=1}^L, \mathbf{a}^0)$
5: $\{\mathbf{G}^l\}_{l=1}^L = \texttt{Backward}(\{\mathbf{W}^l, \mathbf{a}^l\}_{l=1}^L, y)$
6: $\{\mathbf{M}^l = 0.9\mathbf{M}^l + 0.1\mathbf{G}^l\}_{l=1}^L$ ▷ `weighted mean`
7: $\{\mathbf{V}^l = 0.99\mathbf{V}^l + 0.01\Big(\mathbf{G}^l\Big)^2\}_{l=1}^L$ ▷ `weighted variance`
8: $\{\hat{\mathbf{V}}^l = \max(\hat{\mathbf{V}}^l, \mathbf{V}^l)\}_{l=1}^L$
9: $\{\mathbf{G}^l = \frac{\mathbf{M}^l}{\sqrt{\hat{\mathbf{V}}^l + \epsilon}}\}_{l=1}^L$ ▷ `history-based scaling with square root`
10: $\{\mathbf{W}^l = \mathbf{W}^l - \eta\mathbf{G}^l\}_{l=1}^L$

1. Sampling the input pair $(\mathbf{a}^0, y)$ from the dataset $\mathcal{D}$.

2. Using the input $(\mathbf{a}^0)$ to obtain the prediction $\{\mathbf{a}^l\}_{l=1}^L$.

3. Computing the loss between the prediction $\{\mathbf{a}^l\}_{l=1}^L$ and the target output (y), and computing the gradient $\mathbf{G}^l$ for the loss $\ell(\mathbf{a}_i^L, y_i)$ with respect to each of the weights $\mathbf{W}_l$ in the network.

4. Updating the weights using a weighted average of the past gradients, specifically, their first and second moments $\mathbf{M}^l$ and $\mathbf{V}^l$.

Training and Inference with Fixed-Point Numbers.

Motivated by the need to deploy DNNs on embedded and mobile devices with limited memory and power, significant research effort has been devoted to model quantization and compression. Often the goal is to rely solely on fixed-point encoding of real numbers. In fact, Tensorflow offers a lightweight variant to address this goal [205, 206]. These developments are useful for secure prediction. This is because cryptographic techniques scale with the circuit representation of the function being evaluated, and so

a floating-point encoding and subsequent operations on that encoding are extremely costly. However, for the task of training, there are few works that perform all operations in fixed-point [207–211]. We start by reviewing fixed-point encodings, and the MPC techniques that we will consider. Then, in Section 3.3, we describe how this method can be modified and improved for secure training.

Notation for Fixed-Point Encodings. We represent fixed-point numbers as a triple $x = (a, \ell, p)$, where $a \in \{-2^{\ell-1} - 1, \ldots, 2^{\ell-1}\}$ is the fixed-point integer, ℓ is its *range* or *bit-width*, and p is its *precision*. The rational number encoded by x is $a/2^p$. For simplicity, we will often make ℓ implicit and write the rational $a/2^p$ as $a_{(p)}$. As our computations will be over ℓ-bit values in 2's complement, overflows/underflows can happen, resulting in large errors. This requires that our training procedure is very stable, within a very controlled range of potential values.

3.2.1 MPC for Machine Learning

In this section, we introduce MPC techniques, highlighting the trade-offs that inspire our design of secure training and prediction protocols. The goal of MPC protocols is to compute a public function $f(\cdot)$ on private data held by different parties. The computation is done in a way that reveals the final output of the computation to intended parties, and nothing else about the private inputs. MPC protocols work over a finite discrete domain, and thus the function $f(\cdot)$ must be defined accordingly. Generally, MPC protocols can be classified depending on (i) a type of structure used to represent $f(\cdot)$ (generally either Boolean or integer-arithmetic circuits) and (ii) a scheme to secretly share values between parties, namely Boolean-sharing, additive-sharing, Shamir-secret-sharing, and more complex variants (for more information see [145]). The choices for (i) and (ii) define the computational properties of an MPC protocol.

Additive-sharing protocols are very efficient for computations over large integral domains that do not involve comparisons. This includes sequences of basic linear algebra operations, such as matrix-vector multiplications. Specifically, additions are

extremely cheap as they can be performed locally, while multiplications are more expensive. On the other hand, computations involving comparisons require computing a costly bit-decomposition of the values.

In contrast to additive-sharing, protocols based on Boolean-sharing are well-suited for computations easily represented as Boolean circuits, such as division/multiplication by powers of two (via bit shifting), comparisons, and `sign`(). They are slower at addition and multiplication which require adder and multiplier circuits.

These trade-offs lead to a natural idea recently exploited in several works in the secure computation for ML: one should design protocols that are customized to full algorithms, such as the training of linear/logistic regressors and DNNs [18, 192, 193], matrix factorization [194], or k-nearest neighbor classification [196, 197]. Moreover, custom protocols alternate between different secret-sharing schemes as required by the specific computation being implemented. Of course, the transformations between secret-sharing schemes must themselves be implemented by secure protocols[1]. This point is especially relevant to DNN training, as it amounts to a sequence of linear operations (which are naturally represented as arithmetic circuits) interleaved with evaluations of non-linear activation functions such as the ReLU (which are naturally represented as Boolean circuits).

Some MPC frameworks work in the so-called *pre-processing model* (see [18]), where computation is split into a data-independent offline phase and a data-dependent online phase. Random values useful for multiplication can be generated offline, and then used for fast secure multiplication online. In the current case, we consider total time, removing the assumption of an offline phase. This is not a fundamental limitation, as we have variants of our protocols that work in the pre-processing model as explained later.

[1]This aspect was thoroughly investigated in [145].

Key Ingredients

Notation for Secret-Sharing. In this work, we focus on the two-party computation setting, which excludes solutions that rely on a majority of honest parties [19, 212]. Inspired by work on function-specific protocols, we employ both Boolean sharing and additive sharing. We start by fixing two parties $\mathsf{P_1}$ and $\mathsf{P_2}$. We denote the *Boolean-share* of $x \in \{0,1\}$ held by $\mathsf{P_1}$ as $\langle x \rangle_1$, and $\langle x \rangle_2$ for $\mathsf{P_2}$. In Boolean-sharing, the shares satisfy $\langle x \rangle_1 = x \oplus \langle x \rangle_2$, where $\langle x \rangle_2$ is a random bit, and $\oplus$ signifies the `XOR` operation. We denote the *additive-share* of integer $y \in \mathbb{Z}_q$ held by $\mathsf{P_1}$ as $[\![y]\!]_1$, and $[\![y]\!]_2$ for $\mathsf{P_2}$. Here $[\![y]\!]_1 = y - [\![y]\!]_2$, with random $[\![y]\!]_2 \in \mathbb{Z}_q$. In practice, q is 2^σ, for $\sigma \in \{8, 16, 32, 64, 128\}$, as these are word lengths offered in common architectures.

In many our protocols, we use Yao's Garbled Circuits [131] and Oblivious Transfer (OT) [123] (described in Chapter 2) as a subprotocol. OT is a basic primitive in MPC. In fact, any function can be evaluated securely using only an OT protocol [213] and, moreover OT is a crucial component of Yao's Garbled Circuits.

In our protocols, we employ a more efficient primitive—Correlated Oblivious Transfer (COT). COT was introduced in [129] alongside an efficient COT Extension protocol that uses roughly half the communication than the general OT Extension. COT is a particular case of OT where the Sender does not get to choose its messages, but instead chooses a function f relating the two messages m_0 and m_1, as $m_0 = f(m_1)$. This functionality is enough for important applications of OT in MPC such as Garbled Circuits and OT-based triplet generation (see [145]). Below we define the flavour of COT that we need in our application, following the notation from [129].

Definition 3.2.1 ($m \times \mathrm{COT}_\ell$)**.** *Let* $\mathbf{f} = (f_i)_{i \in [m]}$*, be a sequence of* correlation functions*, each with signature* $f_i : \{0,1\}^\ell \mapsto \{0,1\}^\ell$ *held by party* $\mathsf{P_1}$ *(the Sender), and let* $\mathbf{w} \in \{0,1\}^\ell$ *be a sequence of* choice bits *held by party* $\mathsf{P_2}$ *(the Chooser). After an execution of* $m \times COT_\ell(\mathbf{f}, \mathbf{w})$*,* $\mathsf{P_1}$ *obtains* m random *vectors* $(\mathbf{x}_i \in \{0,1\}^\ell)_{i=1}^m$*, and* $\mathsf{P_2}$ *obtains* m *vectors* $(\mathbf{y}_i \in \{0,1\}^\ell)_{i=1}^m$ *such that* $\forall i \in [m] : y_i = (\neg \mathbf{w}_i) \cdot \mathbf{x}_i + \mathbf{w}_i \cdot f_i(\mathbf{x}_i)$.

Figure 3.1: (Left) Training in the two-server model of MPC: (i) Each user i shares their labeled data $(\mathbf{a}_i^0, y_i)$ across two servers P_1 and P_2, by giving one share $[\![(\mathbf{a}_i^0, y_i)]\!]_j$ to each P_j. (ii) Each P_j compiles their share of a training dataset $\mathcal{D}$, by simply accumulating all shares received from users. Finally, (iii) P_1 and P_2 engage in a multi-party computation, in which $\mathcal{D}$ is securely reconstructed, and subsequently used to train a model $\mathbf{W}$, from which each server gets a share. (Right) Private prediction using MPC: A client C and a server S engage in an multi-party computation protocol for the client to obtain y (the prediction of the server's model $\mathbf{W}$ for their client's data $\mathbf{a}^0$) without the parties disclosing anything about $\mathbf{W}$ and $\mathbf{a}^0$ to each other.

3.2.2 Threat Model

In the two-party model of MPC, the training procedure is outsourced to two servers. This framework has been used in several previous works [18, 192–195, 214]. It works by first secret-sharing the training dataset $\mathcal{D}$ across the servers. This is depicted in Figure 3.1 (Left): users secret-share their values $(\mathbf{a}_i^0, y_i)$ across two non-colluding servers, which run the training procedure and obtain the resulting secret-shared model: $[\![\mathbf{W}]\!]_1, [\![\mathbf{W}]\!]_2$. Note that the scenario where two organizations collaborate to build a model of their respective private data in a privacy-preserving way is a particular case of this setting: this corresponds to the stage after the users have shared their data. We present fast MPC protocols to implement $\textsc{TrainModel}(\mathcal{D})$ from Figure 3.1 (Left). Thus, all our protocols are presented as two-party protocols between parties P_1 and P_2 via input and output additive-shares. Our protocols are secure in the semi-honest model for secure computation, as in [18].

Alongside private training, an important related problem is private prediction. This is depicted in Figure 3.1 (Right). Specifically, a client C has private data $\mathbf{a}^0$ for which they wish to have a private prediction y, via the private weights $\mathbf{W}$ of server S. In our setting, a fast protocol for two-party training in MPC immediately

yields a fast protocol for private prediction, as prediction corresponds to the forward pass in training (i.e., Algorithm 2).

3.3 Deep Learning for MPC

In this section, we describe new methods to optimize DNNs that were developed alongside our protocols (in Section 3.4). Our first insight is that recent work on training deep networks in fixed-point [210] can be leveraged for crypto-friendly training. Namely, while this work was originally intended for embedded devices, it contains useful primitives such as repeated quantization to low fixed-point precisions to stabilize optimization. However, out-of-the-box, this work is unsuited for privacy-preserving protocols. We make the following modifications:

(a) We ternarize the network weights: $\mathbf{W} \in \{-1, 0, 1\}$ during the forward and backward passes. This will allow matrix multiplication with $\mathbf{W}$ to be phrased as repeated 1-out-of-2 oblivious transfers (described in Section 3.4).

(b) We construct an MPC-friendly quantization function for the weight gradients in the backward pass, replacing a biased coin flip and truncation steps with a saturation-free quantization, without loss in accuracy.

(c) We replace the backward pass normalization operation, division by the closest-power-of-two, with the *next*-power-of-two.

While a seemingly small change, the latter operation has a very efficient circuit implementation [215] that we leverage in our secure protocols in the next section. Further, we observe empirically in Section 3.5 that this change also has no effect on accuracy. Ultimately, these changes will only speed up the computation of a training iteration in a secure protocol. If training is based on stochastic gradient descent, many iterations will still be necessary for convergence. To address this, we design a new fixed-point adaptive gradient algorithm. It is inspired by a state-of-the-art

floating-point adaptive gradient method, AMSgrad [204]. This optimization allows us to achieve the best accuracy to date on all 5 datasets we consider for DNNs trained in fixed-point (in Figure 3.7). In this section, we describe the work of [210], our changes (a-c) mentioned above, and our new fixed-point adaptive gradient training procedure.

3.3.1 Fixed-Point Encoding for Deep Models

The work by Wu et al. [210] describes an efficient optimization procedure for DNNs operating entirely on fixed-point encodings of real numbers. We describe it in detail here.

Quantize gradients, and quantize frequently. The first idea of WAGE [210] is to introduce functions Q_w, Q_a, Q_g, Q_e that quantize the *weights* $\mathbf{W}$, *activations* $\mathbf{a}$, *weight gradients* $\mathbf{G}$, *activation gradients* $\mathbf{e}$ to a small, finite set of fixed-point numbers. While previous work [216–223] had already introduced the idea of functions Q_w, Q_a to quantize weights and/or activations in the forward pass, they required the weights $\mathbf{W}$ or gradients $\mathbf{G}, \mathbf{e}$ to be represented in floating-point in the backward pass, in order to optimize accurately. In Wu et al. [210], all the quantization functions take fixed-point numbers with some precision p, i.e., $v_{(p)} = v/2^p$ and find the nearest number with another precision q, i.e., $v_{(q)}$:

$$v_{(q)} = N(v_{(p)}, q) = \frac{\lfloor \frac{v}{2^p} 2^q \rceil}{2^q} \tag{3.1}$$

where $\frac{a}{2^p} 2^q$ is in practice either a division or a multiplication by $2^{|p-q|}$ depending on whether $p > q$. Additionally, Wu et al. [210] introduce a saturation function $S(\cdot)$, to yield $Q(\cdot)$ as:

$$v_{(q)} = Q(v_{(p)}, q) = S(N(v_{(p)}, q), q). \tag{3.2}$$

where $S(x, q) = \min(\max(x, -1 + 2^{-q}), 1 - 2^{-q})$ saturates any x to be within $[-1 + 2^{-q}, 1 - 2^{-q}]$.

Weight quantization Q_w. The first function $\mathbf{W}_{(p_w)} = Q_w(\mathbf{W}_{(p)}, p_w)$ takes as input a p-precision fixed-point weight $\mathbf{W}_{(p)}$ and returns the closest p_w-precision weights $\mathbf{W}_{(p_w)}$ using the above function:

$$\mathbf{W}_{(p_w)} = Q_w(\mathbf{W}_{(p)}, p_w) = Q(\mathbf{W}_{(p)}, p_w) \tag{3.3}$$

Activation quantization Q_a. The second function $\mathbf{a}_{(p_a)} = Q_a(\mathbf{a}_{(p)}, p_a)$ is almost identical except it introduces an additional scaling factor $2^{-\alpha}$ as follows:

$$\mathbf{a}_{(p_a)} = Q_a(\mathbf{a}_{(p)}, p_a) = Q(\frac{\mathbf{a}_{(p)}}{2^{\alpha}}, p_a). \tag{3.4}$$

Where α is a fixed integer determined by the dimensionality of $\mathbf{W}$, see [210] eq. (7). The intuition is that this is a simple form of normalization (used in the forward pass), and the authors describe a technique to set α based on the layer size.

Activation gradient quantization Q_e. For the backward pass the first gradient we must consider is that of the activations, which we call $\mathbf{e}$. The strategy to quantize $\mathbf{e}_{(p_e)} = Q_e(\mathbf{e}_{(p)}, p_e)$, is similar to quantizing the activations. Except here the scaling factor depends on the largest value of $\mathbf{e}$:

$$\mathbf{e}_{(p_e)} = Q_e(\mathbf{e}_{(p)}, p_e) = Q(\frac{\mathbf{e}_{(p)}}{2^{\texttt{cpow}(\max\{|\mathbf{e}_{(p)}|\})}}, p_e) \tag{3.5}$$

where $\texttt{cpow}(x) = 2^{\lfloor \log_2(x) \rceil}$ is the closest power of 2 to x. The intuition here is again that normalization helps, but it cannot be constant as the gradient magnitude can potentially fluctuate. Thus normalization needs to be data-specific to stabilize training.

Weight gradient quantization Q_g. The second gradient is the weight gradient $\mathbf{G}$, quantized by $\mathbf{G}_{(p_g)} = Q_g(\mathbf{G}_{(p)}, p_g)$, as follows:

$$\mathbf{G}_{(p_g)} = Q_g(\mathbf{G}^n_{(p)}, p_g) = \frac{\texttt{sign}(\mathbf{G}^n_{(p)})}{2^{p_g - 1}} \left[\lfloor |\mathbf{G}^n_{(p)}| \rfloor + \mathrm{B}\Big(|\mathbf{G}^n_{(p)}| - \lfloor |\mathbf{G}^n_{(p)}| \rfloor \Big) \right]. \tag{3.6}$$

where $\mathbf{G}^n_{(p)} = \eta \mathbf{G}_{(p)} / 2^{\texttt{cpow}(\max\{|\mathbf{G}_{(p)}|\})}$ is a normalized version of $\mathbf{G}_{(p)}$ that is also multiplied by the learning rate η, $\texttt{sign}(a)$ returns the sign of a, and $B(p)$ draws a sample

Algorithm 2: Fixed-Point Forward Pass (`Forward`)

Input: Fixed-point weights $\{\mathbf{W}^l_{(p_w)}\}_{l=1}^L$,
Fixed-point data sample $\mathbf{a}^0_{(p_a)}$,
Layer-wise normalizers $\{\alpha_l\}_{l=1}^L$
Output: Fixed-point activations $\{\mathbf{a}^l_{(p_a)}\}_{l=1}^L$

1: **for** $l = 1, \ldots, L$ **do**
2: $\quad \mathbf{a}^l = f(\mathbf{W}^l_{(p_w)}\mathbf{a}^{l-1}_{(p_a)})$ ▷ pass through layer
3: $\quad \mathbf{a}^l_{(p_a)} = Q_a(\mathbf{a}^l, p_a)$ ▷ activations precision p_a

Algorithm 3: Fixed-Point Backward Pass (`Backward`)

Input: Fixed-point weights: $\{\mathbf{W}^l_{(p_w)}\}_{l=1}^L$,
Fixed-point activations (act) $\{\mathbf{a}^l_{(p_a)}\}_{l=1}^L$,
Fixed-point label $y_{(p_a)}$,
Activation function f,
Saturation function S
Output: Fixed-point gradients $\{\mathbf{G}^l\}_{l=1}^L$

1: $\mathbf{e}^L = \frac{\partial \ell(\mathbf{a}^L_{(p_a)}, y_{(p_a)})}{\partial \mathbf{a}^L_{(p_a)}}$ ▷ act. gradient
2: **for** $l = L, \ldots, 1$ **do**
3: $\quad \mathbf{e}^l_{(p_e)} = Q_e(\mathbf{e}^l, p_e)$ ▷ act. gradient precision p_e
4: $\quad \mathbf{e}^{l-1} = \mathbf{W}^l_{(p_w)}\left(\mathbf{e}^l_{(p_e)} \circ \frac{\partial \mathbf{a}^l_{(p_a)}}{\partial f} \circ \frac{\partial \mathbf{a}^l_{(p_a)}}{\partial S}\right)$ ▷ act. gradient
5: $\quad \mathbf{G}^l = {\mathbf{a}^{l-1}_{(p_a)}}^\top \mathbf{e}^l_{(p_e)}$ ▷ weight gradient

from a Bernoulli distribution with parameter p. This function normalizes the gradient, applies a randomized rounding, and scales it down. The idea behind this function is that additional randomness (combined with the randomness of selecting data points stochastically during training) should improve generalization to similar, unseen data.

3.3.2 Fixed-Point Encoding for *Cryto-Friendly* Deep Models

We propose three changes to the quantization functions that do not affect accuracy, but make them more suitable for MPC.

Weight quantization Q_w. We fix $p_w = 1$ and set $Q_w(\mathbf{W}_{(p)}, p_w) = 2Q(\mathbf{W}_{(p)}, p_w)$ so that our weights are ternary: $\mathbf{W} \in \{-1, 0, 1\}$ as mentioned above.

Algorithm 4: Fixed-Point SGD Optimization (Δ_{sgd})

Input: Fixed-point weights $\{\mathbf{W}^l_{(\overline{p}_w)}\}_{l=1}^L$,
Fixed-point data $\mathcal{D} = \{(\mathbf{a}^0_{(p_a),i}, y_{(p_a),i}\}_{i=1}^n$,
Learning rate η
Output: Updated weights $\{\mathbf{W}^l_{(\overline{p}_w)}\}_{l=1}^L$

1: **for** $t = 1, \ldots, T$ **do**
2: $(\mathbf{a}^0_{(p_a)}, y_{(p_a)}) \sim \mathcal{D}$
3: $\{\mathbf{W}^l_{(p_w)} = Q_w(\mathbf{W}^l_{(\overline{p}_w)}, p_w)\}_{l=1}^L$
4: $\{\mathbf{a}^l_{(p_a)}\}_{l=1}^L = \texttt{Forward}(\{\mathbf{W}^l_{(p_w)}\}_{l=1}^L, \mathbf{a}^0_{(p_a)})$
5: $\{\mathbf{G}^l\}_{l=1}^L = \texttt{Backward}(\{\mathbf{W}^l_{(p_w)}, \mathbf{a}^l_{(p_a)}\}_{l=1}^L, y_{(p_a)})$
6: $\{\mathbf{G}^l_{(p_g)} = Q_g(\mathbf{G}^l, p_g)\}_{l=1}^L$ ▷ `weight gradient precision` p_g
7: $\{\mathbf{W}^l_{(\overline{p}_w)} = S(\mathbf{W}^l_{(\overline{p}_w)} - \eta \mathbf{G}^l_{(p_g)}, \overline{p}_w)\}_{l=1}^L$

Activation gradient quantization Q_e. We make the following change to eq. (3.5) to $Q_e(\mathbf{e}_{(p)}, p_e) = Q(\mathbf{e}_{(p)}/2^{\texttt{npow}(\max\{|\mathbf{e}_{(p)}|\})}, p_e)$, where $\texttt{npow}(x) = 2^{\lceil \log_2(x) \rceil}$ is the *next* power of 2 after x. This is faster than the closest power of 2, $\texttt{cpow}(x)$ as the latter also needs to compute the previous power of 2 and compare them to x to find the closest power.

Weight gradient quantization Q_g. We introduce a different quantization function than [210] which is significantly easier to implement using secure computation techniques:

$$\mathbf{G}_{(p_g)} = Q_g(\mathbf{G}_{(p)}, p_g) = N(\frac{\mathbf{G}_{(p)}}{2^{\texttt{npow}(\max\{|\mathbf{G}_{(p)}|\})}}, p_g). \tag{3.7}$$

We find that the original quantization function in eq. (3.6) needlessly removes information and adds unnecessary overhead to a secure implementation.

Forward/Backward passes. Algorithm 2 corresponds to prediction in DNNs, and Algorithms 2, 3, 4 describe the training procedure. Apart from the quantization functions note that in the backward pass (Algorithm 3) we take the gradient of the activation with respect to the activation function: $\partial \mathbf{a}^l_{(p_a)}/\partial f$ and the saturation function: $\partial \mathbf{a}^l_{(p_a)}/\partial S$. Algorithm 4 describes a stochastic gradient descent (SGD) algorithm for learning fixed-point weights inspired by [210]. We keep around two copies of weights

in different precisions p_w and $\overline{p}_w$. The weights $\mathbf{W}_{(p_w)}$ are ternary and will enable fast secure forward (Algorithm 2) and backward passes. The other weights $\mathbf{W}_{(\overline{p}_w)}$ are at a higher precision and are updated with gradient information. We get the ternary weights $\mathbf{W}_{(p_w)}$ by quantizing the weights $\mathbf{W}_{(\overline{p}_w)}$ in line 3. Note that the forward and backward passes of convolutional networks can be written using the exact same steps as Algorithms 2 and 3 where weights are reshaped to take into account weight-sharing. Similarly, for residual networks the only differences are: (a) line 2 in the forward pass changes and (b) line 4 in the backward pass has an added term from prior layers.

3.3.3 A Fixed-Point Adaptive Gradient Algorithm

One of the state-of-the-art adaptive gradient algorithms is AMSgrad [204]. However, AMSgrad includes a number of operations that are possibly unstable in fixed-point domains: (i) the square-root of a sum of squares, (ii) division and (iii) moving average. We redesign AMSgrad in Algorithm 5 to get around these difficulties in the following ways. First, we replace the square-root of a sum of squares with absolute value in line 9, a close upper-bound for values $\mathbf{v} \in [-1, 1]$. We verify empirically that this approximation does not degrade performance.

Second, we replace division of $\hat{\mathbf{V}}_{(p_v)}$ in line 11 with division by the next power of two. Third, we continuously quantize the weighted moving sums of lines 7 and 8 to maintain similar precisions throughout. These changes make it possible to implement AMSgrad in secure computation efficiently. In the next section we describe the protocols we design to run Algorithms 2-5 privately.

3.4 Oblivious Transfer for Secure Learning

In this section, we present custom MPC protocols to implement private versions of the algorithms introduced in the previous section. Our protocols rely heavily on OT, and crucially exploit characteristics of our networks such as ternary weights and fixed-point (8-bit precision) gradients. We present our MPC protocol for neural network training

Algorithm 5: Fixed-Point AMSgrad Optim. (Δ_{ams})

Input: Fixed-point weights $\{\mathbf{W}^l_{(\overline{p}_w)}\}_{l=1}^L$,
Fixed-point data $\mathcal{D} = \{(\mathbf{a}^0_{(p_a),i}, y_{(p_a),i}\}_{i=1}^n$, learning rate η
Output: Updated weights $\{\mathbf{W}^l_{(\overline{p}_w)}\}_{l=1}^L$

Initialize: $\{\mathbf{M}^l, \mathbf{V}^l\}_{l=1}^L = 0$
for $t = 1, \ldots, T$ **do**
  $(\mathbf{a}^0_{(p_a)}, y_{(p_a)}) \sim \mathcal{D}$
  $\{\mathbf{W}^l_{(p_w)} = 2Q(\mathbf{W}^l_{(\overline{p}_w)}, p_w)\}_{l=1}^L$
  $\{\mathbf{a}^l_{(p_a)}\}_{l=1}^L = \texttt{Forward}(\{\mathbf{W}^l_{(p_w)}\}_{l=1}^L, \mathbf{a}^0_{(p_a)})$
  $\{\mathbf{G}^l\}_{l=1}^L = \texttt{Backward}(\{\mathbf{W}^l_{(p_w)}, \mathbf{a}^l_{(p_a)}\}_{l=1}^L, y_{(p_a)})$
  $\{\mathbf{G}^l = \frac{\mathbf{G}^l}{2^{\texttt{npow}(\max\{|\mathbf{G}^l|\})}}\}_{l=1}^L$ ▷ scale gradients
  $\{\mathbf{M}^l = N(0.9\mathbf{M}^l, p_m) + 0.1\mathbf{G}^l\}_{l=1}^L$ ▷ weighted mean
  $\{\mathbf{V}^l = N(0.99\mathbf{V}^l, p_v) + 0.01\left|\mathbf{G}^l\right|\}_{l=1}^L$
  $\{\hat{\mathbf{V}}^l = \max(\hat{\mathbf{V}}^l, \mathbf{V}^l)\}_{l=1}^L$
  $\{\mathbf{G}^l = \frac{\mathbf{M}^l}{2^{\texttt{npow}(|\hat{\mathbf{V}}^l|+\epsilon)}}\}_{l=1}^L$ ▷ history-based scaling
  $\{\mathbf{G}^l_{(p_g)} = N(\mathbf{G}^l, p_g)\}_{l=1}^L$ ▷ weight gradient precision p_g
  $\{\mathbf{W}^l_{(\overline{p}_w)} = S(\mathbf{W}^l_{(\overline{p}_w)} - \eta\mathbf{G}^l_{(p_g)}, \overline{p}_w)\}_{l=1}^L$

by describing its components separately. First, in Section 3.4.1 we describe our protocol for ternary matrix-vector multiplication, a crucial primitive for training used both in the forward and backward pass. Next, in Sections 3.4.2 and 3.4.3 we describe the protocols for the forward and backward passes, respectively. These protocols use our matrix-vector multiplication protocol (and a slight variant of it), in combination with efficient garbled circuit implementations for normalization and computation of ReLU.

3.4.1 Secure Ternary Matrix-Vector Multiplication

A recurrent primitive, both in the forward and backward passes of the fixed-point neural networks from Section 3.3 is the product $\mathbf{Wa}$, for ternary matrix $\mathbf{W} \in \{-1, 0, 1\}^{n \times m}$ and fixed-point integer vector $\mathbf{a} \in \mathbb{Z}_q^m$. Several previous works on MPC-based ML have looked specifically at matrix-vector multiplication [18, 196]. As described in Section 3.2.1, multiplication is a costly operation in MPC. Our insight is that if $\mathbf{W}$ is

ternary, we can replace multiplications by selections, enabling much faster protocols. More concretely, we can compute the product $\mathbf{z} = \mathbf{Wa}$ as shown in Algorithm 6.

Algorithm 6: Ternary-Integer Matrix-Vector Product

Input: Matrix $\mathbf{W} \in \{-1, 0, 1\}^{n \times m}$ and vector $\mathbf{a} \in \mathbb{Z}_q^m$

1: $\mathbf{z} = (0)_{i \in [n]}$
2: **for** $i \in [n], j \in [m]$ **do**
3: **if** $\mathbf{W_{i,j}} > 0$ **then** z_i += $\mathbf{a}_j$
4: **if** $\mathbf{W_{i,j}} < 0$ **then** z_i −= $\mathbf{a}_j$
5: **return** $\mathbf{z}$

A natural choice for implementing the functionality in Algorithm 6 securely are MPC protocols that represent the computation as a Boolean circuit. In our two-party setting natural choices are garbled circuits and the GMW protocol [213]. In Boolean circuits, the If-Then-Else construction corresponds to a multiplexer, and a comparison with 0 is essentially free: it is given by the sign bit in a two's complement binary encoding. However, a circuit implementation of the computation above will require $|\mathbf{W}|$ additions and the same number of subtractions, which need to implemented with full-adders. Whereas, additions, if computed on additive shares, do not require interaction and thus are extremely fast.

Our proposed protocol achieves the best of both worlds by combining Boolean sharing and additive sharing. To this end, we represent the ternary matrix $\mathbf{W}$ by two Boolean matrices $\mathbf{W}^+ \in \{0, 1\}^{n \times m}$ and $\mathbf{W}^- \in \{0, 1\}^{n \times m}$, defined as $\mathbf{W}^+_{i,j} = 1 \Leftrightarrow \mathbf{W}_{i,j} = 1$ and $\mathbf{W}^-_{i,j} = 1 \Leftrightarrow \mathbf{W}_{i,j} = -1$. Now, the product $\mathbf{Wa}$ can be rewritten as $\mathbf{W}^+\mathbf{a} - \mathbf{W}^-\mathbf{a}$. This reduces our problem from $\mathbf{Wa}$ with ternary $\mathbf{W}$ to two computations of $\mathbf{Wa}$ with binary $\mathbf{W}$. Note that we can use the same decomposition to split any matrix with inputs in a domain of size k into k Boolean matrices, and thus our protocol is not restricted to the ternary case.

Accordingly, at the core of our protocol is a two-party subprotocol for computing additive shares of $\mathbf{Wa}$, when $\mathbf{W}$ is Boolean-shared among the parties and $\mathbf{a}$ is additively shared. In turn, this protocol relies on a two-party subprotocol for computing additive

shares of the inner product of a Boolean-shared binary vector and an additively shared integer vector. As a first approach to this problem, we show in Protocol 7 a solution based on oblivious transfer. We state and prove the correctness of Protocol 7 in the next lemma.

Lemma 3.4.1. *Let* $\mathbf{b}$ *and* $\mathbf{a}$ *be a Boolean and integer vector, respectively, shared among parties* $\mathsf{P}_1, \mathsf{P}_2$*. Given a secure OT protocol, the two-party Protocol 7 is secure against semi-honest adversaries, and computes an additive share of the inner product* $\mathbf{b}^\top \mathbf{a}$ *among* $\mathsf{P}_1, \mathsf{P}_2$*.*

Proof. For the correctness, note that, for all $j \in [m]$, $b_j a_j = b_j [\![a_j]\!]_1 + b_j [\![a_j]\!]_2$, and that each of the OTs in steps 4 and 5 of the algorithm are used to compute an additive share of each of the terms of the sum. Concretely, in the OT in line 4 is used to compute an additive share of $b_j [\![a_j]\!]_1$ as $z_{1,j} + z'_{1,j}$, and the value received by party P_2 is $m_{1,[\![b_j]\!]_2} = -z_{1,j}$ if $[\![b_j]\!]_1 \oplus [\![b_j]\!]_2 = 0$ and $m_{1,[\![b_j]\!]_2} = [\![a_j]\!]_i - z_{1,j}$ otherwise. Security follows easily from the fact that all messages in the OTs are masked with fresh randomness $z_{i,j}$ and thus constructing simulators from a simulator for OT is straightforward. □

One can think of Protocol 7 as a component-wise multiplication protocol. The only modification required is in step 6, i.e. the local aggregation of additive shares of the result of component-wise multiplication. We could directly use this protocol to implement our desired matrix-vector multiplication functionality, and this leads to very significant improvements due to the concrete efficiency of OT Extension implementations. However, we can further optimize this to obtain our final protocol.

The use of OT in Protocol 7 has similarities with the GMW protocol, and is inspired by the OT-based method due to Gilboa for computing multiplication triplets, and discussed in [145]. A similar idea was used in the protocol for computing the sigmoid function by Mohassel and Zhang [18]. Moreover, concurrently to this work, Riazi et al. [224] have proposed OT-based protocols for secure prediction using DNNs. They propose a protocol called Oblivious Conditional Addition (OCA) that is analogous to

Protocol 7: Boolean-Integer Inner Product

Parties: $\mathsf{P_1}$ and $\mathsf{P_2}$

Input: Arithmetic shares of integer vector $\mathbf{a} \in \mathbb{Z}_q^m$ and Boolean shares of binary vector $\mathbf{w} \in \{0,1\}^m$

Output: Arithmetic shares of $z = \mathbf{w}^\top \mathbf{a}$

1: Each $\mathsf{P_i}$ generates random values $(z_{i,j})_{j\in[m]}$.
2: **for** $j \in [m]$ **do**
3: $\mathsf{P_i}$ sets
$$m_{i,0} := \langle \mathbf{w}_j \rangle_i \cdot [\![\mathbf{a}_j]\!]_i - z_{i,j}$$
$$m_{i,1} := \neg\langle \mathbf{w}_j \rangle_i \cdot [\![\mathbf{a}_j]\!]_i - z_{i,j}$$
4: $\mathsf{P_1}$ and $\mathsf{P_2}$ run $\mathrm{OT}(m_{1,0}, m_{1,1}, \langle \mathbf{w}_j \rangle_2)$, with $\mathsf{P_1}$ as Sender and $\mathsf{P_2}$ as Chooser, for $\mathsf{P_2}$ to obtain $z'_{1,j}$
5: $\mathsf{P_1}$ and $\mathsf{P_2}$ run $\mathrm{OT}(m_{2,0}, m_{2,1}, \langle \mathbf{w}_j \rangle_1)$, with $\mathsf{P_2}$ as Sender and $\mathsf{P_1}$ as Chooser, for $\mathsf{P_1}$ to obtain $z'_{2,j}$
6: Each $\mathsf{P_i}$ sets $[\![z]\!]_i = \sum_{j\in[m]}(z_{i,j} + z'_{i,j})$.

Protocol 7. While their work only addresses secure prediction, our improved protocol presented in the next section is relevant in their setting as well.

Optimizations: Correlated OT and Packing.

In the previous section, our protocol assumed a standard OT functionality, but actually, we can exploit even more efficient primitives. Our optimization of Protocol 7, presented in Protocol 8 exploits COT in a way to implement our required inner product functionality. The idea behind Protocol 8 is simple: note that in Protocol 7 parties choose their random shares $z_{i,j}$ of intermediate values in the computation, and they use them to mask OT messages. However, as we only require the $z_{i,j}$'s to be random, one could in principle let the OT choose them. Note also that the parties can choose their share of the result as a function of their inputs, which can be implemented in COT. This is done in lines 4 and 5 of Protocol 8. As the protocol consists of just two executions of $m \times \mathrm{COT}_\ell$ and local additions its security is trivial, while we state and prove the correctness of Protocol 8 in the next lemma. Finally, note that, as in the case of Protocol 7, it is easy to turn Protocol 8 into a protocol for component-wise multiplication.

Protocol 8: Boolean-Integer Inner Product via $m \times \mathrm{COT}_\ell$

Parties: P_1 and P_2

Input: ℓ-bit arithmetic shares of integer vector $\mathbf{a} \in \mathbb{Z}_q^m$ and Boolean shares of binary vector $\mathbf{w} \in \{0,1\}^m$

Output: Arithmetic shares of $z = \mathbf{w}^\top \mathbf{a}$

1: Each party $\mathsf{P_i}$ constructs a vector of correlation functions $\mathbf{f}^{\mathbf{i}} = (f_{i,j}(x))_{j\in[m]}$,

$$f_{i,j}(x) = x - [\![\mathbf{w}_j]\!]_i \cdot [\![\mathbf{a}_j]\!]_i + \neg[\![\mathbf{w}_j]\!]_i \cdot [\![\mathbf{a}_j]\!]_i$$

2: The parties run $m \times \mathrm{COT}_\ell(\mathbf{f}^{\mathbf{1}}, [\![\mathbf{w}_j]\!]_2)$ with P_1 acting as the Sender, and P_1 obtains $\mathbf{x}$ while P_2 obtains $\mathbf{y}$.

3: The parties run $m \times \mathrm{COT}_\ell(\mathbf{f}^{\mathbf{2}}, [\![\mathbf{w}_j]\!]_1)$ with P_2 acting as the Sender, and P_2 obtains $\mathbf{x}'$ while P_1 obtains $\mathbf{y}'$.

4: P_1 sets $[\![z]\!]_1 = \sum_{j\in[m]}([\![\mathbf{w}_j]\!]_1 \cdot [\![\mathbf{a}_j]\!]_1 - \mathbf{x}_j + \mathbf{y'}_j)$

5: P_2 sets $[\![z]\!]_2 = \sum_{j\in[m]}([\![\mathbf{w}_j]\!]_2 \cdot [\![\mathbf{a}_j]\!]_2 - \mathbf{x'}_j + \mathbf{y}_j)$

Lemma 3.4.2. *Let* $\mathbf{w}$ *and* $\mathbf{a}$ *be a Boolean and integer vector, respectively, shared among parties* $\mathsf{P}_1, \mathsf{P}_2$*. Given an* $m \times COT_\ell$ *protocol, the two-party Protocol 8 is secure against semi-honest adversaries, and computes an additive share of the inner product* $\mathbf{w}^\top \mathbf{a}$ *among* $\mathsf{P}_1, \mathsf{P}_2$*.*

Proof. Privacy follows directly from the correctness of the COT subprotocol. To see that the protocol computes the right value, we argue for a $j \in [m]$, and distinguish cases according to all possible values of the shares of w_j.

Case I ($w_j = 0$; $[\![w_j]\!]_1 = 1$, $[\![w_j]\!]_2 = 1$): In this case $f^1(x) = x - [\![a_j]\!]_1$ and $f^2(x') = x' - [\![a_j]\!]_2$. Upon execution of step 2, P_1 obtains x and P_2 obtains $y = x - [\![a_j]\!]_1$. Upon execution of step 3, P_2 obtains x' and P_1 obtains $y' = x' - [\![a_j]\!]_2$. P_1 accumulates $[\![z_j]\!]_1 = [\![a_j]\!]_1 - x + x' - [\![a_j]\!]_2$ into $[\![z]\!]_1$ and P_2 accumulates $[\![z_j]\!]_2 = [\![a_j]\!]_1 - x' + x - [\![a_j]\!]_1$ into $[\![z]\!]_2$. Then $[\![z_j]\!]_1 + [\![z_j]\!]_2 = 0$, which is $w_j a_j$ as $w_j = 0$.

Case II ($w_j = 0$; $[\![w_j]\!]_1 = 0$, $[\![w_j]\!]_2 = 0$): In this case $f^1(x) = x + [\![a_j]\!]_1$ and $f^2(x') = x' + [\![a_j]\!]_2$. Upon execution of step 2, P_1 obtains x and P_2 obtains $y = x$. Upon execution of step 3, P_2 obtains x' and P_1 obtains $y' = x'$ P_1 accumulates $[\![z_j]\!]_1 = -x + x'$ into $[\![z]\!]_1$ and P_2 accumulates $[\![z_j]\!]_2 = -x' + x$ into $[\![z]\!]_2$. Then $[\![z_j]\!]_1 + [\![z_j]\!]_2 = 0$, which is $w_j a_j$ as $w_j = 0$.

Protocol 9: Ternary-Integer Matrix-Vector Product

Parties: P_1 and P_2

Input: Arithmetic shares of integer vector $\mathbf{a} \in \mathbb{Z}_q^m$ and Boolean shares of binary matrices $\mathbf{W}^+, \mathbf{W}^- \in \{0,1\}^{n,m}$

Output: Arithmetic shares of $\mathbf{z} = \mathbf{W}^+\mathbf{a} - \mathbf{W}^-\mathbf{a}$

1: P_1 and P_2 compute $[\![\mathbf{W}^+\mathbf{a}]\!]$ using n executions of Protocol 8.
2: P_1 and P_2 compute $[\![\mathbf{W}^-\mathbf{a}]\!]$ using n executions Protocol 8.
3: P_i sets $[\![\mathbf{z}]\!]_i := [\![\mathbf{W}^+\mathbf{a}]\!]_i - [\![\mathbf{W}^+\mathbf{a}]\!]_i$.

Case III ($w_j = 1$; $[\![w_j]\!]_1 = 0$, $[\![w_j]\!]_2 = 1$): In this case $f^1(x) = x + [\![a_j]\!]_1$ and $f^2(x') = x' - [\![a_j]\!]_2$. Upon execution of step 2, P_1 obtains x and P_2 obtains $y = x + [\![a_j]\!]_1$. Upon execution of step 3, P_2 obtains x', P_1 obtains $y' = x'$. P_1 accumulates $[\![z_j]\!]_1 = -x + x'$ into $[\![z]\!]_1$ and P_2 accumulates $[\![z_j]\!]_2 = [\![a_j]\!]_2 - x' + x + [\![a_j]\!]_1$ into $[\![z]\!]_2$. Then $[\![z_j]\!]_1 + [\![z_j]\!]_2 = a_j$, which is $w_j a_j$ as $w_j = 1$.

Case IV ($w_j = 1$; $[\![w_j]\!]_1 = 1$, $[\![w_j]\!]_2 = 0$): In this case $f^1(x) = x - [\![a_j]\!]_1$ and $f^2(x') = x' + [\![a_j]\!]_2$. Upon execution of step 2, P_1 obtains x and P_2 obtains $y = x$. Upon execution of step 3, P_2 obtains x' and P_1 obtains $y' = x' + [\![a_j]\!]_2$. P_1 accumulates $[\![z_j]\!]_1 = [\![a_j]\!]_1 - x + x' + [\![a_j]\!]_2$ into $[\![z]\!]_1$, P_2 accumulates $[\![z_j]\!]_2 = -x' + x$ into $[\![z]\!]_2$. Then, $[\![z_j]\!]_1 + [\![z_j]\!]_2 = a_j$, which is $w_j a_j$ as $w_j = 1$. □

Protocol 9 achieves our goal of computing an arithmetic share of $\mathbf{Wa} = \mathbf{W}^+\mathbf{a} - \mathbf{W}^-\mathbf{a}$, for an $n \times m$ matrix. This is easily achieved using $2n$ calls to Protocol 8. This translates into $4n$ executions of $m \times \mathrm{COT}_\ell$, plus *local* extremely efficient additions. Note that this protocol is fully paralellizable, as all the COT executions can be run in parallel.

Overall, our approach exploits the fact that $\mathbf{W}$ is ternary without having to perform any Boolean additions in secure computations. Our experiments in Section 3.5 show concrete gains over the prior state-of-the-art.

Communication costs. Using the $m \times \mathrm{COT}_\ell$ Extension protocol from [129], the parties running Protocol 8 send $m(\tau + \ell)$ bits to each other to compute inner products of length m, where τ is the security parameter (128 in our implementation). This

results in $nm(\tau + \ell)$ bits being sent/received by each party for the whole matrix-vector multiplication protocol. In contrast, the OT-based approaches to matrix-vector multiplication entirely based on arithmetic sharing from [18] would require at least $nm\ell(\tau + \ell)$ (assuming optimizations like packing and vectorization).

Packing for matrix-vector multiplication. While the forward pass of quotient operates over 8-bit vectors (and thus $q = 8$ in Protocol 9), the value of ℓ in implementations of $m \times \text{COT}_\ell$ is 128, i.e., the AES block size. However, we can exploit this to pack $128/8 = 16$ vector multiplications against the same matrix for the same communication and computation. This is very useful in batched gradient descent, as this results in 16x additional savings in communication and computation. This packing optimization was also used in [18] for implementing an OT-based offline phase to matrix-vector multiplications occurring in batched gradient descent.

3.4.2 Secure Forward Pass

Our protocol for the forward pass (Algorithm 2) is a sequential composition of Protocol 9, and a garbled circuit protocol with three components: (i) Evaluation of ReLU, (ii) normalization by a public data-independent value α_l (line 3 in Algorithm 2), and (iii) quantization function $Q(\cdot)$ in eq. (3.2) in Section 3.3.1. The protocol is depicted in Figure 3.2. Note that we only show a forward pass for a single layer, but the protocol trivially composes sequentially with itself, as input $\mathbf{a}^{\ell-1}$ and output $\mathbf{a}^{\ell}$ are both secret-shared additively. Security follows directly from the security of the subprotocols, as their outputs and inputs are always secret-shares. For scalability, we describe an efficient circuit implementation of (i)-(iii). In our proposed circuit P_1 inputs its share $[\![\mathbf{z}]\!]_1$, and a random value chosen in advance that will become its share of the output, denoted $[\![\mathbf{a}^{\ell}]\!]_1$. In the garbled circuit, $\mathbf{z}$ is first reconstructed (this requires $|\mathbf{z}|$ parallel additions). For component (i), ReLU, we exploit the fact that the entries in $\mathbf{z}$ are encoded in binary in the circuit using two's complement as $z_i = (b_{i,k} \cdots b_{i,1})$, where $b_{i,k} = 1 \Leftrightarrow z_i < 0$. Hence $\text{ReLU}(z_i) = (\neg b_{i,k}) \cdot b_{i,k}$. Note that this is very efficient,

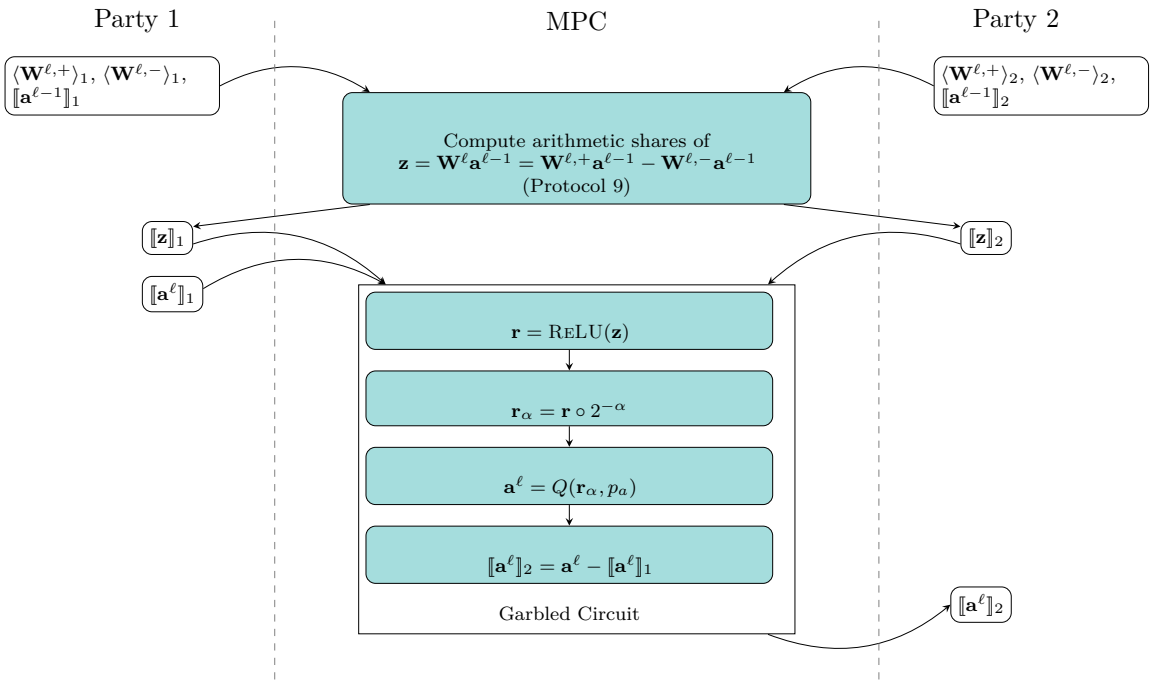

Figure 3.2: Our MPC protocol for private prediction (forward pass). We compose three protocols to evaluate one layer of the form $f(\mathbf{Wa})$, with ternary $\mathbf{W}$, and where $f = \text{ReLU}$.

as it only requires to evaluate $k|\mathbf{z}|$ NOT and AND gates. The next two steps, (ii) normalization and (iii) quantization, are extremely cheap, as they can be implemented with logical and arithmetic shifts without requiring any secure gate evaluation. Finally, to construct P_2's output we need to perform a subtraction inside the circuit to compute $[\![\mathbf{a}^{\ell}]\!]_2 = \mathbf{a}^{\ell} - [\![\mathbf{a}^{\ell}]\!]_1$. Altogether this means that our forward pass requires execution of Protocol 9 and a garbled circuit protocol to evaluate a vector addition, a vector subtraction, and a linear number of additional gates. Moreover, note that this garbled circuit evaluation can be parallelized across components of the vector $\mathbf{z}$.

3.4.3 Secure Backward Pass

Figure 3.3 shows our protocols for the backward pass (Algorithm 3). Analogous to the forward pass, we depict the pass for one layer, and observe that all its inputs and outputs are secret-shared. Hence, it can be easily composed sequentially across layers,

and with the forward pass, and its security follows trivially from the security of each of the subprotocols.

The protocol works via a sequence of subprotocols. Each of them produces a result shared among the parties $\mathsf{P}_1, \mathsf{P}_2$ either as a Boolean-share or an additive share. As in the forward pass, this protocol leverages Protocol 9, as well as Protocol 8 for component-wise multiplication. Recall that the goal of the backward pass is to recompute ternary weight matrices $\{\mathbf{W}^\ell\}_{\ell=1}^L$ by means of a gradient-based procedure. As $\{\mathbf{W}^\ell\}_{\ell=1}^L$ is represented in our MPC protocol by pairs of binary matrices, the protocol to be run for each layer ℓ takes as inputs the Boolean-shares of such matrices, i.e., $\mathbf{W}^{\ell,-}$ and $\mathbf{W}^{\ell,+}$, from each party. Moreover, the parties contribute to the protocol arithmetic shares of the input to each layer a^ℓ computed in the forward pass, as well as the target values $\mathbf{y}$.

The first step of the backward pass is a data-dependent normalization of the activation gradient $\mathbf{e}^\ell$, followed by the quantization step that we described in the forward pass, as shown in Figure 3.3. We now describe the design of the Boolean circuit used to compute these steps.

Normalization by the infinite norm. Our goal is to design a (small) circuit that, given $\mathbf{e}$, computes $\mathbf{e}/2^{\mathtt{npow}(\max\{|\mathbf{e}|\})}$. A naive circuit would compute the absolute value of every entry $|\mathbf{e}|$, compute the maximum value $\max\{|\mathbf{e}|\}$, compute $2^{\lceil \log_2(\max\{|\mathbf{e}|\})\rceil}$, and finally compute a division. However, computing exponentiation and logarithm in a garbled circuit would be prohibitively expensive. Such circuits are large, and we have to do this computation in each layer ℓ, as many times as the number of total iterations. To overcome this we apply two crucial optimizations: (i) approximate $\max\{|\mathbf{e}|\}$ by bitwise OR of all values $|\mathbf{e}_i|$, and (ii) compute `npow` with an efficient folklore procedure for obtaining the number of leading zeros in a binary string. This requires only b OR gates and $\log(b)$ arithmetic right shifts and additions, where b is the bitwidth of the entries of $\mathbf{e}$ (8 in our applications) [215]. Also note that the division can be computed as an arithmetic right shift. An important remark is that the denominator is a private value in the circuit, which means that, although we can use right shift for division,

our circuit needs to first compute all possible right shifts and then select the result according to the value of $\texttt{npow}(\max\{|\mathbf{e}|\})$. This involves a subcircuit linear in b, and since in our case $b=8$ we once again benefit from having small bitwidth, by trading a small overhead for costly divisions, logarithms, and exponentiations.

Derivatives of ReLU and saturation. Computing derivatives of ReLU and saturation $S(\cdot)$ can be done efficiently in a Boolean representation, as they lie in $\{0,1\}$. Specifically, computation only involves extracting the sign bit for ReLU, and ANDing a few bits for saturation. Ultimately we need to compute $\mathbf{e} \circ \mathbf{d}$, for $\mathbf{d} = \text{ReLU}'(\mathbf{a}) \circ S'(\mathbf{a})$ (line 4 in Algorithm 3). Note that this is a Boolean combination of integers, so we can alternate between Boolean and arithmetic shares and compute it with Protocol 8 (for component-wise product). We can further optimize the procedure by computing $\langle \mathbf{d} \rangle_i$ in the forward pass, since ReLU is already used there. This is commonly done in ML implementations in the clear.

The remainder of the backward pass involves (i) computing $\mathbf{e}^{l-1}$ (the rest of line 4 in Algorithm 3), for which we use Protocol 9, and (ii) an outer product between 8-bit vectors $\mathbf{a}^{l-1^\top}\mathbf{e}$ (line 5). For (ii) we use a vectorized version of the OT-based multiplication protocol presented originally in [225], and used in [18, 145].

Overall, this makes our backward pass very efficient, involving three small garbled circuits (two can be parallelized), and relies heavily on oblivious transfer computations.

SGD. To implement Algorithm 3 we need to additionally keep higher precision 8-bit matrices $\{\mathbf{W}^\ell_{(\overline{p}_w)}\}_{\ell=1}^L$ as arithmetic shares. We ternarize these to obtain our weights $\mathbf{W}$, which we implement in the natural way with a small garbled circuit involving 2 comparisons. The same operations for quantization and normalization of $\mathbf{e}$ can be used for $\mathbf{G}$.

AMSgrad. Almost all of the operations in AMSgrad (Algorithm 5): quantization, normalization, saturation, and absolute value have been described as part of the

previous protocols. The only addition is element-wise maximum (line 10), which we do via a comparison of Boolean shares.

Convolutional & residual layers. Although we have only described a fully connected layer, extending this protocol to convolutional layers is straightforward. The forward and backward passes of convolutional layers can be written using the same steps as Algorithms 2 and 3 with weight-reshaping. And max-pooling operations are simply comparisons, efficiently implemented in Boolean shares. Similarly, for residual networks, we only introduce integer additions in the forward pass (which we can perform on additive shares) and another computation of $\mathbf{d}$ for the backward pass.

3.5 Experiments

In this section, we present our experimental results for secure DNN training and prediction.

Experimental Settings. The experiments were executed over two Microsoft Azure Ds32 v3 machines equipped with 128GB RAM and Intel Xeon E5-2673 v4 2.3GHz processor, running Ubuntu 16.04. In LAN experiments, machines were hosted in the same region (West Europe) with an average latency of 0.3ms and a bandwidth of 1.82GB/s. For WAN, the machines were hosted in two different regions (North Europe & East US), with an average latency of 42ms and a bandwidth of 24.3 MB/sec. The machine specifications were chosen to be comparable with the ones in [18], hence enabling direct running time comparisons.

Implementation. We use two distinct code bases. We use the EMP-toolkit [130] to implement our secure protocols for forward and backward passes, as described in Section 3.4. EMP is written in C++ and offers efficient implementations of OT and COT extension [129]. We extended the semi-honest COT implementation to the functionality required for Protocol 8, as it is currently limited to correlation functions of the form $f(x) = x \oplus \Delta$ (the ones required by Yao's garbled circuits protocol). This code base was

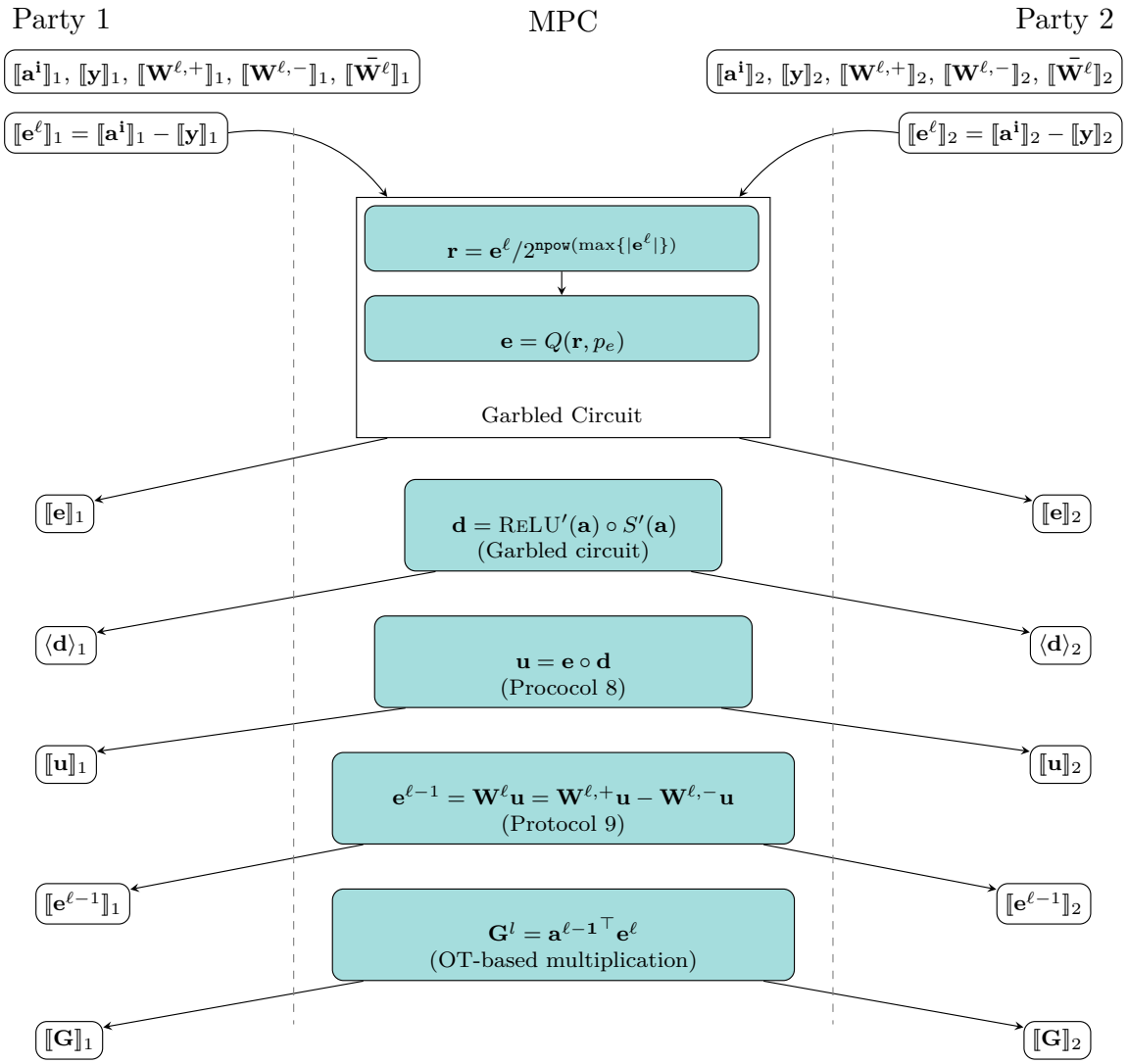

Figure 3.3: Our protocol for the backward pass, corresponding to Algorithm 3 from Section 3.3.

used for timing results. We developed a more versatile insecure Python implementation based on Tensorflow[226] for accuracy experiments. While this implementation does not use MPC, it mirrors the functionality implemented using the EMP-toolkit.

Evaluations. For training over QUOTIENT, we use two weight variables as described in Section 3.3: (i) ternary (2-bit) weights for the forward and backward passes, and (ii) 8-bit weights for the SGD and AMSgrad algorithms. We use 8-bits for the quantized

Network	k	QUOTIENT (s)	GC (s)	SecureML (OT) (s)	SecureML (LHE) (s)
LAN	10^3	0.08	0.025	0.028	5.3
	10^4	0.08	0.14	0.16	53
	10^5	0.13	1.41	1.4	512
	10^6	0.60	13.12	14*	5000*
	10^7	6.0	139.80	140*	50000*
WAN	10^3	1.7	1.9	1.4	6.2
	10^4	1.7	3.7	12.5	62
	10^5	2.6	20	140	641
	10^6	7.3	148	1400*	6400*
	10^7	44	1527	14000*	64000*

Table 3.1: Comparison of our COT-based component-wise multiplication of k-dimensional vectors with ternary fixed-point multiplication using garbled circuits (GC) and SecureML [18] (OT, LHE). One of the vectors hold only ternary values.

weight gradients (g), activations (a) and activation gradients (e).

As our protocols are online, to compare with other approaches employing an offline phase, we take a conservative approach: we compare their offline computation time with our total computation time using similar computational resources. We adopt this strategy because online phases for these approaches are relatively inexpensive and could potentially involve a set-up overhead. Additionally, our model could easily be divided into offline/online phases, but we omit this for simplicity.

We employ a naive parallelization strategy: running independent processes over a mini-batch on different cores. This speeds up the computation on average by 8-15× over LAN and by about 10-100x over WAN depending on the number of parallelizable processes. We leave more involved parallelization strategies to future work.

3.5.1 Data-independent benchmarking

In this section, we present the running times of the basic building blocks that will be used for DNN prediction and training.

Component-wise Multiplication. Table 3.1 compares the running times of our COT-based approach from Protocol 8 for computing component-wise product with (i)

Network	n	QUOTIENT (s)	SecureML (OT Vec) (s)	SecureML (LHE Vec) (s)
LAN	100	0.08	0.05	1.6
	500	0.1	0.28	5.5
	1000	0.14	0.46	10
WAN	100	1.7	3.7	2
	500	2	19	6.2
	1000	2.7	34	11

Table 3.2: Performance comparison of our matrix-vector multiplication approach with the vectorized approaches of SecureML [18] (OT, LHE). Here we multiply a $128 \times n$ ternary matrix with an n-dimensional vector.

an implementation of Algorithm 6 in a garbled circuit and (ii) two protocols proposed in SecureML [18] for the offline phase. Their first protocol corresponds to Gilboa's method [227] for oblivious product evaluation (the OT-based variant implemented with a packing optimization). Their second is a sequence of Paillier encryptions, decryptions, and homomorphic additions (the LHE variant) [81]. For QUOTIENT and GC, one vector is ternary and the other holds 128-bit values, while for OT and LHE both vectors hold 32-bit values. Although our protocols for multiplication are independent and suitable for parallelization, here we benchmark without parallelization. QUOTIENT clearly outperforms all other approaches as soon as we move past the set-up overhead of the base OTs. Note that most DNN layers involve greater than 10^4 multiplications, which makes our approach more suitable for those applications.

Matrix-Vector Product. As discussed in Section 3.4, our component-wise multiplication directly translates into matrix-vector products with local additions as described in Protocol 9. However, the protocols from [18] benefit greatly from a vectorization optimization, and thus a comparison of the matrix-vector multiplication tasks is important. Table 3.2 compares the performance of implementation of Protocol 9, for a ternary $128 \times n$ matrix and an n-dimensional vector. Similar to Table 3.1, we populate the matrix with ternary values (2-bit values) and the vector with up to 128-bit values. Our approach is at least $5\times$ faster than the vectorized LHE protocol from [18] on LAN,

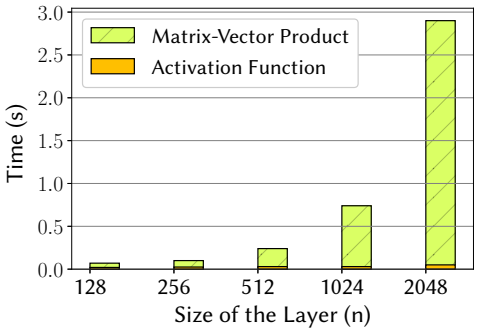

Figure 3.4: Forward pass time for single prediction over an $n \times n$ fully connected layer.

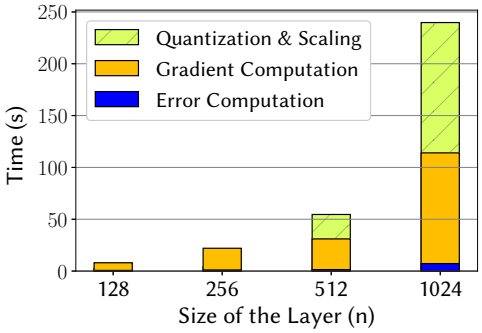

Figure 3.5: Forward and Backward pass time over an $n \times n$ fully-connected layer for 1 batch. Here batch size = 128.

and is roughly 10× faster than the OT protocol from [18] on WAN for $n \geq 500$. In general, the speedup increases as we increase the number of computations.

Layer Evaluation. Furthermore, we benchmark the basic building blocks of secure DNN training framework—forward and backward pass for a variety of different layer sizes. Figure 3.4 shows the running time of QUOTIENT for the forward pass as we increase the layer size. We split the total time into time spent on the matrix-vector product (Protocol 7) and computation of activation function (ReLU) using garbled circuits. Figure 3.5 shows the running time of the forward and backward pass, over a single batch of size 128. We report the running time of each of the three required functionalities: quantization & scaling, gradient computation, and error computation. As we increase the size of the layers, the garbled circuit for the quantization phase starts to dominate over the COT-based matrix-matrix product required for the gradient computation. This can primarily be attributed to the quantization and scaling of the gradient matrix. Our COT-based matrix-vector multiplication shifts the bottleneck from the multiplication to the garbled circuits based scaling phase. Finding efficient protocols for that task is an interesting task for future work. In particular, parallelized garbled circuits and optimization of the matrix-matrix multiplication in the gradient computation phase could be explored.

3.5.2 Experiments on Real-World Data

In this section, we evaluate our proposed QUOTIENT on real-world datasets. We show that: (i) QUOTIENT incurs only a small accuracy loss with respect to training over floating point, and (ii) it is more accurate than the state-of-the-art in fixed-point NN training, namely WAGE [210]. Both (i) and (ii) hold for several state-of-the-art architectures including fully connected, convolutional, and residual layers, and for different types of data. For our 2PC protocols, we show that (iii) 2PC-QUOTIENT outperforms the existing secure training approach for DNNs SecureML [18], both in terms of accuracy and running time, and both in the LAN and WAN settings. We first report the accuracy across a variety of datasets to show (i) and (ii) above. Then we report running times for 2PC-QUOTIENT training and prediction to argue (iii).

Datasets and Deep Neural Network Architectures

We evaluate QUOTIENT on six different datasets. Five of these are privacy sensitive in nature, as they include health datasets—Thyroid [228], Breast cancer [229], MotionSense [230], Skin cancer MNIST [231] and a financial credit dataset—German credit [232]. We also evaluate our approach on MNIST [164] for the purpose of benchmarking against prior work.

MNIST contains 60K training and 10K test grayscale (28×28) images of 10 different handwritten digits. We adopt the state-of-the-art floating point convolutional neural network LeNet [164] (32C5-BN-32C5-BN-MP2-64C5-BN-64C5-BN-MP2-512FC-BN-DO-10SM)[2] into a fixed-point equivalent of the form 32C5-MP2-64C5-MP2-512FC-10MSE for secure training & inference. In addition, we explore a variety of fully-connected neural networks 2×(128FC)-10MSE, 3×(128FC)-10MSE, 2×(512FC)-10MSE & 3×(512FC)-10MSE. We set the learning rate to 1 for both SGD and AMSgrad optimizers.

[2]BN, DO, and SM are batch-normalization, DropOut, and SoftMax respectively.

MotionSense contains smartphone accelerometer and gyroscope sensor data for four distinct activities namely walking, jogging, upstairs, and downstairs. For each subject, the dataset contains about 30 minutes of continuously recorded data. We use a rolling window, of size 2.56 seconds each for extracting around 50K samples for training and 11K for testing. As proposed in [230] we use floating point convolutional neural network 64C3-BN-MP2-DO-64C3-BN-MP2-DO-32C3-BN-MP2-DO-32C3-BN-MP2-DO-256FC-BN-DO-64FC-BN-DO-4SM and its fixed-point analogue of the form 64C3-MP2-64C3-MP2-32C3-MP2-32C3-MP2-256FC-64FC-4MSE. We furthermore explore a fully connected architecture of the form 3×(512FC)-4MSE.

Thyroid contains 3.7K training sample and 3.4K test samples of 21 dimensional patient data. The patients are grouped into three classes namely normal, hyperfunction and subnormal based on their thyroid functioning. We use a fully-connected neural network of the form 2×(100FC)-3SM for this dataset and its analogue fixed-point network 2×(100FC)-3MSE.

Breast cancer contains 5547 breast cancer histopathology RGB (50×50) images segregated into two classes—invasive ductal carcinoma and non-invasive ductal carcinoma. We use the 90:10 split for training and testing. We use a convolutional neural network with 3×(36C3-MP2)-576FC-2SM and a fully-connected network of the form 3×(512FC) along with their fixed point analogues.

Skin Cancer MNIST contains 8K training and 2K (28×28) dermatoscopic RGB images. They have been grouped into seven skin lesion categories. We use floating point network ResNet-20 [42]. ResNet-20 is made up of batch-normalisation, dropout, SoftMax layers and employs cross-entropy loss for training in addition to the residual layers. For the fixed-point version of ResNet-20 [42], we exclude batch normalisation, average pooling, and SoftMax layers. In addition, we use a fully connected architecture of the form 2×(512FC)-7MSE for secure training.

Dataset	QUOTIENT (%)	Floating point (%)
MNIST	99.38	99.48
MotionSense	93.48	95.65
Thyroid	97.03	98.30
Breast cancer	79.21	80.00
German credit	79.50	80.50

Table 3.3: Accuracy comparison of training over state-of-the-art floating point neural networks with their fixed point equivalents using QUOTIENT.

German credit contains 1k instances of bank account holders. They have been divided into two credit classes—Good or Bad. Each individual has 20 attributes (7 quantitative and 13 categorical). As a pre-processing step, we normalize the quantitative variables and encode the categorical variables using one-hot encoding. This amounts to 60 distinct feature for each individual in the dataset. We use the 80:20 split for training and testing. We use a fully-connected neural network of the form 2×(124FC)-2SM and its fixed point analogue for this dataset.

Accuracy

We evaluate the accuracy of QUOTIENT on different datasets and architectures. Also, we judge the impact of our MPC-friendly modifications on accuracy. To do so we consider four variants of QUOTIENT: (i) Our proposed QUOTIENT, with secure AMSgrad optimizer (`QUOTIENT[AMSgrad=Ours, Norm=Ours]`); (ii) QUOTIENT with the standard AMSgrad optimizer (`QUOTIENT[AMSgrad=Std, Norm=Ours]`) (Algorithm 1); (iii) QUOTIENT with our proposed AMSgrad with the closest power of 2 (C-Pow2) functionality instead of next power of 2 for quantization (`QUOTIENT[AMSgrad=Ours, Norm=WAGE]`); and (iv) QUOTIENT with the standard SGD optimizer (`QUOTIENT[SGD=Std, Norm=Ours]`).

Comparison with Floating Point Training. As a baseline evaluation of our training using `QUOTIENT[AMSgrad=Ours, Norm=Ours]`, we compare its performance (upon convergence) with the floating point training counterparts in Table 3.3. For

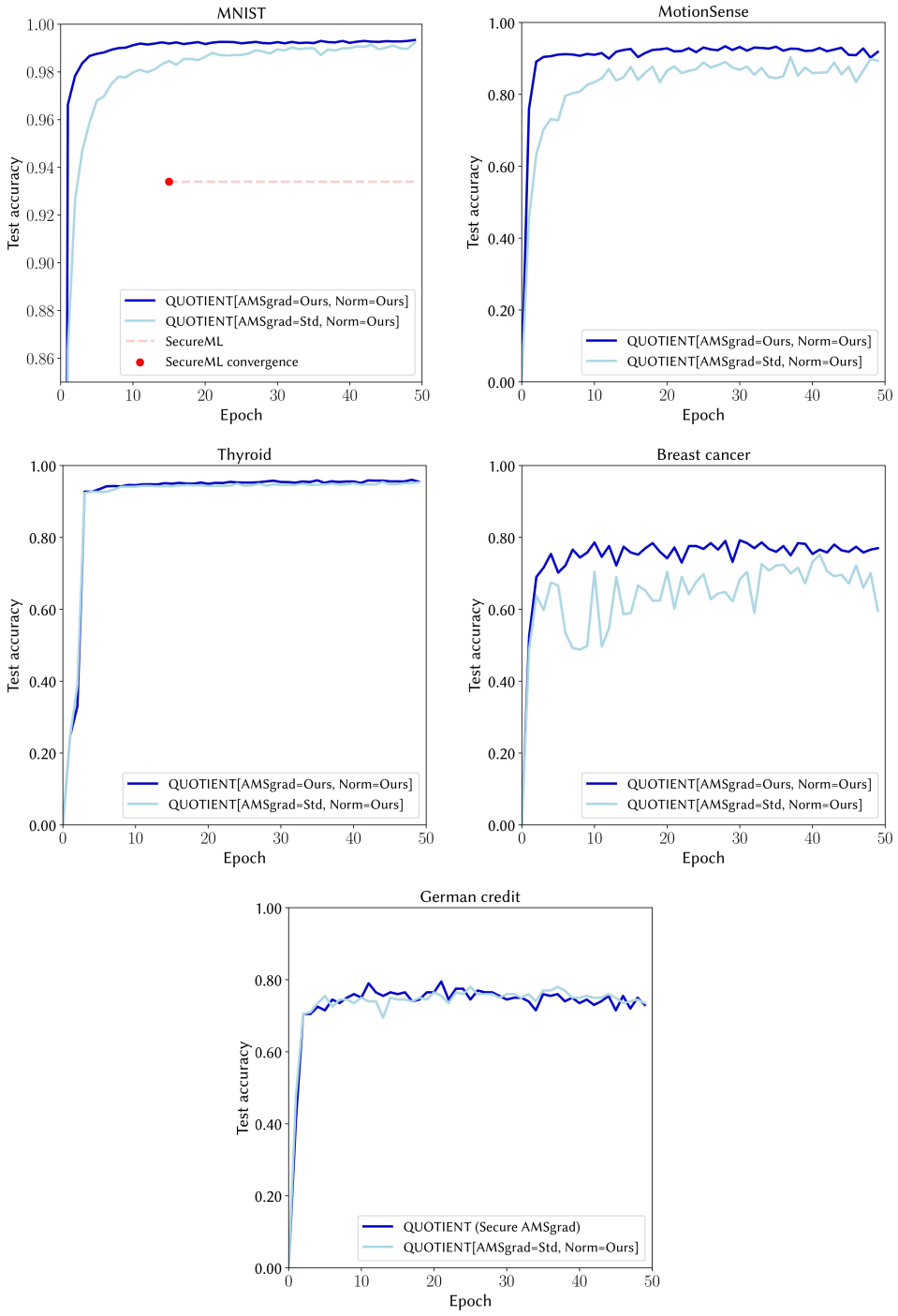

Figure 3.6: Performance comparison of secure AMSgrad and secure SGD for QUOTIENT. The plots compare training curves over MNIST (using CNN), MotionSense, Thyroid, Breast cancer and German credit datasets.

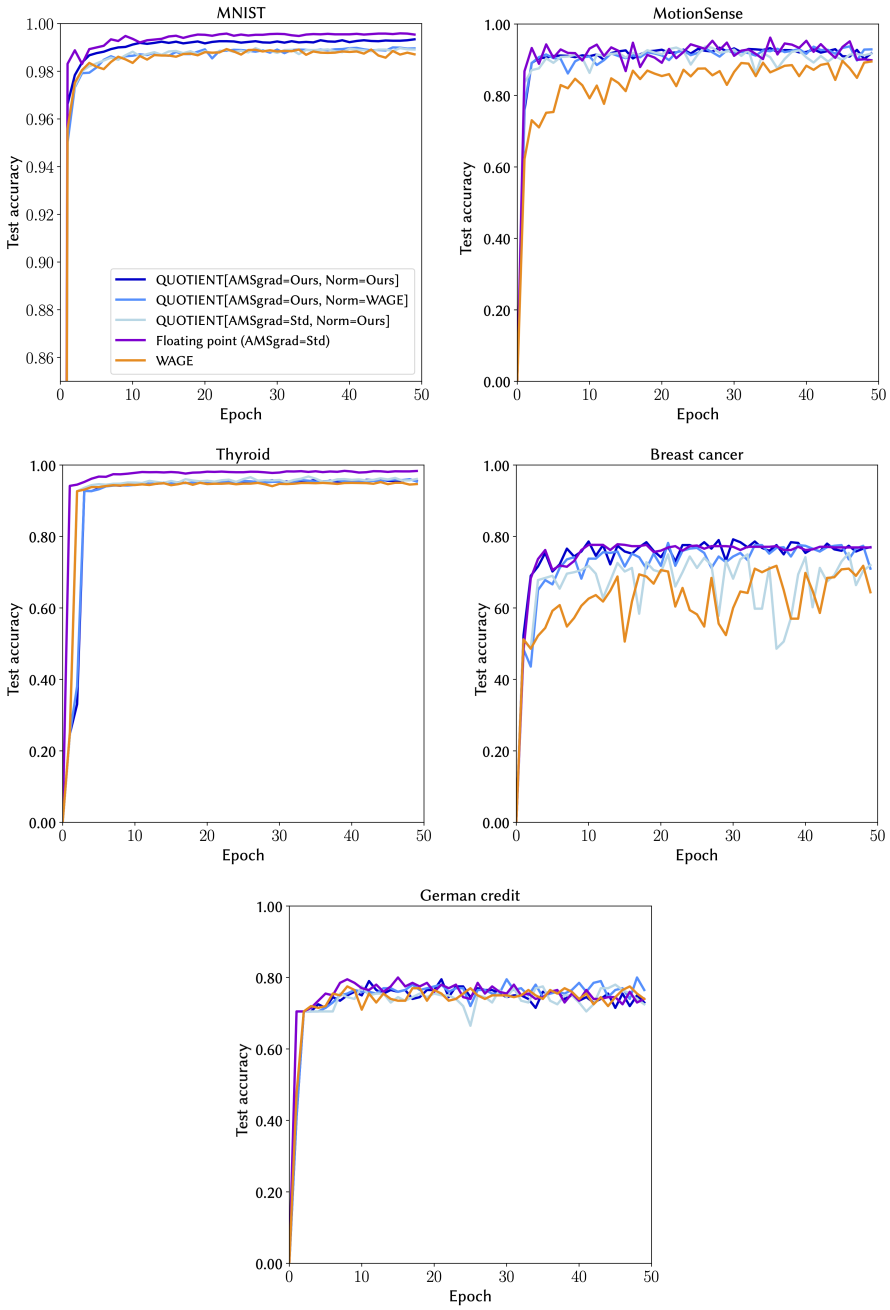

Figure 3.7: Performance comparison of three different variants of QUOTIENT training with floating point and WAGE [210] training on MNIST, MotionSense, Thyroid, Breast cancer and German credit datasets. `QUOTIENT[AMSgrad=Ours, Norm=WAGE]` and `QUOTIENT[AMSgrad=Std, Norm=Ours]`) differ from (`QUOTIENT[AMSgrad=Ours, Norm=Ours]` in using next power of 2 vs the closet Power of 2 (affecting lines 9 and 11 of secure AMSgrad in Algorithm 5, respectively.

MNIST and Breast cancer datasets `QUOTIENT[AMSgrad=Ours, Norm=Ours]` training achieves near state-of-the-art accuracy levels for MNIST and Breast Cancer, while we differ by at most ∼2% for German credit, MotionSense and Thyroid datasets.

Secure AMSgrad vs SGD. Figure 3.6 shows the training curves for QUOTIENT over all the datasets. In particular, we compare `QUOTIENT[AMSgrad=Ours, Norm=Ours]` and `QUOTIENT[SGD=Std, Norm=Ours]`. The secure AMSgrad optimizer converges faster than the secure SGD optimizer especially for convolutional neural networks (used for MNIST, MotionSense, and Breast cancer).

Effects of our Optimizations. In order to evaluate the impact of our optimizations, we compare training with floating point and WAGE [210] with the first three variations of QUOTIENT in Figure 3.7. For comparison with WAGE, we use the same fixed-point architecture as the ones used with QUOTIENT with an exception of the choice of optimizer— we employ our proposed secure AMSgrad, while WAGE uses a SGD optimizer in conjunction with randomized rounding. Moreover, the floating point analogues employ (a) batch normalization and dropout in each layer, (b) softmax for the last layer, and (c) cross-entropy loss instead of MSE. We observe that `QUOTIENT[AMSgrad=Ours, Norm=Ours]` outperforms all the other variants and is very close to the floating point training accuracy levels. We believe this is due to our modified AMSgrad, our normalization scheme, and use of the next power of two (used in `QUOTIENT[AMSgrad=Ours, Norm=Ours]`) which may act as an additional regularization compared with the closest power of two.

End-to-End Running Times

In the previous section, we demonstrated that large networks can match and improve upon the state-of-the-art in fixed-point deep networks. However, often one can use much simpler networks that are much faster and only sacrifice little accuracy. In order to balance accuracy and run-time, we design practical networks for each dataset and evaluate them here.

LAN																
	MNIST								MotionSense		Thyroid		Breast cancer		German credit	
	2 × (128FC)		3 × (128FC)		2 × (512FC)		3 × (512FC)		2 × (512FC)		2 × (100FC)		3 × (512FC)		2 × (124FC)	
Epoch	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc
1	8.72h	0.8706	10.05h	0.9023	27.13h	0.9341	38.76h	0.9424	10.07h	0.8048	0.08h	0.2480	14.51h	0.4940	0.03h	0.4100
5	43.60h	0.9438	50.25h	0.9536	135.65h	0.9715	193.80h	0.9745	50.35h	0.8847	0.40h	0.9341	72.55h	0.7360	0.15h	0.725
10	87.20h	0.9504	100.50h	0.9604	271.30h	0.9772	387.60h	0.9812	100.70h	0.8855	0.80h	0.9453	145.10h	0.7600	0.30h	0.7900
WAN																
	MNIST								MotionSense		Thyroid		Breast cancer		German credit	
	2 × (128FC)		3 × (128FC)		2 × (512FC)		3 × (512FC)		2 × (512FC)		2 × (100FC)		3 × (512FC)		2 × (124FC)	
Epoch	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc
1	50.74h	0.8706	57.90h	0.9023	139.71h	0.9341	190.10h	0.9424	44.43h	0.8048	0.52h	0.2480	74.10h	0.4940	0.15h	0.4100
5	253.7h	0.9438	289.5h	0.9536	698.55h	0.9715	950.5h	0.9745	222.15h	0.8847	2.6h	0.9341	370.5h	0.7360	0.75h	0.725
10	507.4h	0.9504	579h	0.9604	1397.1h	0.9772	1901h	0.9812	444.3h	0.8855	5.2h	0.9453	741h	0.7600	1.5h	0.7900

Table 3.4: Training time and accuracy values for various datasets and architectures after 1, 5 & 10 training epochs using QUOTIENT over LAN and WAN.

	MNIST					MotionSense		Thyroid	Breast cancer		German credit
	2 × (128FC)	3 × (128FC)	2 × (512FC)	3 × (512FC)	Conv	2 × (512FC)	Conv	2 × (100FC)	3 × (512FC)	Conv	2 × (124FC)
Single Prediction(s)	0.356	0.462	0.690	0.939	192	0.439	134	0.282	3.58	62	0.272
Batched Prediction (s)	2.24	2.88	4.79	6.50	2226	2.91	1455	1.83	44.02	447	1.77

Table 3.5: Prediction time for various datasets and architectures using QUOTIENT over LAN. Here batch size = 128.

2PC-QUOTIENT Training. Table 3.4 shows the running time of 2PC-QUOTIENT for practical networks across all datasets over LAN and WAN. We report accuracy and timings at 1,5 and 10 epochs. We note that the training time grows roughly linearly with the number of epochs. In all cases except the largest MNIST model, 10 epochs finish in under 12 days. Note that standard large deep models can easily take this long to train.

Our training protocols port nicely to the WAN settings. On average, our networks are only about 5× slower over WAN than over LAN. This is is due to the low communication load and round-trip of our protocols. As a result, we present the first 2PC pragmatic neural network training solution over WAN.

2PC-QUOTIENT Prediction. In addition to secure training, forward pass of QUOTIENT can be used for secure prediction. Table 3.5 presents prediction timings for all datasets over LAN. In addition to fully connected networks, we also report the prediction timings over convolutional neural networks. The timings have been reported for a single point as well as 128 parallel batched predictions (this corresponds to classifying many data points in one shot). Except for the multi-channel Breast

	MNIST				MotionSense	Thyroid	Breast cancer	German credit
	2 × (128FC)	3 × (128FC)	2 × (512FC)	3 × (512FC)	2 × (512FC)	2 × (100FC)	3 × (512FC)	2 × (124FC)
Single Prediction(s)	6.8	8.8	14.4	19.9	9.46	5.99	33.3	5.1
Batched Prediction (s)	8.3	10.9	22.6	29.9	12.08	6.89	69.1	7.3

Table 3.6: Prediction time for various datasets and architectures using QUOTIENT over WAN. Here batch size = 128.

	LAN		WAN
	2 × (512FC)	ResNet	2 × (512FC)
Single Prediction(s)	1.269	1982	17
Batched Prediction (s)	14.31	18122	39.9

Table 3.7: Prediction time for Residual and fully-connected neural networks on Skin cancer MNIST using QUOTIENT over LAN. Here batch size = 128.

cancer dataset, all single predictions take less than 1s and all batched predictions take less than 60s.

Table 3.6 presents an equivalent of Table 3.5 under WAN settings. Here we limit ourselves to only fully connected architectures. While still practical, per prediction costs over WAN are, on average, 20× slower than over LAN. This is due to the high initial setup overhead over WAN. However, batched predictions are only about 4-6× slower over WAN.

In addition to fully-connected and convolutional layers, we also evaluate 2PC-QUOTIENT on residual neural networks. Table 3.7 shows the prediction time of 2PC-QUOTIENT on Skin cancer MNIST dataset for both fully-connected and residual neural networks, while we show its training timing using practical fully-connected neural networks in Table 3.8.

Comparison with Previous Work. The only prior work for 2PC secure training of neural networks that we are aware of is SecureML [18]. They report the results for only fully connected neural network training on the MNIST dataset. 2PC-QUOTIENT is able to achieve its best accuracy levels in less than 16 hours over LAN and less than 90 hours over WAN. This amounts to a speedup of 5× over LAN and a speedup of 50× over WAN. In particular, QUOTIENT is able to make 2PC neural network

	2 × (512FC)		
Epoch	LAN	WAN	Acc
1	8.57h	38.44h	0.2078
5	42.85h	192.2h	0.7026
10	85.70h	384.4h	0.7157

Table 3.8: Training time and accuracy values of QUOTIENT on Skin cancer MNSIT dataset after 1, 5 & 10 training epochs using fully-connected neural network over both LAN and WAN.

training over WAN practical. Moreover, our 2PC-QUOTIENT training is able to achieve near state-of-the-art accuracy of 99.38% on MNIST dataset, amounting to an absolute improvement of 6% over SecureML's error rates upon convergence. In terms of secure prediction, 2PC-QUOTIENT is 13× faster for single prediction and 7× faster for batched predictions over LAN in comparison to SecureML. Furthermore, 2PC-QUOTIENT offers a 3× speed-up for a single prediction and 50× for batched predictions versus SecureML over WAN.

3.6 Conclusion

In this chapter, we introduced QUOTIENT, a new method to train deep neural network models securely that leverages oblivious transfer (OT). In an attempt to design an efficient protocol, we show that by simultaneously designing and adapting new machine learning techniques and new privacy-preserving protocols tailored to machine learning training, we improve upon the state-of-the-art in both accuracy and speed.

QUOTIENT is the first we are aware of to enable secure training of convolutional and residual layers, key building blocks of modern deep learning. However, training over convolutional networks is still slow, and incurs a large communication load with our methods. Finding dedicated MPC protocols for fast evaluation of convolutional layers is an interesting venue for further work. Moreover, the protocol is designed with two parties in mind. In summary, we proposed generalizable insights for designing

efficient crypto-friendly architectures for secure deep learning, in addition to introducing QUOTIENT, a tailored protocol for privacy preserving deep learning. Thus, we addressed our first sub-question *"How can we design crypto-friendly protocols and architectures for secure deep learning?"*. In the next chapter we take a slightly different approach to design a protocol suited for a large number of parties.

4

Secure Federated Learning for Unlabeled Data using Generational Distillation

4.1 Introduction

In the previous chapter, we proposed a two-party protocol for secure training using quantized networks. However, training over data accumulated from IoT devices, such Alexa, Google Home, mobile phones, involves a large number of parties. While relatively efficient, the approach is not suitable for a large scale setting or when more complex DNN architectures are used. This is because of the resulting compute and communication overhead. In this chapter, we take a slightly different approach and focus on training in such a setting. In this process, we address the sub-research question *"Can secure deep learning approaches be decentralised and/or federated?"*. We focus on a scenario where a number of devices (or users) are accumulating their local data i.e. data is (i) *distributed* and (ii) *private*. As a consequence, the data is usually (iii) *unlabeled*, because it is ordinarily not feasible or ideal for users to manually label the data and the sharing of data with external agents for labeling would raise privacy concerns.

Recently, distributed training approaches, particularly Federated Learning (FL)

[25, 43], have emerged as an alternative to training over distributed devices where the data never leaves the device, thereby addressing (i) and (ii)—with some limitations. Federated learning is an on-device distributed learning paradigm for a large number of nodes to jointly train a shared model. Individual nodes use their local data to train their local models which are then aggregated together to update the global model using a variety of aggregation strategies [25, 233]. In this process, the underlying data never leaves any of the nodes and only local models or their updates are sent to a central server (aggregator). This vanilla version of FL training, albeit more efficient and secure than centralized training, still faces a number of challenges:

- *Privacy* (ii): Model updates being sent by the client nodes have been shown to leak significant sensitive information [234, 235] making the nodes susceptible to a variety of attacks [58, 59].

- *Unlabeled data* (iii): FL learning cannot, on its own, enable training over unlabeled client data.

Due to these challenges, FL is of little independent utility for training models on large scale frameworks, such as IoT devices. To address (iii), we turn to semi-supervised learning approaches. In particular, we draw on recent work on self-distillation [236], which proposes an approach for generational improvement of models using self-training. A number of publications [237, 238] have also explored how diverse models could effectively compose into *ensembles*, enabling superior performance as compared to each of the constituent models. Drawing motivation from a combination of these approaches, we propose a multi-model knowledge distillation approach for federated learning over unlabeled data. To further strengthen the guarantees in (iii), we make use of secure model aggregation using additive shares for aggregating the local FL models, thus also addressing (ii).

4.2 Preliminaries

4.2.1 Federated Averaging

Federated Averaging is a scheme for coordinating and aggregating model updates within the broad FL training pipeline. It starts with a subset of nodes N available to participate in the current round of iteration r. Each of the nodes $i \in N$ procure a local copy of the global model $w_i^r = w_r$ for updating the local models with their local data to obtain w_i^{r+1}. These model updates comprise of training the local models using batched SGD for a pre-defined number of epochs over n_i batches of local data. The individual client updates are aggregated to form the new global model as $w_{r+1} = \Sigma_{i=1}^{N} \frac{n_i}{n} w_i^r$, where $n = \Sigma_{i=1}^{N} n_i$.

4.2.2 Knowledge Distillation

A trained deep neural network model encompasses significant knowledge pertaining to a particular task and dataset. Such a model can be used to train a new model or improve a sub-optimal simpler model. This process is called knowledge distillation. Knowledge distillation had been introduced in its full complexity in [239] involving advancements for unlabeled data and temperature switching for smoothing the logits. However, it was originally proposed as a technique for model compression in [240] and was further developed upon in [241].

When a new simpler model is trained by distilling knowledge from a pre-trained cumbersome model, the resulting model could achieve similar performance. This is usually achieved through the use of class probabilities i.e. softmax-ed class scores obtained from the pre-trained model, more commonly referred to as the *teacher model*. These class probabilities (soft labels) contain relevant knowledge of the network and can be used to teach a new model, more commonly referred to as the *student model*.

Knowledge distillation is effected through a training dataset, referred to as transfer set (usually different from the dataset used to train the teacher model). When ground truth labels are available for the transfer set, a weighted combination of the soft label

loss and hard label loss is often used. For an input x and an $n-$dimensional label vector y, a model produces logit vector $V(x) = (v_1(x), v_2(x), \ldots, v_n(x))$ and the softmax-ed $V(x)$ i.e class probabilities as $P(x) = (p_1(x), p_2(x), \ldots, p_n(x))$. In order to soften these probabilities, Hinton et al. [239] suggest using temperature (t)

$$p'_i(x;t) = \frac{exp(v_i(x)/t)}{\Sigma_{j=1}^{n}(v_j(x)/t)} \tag{4.1}$$

For distilling knowledge, scaled softmax probabilities from the teacher, $P^T(x)$, and the student, $P^S(x)$, are used to construct the loss function, L

$$L(x,y) = (1-\sigma)CE(y, P^S(x)) + \sigma t^2 CE(P'^T(x;t), P'^S(x;t)) \tag{4.2}$$

where CE is the cross-entropy loss and σ is a hyperparameter.

The basic teacher-student knowledge distillation paradigm has been built upon in many ways such as using a number of teachers for training [242] or multiple networks helping each other to optimize better [243] amongst others. More recently, an idea of *generations* for self-distillation was proposed in [236]. Furthermore, Furlanello et al. [244] and Yang et al. [245] explore a similar idea where an ensemble of models is used by including a new distilled model in each generation. Concurrently, [246] explores the idea of ensemble knowledge distillation for multiple weak teachers distilling their knowledge into a single student. In this work, we build upon on these ideas of generational distillation and ensemble model distillation with multiple FL models.

Secure aggregation using MPC

Secure aggregation schemes can significantly improve the security and privacy guarantees of Federated Learning. In particular, secure aggregation seeks to protect update leaks during the aggregation phase of the model where the central server has access to individual local updates. These local updates have been studied to be susceptible to a variety of attacks [58, 59, 234, 235]. Multi-party computation schemes for FL aggregation such as [47] enable the server to perform aggregation over local model

updates without learning anything about the individual updates. To further enhance the privacy guarantees, combination of secure aggregation with differential privacy [247, 248] has also been explored.

4.3 Multi-model Federated Generational Distillation

Multi-model federated generational distillation (MFGD) is a novel semi-supervised learning approach for training models in a federated learning setting using unlabeled data. The approach builds upon the idea of generational knowledge distillation together with multiple FL models to improve the individual models using unlabeled data as the transfer set.

4.3.1 Overview of the Approach

In contrast to a vanilla federated learning approach [43], we start with multiple global models. These global models are associated with a mutually exclusive subset of nodes as showcased in Figure 4.1. We refer to these individual subsets as *Silos*. The global model are pre-trained on either the labeled data present on the nodes or on the data available centrally. These pre-trained models then undergo a mutual improvement through generational distillation process using the locally generated unlabeled data on the nodes as the transfer set. After each generation (i.e. local model training and aggregation of the local models into their corresponding global models), each node in the network procure a copy of all the global models. The ensemble of these global models is then used to generate soft labels, for the unlabeled data, for that particular generation on each node. The ensemble uses simple averaging to produce the labels. These newly generated labels eventually allow for updating of the local models and subsequently the global models. The local models update are aggregated into the global models using federated averaging [25].

An indicative representation of the approach is presented in Figures 4.2 and 4.3 where model $S1$ is associated with devices A and B and model $S2$ is associated with devices C and D. After each generation of knowledge distillation driven labeling, global models are updated and redistributed (Models $S1'$ and $S2'$ in Figure 4.3).

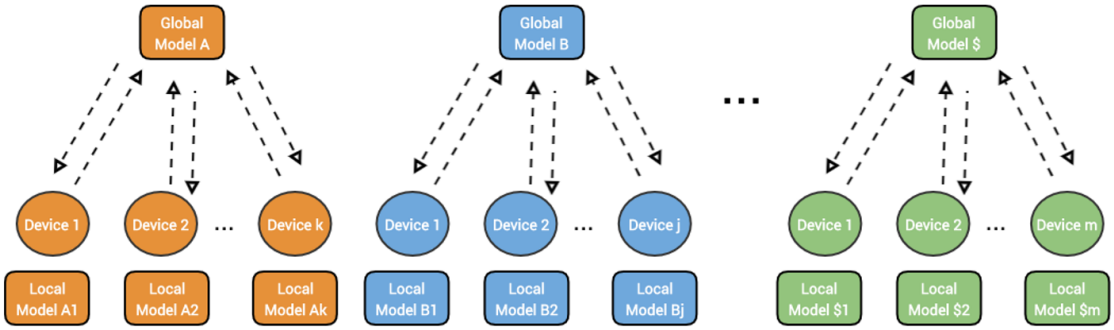

Figure 4.1: *Silos*: Nodes are divided into subsets with their own global models for training.

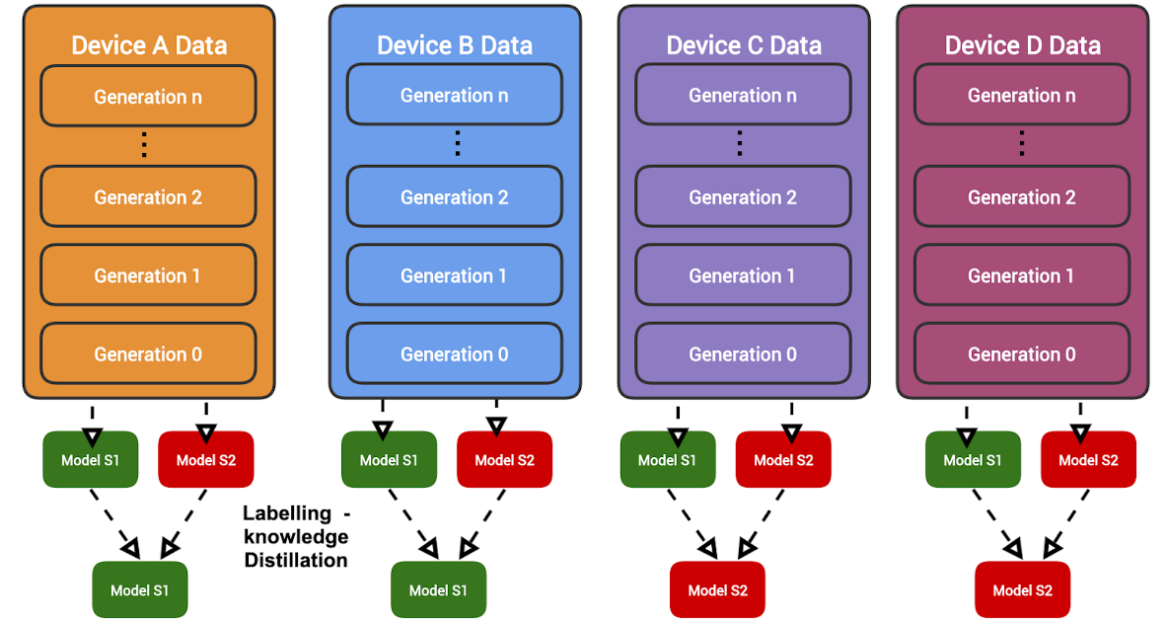

Figure 4.2: How the *generations* work : model $S1$ is associated with devices A and B and model $S2$ is associated with devices C and D.

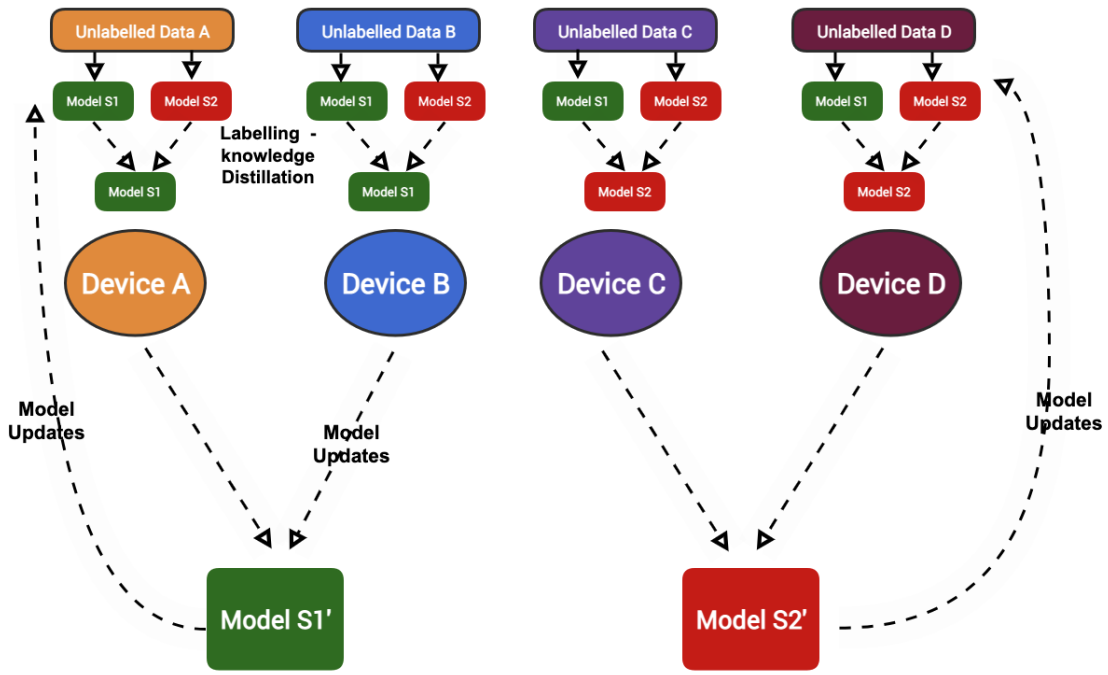

Figure 4.3: Multi model generational distillation for Federated Learning : model $S1$ is associated with devices A and B and model $S2$ is associated with devices C and D. Global models $S1'$ and $S2'$ signify the next generation of the global models.

4.3.2 Algorithm description

We formally state the MFGD procedure for a single generation within a silo in Algorithm 10. We isolate this description for a single generation of training where the procedure uses the local data for that generation $[X_g^1 \dots X_g^n]$, as an input, corresponding to the nodes $1 \dots n$ for a particular silo along with a copy of all the global models from the previous generation $[\theta_{g-1}^1 \dots \theta_{g-1}^{|C|}]$. Here $C = \cup_{i=1}^{M} S_i$ represents the union of all the silos (S_i is a set of indices of all the nodes in the i^{th} silo). Along with that, the procedure also ingests the learning rate, η and the loss function to be optimized, L. The same procedure is replicated across all the silos.

Algorithm 10: Multi-model federated generational distillation.

Input: $[X_g^1 \dots X_g^n], C, [\theta_{g-1}^1 \dots \theta_{g-1}^{|C|}], \eta, L$

▷ where $C = \cup_{i=1}^{M} S_i$

1: **for** $S_p \in C$ **do**
2: **for** $i \in S_p$ **do**
3: $\theta_{g,i} = \theta_{g-1}^p$
4: $\phi_{g,k}^i = f_{\theta_{g-1}}^k(X_g^i), \forall k \in 1 \dots |C|$
5: $Y_g^i = \sum_{j=1}^{|C|} \phi_{g,j}^i$
6: $\theta_{g,i} = \theta_{g-1,i} - \eta \nabla L(X_g^i, Y_g^i | \theta_{g-1,i})$
7: $\theta_g^p = FedAvg_{Secure}(\theta_{g,i}), \forall i \in S_p$
8: **return** $[\theta_g^1 \dots \theta_g^{|C|}]$ ▷ return the updated global models

4.4 Privacy and Security Considerations

We touch upon the privacy and security considerations of our MFGD approach for unlabeled data with respect to the vanilla federated learning for pre-labeled data.

4.4.1 Across the Silos

Each of the silos perform their own replication of Algorithm 10. In contrast to single-model federated learning, for the nodes outside a silo, the aggregating server does not get a copy of the individual model updates. The only interaction between the silos is through knowledge distillation and global models. This motivates superior privacy guarantees similar to the setting discussed in [249]. The setting also open up an arena for employing privacy-preserving model inference techniques [24, 26, 86, 224, 250] to enable secure knowledge distillation without sharing local data or the global models outside the silos.

4.4.2 Within a Silo

As in vanilla federated learning [43], in MFGD, the individual model updates are sent to the central server for aggregation. This makes the updates susceptible to inspection by the server, leaking the updates. These updates have been studied to be a privacy threat

[234, 235]. To mitigate these issues, multi-party computation based secure aggregation strategies have been proposed in the literature [47, 251]. Drawing motivation from those, we adopt an additive secret sharing based approach for model aggregation, thereby safeguarding the updates from leakage. In particular, the individual local models are *secret shared* with all the nodes in a silo. The local shares are then aggregated by each of the parties before sending the updates to the central server. This secure aggregation strategy enables model aggregation without revealing anything about the local models.

4.5 Experiments

In this section, we present experiments with our MFGD approach on a number of textual datasets for a classification task. In the following subsections, we describe the datasets, models, base model training and model improvements. We use an implementation built upon Pysyft[1] [252] and Pytorch[2] for our experiments. We experiment with both IID and non-IID data to evaluate our approach for varied settings.

4.5.1 Datasets

We evaluate our MFGD approach on five public datasets [253] for different classification tasks.

Yelp Reviews contains customer reviews obtained from the Yelp Dataset Challenge, 2015. The dataset contains 560k training instances and 30k test instances. The data is equally distributed between 2 classes. The goal here was to predict the polarity label.

AG News[3] contains titles and descriptions of news articles from 2000 different news sources. The dataset consists of 120k training instances and 7.6k test instances, equally divided across 4 classes. The task was to predict the correct category for the news articles.

[1] https://github.com/OpenMined/PySyft
[2] https://pytorch.org/
[3] http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

Dataset	Data samples per generation
Yelp Reviews	11,200
AG News	12,000
DBPedia	11,200
Sogou News	9,000
Yahoo Answers	22,000

Table 4.1: Number of data samples used for training in each generation.

DBPedia consists of 560k training instances and 70k test instances. The dataset is a collection of titles and abstracts of Wikipedia articles equally distributed across 14 ontology classes. The task was to correctly assign ontology classes to the abstracts.

Sogou News contains the title and the content of news articles divided across 5 domains. The dataset consists of 90k training instances per class (45k in total) and 60k test instances. Similar to AG News, the task was to predict the correct category for the news articles.

Yahoo Answers consists of questions posted on Yahoo Answers. The corpus consists of 10 classes with 140k training instances per class and 50k test instances in total. The goal here was to classify the questions into the the right topic.

In our experiments, we use only a subset of these datasets for training. This subset is 10% of the original strength for Yelp Reviews, DBPedia, Sogou News and Yahoo Answers datasets while it is 50% for the AG News Dataset.

4.5.2 Models

We experimented with an ensemble of neural networks with one embedding layer and one fully connected layer. For all our experiments in this section, we use 10 nodes and 3 global models. The first two models are linked with 3 nodes each, while the third model is linked with 4 nodes. Each node is distributed and has a private local dataset.

Dataset	Accuracy (%)
Yelp Reviews	72.03
AG News	48.15
DBPedia	32.63
Sogou News	32.81

Table 4.2: Single model Vanilla FL training on labeled data.

4.5.3 Base Model Training

We train the base models using an instantiation of vanilla FL within each of the silos. We tabulate the number of samples used in each of the generations for each of the datasets in Table 4.1. Each of the base models use the same number of labeled samples. In this phase, there is no interaction across the silos. For IID data experiments, each node processes an equal number of random samples from each of the classes. For the non-IID data experiments, the training instances are instead sampled using Dirichlet distribution.

4.5.4 Experiments with IID data

For the IID data experiments, we first pre-train the base model using IID data. Once we obtain the pre-trained base models, we discard the labels for the rest of the dataset. The rest of the dataset is used as unlabeled data to train the models using MFGD approach for a number of generations. The cardinality of the samples used for each generation is the same as the one presented in Table 4.1. Each node is sampled with an equal number of random instances. We present the findings in Table 4.3. We observe that all individual models benefit from MFGD training on unlabeled data and are able to help each other to improve through generations.

We also see an overall improvement in the model accuracy i.e. we surpass the accuracy of the best performing individual model from the pre-training phase (base). For example, on DBpedia dataset, the best performing base model improves from 45.83% base accuracy to 51.54% after 4 generations of training on unlabeled data. For

individual models we observe a moderate relative improvement of 4.75% for polarity classification on Yelp and maximum relative improvement of 52% on the Sogou News dataset. In Table 4.2, we present the performance of single model FL training on the labeled section of the data. We train it on the same labeled data and for the same number of epochs as used for pre-training the MFGD models distributed across the nodes. In most cases, these models perform sub-optimally compared to the MFGD base models. We posit that this could be due to the digression as a result of large number of nodes involved in the aggregation. Whereas for MFGD, only a subset of local models participate in the aggregation.

	Yelp			**AG News**			**DBPedia**			**Sogou News**		
Accuracy(%)	**Base**	**Final**	**Gen.**	**Base**	**Final**	**Gen.**	**Base**	**Final**	**Gen.**	**Base**	**Final**	**Gen.**
Model 1	71.75	75.15		54.04	56.18		41.89	48.94		51.46	66.89	
Model 2	72.47	75.23	4	52.45	54.59	4	44.67	50.53	4	43.36	66.07	4
Model 3	74.77	74.84		51.53	52.72		45.83	51.54		69.73	70.36	
Lift	-	4.734		-	4.08		-	16.85		-	52.36	

Table 4.3: Performance for the models trained using MFGD approach on IID data. *Base* represents the model accuracy upon base training with labeled data, *Final* represents the model accuracy upon *Gen.* generations of training over unlabeled data. *Lift* indicates the maximum relative improvement for any model.

4.5.5 Experiments with non-IID distribution

In a realistic FL environment with unlabeled data, it is quite likely that the data being generated by individual nodes is not uniform. For example, different Alexa users are likely to query different domains and as a result they might generate skewed data distributions. To mimic this setting we experiment with non-IID distributions.

For these experiments, we use the Dirichlet distribution to sample elements from different classes for each of the nodes. We use the same mechanism for training the MFGD base models on labeled data as explained in Section 4.5.3. The dataset size for each generation is still kept the same as described in Table 4.1. For pre-training we use one generation worth of labeled data followed by MFGD-based training on unlabeled

data. We present the model performance for non-IID data in Table 4.4. We observe far greater improvement for individual models as well as an overall improvement in accuracy. For example, on Yelp dataset where we observe a relative improvement of $\sim 4.7\%$ on IID data, the non-IID data driven models improve by $\sim 40\%$ relatively. The maximum model accuracy at base level improves from 63.86% to 72.75% after training over unlabeled data. We observe similar effects across other datasets as well, with a maximum relative improvement of 54.7% for the Yahoo Answers dataset. This behaviour could be attributed to the fact that the global models are trained on skewed samples at base level which results in their sub-optimal performance on IID test set. However, this diversity of models results in far superior final models when they share their knowledge through distillation. In particular, MFGD helps models generalize well even when trained on skewed data.

	Yelp			**AG News**			**DBPedia**			**Yahoo Answers**		
Accuracy (%)	**Base**	**Final**	**Gen.**	**Base**	**Final**	**Gen.**	**Base**	**Final**	**Gen.**	**Base**	**Final**	**Gen.**
Model 1	51.94	72.02		35.05	42.76		34.42	40.77		19.44	24.40	
Model 2	55.08	72.75	3	47.34	50.28	3	38.10	45.93	4	16.29	25.21	2
Model 3	63.86	71.18		44.18	47.41		26.04	33.83		20.92	24.38	
Lift	-	39.64		-	21.10		-	29.92		-	54.70	

Table 4.4: Performance for the models trained using MFGD approach on non-IID data. *Base* represents the model accuracy upon base training with labeled data, *Final* represents the model accuracy upon *Gen.* generations of training over unlabeled data. *Lift* indicates the maximum relative improvement for any model.

4.6 Conclusion

This chapter ties back to and addresses our sub-question *"Can secure deep learning approaches be decentralised and/or federated?"*. We answer the question in the affirmative. We propose a multi model federated generational distillation approach for secure federated learning over unlabeled data. With this framework, we propose and evaluate a strategy for training in a decentralized manner where a large number of parties are involved.

We also motivate the idea of silos and expand on the benefits of using multiple models. Moreover, there is a potential to explore privacy-preserving inference approaches for secure knowledge distillation and ensuring model privacy. The approach presents a significant improvement over base models when trained exclusively with unlabeled data over generations. We evaluate the approach for both IID and non-IID data distributions and report relative improvements of up to $\sim 54\%$ in individual models performance.

The work has its limitations. It is an early exploration into federated learning for unlabeled data. It does, however, set the context for further research and technical analysis that could lead to real world uptake. We do not explore, in detail, the privacy guarantees entailed by the silos. We provide some intuition into this, but a stronger theoretical analysis is needed. Furthermore, we have used one measure of skewness while experimenting with non-IID data. There are other ways to trigger such distributions. Nonetheless, the benefits of our methods should be apparent for any data distribution. Finally, we do not address issues around data trustworthiness in this chapter. Data contamination is a significant challenge in collaborative training over distributed data. This could allow additional attacks [254]. However, this is outside of the scope of this thesis. In the next chapter, we move away from the challenge of securely training networks to the next stage in the process for end to end privacy-preserving machine learning. We focus on the problem of securely enforcing constraints and ensuring model integrity while using a machine learning model.

5

MPC-Friendly Commitments for Publicly Verifiable Covert Security

Contents

5.1 Introduction

Chapters 3 and 4 focused on privacy-preserving training for a variety of settings. In this chapter we move to the next ingredient in delivering an end-to-end secure solution for deep learning—*"How can we ensure the integrity and security of the derived models?"*. In particular, we focus on securely establishing the identity of the input model while performing secure inference. In this process, we start with a generalized approach to be used for not just machine learning models but for ensuring integrity of inputs to an arbitrary MPC computation followed by motivating and explaining how this applies to ML predictions.

Generally speaking, the guarantee of an MPC computation is that the inputs of the participants remain private to other parties, but that does not prevent parties from choosing their inputs in an arbitrary way, e.g., in the well-known millionaires' problem [53], nothing prevents a millionaire from lying about their wealth. Going back to the private prediction application mentioned above: nothing prevents the model owner from modifying the model arbitrarily. This is a problem in settings where the model has to satisfy certain non-functional constraints such as safety, fairness, or privacy. These constraints undermine accuracy (as often measured in ML) and thus the model owner may have an incentive to switch the model. This exact problem and, more generally, model certification, was tackled recently by Kilbertus et al. [195] and Segal et al. [255]. Both of these pieces of research show that commitments verified in MPC can help here. For example, consider a service provider offering dietary or exercise recommendations based on personal data. Users may require the service provider to commit to a recommendation algorithm that is certified not to make harmful recommendations (which could have been verified and signed by a regulator). More formally a user requires the following 2-party functionality: the service provider (P1) commits to an input x by producing commitment c and sends c to the user (P2).

Later, P2 uses c to verify that x is being used by P1 inside an MPC protocol between both parties. We call this *MPC on committed data*. We describe an application of this framework to certified predictions in Section 5.7, along with an empirical example on realistic data showing that heuristic methods that do not ensure that the model does not change will fail. Concretely, we show that changing a single parameter in a fair model results in an unfair model with increased accuracy.

So far current work on this uses standard collision-resistant hash functions such as SHA-256 [195] and SHA-3 [255] to produce and then verify commitments in MPC. However, these methods do not take advantage of two key properties of this setting: 1. *Interactivity*: given that an MPC protocol needs to be run between the user and service provider to compute some functionality (e.g., a recommendation), it is possible to leverage the interactivity of the protocol to construct a commitment; 2. *Reputation of service provider*: as this computation involves a service provider who relies on users for profit, a protocol can be constructed so that cheating would harm the reputation of the service provider, using ideas from Publicly-Verifiable Covert (PVC) security [256].

Based on these properties we make a simple observation: one can detect if an input x to a Boolean MPC protocol has been changed with probability $1/2 - \epsilon$ (for arbitrarily small $\epsilon > 0$) using a simple additional Boolean circuit as part of the protocol (more details on such circuits are in Figure 5.2). The idea is that in MPC a hash can be efficiently constructed by using inputs from both parties, an idea we call *indexed hash functions*. These functions allow one to build MPC commitments in the PVC setting that analytically and experimentally outperform prior approaches by as much as 60× in runtime and 36× in communication.

Other Related Work. Baum [257] also discusses the problem of input validity, using efficient secure function evaluation (SFE) protocols. In particular, the solution utilizes universal hash functions and committed OT. The protocol specifically improves the performance of garbled circuit based secure function evaluation for cases where sub-circuits depend on only one party's input. Concurrent with [257], Katz et al. [258]

propose a solution performing predicate checks followed by secure evaluation of an arbitrary function, if the inputs pass the verification. However, both these works focus on malicious security, different from our proposal in PVC security model, utilizing properties 1 and 2. In particular, our approach does not fit their paradigm because our commitment verification has a private input. However, our results for maliciously secure setting (Section 5.9) are compatible with their optimizations.

5.2 Preliminaries

We give a brief background on key ideas we will use in this chapter.

Hash functions and pseudo-randomness. We consider a hash function to be a function $h : \{0,1\}^* \to O$ for some finite output space O. Informally, h is *collision resistant* if no adversary is capable of finding two distinct inputs with the same image except with negligible probability. The formal definition requires talking about families of hash functions [259]. A pseudo-random number generator, or PRNG, is a function $\mathsf{prng} : K \times \mathbb{N} \to \{0,1\}^*$ which maps (k, b) to a bit-string of length b. Both hash functions and PRNGs can be modelled as random oracles. That is, as a uniformly random mapping from their inputs to their outputs.

Publicaly-verifiable covert (PVC) security. Covert security [260] weakens the malicious security setting by guaranteeing that a cheating party (who may behave arbitrarily) will be caught by the other party with a probability, p, referred to as the covert security parameter. The motivation for covert security is that, if certain parties have a reputation to preserve, then the risk associated with being caught outweighs the benefit of cheating. This allows faster protocols than malicious security [260–263]. PVC security was introduced by Asharov and Orlandi [256]. It, with probability p, provides a publicly-verifiable proof of cheating, which allows greater reputational harm and possibly legal repercussions for a cheater.

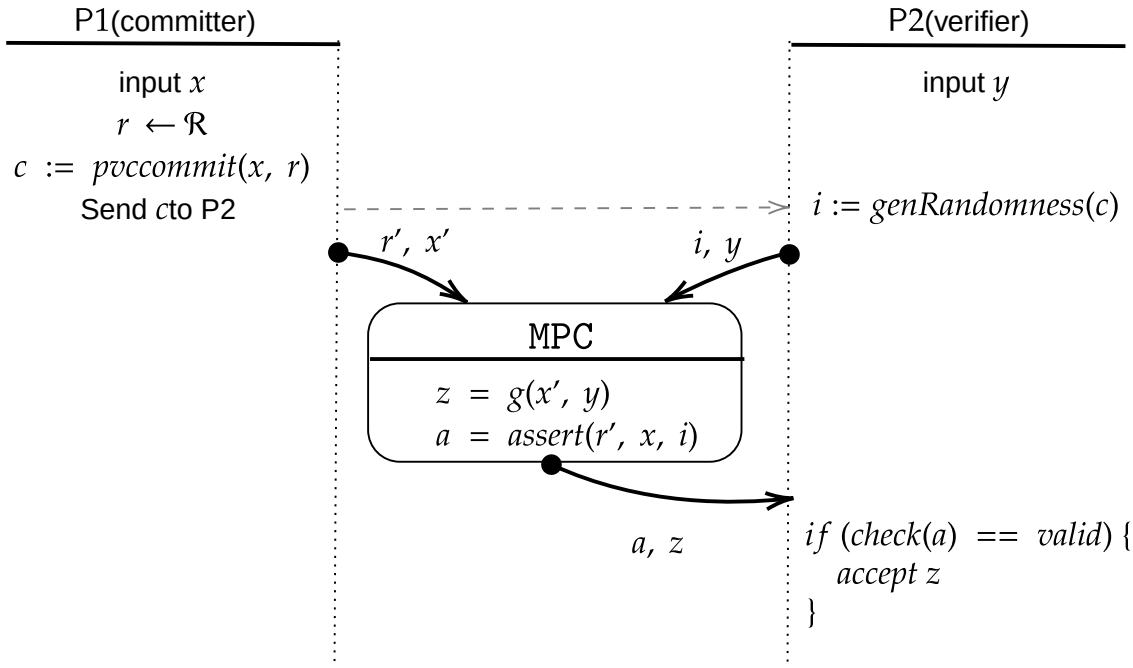

Figure 5.1: The diagram shows a multi-party computation with a committed input x, as enabled by our constructions. Party 1 (the committer) holds an input x, for which it generates a commitment c and sends it to party 2 (the verifier). The commitment is randomized using r to ensure privacy for x, i.e., that the commitment is hiding. Later on, the parties engage in a secure computation of a generic function g, where party 1 inputs x', and party 2 inputs input y. For the purpose of verifying that $x = x'$, party 2 derives a challenge i from c, and the MPC returns a certificate a that can be checked by P2. This guarantees to party 2 that g is evaluated on the value x to which party 1 had previously committed.

Commitment schemes. A commitment scheme is a two-party cryptographic primitive run between a committer C and a verifier V. It allows C to commit to a value m and reveal it to V at a later time. Party V can then verify that the value m' sent by the C is the one they originally committed to, i.e., $m' = m$. We provide here a version of this notion with PVC security, which only makes sense if the verification is occurring in MPC. We will also use the original notion as an optimization of the run time.

5.3 Overview

Hashing is a useful primitive to implement in MPC, as generally speaking it enables privacy-preserving consistency checks, and thus *MPC on committed data*. This work focuses on efficient implementation of this functionality, depicted abstractly in Figure 5.1. The high-level goal is to enable a party P1 to commit to a *private* value x at some point in time and later, when engaging in a secure computation of a function g with a second party P2, provide the assurance to P2 that P1 inputs the committed value x into the computation and not some other value. This is modelled, analogously to commitment schemes, by three algorithms: pvccommit, assert, and check. As shown in Figure 5.1, pvccommit outputs a commitment to an input, later in the MPC assert is run. As we will see next, in a naive instantiation of this scheme assert could simply compute pvccommit, but our constructions will leverage randomness in assert. Otherwise the algorithms correspond to a standard commitment scheme, in that they should satisfy the standard *binding and hiding* properties. The function check is used to interpret the results and see whether cheating has occurred. The verification is split between assert and check.

The functionality of Figure 5.1 has appeared in previous works on *certified predictions* [195, 255] where x is a confidential machine learning model owned by P1 that has been checked by a certifying authority to have certain properties, e.g., fairness. In those applications, g corresponds to model evaluation and users receive predictions using the certified model x. MPC on committed data directly enables this functionality, with the model owner and the users playing the roles of parties P1 and P2 in Figure 5.1, respectively. We describe and motivate in more detail this application in Section 5.7.

Baseline protocol. To see how a hash function h, e.g., SHA3 in practice, can be used to implement MPC on committed data, consider the following instantiation of pvccommit, assert, and check. P1 can just choose a random r and have pvccommit(x, r) and assert(x, r) be $h(x||r)$. Then, check just checks that they are equal. Both

pvccommit and check are efficient in this instantiation, so one would want h to have an efficient MPC protocol.

MPC-friendly hashing. The works on fairness certification of Segal et al. [255] and Kilbertus et al. [195] propose the above baseline construction. Segal et al. [255] concretely instantiate h with the Keccak-F function, which is the basis of the SHA3 standard. That function takes a 1600 bit input and can be implemented by a Boolean circuit of 38400 AND gates i.e. 24 AND gates per input bit [255][1]. Using a Merkle tree for succinctness of c, the total number of hashes is twice the number of input blocks, resulting in 48 AND gates per input bit of x. Thus assert would result in an overhead of 48 AND gates/bit in this instantiation. Alternatively, using SHA3-256 in sponge mode results in an overhead of roughly 35 AND gates/bit. On the other hand, using an MPC-optimized hash (but new and susceptible [264]) LowMCHash-256 [265] in sponge mode roughly take up 14 AND gates/bit. Note that AND gate counts, and non-linear gate counts in general, are a standard reference for computation time in MPC, and secure computation in general.

In this work we propose efficient MPC-friendly commitments schemes based on hashing, with a focus on Publicly Verifiable Covert (PVC) security.

Our starting observation is that executing a collision-resistant hash function such as SHA-256 in a PVC-secure protocol is an overkill: Note that ensuring that commitments are binding, i.e., that P1 in Figure 5.1 cannot generate x', r' such that $x \neq x'$ and the verification passes, up to negligible probability, but then ensuring that the subsequent MPC is secure only up to a constant probability p leaves some potential room for weakening the binding guarantee of the commitment scheme to favor efficiency. To take advantage of the PVC setting, we observe that assert must in some way receive randomness from P2, as it is against this randomness that P1 will have probability p of being caught. We design assert functions that leverage this observation, and result

[1] `https://homes.esat.kuleuven.be/~nsmart/MPC`

in an overhead of the assert circuit as low as *half an AND gate per bit of the input* x, even when p is close to 1, including $1 - 2^{-\sigma}$ for a statistical security parameter σ.

Contributions

We introduce the notion of an *indexed hash function*. This, roughly speaking, is a function that produces a hash of an input x, given a random value r and an index i from a domain $\mathcal{I}$. The index of the hash function plays the role of the randomness chosen by P2 mentioned above. We build indexed hash functions from any collision-resistant hash function h and prove properties related to collision resistance that allow us to construct PVC commitments from indexed hash functions, and use them for achieving the functionality in Figure 5.1 with PVC security.

Given an indexed hash function H, our proposed PVC commitment schemes follow the following high-level structure: pvccommit computes $(H(j, r, x))_{j \in I}$, i.e. the hash evaluated at all indices, the verifier selects an index $i \in \mathcal{I}$, assert computes $H(i, r, x)$, and check checks that this value is correct. This check fails (in the sense of giving a false positive) with a probability upper bounded by $1 - p$ (p is called the covert security parameter). This fact is formalized by reducing an appropriate notion of collision resistance of H to the collision resistance of h.

The efficiency of our scheme relies on the fact that our constructions for indexed hash functions are very efficient to be evaluated in MPC, requiring a very small number of XOR and AND gates with respect to the input size and thus inducing a very small overhead when evaluated in MPC. Next, we summarize key contributions of each section.

A Construction for Boolean circuits (Section 5.4). We give a construction for Boolean circuits that can achieve a covert security parameter arbitrarily close to 1/2, which asymptotically requires only half an AND gate per bit. Furthermore, in practice, it requires less than one AND gate per bit for moderately sized inputs.

PVC (Section 5.5). We define PVC commitment schemes and their security properties, propose a secure instantiation based on indexed hash functions, and show how to use them for committed PVC MPC.

Experimental Evaluation (Section 5.6). We fully implement our most practical construction, and compare it with the baseline approach (both instantiated with SHA3 and an MPC-friendly hash function LowMCHash). Our experiments show a 60× speed up and 36× less communication in the resulting assert compared to SHA3. It takes $\sim$ 1 MB and $<$ 15 minutes to commit million bit inputs.

Certified Predictions. (Section 5.7). We describe an integral machine learning application of our framework— certified predictions. To motivate the application we show how manipulating even a single parameter in a neural network can significantly perturb the fairness and increase the accuracy of a neural network model. We also discuss other related solutions for this application.

Lower bounds (Section 5.8). We show the optimality of our construction by proving lower bounds for both our approach and the baseline hash based approach. In particular, we show that if the resulting hash is required to be even remotely succinct (to have size $< 99x/100$) then (i) the indexed hash function must require at least half an AND gate per bit asymptotically, and (ii) with an ordinary hash function at least one AND gate per bit is required. We also show that even if we remove the assumption on the size of the hash at least 1/5 of an AND gate per bit would be required.

A Way to Amplify Security (Section 5.9). We propose a way to achieve covert security parameter $1 - 2^{-\sigma}$, and thus full statistical security, in both of the Boolean and arithmetic cases. The method relies on repeating our constructions to amplify their probabilistic guarantees.

A Construction for Arithmetic Circuits (Section 5.10). We propose an analogue to our Boolean construction for arithmetic circuits which can achieve a covert security parameter arbitrarily close to 1 with only half a MULT gate per input element

asymptotically. This is also a practical improvement over the best known ordinary hash functions [265]. However the security parameter cannot be taken to be ~ 1

5.4 Indexed Hash Functions

In this section we will introduce the primitive we will use to build our commitments: *indexed hash functions*. We do so informally and then formally, and give some examples of indexed hash functions. The examples will show that secure indexed hash functions have much smaller circuits than ordinary secure hash functions.

Like an ordinary secure hash function, an indexed hash function takes an input from some space $\mathcal{X}$ and produces an output in a space $\mathcal{O}$. When working with a specific $x \in \mathcal{X}$ we will denote by n the bitlength of x.

The whole idea is to take advantage of the fact that the verifier (P2) can have an input to the indexed hash. We call this input the *index* of the hash function and denote it by i, drawn from an index set $\mathcal{I}$. If the wrong input from $\mathcal{X}$ is used during verification, then at least a fixed fraction of indices i will result in an incorrect hash.

We need to ensure that the hash is hiding, otherwise if $x_1, x_2 \in \mathcal{X}$ were the possible inputs by the committer, then by computing the hash of x_1 and x_2 the verifier could learn which was the true input from the output. Thus the committer must provide an extra random input r from some set $\mathcal{R}$ containing enough entropy to hide the true input. Hence we define indexed hash functions as functions taking an index i, a random nonce r, and an input x.[2]

Definition 5.4.1. *An* indexed hash function *is a function* $H : \mathcal{I} \times \mathcal{R} \times \mathcal{X} \to \mathcal{O}$.

[2]We call these hash functions because they have output smaller than their input and, even with the insertion of randomness that is not technically part of the function, a single evaluation of this function would not create a commitment.

5.4.1 Collision resistance

Next, we formally define two properties for indexed hash function that we require: (i) the hiding property, i.e., not leaking information about x, and (ii) a notion of collision resistance which we call *q-collision boundedness*. For (i) we can use the definition from classical commitment schemes: we say H is *hiding* if an adversary learns a negligible amount about an input x from learning the value of $H(i, r, x)$ for all $i \in \mathcal{I}$, and a uniformly random (and secret from the adversary) $r \in \mathcal{R}$.

For (ii), we start by defining the concept of a q-collision, which denotes a pair of inputs on which H collides for at least a fraction q of all possible indices $|\mathcal{I}|$.

Definition 5.4.2. *Let $q \in [0, 1]$. We call a quadruple r, x, r', x' a* q-collision *of H if $x \neq x'$ and*

$$|\{i \in \mathcal{I} | H(i, r, x) = H(i, r', x')\}| \geq q|\mathcal{I}|$$

We can now define our notion of collision resistance. Informally, H is *q-collision resistant*, if adversaries are unable to find a q-collision of H except with negligible probability. We formalize this using families of indexed hash functions, in turn indexed by a key $k \in K$ generated by a generator G taking a computational security parameter λ. This is similar to the standard definition of a family of collision-resistant hash functions. Moreover, we say that H is *q-collision bounded* if it is q'-collision resistant for every $q' > q$.

The security parameter λ. We use a single computational security parameter λ for all aspects of our constructions. This includes their underlying collision resistant hash function, as well as the size of the source of randomness $\mathcal{R}$ and the set of indices $\mathcal{I}$.

In particular, $|\mathcal{I}|$ is polynomial in λ and $|\mathcal{R}|$ is exponential in λ in all constructions. Thus our security is formalized in terms of polynomial time adversaries w.r.t. λ, and whose advantage should be bounded by a negligible function in λ. Note that this implies that an attacker is allowed to iterate over $\mathcal{I}$, and in fact in practice we will ensure that $\mathcal{I}$ is as small as possible for efficiency.

Definition 5.4.3. *Given a generator G, security parameter λ, and key $k = G(\lambda)$, a family $\{H_k\}_{k\in K}$ is q-collision resistant if, for any probabilistic polynomial time algorithm A, we have that*

Definition 5.4.4. *A family $\{H_k\}_{k\in K}$ is q-collision bounded if it is q'-collision resistant for all $q' > q$.*

This property will be useful later because by choosing an index uniformly at random we can distinguish between any two inputs $x, x' \in \mathcal{X}$, s.t. with probability at least $1 - q$ by looking at a hash. We can now make the hiding property precise. That H is hiding will be proved under the assumption that h is a random oracle.

Definition 5.4.5. *A family of indexed hash functions $\{H_k\}_{k\in K}$ is hiding if for any polynomial time algorithm A and any $x, x' \in \mathcal{X}$, for a uniformly random choice of $r \in \mathcal{R}$ we have*

$$\mathbb{P}(A(k, (H_k(i, r, x))_{i\in\mathcal{I}}) = 1) =$$

$$\mathbb{P}(A(k, (H_k(i, r, x'))_{i\in\mathcal{I}}) = 1) + \mathit{negl}(\lambda)$$

As mentioned above, we will construct our indexed hash functions by building them from an ordinary secure hash function h. We do so because it will allow us to prove q-collision resistance of H via a (ptime) reduction to collision resistance of h. To argue about collision boundedness we formalize the notion of a construction.

Definition 5.4.6. *A* construction *of an indexed hash function is a function $\mathcal{C}$ which given a hash h and the security parameter λ, returns an indexed hash H.*

We say that $\mathcal{C}$ *preserves q-collision boundedness* if there is an (efficient) algorithm which, given a q-collision of $\mathcal{C}$, returns a collision of h. Thus if a powerful adversary is unable to find a collision in some fixed hash function h, then it is reasonable to assume they cannot find a $q'|\mathcal{I}|$-sized collision in $\mathcal{H}$ for any $q' > q$.

Definition 5.4.7. *We say that $\mathcal{C}$* preserves q-collision boundedness *if $\{h_k\}_{k\in K}$ being collision resistant implies that $\{H_k = \mathcal{C}(h_k)\}_{k\in K}$ is q-collision bounded.*

Construction	$\mathcal{C}_0$ (baseline)	$\mathcal{C}_1$	$\mathcal{C}_2$	$\mathcal{C}_3$ (main)
# ANDs	$C_N n$	$n + C_N \frac{n}{b}$	$\frac{n}{2} + C_N \frac{n}{b}$	$\frac{n}{2} + C_N \frac{n}{b}$
# XORs	$C_L n$	$n + C_L \frac{n}{b}$	$\frac{3n}{2} + C_L \frac{n}{b}$	$\frac{3n}{2} + C_L \frac{n}{b}$
$\lvert\mathcal{I}\rvert$	1	2^b	2^b	$\frac{b+\lambda+1}{2\epsilon^2}$
q	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2} + \epsilon$

Table 5.1: Let H be an indexed hash function resulting from a construction, using h as the underlying collision resistant hash function. This table shows (i) the size of a Boolean circuit for H in terms of number of bits n of the input (omitting lower order $O(n)$ additive terms), where C_N and C_L denote the number of AND and XOR gates of h, respectively; (ii) the size of $\mathcal{I}$, and (iii) the value of q for which q-collision resistance holds. The parameter b can be chosen to be any even positive integer and ϵ can be taken to be any positive value.

5.4.2 Constructions

We will now give four constructions of indexed hash functions denoted $\mathcal{C}_0$ through $\mathcal{C}_3$. Our construction 3 is the most practical and the one we would recommend to use (we will use this construction in our experiments in Section 5.6), but we include all four to build up ideas incrementally. Our goal is to derive indexed hash functions that:

1. *Are efficient to implement in MPC*: this aspect of the constructions in this section is captured in terms of *the number of AND and XOR gates* of their corresponding Boolean circuit implementations (in Section 5.10 we give a construction for arithmetic circuits).

2. *Have a small index domain $\mathcal{I}$*: this directly corresponds to the commitment size of the PVC commitment schemes that we will build on top of them.

For these constructions the value of q (i.e., for which constructions are q-collision bounded) is one minus the security parameter of the PVC commitment scheme derived later. We summarize computation, size, and allowed q values in Table 5.1. In this section we have $q = 1/2$ and $q = 1/2 + \epsilon$ (for arbitrarily small $\epsilon > 0$), but in Section 5.9 we show how to achieve arbitrarily small q.

Blueprint for our constructions. Consider an indexed hash function H taking an index i, randomness r, and input x. All our constructions are parametrized by a *block size* $b \in [|x|]$ and a *block digest function* d. The latter takes (i) a binary encoding of i and (ii) a bitstring of size b, and outputs a *single* bit, i.e. $d : \mathcal{I} \times \{0,1\}^b \to \{0,1\}$. The indexed hash function is defined to be the result of

1. splitting x into n/b consecutive blocks of b bits (if $|x|$ is not a multiple of b, it can be padded with zeros),
2. applying $d(i, \cdot)$ to each block x_j, and
3. outputting the length n/b bitstring resulting from concatenating all digested bits $d(i, x_j)$.

We denote the result of processing an input i, x as in steps 1-3 by $\texttt{process}_{b,d}(i, x)$. Digest function d determines the size of the index set $\mathcal{I}$.

Then, each construction $\mathcal{C}$ is defined by a block size b, digest function d, an ordinary collision resistant hash function h, and a set of random masks $\mathcal{R}$ (which we always take to be $\{0,1\}^\lambda$) as:

$$\mathcal{C}(h, \lambda)(i, r, x) = h(r||i||\texttt{process}_{b,d}(i, x)) \tag{5.1}$$

When presenting the three constructions in this section we will denote the digest function associated with $\mathcal{C}_j$ by d_j. The motivation behind this presentation is simplicity, as it is now enough to define d_1, d_2, d_3, and the arguments in our proofs only need to refer to d_j.

The hiding property. We can prove the hiding property (Def. 5.4.5) without knowing anything about d and thus the size of $\mathcal{I}$, so we do this in generality for all the constructions.

Theorem 5.4.8. *Suppose $\mathcal{C}$ is given by Equation 5.1. If $\{h_k\}_{k \in K}$ is a family of random oracles then $\{\mathcal{C}(h_k, \lambda)\}_{k \in K}$ is hiding.*

Proof. Let $r \leftarrow \mathcal{R}$. Suppose that a polynomial time algorithm A is given input k, $(H_k(i,r,x))_{i\in\mathcal{I}}$. For fixed k, the $H_k(i,r,x)$ are independent uniform random variables irrespective of the value of x or r, so without querying the oracle the adversary can learn nothing about x or r.

When the adversary requests the value of the random oracle on an input beginning with $r' \in S$ suppose it is also told whether or not $r' = r$.

When the adversary queries with $r' \neq r$ it learns nothing about r except that $r \neq r'$. Thus the probability of using the right salt on the jth query is at most $1/(|\mathcal{R}| - j + 1)$ and so the probability of querying the correct r with a guesses is at most $a/|\mathcal{R}|$. As the adversary has time for only polynomially many queries and $|\mathcal{R}|$ grows exponentially in λ it will query with r as the randomness with negligible probability.

Conditioned on A never querying the correct randomness, its view is independent of x and thus so is the probability of it outputting 1. □

Collision boundedness. For each C_j we propose, we will show q-collision boundedness of $H = \mathcal{C}_j(h,\lambda)$. This will be done by showing that for at most $q|\mathcal{I}|$ indices i we have $\texttt{process}_{b,d}(i,x) = \texttt{process}_{b,d}(i,x')$. That is a sufficient condition for q-collision boundedness is shown in the following theorem.

Theorem 5.4.9. *Let $q \in [0,1]$. Suppose that for any $x \neq x'$ and for any sufficiently large λ,*

$$|\{i \in \mathcal{I} | \texttt{process}_{b,d}(i,x) = \texttt{process}_{b,d}(i,x')\}| \leq q|\mathcal{I}|$$

then the construction in Equation 5.1 preserves q-collision boundedness.

Proof. Let $\{h_k\}_{k\in K}$ be a collision resistant family of hash functions, and $H_k = \mathcal{C}(h_k,\lambda)$. Suppose that the hypothesis of the statement holds but there exists a polynomial time algorithm A which finds a q-collision in H_k with non-negligible probability. We show next that $\{h_k\}_{k\in K}$ is not collision resistant: a contradiction.

Let $k = G(\lambda)$. For sufficiently large λ, a probabilistic ptime (in λ) algorithm B for finding a collision in h with non-negligible probability is given by the following.

Given k, B computes $(r, x, r', x') = A(k)$. If (r, x, r', x') is a q'-collision for some $q' > q$ (this happens with non-negligible probability) then B computes an index i such that $\texttt{process}_{b,d}(i, x) \neq \texttt{process}_{b,d}(i, x')$ but $H_k(i, r, x) = H_k(i, r', x')$. Note that such an i must exist with probability 1 and can be found by exhaustion in time $O(\mathcal{I})$ (and thus $O(\lambda)$). By Equation 5.1 we now have that $r||i||\texttt{process}_{b,d}(i, x)$ and $r'||i||\texttt{process}_{b,d}(i, x')$ form a collision in h_k. The algorithm B outputs this collision. The fact that B succeeds with non-negligible probability contradicts the collision resistance of the family $\{h_k\}_{k \in K}$. □

We now proceed to present each construction $\mathcal{C}_j$ by specifying the length of $\mathcal{I}$ and the digest function d_j to sub into Equation 5.1. Recall that all constructions are summarized in Table 5.1.

Construction 0**.** Firstly let us consider a trivial construction. Let $\mathcal{I}$ be the set containing only the empty string ϵ and let $d_0(i, x) = x$ and the block size be $b_0 = |x|$. Let

$$h(r||\epsilon||\texttt{process}_{b_0,d_0}(i, x)) = h(r||x).$$

Theorem 5.4.10. *$\mathcal{C}_0$ preserves 0-collision boundedness.*

Proof. By Theorem 5.4.9 this is immediate as the identity function has no collisions, and thus the condition of that theorem holds for $q = 0$. □

This construction is very simple and d_0 is trivial to compute. However, we can improve this by making the digest function d compress the input so that h only has to be computed on an input much smaller than x, resulting in a more efficient circuit.

Construction 1**:** n **AND gates.** The digest function for construction 1 is shown in Figure 5.2 (left). As above, this construction is parameterised by a block size b. Let $\mathcal{I} = \{0, 1\}^b$ and let x_j be the block containing bits jb through $jb + b - 1$, inclusive, of x (padding x with zeros to length a multiple of b). We use & to denote bitwise AND, and let parity map bit strings to the XOR of all their bits. Then we define $d_1(i, x_j) = \mathsf{parity}(x_j \& i)$.

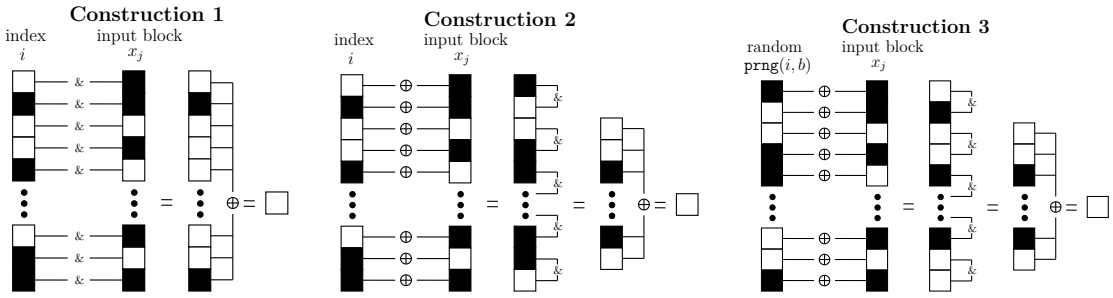

Figure 5.2: A depiction of digest functions d_1 (left), d_2 (center), and d_3 (right).

Theorem 5.4.11. *$\mathcal{C}_1$ preserves 1/2-collision boundedness.*

Proof. If x and x' differ, they differ in some block j. Let $\texttt{process}_{b,d}(i,x)_j$ be the jth bit of $\texttt{process}_{b,d_1}(i,x)$. Note that $\texttt{process}_{b,d_1}(i,x)_j = \texttt{process}_{b,d_1}(i,x')_j$ if and only if $d_1(i,x_j) = d_1(i,x'_j)$, which happens if and only if $\mathsf{parity}((x_j \oplus x'_j)\&i) = 0$ (as conjunction distributes over exclusive or). As $x_j \oplus x'_j \neq 0$, we have that $\texttt{process}_{b,d_1}(i,x) = \texttt{process}_{b,d_1}(i,x')$ holds for exactly half of the possible values of i. The result is then immediate from Theorem 5.4.9. □

Note that d_1 (and thus $\texttt{process}_{b,d_1}(i,x)$) has only one AND gate per bit of input and the output of $\texttt{process}_{b,d_1}(i,x)$ has size $\lceil n/b \rceil$ (see Figure 5.2). Thus for large n and b this construction can asymptotically be computed with a number of AND gates arbitrarily close to n (and n XOR gates). However, as we will see in the lower bounds section (Section 5.8) only $n/2$ AND gates are needed for any collision resistant indexed hash function. This motivates looking for the following construction which closes this gap.

Construction 2: $\frac{n}{2}$ AND gates. We wish to avoid using an AND gate for each bit of the input but still need some nonlinearity in d. So the idea is to combine two bits of the input together using a single AND gate. If the raw input bits went directly into the AND gate then the adversary would sometimes be able to change them in a way that definitely would not change the output of the gate. Instead we

will first XOR each bit with a bit from i. Now the adversary cannot tell whether changing a certain bit will change the output.

Let $\oplus$ denote bitwise XOR. Let y be a bitstring of length $2m$ and let y_j be the jth bit of y, we define $\mathsf{andreduce}(y)$, as the concatenation of $y_{2j} \& y_{2j+1}$ for all $j \in (0, ..., m-1)$. Then we define $d_2(i, x_j)$ to be $\mathsf{parity}(\mathsf{andreduce}(x_j \oplus i))$, see Figure 5.2 (center).

A *bent function* has the property that for a fixed linear change to its input, the output of the function would change for exactly half of all starting inputs [266]. The following theorem boils down to showing that $\mathsf{parity} \circ \mathsf{andreduce}$ is a bent function.

Theorem 5.4.12. *$\mathcal{C}_2$ preserves 1/2-collision boundedness.*

Proof. By Theorem 5.4.9 it suffices to show that if $x_j \neq x'_j$, i.e. x and x' differ in the jth block, then $e_j := d_2(x_j) \oplus d_2(x'_j) = \mathsf{parity}(\mathsf{andreduce}(x_j \oplus i)) \oplus \mathsf{parity}(\mathsf{andreduce}(x'_j \oplus i))$ is a uniformly random bit for a randomly chosen $i \in \mathcal{I}$.

Given that $x_j \neq x'_j$ we assume WLOG that they differ in at least one of the first two bits. The first bit of $\mathsf{andreduce}(x_j \oplus i)$ is 1 if and only if the first two bits of i are the bitwise not of the first two bits of x_j, therefore it is 1 with probability 1/4. Similarly the first bit of $\mathsf{andreduce}(x'_j \oplus i)$ is 1 with probability 1/4 and they cannot both be 1 at once. Therefore they differ with probability 1/2. Further they are independent of all but the first two bits of i and thus of the rest of $\mathsf{andreduce}(x_j \oplus i)$ and $\mathsf{andreduce}(x'_j \oplus i)$. It follows that e_j is uniformly random. □

This construction can approach arbitrarily close to half an AND gate per bit of input by choosing b appropriately (see Table 5.1). However, suppose h is SHA3-256, which requires $\approx$35 AND gates per bit of input (and thus $C_N \approx 35$ in Table 5.1). Then for H to have less than one AND gate per bit we would have to take $b \geq 70$. This would give $|\mathcal{I}| \geq 2^{70}$. Recall that $|\mathcal{I}|$ directly corresponds to the size of our commitments, and thus $|\mathcal{I}| = 2^b$ is impractical in computation and communication/storage. Construction 3 sacrifices a small amount of collision resistance in order to reduce the size of $\mathcal{I}$.

Construction 3: main result. The digest function d_3 is shown in Figure 5.2 (right), and is analogous to d_2, but instead of using $i \in \mathcal{I} = \{0,1\}^b$ directly, we use a pseudo-random number generator prng to expand i into length b strings to xor with the blocks of x. This corresponds to re-interpreting the set of indices $\mathcal{I}$ as the set of seeds of prng. This replaces the need for $|\mathcal{I}| = 2^b$ from construction 2 by a much smaller $\mathcal{I}$ of size linear in b and λ.

Concretely, let $\mathsf{prng} : \mathcal{Q} \to \{0,1\}^b$ be a pseudo-random number generator, with an arbitrarily large keyspace $\mathcal{Q}$. Let an evaluation of prng on key i as $\mathsf{prng}(i,b)$ denote that the prng stretches the input to length b. For practical purposes we can think of $\mathcal{Q} = \{0,1\}^{128}$ if, for example, we instantiate prng with AES (in counter mode) with 128-bit keys. Let $\mathcal{I}$ be a subset of $\mathcal{Q}$. For construction 3 we define $d_3(i,x)$ to be $\mathsf{parity}(\mathsf{andreduce}(x_i \oplus \mathsf{prng}(i,b)))$.

Because we do not use a uniformly distributed mask on each block we do not get 1/2-collision boundedness. However, we can get arbitrarily close to that by increasing the size of $\mathcal{I}$. In particular $|\mathcal{I}|$ need only grow linearly in b, as shown in the following theorem.

Theorem 5.4.13. *If* prng *is a random oracle, then given any* $q > 1/2$*, there exists a choice of* $|\mathcal{I}|$ *such that with probability* $1-2^{\sigma}$ *(over the randomness of* prng*), construction 3 preserves* q*-collision boundedness. Specifically, it suffices to take*

$$|\mathcal{I}| \geq \frac{1}{2(q-1/2)^2}(\sigma + b + 1).$$

Proof. Unlike in the proof of Theorem 5.4.12 we will make use here of the fact that the hypothesis in Theorem 5.4.9 is only required to hold for sufficiently large λ.

Let $m = \mathsf{prng}(i,b)$. Here we must show that with all but negligible probability, for sufficiently large λ, $x_j \neq x'_j$ implies $e_j := d_2(x_j) \oplus d_2(x'_j) = \mathsf{parity}(\mathsf{andreduce}(x_j \oplus m)) \oplus \mathsf{parity}(\mathsf{andreduce}(x'_j \oplus m))$ is equal to one with probability at least $1-q$.

We will show that with the choice of $|\mathcal{I}|$ given in the statement, the above will hold with probability $2^{-\sigma}$.

Given $x_j \neq x'_j$, let $y = x_j$ and $y' = x'_j$ to avoid extra subscripts. Let y_l and m_l be the lth bit, with *one* indexing, of y and m respectively. Leaving AND implict (like multiplication) and using $\sum$ to denote XOR, we can rearrange the definition of e_j as follows.

$$e_j = \sum_{l=1}^{b/2} \Big(m_{2l-1} \left(y_{2l} \oplus y'_{2l}\right) \oplus \\ m_{2l} \left(y_{2l-1} \oplus y'_{2l-1}\right) \oplus y_{2l} y'_{2l-1} \oplus y_{2l-1} y'_{2l} \Big)$$

Let $v(y, y')$ be the vector with entries $y_l \oplus y'_l$ for all $l \in (1, ..., b)$ plus one entry containing $\sum_{l=1}^{b/2} y_{2l} y_{2l-1} \oplus y_{2l-1} y_{2l}$. Note that there are only 2^{b+1} possible values for v.

Now e_j is a function of v and m, we write it as $e(m, v)$. For a fixed value of v let $q(v)$ be the fraction of the key space for which $e(m, v) = 1$. Note that this value has distribution $\text{Bin}(|\mathcal{I}|, 1/2)$ with respect to the randomness of prng. Therefore by a Chernoff bound we have that $\mathbb{P}(q(v) < p) \leq e^{-2(q-1/2)^2|\mathcal{R}|}$. As there are 2^{b+1} possible values of v, a union bound over v yields $\mathbb{P}(\exists v \text{ s.t. } q(v) < p) \leq 2^{b+1} e^{-2(q-1/2)^2|\mathcal{R}|}$.

Rearranging, it follows that it suffices to take $|\mathcal{I}| \geq \frac{1}{2(q-1/2)^2}(\sigma + b + 1)$. □

One should think of the σ as a statistical security parameter, thus 40 would be a standard choice. The prng can be thought of as a fixed function on the set $\mathcal{I}$. We need this function to have the property that any lie v will be caught with probability at least p. This might not need to be a random oracle, but we cannot prove any fixed function works, thus we instead show that a randomly selected function works with high probability. However, it is important here that the randomness for the prng and the parameter b are not chosen adversarially. If they are then the result could still be recovered by increasing σ by however many bits of information about b and the output of prng the adversary was able to control. We will use $\sigma = 40$ when presenting our results.

It would be convenient if given a specific b and prng we could check whether the resulting construction preserves q-collision boundedness. Unfortunately, the problem of determining whether this is the case is as hard as the learning parity with noise problem [267], which is conjectured to be hard.

This does not rule out the idea of replacing the prng with a process that generates an output that is specially structured to guarantee preservation of q-collision boundedness. Indeed this is done in the analogous construction 4 of Section 5.10 over large fields. However we were unable to find such a construction in the binary case.

The expansion of prng requires $O(b)$ gates, the evaluation of h requires $O(n/b)$ gates, and f requires $n/2$ AND gates and $3n/2$ XOR gates as in construction 2. Thus by taking $b \approx \sqrt{n}$ the total cost is $n/2 + O(\sqrt{n})$ AND gates and $3n/2 + O(\sqrt{n})$ XOR gates.

However, in practice, as $O(n/b)$ is small compared to $n/2$ once b is moderately large we advise taking $b \approx \min(\sqrt{n}, 1024)$ so that for large n the size of $|\mathcal{I}| = O(\sigma + b)$ does not become prohibitive.

The choice of q is somewhat arbitrary but it is a trade-off between wanting something close to $1/2$ whilst not wanting $|\mathcal{I}|$ to be too large. Taking $q = 5/8$ is the compromise we work with.

With $\sigma = 40$, $q = 5/8$ and $b = 1024$ we have $|\mathcal{I}| = 34080$ indices. We will explore these values more in Section 5.6.

5.5 PVC Committed MPC from Indexed Hashes

In this section we introduce PVC commitments and the required properties for them to be secure, instantiate them using indexed hash functions, and propose a protocol for committed MPC with PVC security that directly leverages PVC commitments. We will start by defining what a PVC commitment scheme is, then we will explain how to construct one using a collision bounded indexed hash function. We will express the guarantees provided in a theorem and assess how the computational cost of the scheme depends on the indexed hash function. Throughout sk, pk is a public key pair belonging to the committing party that can be thought of as the identity of the input; it should only be used by one input. It is important this public key is associated to the committer (possibly by being signed with another key) by anyone to whom the verifier

wishes to prove cheating, e.g. a regulatory authority. The values $i \in \mathcal{I}$ and $r \in \mathcal{R}$ will be randomly chosen as inputs to provide security. For simplicity, we omit the security parameter λ in some of our statements, and when we say that an adversary cannot be successful at a task, we mean that they stand a negligible chance of doing so.

5.5.1 Definitions

We now define PVC commitments in terms of the three functionalities mentioned above.

Definition 5.5.1. *A PVC commitment scheme with covert security parameter $p \in [0, 1]$ consists of three functions* $\mathsf{pvccommit}$, assert, check, *the last of which is deterministic, satisfying four security properties defined below (correctness, general binding property with parameter p, hiding property, and defamation freeness).*

Let us first describe the form of the three functions $\mathsf{pvccommit}$, assert, and check:

1. A commitment function which commits to a value x,

$$c = \mathsf{pvccommit}(x, \mathsf{sk}, r).$$

2. An assertion function which is applied to the x we later wish to check was committed to,

$$a = \mathsf{assert}(x, \mathsf{sk}, r; i, \mathsf{pk}).$$

3. And a checking function, which interprets the output from the other two functions,

$$\mathsf{output} = \mathsf{check}(c, a, \mathsf{pk}).$$

With output satisfying $\mathsf{output} \in \{\mathsf{valid}, \mathsf{cheated}, \mathsf{inconclusive}\}$.

Intuitively, $\mathsf{output} = \mathsf{valid}$ means that the commitment opened to the expected value, $\mathsf{output} = \mathsf{cheated}$ means that the check did not pass because the committed and asserted values do not match, and $\mathsf{output} = \mathsf{inconclusive}$ denotes situations where the result of the verification is inconclusive because of a malformed message, or more

generally an abort by the committer. This latter situation cannot be avoided in general when evaluating PVC commitments in MPC, as a corrupted committer could send invalid messages or stop responding, similar to the role of aborts in MPC security with aborts. The first of the properties is correctness.

Definition 5.5.2 (Property 1: Correctness)**.** *For any* i, r, x *and valid key pair* sk, pk*, if* $c = \mathsf{pvccommit}(x, \mathsf{sk}, r)$ *and* $a = \mathsf{assert}(x, \mathsf{sk}, r; i, \mathsf{pk})$ *then* $\mathsf{check}(c, a, \mathsf{pk}) = \mathsf{valid}$.

The second is binding, a guarantee that a cheating committer will be caught with reasonable probability. P1 can avoid being caught cheating by refusing to sign anything, which is fine so long as they cannot possibly get a valid result either. Thus we require that they be caught with probability p only conditioned on the result not being inconclusive. A simple version of this is the following.

Definition 5.5.3 (Honest Binding)**.** *No polynomial time adversary can find* $x, \mathsf{sk}, r, x', \mathsf{sk}', r', \mathsf{pk}$ *such that (i)* $x \neq x'$ *and (ii) if* $i \leftarrow \mathcal{I}$*,* $c = \mathsf{pvccommit}(x, \mathsf{sk}, r)$*,* $a = \mathsf{assert}(x', \mathsf{sk}', r'; i, \mathsf{pk})$ *and* $\mathsf{output} = \mathsf{check}(c, a, \mathsf{pk})$ *then* $\mathbb{P}(\mathsf{output} = \mathsf{inconclusive}) < 1$ *and*

$$\mathbb{P}(\mathsf{output} = \mathsf{cheated} | \mathsf{output} \neq \mathsf{inconclusive}) < p$$

The above allows us to prove PVC security with parameter p only if the commitment is made honestly. If the commitment might be arbitrarily generated then we need the following strictly stronger version of binding. As this version is stronger it is the only one we include in the definition of a PVC commitment scheme, the previous definition will be referenced later in proofs though.

Definition 5.5.4 (Property 2: General Binding)**.** *No polynomial time adversary can find* $x, \mathsf{sk}, r, x', \mathsf{sk}', r', \mathsf{pk}$ *and* c *such that (i)* $x \neq x'$ *and (ii) if* $i \leftarrow \mathcal{I}$*,* $a = \mathsf{assert}(x, \mathsf{sk}, r; i, \mathsf{pk})$*,* $a' = \mathsf{assert}(x', \mathsf{sk}', r'; i, \mathsf{pk})$*,* $\mathsf{output} = \mathsf{check}(c, a, \mathsf{pk})$*, and* $\mathsf{output}' =$

$\mathsf{check}(c, a', \mathsf{pk})$ *then*

$\mathbb{P}(\mathsf{output} = \mathsf{inconclusive}) < 1$, $\mathbb{P}(\mathsf{output}' = \mathsf{inconclusive}) < 1$ *and*

$$\mathbb{P}(\mathsf{output} = \mathsf{cheated} | \mathsf{output} \neq \mathsf{inconclusive})$$
$$+\mathbb{P}(\mathsf{output}' = \mathsf{cheated} | \mathsf{output}' \neq \mathsf{inconclusive}) < p \tag{5.2}$$

To see this is stronger, note that if a scheme is not honestly binding the same counterexample but with $c = \mathsf{pvccommit}(x, \mathsf{sk}, r)$ will show it is not generally binding.

The final two properties prevent the verifier from cheating, so consider sk, pk to be fixed. It is useful to define an oracle $\mathcal{O}_{\mathsf{sk}}(x)$ which when called samples $r \leftarrow \mathcal{R}$ and returns

$$\mathsf{pvccommit}(x, \mathsf{sk}, r)$$

and

$$(\mathsf{assert}(x, \mathsf{sk}, r; i, \mathsf{pk}))_{i \in \mathcal{I}}\,.$$

The third property is the hiding property which guarantees the verifier cannot learn anything about x from the outputs of $\mathsf{pvccommit}$ or assert.

Definition 5.5.5 (Property 3: Hiding)**.** *For any x, x' and polynomial time adversary $\mathcal{A}$*

$$\mathbb{P}(\mathcal{A}(\mathcal{O}_{\mathsf{sk}}(x)) = 1) = \mathbb{P}(\mathcal{A}(\mathcal{O}_{\mathsf{sk}}(x')) = 1) + \mathit{negl}(\lambda).$$

The final property is defamation freeness which guarantees the verifier cannot frame an honest committer.

Definition 5.5.6 (Property 4: Defamation Freeness)**.** *No polynomial time adversary can choose an x and then when given $\mathcal{O}_{\mathsf{sk}}(x)$ find c and a such that*

$$\mathsf{check}(c, a, \mathsf{pk}) = \mathsf{cheated}$$

Note it is important that each secret key is only used for one choice of x, r. This could be achieved by deriving the secret key from (x, r) by a one way function (possibly with extra randomness).

5.5.2 PVC commitment from indexed hashes

Let H be an indexed hash function with index space $\mathcal{I}$ and randomness space $\mathcal{R}$. Let $m_{\mathsf{sgn(sk)}}$ denote m together with a signature of m by secret key sk. Consider the following three functions.

Let H be an indexed hash function with index space $\mathcal{I}$ and randomness space $\mathcal{R}$. Consider the following three functions.

$$\mathsf{pvccommit}(x, \mathsf{sk}, r) = ((H(i, r, x))_{i\in\mathcal{I}})_{\mathsf{sgn(sk)}}$$

$$\mathsf{assert}(x, \mathsf{sk}, r; i, \mathsf{pk}) = \begin{cases} (i, H(i, r, x))_{\mathsf{sgn(sk)}} & \text{if } (\mathsf{sk}, \mathsf{pk}) \text{ is a valid keypair} \\ \perp & \text{otherwise} \end{cases}$$

For check let G be the event that the signatures are valid.

$$\mathsf{check}(c, a, \mathsf{pk}) = \begin{cases} \mathsf{valid} & \text{if } G \text{ and } c[a[0]] = a[1] \\ \mathsf{cheated} & \text{if } G \text{ and } c[a[0]] \neq a[1] \\ \mathsf{inconclusive} & \text{Otherwise} \end{cases}$$

We require one slightly unusual property of the signature scheme. This is a technicality, as (a) lots of schemes have this property and (b) in the next subsection we will introduce a computational optimization which has the side effect of guaranteeing this property from any scheme.

Definition 5.5.7. *Call a signature scheme* discrimination resistant *if no polynomial time adversary can find m, m', sk and pk, ($(\mathsf{sk}, \mathsf{pk})$ not necessarily a valid key pair), such that $m_{\mathsf{sgn(sk)}}$ and $m'_{\mathsf{sgn(sk)}}$ are valid with non-negligibly different probabilities.*

We also require that the signature scheme has a deterministic verification function. This could be lifted at the expense of complicating the definitions with extra negligible terms. However, whilst not implied by the definition of a signature scheme, all the most popular schemes satisfy this assumption so we will make it for simplicity.

Theorem 5.5.8. *If H is hiding and q-collision bounded and the signature scheme has deterministic verification and is discrimination resistant, then the above functions form a PVC commitment scheme with covert security parameter $p = 1 - q$ (Definition 5.5.1).*

Proof. To show that the functions pvccommit, assert and check for a PVC commitment scheme with parameter p, we must check that check is deterministic and that the four properties hold.

As the function check is given by a decision tree depending on checking whether (deterministic) parts of the input are equal and whether signatures are valid (which is deterministic by the assumption on the verification function) it is deterministic.

Correctness Given i, r, x and a valid key pair sk, pk, let $c = \mathsf{pvccommit}(x, \mathsf{sk}, r)$ and $a = \mathsf{assert}(x, \mathsf{sk}, r; i, \mathsf{pk})$. Consider the definition of $\mathsf{check}(c, a, \mathsf{pk})$. As the key pair is valid both of the signatures will check out thus the result is not inconclusive. Furthermore, both $c[a[0]]$ and $a[1]$ are equal to $H(i, r, x)$, thus the check will return valid.

General Binding Suppose these functions do not satisfy general binding. Then there exists a polynomial time adversary, $\mathcal{A}$, contradicting Definition 5.5.4. Let $x, \mathsf{sk}, r, x', \mathsf{sk}', r', \mathsf{pk}$ and c be the output of this adversary. Further, let $i, a, a', \mathsf{output}$ and output' be as in the definition. As the signature scheme is discrimination resistant the distribution of i conditioned on G (and thus on output or output' being inconclusive) is still uniform. It follows that in order for Inequality 5.2 to hold we must have

$$\mathbb{P}(H(i, r, x) \neq c[i]) + \mathbb{P}(H(i, r', x') \neq c[i]) < p.$$

Thus for greater than a $1 - p$ fraction of the choices of i we must have $H(i, r, x) = c[i] = H(i, r', x')$. This would mean that r, x, r', x' is a q'-collision for some $q' > q$. The above process then gives a polynomial time algorithm contradicting the q-collision boundedness of H. So the general binding property must hold.

Hiding Suppose $\mathcal{A}$ is a polynomial time adversary contradicting Definition 5.5.5. Consider the polynomial time algorithm that takes as input $(H(i, r, x))_{i \in \mathcal{I}}$, computes $\mathcal{A}(\mathcal{O}_{\mathsf{sk}}(x))$ (using a hard-coded sk) and outputs the result. This adversary contradicts the hiding property of H (Definition 5.4.5).

Defamation Freeness Let $\mathcal{A}$ be a polynomial time adversary. In order to have $\mathsf{check}(c, a, \mathsf{pk}) = \mathsf{cheated}$ both c and a must be correctly signed. As the signature scheme is chosen-plaintext secure $\mathcal{A}$ can only achieve this with non-negligible probability by using the contents of $\mathcal{O}_{\mathsf{sk}}(x)$ as c and a (they cannot even be switched as they have different formats). But with that choice of c and a, $c[a[0]] = a[1]$, and thus the check would return valid. Therefore the functions are defamation free. □

5.5.3 PVC Committed MPC from a PVC commitment scheme

In this section we define formally PVC committed MPC, for the two party case, and propose protocols to efficiently realize this functionality, which corresponds to the intuitive idea from Figure 5.1.

We follow the definitions by Asharov and Orlandi [256] to prove PVC security of our protocols. This involves proving (i) simulatability (in the ideal vs. real worlds framework) for the covert security part, along with (ii) accountability and (iii) defamation freeness for the public verifiability. For (ii) and (iii) we use the definitions by Asharov and Orlandi and for (i) our ideal world is presented in detail in Appendix A.1 as an extension of theirs, to handle the commitment phase. Without loss of generality, we describe our ideal world for only two parties `P1` and `P2`. Moreover, as in our protocols, the first party gets malicious security, while the second party gets PVC security. This matches the guarantee in the generic PVC protocol by Wang [268] that we use in the experimental evaluation.

Our ideal world is parameterized by two values $p_{\texttt{exec}}, p_{\texttt{commit}} \in [0, 1]$ denoting lower bounds on the probabilities with which `P1` can get caught when (i) cheating

in the protocol execution and (ii) breaking the commitment, respectively. Note that Asharov and Orlandi only formalize (i), and they denote $p_{\texttt{exec}}$ as ϵ. Moreover, our ideal world is parametrized by an arbitrary distribution $\mathcal{E}$ which we refer to as *the environment* (this is similar to the notion used in the UC framework). A sample from the environment is included in the parties' view as an auxiliary input that is received only *after* the commitment phase has finished. This limits the ability of the ideal world adversary (the simulator) to rewind the adversary beyond the commitment phase (similar to the role of the environment in UC), and models information that the adversary might get after committing.

We summarize the ideal world execution next. First, P1 receives its prescribed input and commits to it (if honest) or an arbitrary value (if corrupted) by sending it to the trusted party. This constitutes the commitment phase, and captures the situation where P1 commits to using an input, e.g., an ML model, to be used at an undetermined time in the future in a secure computation with a second party P2. Then, party P1 receives an input from the environment, in the form of a sample from $\mathcal{E}$, which is also given to P1 in the real world, as explained above. This determines the beginning of the secure computation phase, which starts with P2 receiving its prescribed input and with P1 notifying the trusted party of their desire to cheat in the execution. This attempt will succeed with probability $1 - p_{\texttt{exec}}$, in which case P1 gets to completely break the protocol, i.e. learn P2's input and choose their output. If P1 fails, P2 receives output `corrupted`. If a corrupted P1 decided to not cheat in this way, they still get a chance to cheat in switching the input of the secure computation from the committed value w to a different one. If this attempt fails (which happens with probability at least $p_{\texttt{commit}}$), P2 gets notified. For simplicity in the presentation we allow P1 to abort after receiving their output, and before P2 gets to observe theirs, but this assumption can be lifted by ensuring that in the underlying PVC protocol P2 gets the output first.

Let $(\mathsf{pvccommit}, \mathsf{assert}, \mathsf{check})$ be a PVC commitment scheme with parameter p. Let $\texttt{Blame}_{\texttt{commit}}$ be the function which when given a view of P2 (honestly) running the

Public Parameters: A PVC commitment scheme (Definition 5.5.1) and a public key pk.
Inputs: input x and secret key sk matching pk.
Outputs: Commitment c.

Algorithm:

1. Sample $r \leftarrow \mathcal{R}$.
2. Compute $c = \mathsf{pvccommit}(x, \mathsf{sk}, r)$.
3. Store r as a secret and return c.

Figure 5.3: PVC Committed 2PC (commitment algorithm).

Parties: P1, P2.
Public Parameters: A PVC commitment scheme (Definition 5.5.1), a commitment c, and a public key pk.

The protocol uses a PVC secure protocol Π offering PVC security to P2 and malicious security to P1.
Inputs: P1: x, r; P2: y.
Outputs: $\mathtt{P1} : g_1(x, y)$; $\mathsf{P}_2 : g_2(x, y)$, or a proof of cheating a.

Protocol:

1. P2 samples $i \leftarrow \mathcal{I}$.
2. $\mathtt{P1}, \mathsf{P}_2$ run Π to compute $(o_1; o_2, a) = (g_1(x, y); g_2(x, y), \mathsf{assert}(x, \mathsf{sk}, r; i, \mathsf{pk}))$.
3. P2 computes $\mathsf{output} = \mathsf{check}(c, a, \mathsf{pk})$ and
   If $\mathsf{output} = \mathsf{valid} \longrightarrow$ accepts o_2 as $g_2(x, y)$.
   If $\mathsf{output} = \mathsf{cheated} \longrightarrow$ accepts a as proof of cheating.
   Otherwise $\longrightarrow$ aborts and sets result to *inconclusive*.

Figure 5.4: PVC Committed 2PC for functionality $g(x, y) = (g_1(x, y), g_2(x, y))$ (integrity check).

protocol in Fig. 5.4, in which $\mathsf{output} = \mathsf{cheated}$ returns the commitment c and the resulting a and otherwise returns $\perp$. Let $\mathtt{Judgement}_{\mathtt{commit}}$ be the function check with the public key of P1 hard coded. Let Commit be the commitment algorithm in Fig. 5.3 and $\mathcal{P}$ be the protocol in Fig. 5.4, with Π instantiated with the protocol of Wang [268]. Finally, let $\mathtt{Blame}_{\mathtt{exec}}$ and $\mathtt{Judgement}_{\mathtt{exec}}$ be the blame and judgement functions from Π, and define $\mathtt{Blame}(x)$ to be $\mathsf{cheated}$ if either $\mathtt{Blame}_{\mathtt{exec}}(x)$ or $\mathtt{Blame}_{\mathtt{commits}}(x)$

Parties, inputs, outputs, and public parameters are as in Figure 5.4, and the PVC commitment scheme is instantiated by an indexed hash function H (as in Theorem 5.5.8).

Protocol:

1. P2 samples $i \leftarrow \mathcal{I}$ and $\tilde{r} \leftarrow \mathcal{R}$.
2. P1, P2 run Π to compute $(g_1(x,y), h(m|\tilde{r}); g_2(x,y), m)$, where $m = (i, H(i,r,x))$.
3. P1 computes $s = \mathsf{sign}(h(m|\tilde{r}), \mathsf{sk})$ and sends it to P2.
4. P2 aborts if s is not the valid signature of $h(m|\tilde{r})$.
5. P2 computes $\mathsf{output} = \mathsf{check}(c, a, \mathsf{pk})$ and
   If $\mathsf{output} = \mathsf{valid} \longrightarrow$ accepts o_2 as $g_2(x,y)$.
   If $\mathsf{output} = \mathsf{cheated} \longrightarrow$ accepts a as proof of cheating.
   Otherwise $\longrightarrow$ aborts and sets result to *inconclusive*.

Figure 5.5: PVC Committed 2PC for functionality $g(x,y) = (g_1(x,y), g_2(x,y))$ (Optimized integrity check).

equals cheated, and analogously for a function Judgement. We are now ready to state our main result (Proof in Appendix A.2).

Theorem 5.5.9. *The quadruple* $(\mathtt{Commit}, \mathcal{P}, \mathtt{Blame}, \mathtt{Judgement})$ *securely computes* g *with committed first input in the presence of a malicious* P1 *or a covert* P2 *with* $p/2$*-deterrent and public verifiability.*

If pvccommit *and* check *are used as given in the previous section then we can replace the* $\mathcal{P}$ *with the protocol in Fig. 5.5 and still have the same security guarantee.*

Furthermore, if in either case it can be guaranteed that P1 *is honest in running the commitment algorithm in Fig. 5.3, then the deterrent factor improves from* $p/2$ *to* p*.*

Non-committed output at no risk. P1 can in the above ideal world, and thus in the protocol, get $g_1(x', y)$ for a non-committed x' at no risk by aborting afterwards. This could be avoided by opening up the PVC blackbox and holding back this output until P2 has checked the result of assert (or optimized equivalent).

	SHA3-256	LowMCHash-256	Ours		
No. of bits	# of ANDs	# of ANDs	# of ANDs	Improvement over SHA3-256	Improvement over LowMCHash-256
2^{14}	6.14×10^5	2.32×10^5	5.17×10^4	12×	4×
2^{18}	9.29×10^6	3.65×10^6	1.90×10^5	49×	19×
2^{22}	1.48×10^8	5.84×10^7	2.29×10^6	65×	25×
2^{26}	2.37×10^9	9.34×10^8	3.42×10^7	69×	27×
2^{30}	3.79×10^{10}	1.49×10^{10}	5.39×10^8	70×	28×

Table 5.2: Analytical comparison of the number of AND gates (circuit size $|\mathcal{C}|$) for the assert functionality using LowMCHash-256 and SHA3-256 with our scheme. These values are for a single call to assert i.e. using a single index for our scheme. Here $p_c = 1/2$.

Computational costs. The cost of the commit operation in the clear is computing H, $|\mathcal{I}|$ times. The cost of the assert is dominated asymptotically by the cost of computing H once i.e. requires $n/2 + O(n)$ AND gates. The check are $O(1)$ and relatively very cheap.

5.6 Evaluation

In this section we compare our method for committed MPC to the baseline using SHA3-256. We evaluate both computation time and communication for the assert functionality as the size of the input n increases. We also analytically compare our method against the hash function based on LowMCHash-256, an MPC friendly hash [265]. Finally, we evaluate the practicality of our proposed scheme in terms of the compute requirement for the committer performing the commitment using the pvccommit functionality and the size of the commitment. As a result, we show, for our scheme: (a) Verification (assert functionality) in MPC is significantly faster than optimized standards such as SHA3-256 as well as MPC optimized hashes such as LowMCHash-256, (b) the size of the commitment is practical and (c) the computation required from the committer (pvccommit functionality) is practical. We use the circuit sizes and real experimental data for (a). Similarly, we analyze the size of the commitment to prove (b) and use actual computation time data to show (c). We begin by describing the experimental and implementation details.

Experimental Settings. The experiments were executed on two Microsoft Azure D32s v3 machines running Ubuntu 16.04, equipped with Intel Xeon E5-2673 v4 2.3GHz processors and 128 GB RAM. The machines were hosted in the same region with a bandwidth of 1.7 GB/s and an average latency of 0.90ms. This arrangement was chosen to be representative of a LAN setting.

Implementation. We use the EMP-toolkit [130] to implement our secure protocols as well as the baselines. In particular, we use the covert-security-with-public-verifiability framework of Hong et al. [269], which makes use of garbled circuits. We set the covert security parameter p_c of this underlying implementation to 1/2. Note this is different from the covert security parameter p used in our scheme. Since $p \leq 1/2$, p_c could be set to 1/2. As one could infer, the effective covert security parameter for our scheme with this implementation would be $min(p, p_c)$.

Baselines. We use two baselines for comparison: SHA3-256 and LowMCHash-256. For SHA3-256, we use the sponge framework [270] with an input block size of 1600. Using the security parameters defined in the standard we get the rate as 1088 and the capacity as 512. This results in a computation cost of ~ 35 AND gates per input bit. For LowMCHash-256, we use LowMC permutations together with the sponge framework using an input block size of 512. We reserve 256 bits for the rate and another 256 bits for the capacity (128 bit security). This results in ~ 14 AND gates per input bit. LowMC is relatively new and has been shown to be susceptible to attacks [264]. However, we include it in this comparison, it being one of the most MPC optimized hashing schemes for Boolean circuits. Both these baselines are implemented on the top of the EMP-toolkit's PVC framework.

Our scheme. For our scheme we implement the idea around PVC commitment from indexed hashes as described in the previous section. We use Construction 3 defined in Section 5.4. In particular we implement the assert functionality in MPC (and pvccommit, check in the clear). Our scheme costs ~ 0.5 AND gate per input bit.

Description	Symbol	Value
Our Indexed Hash Function		
Length of the input ($\|x\|$)	n	no. of bits (variable)
Pseudorandom number generator	$prng$	AES (counter mode)
Underlying collision resistant hash	h	SHA3-256
Statistical security parameter	σ	40
Collision boundedness parameter	q	5/8
Block size	b	$min(\sqrt{n}, 1024)$
Our PVC Commitment		
Covert security parameter	p	$1 - q = 3/8$
Underlying PVC 2PC Protocol (EMP-PVC) For assert		
Covert security parameter	p_c	1/2

Table 5.3: Parameters used in our experiments.

In order to effect signed public verifiability during the assert phase, we use SHA3-256 to commit the hash corresponding to the input index and the index. We summarise the parameters for our scheme in Table 5.3. Note that the table reports the covert security parameter of the commitment scheme for the honest committer case. In the general case this parameter's value would be $p/2 = 3/16$. Similar to the baselines, our scheme is also implemented on top of the EMP-toolkit's PVC framework.

5.6.1 Analytical Performance

Table 5.2 compares the circuit size $|\mathcal{C}|$ (no. of AND gates) for the assert functionality for the LowMCHash-256 and SHA3-256 baselines with our scheme. As we increase the size of the input, the scheme starts to show its full potential. For a small input size, the initial overhead of signing the commitments and the index tends to shadow the improvement. But as we increase the size of the input, we can see a marked 70× improvement over SHA3-256 and 28× improvement over LowMCHash-256. We show that these improvements directly translate into real world experiments, when compared against the actual implementation of SHA3-256, in Section 5.6.2.

No. of bits	Ours (s)	SHA3-256 (s)	Improvement
2^{14}	0.07	0.57	8×
2^{18}	0.22	8.16	36×
2^{22}	2.67	133.23	50×
2^{26}	39.14	2200*	56×
2^{30}	590.70	35500*	60×

Table 5.4: Comparison of running time for SHA3-256 baseline and our scheme executing the assert functionality. Here $p_c = 1/2$. * means estimated via extrapolation.

5.6.2 Experimental Performance

Running time for assert. Table 5.4 shows the running time for executing the assert functionality to verify the commitments using SHA3-256 and our scheme. As we increase the size of the input to practical sizes, we observe that our scheme is 60× faster than the SHA3-256 baseline. This is directly correlated with the 70× improvement in the circuit sizes above. We do not perform actual experiments with LowMCHash-256, but it is similarly expected to be around 25× slower than our scheme as indicated by the circuit sizes. Also, in practice, nothing prohibits us from replacing our underlying collision resistant hash h with LowMCHash to amplify this improvement. We compare the following:

Communication for assert. Table 5.5 shows the amount of communication needed for executing the assert functionality using SHA3-256 and our scheme. We observe that our scheme requires 36× less communication for the committer and the verifier than the SHA3-256 based baseline.

Computation load for pvccommit. In Table 5.6, we show the number of indices $|\mathcal{I}|$ for the commitment that needs to be computed alongside the size of the entire commitment that a committer needs to prepare in order to commit its input. In these experiments $p = 3/8$ ($q = 5/8$) and block size $b = min(\sqrt{n}, 1024)$. The size of the commitment results in a very limited communication and space requirement. Block size limit of 1024 bits, limits the size of the commitment to just 1.09 MB. We use the

No. of bits	Ours (MB)	SHA3-256 (MB)	Improvement
2^{14}	2.51	19.93	8×
2^{18}	10.90	300.34	28×
2^{22}	141.25	4805.39	34×
2^{26}	2169.02	76900*	35×
2^{30}	34022.77	1230200*	36×

Table 5.5: Comparison of communication for SHA3-256 baseline and our scheme for executing the assert functionality. Here $p_c = 1/2$. * means estimated via extrapolation.

formulation, upon ceiling to the next nearest integer, defined in Theorem 5.4.13 to compute $|\mathcal{I}|$. In Figure 5.6, we plot this formulation for $\sigma = 40$, $b = 1024$ and different values of q (and the covert security parameter p i.e $1 - q$) to show how the number times $|\mathcal{I}|$ that the committer needs to compute H varies with the security parameters.

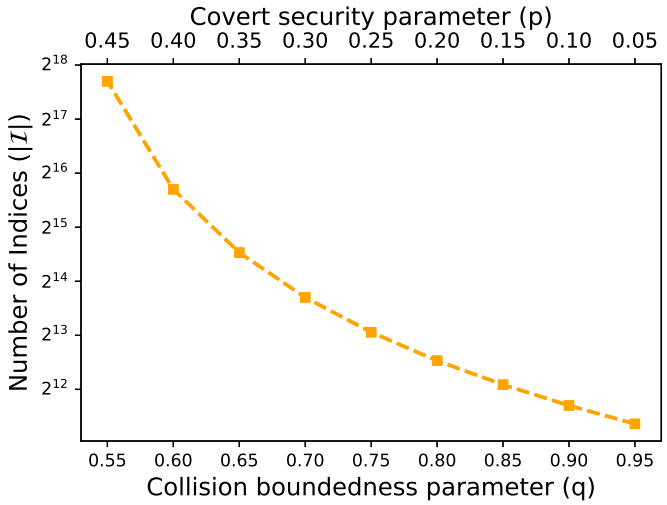

Figure 5.6: Number of indices (hashes) $|\mathcal{I}|$ needed to be computed by the committer as a function of q (and the covert security parameter p i.e. $1 - q$). Here block size $b = 1024$ and statistical security parameter $\sigma = 40$.

In Table 5.6 we also show the computation load of the committer for committing its input. In particular, we evaluate the time needed to perform the pvccommit functionality. This only needs to be performed once for a given input, in the clear. We see that the computation load is very limited even for large input sizes. For these results we use only a single process, however this computation is trivially parallelizable. Several

No. of bits	# of Hashes	Size of the Commitment (MB)	Time
2^{14}	5408	0.17	1.61s
2^{18}	17696	0.57	28.21s
2^{22}	34080	1.09	5.06m
2^{26}	34080	1.09	36.92m
2^{30}	34080	1.09	5.91h

Table 5.6: Computation time and size of the commitment using our scheme for executing the pvccommit functionality. Here $p = 3/8$, statistical security parameter $\sigma = 40$ and block size $b = min(\sqrt{n}, 1024)$ where n is the size of input.

	Accuracy			Average Odds Difference		
Dataset	Unfair	Fair	Changed	Unfair	Fair	Changed
Credit	69.3%	62.7%	64.3%	−0.341	0.359	0.122
COMPAS	67.6%	55.9%	65.0%	−0.181	0.369	0.073
Adult	80.4%	75.8%	78.8%	−0.270	0.261	0.111

Table 5.7: Accuracies of a fair prediction method [271] (Fair), the same model changed by a single weight to maximize accuracy (Changed), compared to the model trained without any constraints (Unfair).

hashes can be computed in parallel. For example a 128 threaded implementation should enable the committer to commit 2^{30} bits in less than 3 minutes. Furthermore, we perform these computations in Python using standard libraries and there is scope for further significant optimization by using a low-level language.

5.7 Certified Predictions

In this section we describe how PVC committed MPC enables a key application, *certified predictions*: obtaining secure predictions by a private model that is certified to have certain properties (more on such properties below). We show by means of a real-world example how heuristic approaches that are sublinear in the input size fail. We do this by training a fair machine learning model and showing how, by modifying a single parameter of the model, it can be made unfair and more accurate. We describe previous work on the problem of obtaining predictions by a certified private model, and discuss an efficient solution enabled by our results.

Recently, ML models have started to be deployed into high-impact, real-world decision-making settings such as medicine [272], self-driving cars [273], and college admissions [274]. However, this has led to problems: many of these settings have key constraints that ML models were not originally designed to handle. Current models lack *interpretability* [275], *safety* [276], and *fairness* [277]. To address this, there has been a wealth of recent work aimed at formalizing these constraints and creating ML models that satisfy them [271, 278–281].

However, models that satisfy these constraints often have reduced accuracy as the constraints restrict the model's predictions in accuracy-agnostic ways. As model accuracy is often directly tied to beneficial outcomes (e.g., monetary investment, company profit, likelihood of publication), real-world constraints can incentivize service providers to cheat. To prevent this a natural question arises: *What is the minimal computation required to ensure cheating does not occur?* One may be tempted to try to construct a procedure that is sublinear in the size of the model. Recent work has proposed to generate a small series of tests to identify small changes to a model [282]. However, we show with a simple example that any protocol must ensure that nothing about the model is changed, requiring a linear time procedure.

Any change may sacrifice fairness. We investigate a popular real-world constraint placed on ML models: *fairness constraints*. In general, the most popular formulation of fairness constraints minimizes the difference between (functions of) predictions made on different demographic groups. Because these techniques constrain predictions across groups, their accuracy is less than unconstrained models. We investigate a popular fair prediction model [271] applied to three fair prediction problems: judging credit risk (Credit[3]); predicting parole violators (COMPAS[4]); inferring income

[3] `https://tinyurl.com/cm-credit`
[4] `https://tinyurl.com/cm-compas`

(Adult[5]). We consider the following *average odds difference* fairness criterion

$$\Big(\mathbb{E}[\hat{Y} \mid A=0, Y=y] - \mathbb{E}[\hat{Y} \mid A=1, Y=y]\Big) \geq \tau, \quad \forall y \in \{0,1\},$$

where Y is the true outcome (e.g., $Y=1$ signifies good credit in Credit, while $Y=0$ signifies bad credit) and $\hat{Y}$ is the prediction. Here A indicates demographic group (e.g., race, gender, sexual orientation, among others). Specifically $A=0$ indicates a *disadvantaged group* and $A=1$ indicates a *privileged group*. Thus the above constraint says that the average outcome for the disadvantaged group has to be at least τ-larger than the average outcome for the advantaged group. This is to combat predictors $\hat{Y}$ that benefit the privileged group (such predictors will arise from unconstrained training). These expectations are computed over a training dataset. Table 5.7 shows the accuracy and average odds difference of the model in Zhang et al. [271] using the fairness constraint (Fair), compared to the model without the fairness constraint (Unfair).

Now we imagine that a cheating service provider wants to take the fair model and only change a single element of the model to maximize accuracy. We imagine they test every single element, optimizing for accuracy alone, while fixing the remaining parameters. They then take the model which has the maximum improvement in accuracy across all single-parameter-changed models (Changed). We report the accuracy and fairness of this model in Table 5.7.

These results show that changing just a single element can significantly improve the accuracy over the fair model (by as much as 9.1% on COMPAS). Further, the changed model has significantly lower average odds difference than the fair model, unfairly benefiting the privileged group at the expense of the disadvantaged group. Thus, to ensure a service provider cannot surreptitiously improve accuracy at the expense of real-world constraints, a protocol must ensure that the entire model remains the same.

Related work. To prevent this, a number of works have proposed techniques to verify ML models [195, 255, 282, 283]. SafetyNets [283] propose an interactive

[5] `https://tinyurl.com/cm-census`

proof protocol for verifying deep neural network predictions. This protocol only has a verification guarantee and leaves a security guarantee to future work. Further it is limited to models expressible as arithmetic circuits. VerIDeep [282] describe a method to generate inputs for which small changes to the ML model would yield very different outputs. However, this model does not guarantee that the entire model remains the same and thus would be vulnerable to attacks similar to that described above.

Recent work with security guarantees [195, 255] propose to use hash functions (SHA-256, SHA-3 in sponge mode) to verify a model has not been altered. Specifically these works generate and verify a hash within MPC. In MPC the protocol cost is dominated by AND gate computations and the most efficient method requires asymptotically 35 AND gates per input bit [255]. While there exists an MPC-optimized hash called LowMCHash-256 [265], it is new and susceptible [264]. Our constructions above enable secure predictions with verified inputs that asymptotically require 0.5 AND gates per input bit and derives security from the well-known random oracle assumption.

Our approach. To enable certified predictions we propose the following procedure. First the service provider (committer, P1) makes a commitment $c = \mathsf{pvccommit}(x, \mathsf{sk}, r)$ to a model x. P1 then engages in an MPC protocol with a regulatory agency (P2') where P2' verifies the model x satisfies the required guarantee (e.g., fairness), and that c is a commitment to that model. If these checks pass then P2' signs the commitment c with their private key and sends it to P1. When a user (verifier, P2) wishes to obtain a certified prediction from P1, they engage in a PVC commitment. Here P1 sends c to P2. If (a) P2 can verify that c is signed by the regulatory agency P2' (e.g., this could be done if regulator's public key is publicly available) and (b) the PVC commitment is verified (via assert and check as described in Figure 5.1), then the output is a certified prediction.

5.8 Lower Bounds

In this section we provide lower bounds on how many AND gates are required for a collision resistant indexed hash function and an ordinary hash function. Recall that our construction 3 from Section 5.4 requires half an AND gate per bit of input. In this section we show that

1. Construction 3 is optimal amongst hash functions whose output's size is sublinear w.r.t. their input's size (Corollary 5.8.1).

2. For ordinary hash functions we show that every collision resistant hash function requires at least one AND gate per input bit (Proposition 2).

3. Assuming that we want our hash functions to be hiding, we show, both for indexed and ordinary hash functions, that allowing their output to be large does not help much to reduce the number of required nonlinear gates (Proposition 3).

Moreover, although we state the above results in terms of Boolean circuits, it is not hard to see that the arguments extend to any field. The following lemma and corollary correspond to item 1 above. The proof, given in Appendix A.2, constructs an algorithm to find a 1-collision on any H with small set of nonlinear gates by casting that problem as that of solving a linear system S on $\mathbb{F}_2$, and showing that S always has a solution. Recall that indexed hash functions have three inputs i, r, x, in the statement by *main input* we mean x.

Proposition 1. *Given any non-trivially collision bounded family of indexed hash functions $\{H_k\}_{k\in K}$ with H_k given by the (polynomial size) circuit C_k with n-bit main input, and m-bit output. With all but negligible probability over the generation of $k = G(\lambda)$, the circuit C_k must have at least $\lceil (n-m)/2 \rceil$ nonlinear gates.*

Note that in practice the lower bound on the nonlinear gate count will apply (with all but negligible probability) for any λ large enough to be considered secure. In particular we have the following corollary which says that, in order to beat our constructions asymptotically, an indexed hash function must have large output.

Corollary 5.8.1. *Any family of covertly collision resistant hash function circuits, indexed by n, with main input in $\{0,1\}^n$ must either have at least $n/2+O(n)$ nonlinear gates or must have output size that is not $O(n)$.*

A stronger result can be achieved in the case of an ordinary secure hash function, by relying on the fact that they do not take auxiliary inputs. The idea of the proof is similar to that of Proposition 1.

Proposition 2. *Let $\{h_k\}_{k\in K}$ be a collision resistant family of hash functions with h_k given by the circuit C_k with n-bit input and m-bit output. With all but negligible probability with respect to the generation of $k = G(\lambda)$, the circuit C_k must have at least $n - m$ nonlinear gates.*

These results show that our constructions have asymptotically half the verification cost of the baseline with any ordinary secure hash function. However, recall that we designed our construction 3 for the output of H to be small, i.e. $O(n)$, for *efficiency and not security reasons*. One may thus wonder whether dropping this requirement allows to significantly overcome the above lower bounds. Next, we show that the answer is negative by leveraging the fact that we do require a hiding property *for security*, which we show implies a linear lower bound on the required number of AND gates.

The proof of the following result can be found in Appendix A.2. It relies on the fact that if you have a small number of AND gates then only a small amount of the entropy in the randomness can affect their inputs. The rest of the randomness cannot be used for hiding the output without giving too much leeway for finding collisions. Thus only a small amount of randomness and a small number of output

wires from AND gates can hide the output. Thus the output must be effectively small and the above propositions can be applied.

Proposition 3. *Suppose that $\{H_k\}_{k\in K}$ is a non-trivially collision bounded and hiding family of hash functions. Let H_k be given by C_k with an n-bit main input and d nonlinear gates, then with all but negligible probability, $d \geq n/5$. Further if $|\mathcal{I}| = 1$, then $d \geq n/3$.*

5.9 From Covert Security to Malicious Security

A natural idea is to amplify the statistical guarantee of an indexed hash function H by computing it at several indices. This would in turn lead to a PVC commitment scheme with improved parameters where H is run on several indices. More concretely, given a collision resistant indexed hash function H we can compute an indexed hash function H^κ with stronger security by computing H multiple, say κ, times with different indices. Formally, with $i_j \in \mathcal{I}$ for $j \in \{1, ..., \kappa\}$

$$H^\kappa((i_j)_{j=1}^\kappa, r, x) = (H(i_j, r, x))_{i=1}^\kappa . \tag{5.3}$$

This new function requires no more hashes to be prepared by the committer and, if $\{H_k\}_{k\in K}$ is q-collision bounded, then $\{H_k^\kappa\}_{k\in K}$ is q^κ-collision bounded.

5.10 Arithmetic Circuits

We have mainly focused on binary circuits because they are more flexible and there are more reasonably fast hash functions for them. However our main idea will also work to construct indexed hash functions to be computed in arithmetic circuits. As before our constructions are in terms of a secure hash function h which could be implemented using MiMC [284] or any other arithmetic circuit hash function. We will assume this arithmetic is in a field $\mathbb{F}$.

Analogues of constructions 2 and 3 would work in this setting with XOR and AND gates replaced by ADD and MUL gates. Indeed, these would also work, with worse parameters, over arbitrary rings. These can be analysed analogously and relevant theorems deduced. However we will not detail these changes here and will instead provide a further development that was not possible in the binary case.

The idea of construction 4 presented in this section is much like the analogue of construction 3, however instead of using a prng to generate the random masks to be added to index, we will generate them in a more structured fashion. Hence, construction 4 still follows the blueprint given in Equation 5.1. The index space $\mathcal{I}$ will be a subset of $\mathbb{F}$, this requires the field to be moderately large and rules out this construction in the binary case.

As in Section 5.4, we have an even block size parameter b, and define the indexed hash by means of a digest function d_4 that takes an index i and b field elements as input and returns a single field element. Given y a fixed block of b elements denoted by $y_1, ..., y_b$,

$$d_4(i, y) = \sum_{j=1}^{b/2} (i^{2j-1} + y_{2j-1})(i^{2j} + y_{2j}) \tag{5.4}$$

The value of the hash is given by $\mathcal{C}_4(h, \lambda)(i, r, x) = h(r||i||\texttt{process}_{b,d_4)}(i, x))$ given functions process and h, as described in Equation 5.1 and Section 5.4.

The following theorem states the guarantee of construction 4. While its full proof is given in Appendix A.2, the basic idea is that there will be a collision so long as some vector determined from x and x' is not perpendicular to $(1, i, i^2, ..., i^b)$. The powers of i come from the definition of d_4 and have been chosen (to replace the prng) so that these vectors form Vandermonde matrices, thus any $b + 1$ of them span and so at most b are perpendicular to any given vector.

Theorem 5.10.1. *Construction 4 is $b/|\mathcal{I}|$-collision bounded.*

Note that this construction only works for fields larger than the block size b, but this is the case for a lot of standard hashes based in field arithmetic. If the field is

very large then the covert security parameter can be made ≈ 1 by taking $|\mathcal{I}|$ to be big. However this would be very impractical to prepare the hashes, and thus in that case it would be more practical to combine construction 4 presented in this section with the amplification ideas from Section 5.9.

5.11 Conclusion

We propose an MPC-friendly commitment mechanism to effect certified predictions in 2PC publicly verifiable covert (PVC) setting. The standard simulation-based security definitions used in MPC allow a malicious adversary controlling one of the parties to provide arbitrary inputs. We introduced a method for securely committing an input PVC model for Boolean circuits. PVC security is valuable when the reputation of the committing party is at stake. Our methods are based on our introduction of indexed hashes and $q-$collision resistance and make use of the covert security guarantees and interactivity in MPC. Our work improves upon ordinary hash functions both in speed and communication. Our work is the first we are aware of to enable commitments in MPC for PVC security. We also extend our methods to a setting where the parties are secure against malicious adversaries. Furthermore, we explore the suitability of our methods for arithmetic circuits. This chapter has addressed our sub-question *"How can we ensure the integrity and security of the derived models?"* by proposing, developing and benchmarking a secure commitment mechanism for certified predictions. Thus far, in this thesis, we have discussed technical contributions in terms of protocols and frameworks. In the next chapter, we will look at design and governance challenges around privacy-preserving computation as a whole. This, we feel, is an integral step towards real world uptake for privacy enhancing technologies in general, and our work in particular.

6

Design and Governance Challenges in Privacy-Preserving Computation

Contents

6.1 Introduction

So far we have discussed how privacy-preserving computation could be realised for secure deep learning in an efficient manner. In this chapter, we focus on the broader picture

and take a human-centered approach. We address the final research sub-question *"How do key stakeholders perceive, understand and evaluate cryptographic primitives such as secure multiparty computation, homomorphic encryption and differential privacy?"*.

Human-centered research on privacy has traditionally focused on end-users: understanding their privacy attitudes and mental models, studying their privacy-related behaviours, and designing tools to help them manage data disclosure according to their preferences. While important, this paradigm of end-user privacy also has limitations. First, individuals may have their data processed in remote and opaque ways by dint of being taxpayers, credit risks, or suspected terrorists — not 'end users' as traditionally conceived. In such cases we still need to understand how privacy as a human right and public good can be reflected and governed in such systems. Second, the end-user privacy paradigm neglects the many other entities who play an important role in articulating, navigating, and embedding privacy in a range of contexts. If HCI is about reflecting human values in computer systems more broadly [285], it is equally important to study those people, whether they are developers [286], designers, risk managers, or policy makers. Finally, addressing privacy as a problem of end-user interaction often yields depressing results due to the sheer complexity of personal data processing making it difficult for end-users to comprehend the choices and tools available. This is perhaps especially true when such complexity is the result of modern cryptographic techniques designed to *protect* privacy [287].

These three limitations are particularly salient in the context of this chapter, which addresses technologies for *privacy-preserving computation*. These are a subset of Privacy Enhancing Technologies (PETs) which have emerged in recent years. These include Homomorphic encryption (HE), secure multi-party computation (SMPC), and differential privacy (DP). These foundational technologies share a common promise: to preserve privacy while also obtaining the benefits of computational analysis. While these technologies have been available in some form for years, and to some extent already

are in deployment, recent progress in their foundational techniques and computational tractability has led some to anticipate their imminent adoption.[1]

Compared to systems and contexts typically studied in privacy-related HCI research, these privacy-preserving computation techniques may be far removed — conceptually, operationally and experientially — from the entities whose privacy they purport to protect. While HE, SMPC and DP are sometimes touted in the marketing campaigns of some device makers[2] for the most part these technologies are deployed as invisible infrastructure rather than being positioned as features which end-users are expected to value, let alone understand or control themselves. In many other (actual or envisioned) deployment contexts, the data being kept private may relate to individual data subjects who are not informed or engaged with its processing; and, even if they were aware, they may have no ability to distinguish between whether such processing was genuinely 'privacy-enhancing' or not. Furthermore, the mathematical and computational complexity underpinning these techniques raises particular challenges to explaining them to various stakeholders; not only end users and/or data subjects, but also developers, investors, product managers, and policymakers.

These differences make privacy-preserving computation technologies a prime case study for an expanded understanding of privacy within HCI beyond traditional paradigms of user attitudes and behaviours [291], to consider developers [292], managers, policymakers and others [293], and the roles they play in defining and operationalising goals like security and privacy. For better or worse, the development and adoption of these technologies, and the political values and consequences they reflect, may ultimately have relatively little to do with 'end users' as traditionally conceived. With these considerations in mind, this work aims to explore the following:

[1]Recent industry analyst reports have suggested that PETs are 'experiencing a renaissance' [288], and that 2020 was the 'the year of PETs' [289].

[2]See e.g. Apple `https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pd`, and Google's [290] DP initiatives.

1. What challenges are associated with the adoption of privacy-preserving computation techniques for different stakeholders?

2. What are the motivations for adopting them?

3. Why and how should privacy-preserving computation technologies be explained, governed, and made accountable to data subjects and wider society?

To gain insight into these questions, we undertook a series of interviews with a variety of stakeholders involved in various ways in the development and adoption of privacy-preserving computation technologies (PPCTs). These included cryptographers and theoretical computer scientists working on foundational PPC techniques, developers of practical tools and libraries for non-expert developers, senior managers and policymakers assessing and identifying real-world use cases, practitioners building PPC products, and designers working with PPCs as a design material. Our aim was to draw out implications for HCI and design raised by this new class of technologies.

In this study, we focussed on three different PPCTs — Homomorphic Encryption, Multi-Party Computation and Differential Privacy. Despite their differences, these technologies have all been classed as 'tools for privacy-preserving computation' [294], which enable 'the derivation of useful results from data without giving other people access' to such data [295]. These privacy-preserving computation technologies are still an emerging technology. While significant theoretical progress has been made, this has yet to be translated into widespread adoption. However, numerous libraries exist for HE, SMPC and DP[3], the number of government-funded PPC projects is increasing [294], and various industry and policymaking forums have publicly heralded their

[3]For HE: Microsoft SEAL [296], HElib (`https://github.com/homenc/HElib`), PALISADE (`https://palisade-crypto.org`); for SMPC: Crypten (`https://crypten.ai`), emptoolkit [130], SPDZ (`https://github.com/bristolcrypto/SPDZ-2`); for DP: Google's DP framework (`https://github.com/google/differential-privacy`), Diffprivlib (`https://github.com/IBM/differential-privacy-library`), Pysyft [252].

potential. Recent reports and working papers have catalogued actual or potential use-cases, as well as noting possible usability barriers including programming complexity, computational overhead, and parameter selection [294, 297].

6.1.1 Related work on PETs in HCI

Much research on privacy in HCI is concerned with how end-users value, negotiate, and manage privacy in the context of their interactions with computers. Work in this vein involves: understanding the attitudes [298–302], expectations [303, 304] and mental models [305] of end-users regarding how their data is collected and used; studying privacy-related behaviours such as willingness to share data [306, 307] and use of protective measures [308]; and evaluating and designing tools for privacy management such as permission settings [309, 310], privacy notices [311, 312], and privacy assistants [313]. Related research in usable privacy and security addresses the usability of various end-user PETs tools. These include privacy and security aspects of ubiquitous tools e.g. web browsers [314, 315], as well as more advanced specialist tools, such as end-to-end encryption [287] and anonymous communication and routing tools [316]. Such work is highly relevant to contexts in which end users directly interact with systems in ways that may affect their privacy, and where there are opportunities to (re)design tools and interfaces to give them more control. Such work is premised on the ideal of individual users being able to understand at least some aspects of how their data is processed, and having the potential to exert some meaningful choices over it.

In some cases, privacy-preserving computation technologies might be usefully studied from this end-user perspective. Bullek et al. [317] studied people's comprehension of the randomized response method for local differential privacy [318]. Participants were asked a series of questions, the answers to which were perturbed with noise to provide privacy. In response to a final question about a particularly sensitive topic, they were able to choose how much perturbation to add (i.e. the value of ϵ). While most participants selected the lowest (most privacy-preserving) value for ϵ, surprisingly, 20%

chose the highest (least privacy-preserving) value for ϵ. Some participants explained this was because adding more noise felt like lying. Xiong et al. [319] also studied participants' willingness to share data with a hypothetical differentially private system. They examined the effect of different descriptions of differential privacy (including real descriptions provided by technology companies and the U.S. Census bureau) on willingness to share, and their findings suggest that certain descriptions (in particular, implication descriptions) are more understandable and increase willingness to share data as a result. Finally, Qin et al. [320] explored usability and understanding in the context of privacy-preserving data aggregation initiatives based on MPC, finding that using various analogies to explain the process of additive secret sharing increased participants' confidence in the scheme.

However, in many contexts, the 'data subjects' are not co-extensive with the 'users'. In the case of the PPCTs mentioned above, there may be several primary users (which may include developers and others) and many wider 'stakeholders' (e.g. commercial and government partners, the wider public). Rather than studying end users who are also data subjects, then, we might instead follow previous HCI research on privacy and security which focuses instead on other actors, such as developers (e.g. [286, 292, 321, 322]). Balebako et al. [286] note that while users may be concerned about privacy, they are generally not 'empowered to protect themselves'; by contrast 'the decisions made by app developers have great impact'. Studying developers, designers, and others can reveal both practical and organisational challenges hindering the deployment of privacy and security technologies [292, 323], highlight discrepancies between privacy research and privacy engineering [324, 325], as well as elucidate the moral dimensions of design. While some studies of software developers suggest that they may 'not have sufficient knowledge and understanding of the concept of informational privacy' [326], others do explicitly engage with the ethical and political ramifications of their work; e.g. Rogaway [327] who acknowledges how the field of cryptographic privacy technologies has 'an intrinsically *moral* dimension'.

This kind of reflexivity on the part of developers and designers is something acknowledged and addressed in approaches like Value Sensitive Design (VSD), which aim to 'illuminate the ethical and moral responsibility on the part of the designer rather than the user' [291]. To understand how particular technologies are imagined as solutions to problems [328], we may need to study a wide variety of actors involved in their development, not only engineers but also those involved in the business of marketing them [293]. By encompassing the full breadth of different actors involved in creating and deploying these systems, we are also able to grapple in different ways with the trade-offs and tensions inherent in the field of privacy-preserving computation, and ask questions like '"Who is making the design decision?", "Who is paying for it?", "What is this saying about the user?"' [329].

Finally, there is also work which critically addresses Privacy-Enhancing Technologies from a philosophical and conceptual perspective. For instance, Tavani and Moor [330] assess how earlier PETs such as PGP and anonymity tools may address privacy as individual control, but do not provide 'external' control beyond the user, which they argue is necessary to protect privacy in the round. Gürses and Berendt [331] point to the limitations of PETs that stem from understanding privacy solely in terms of confidentiality [331]. Stalder [332] points to the ways that PETs designed for individual use may occlude broader social meanings of privacy, while Phillips notes how PETs designed to assist businesses with automating compliance with privacy laws reinforce a restricted notion of privacy as unwanted intrusion [333].

6.2 Research Approach

Given that the technologies being addressed here are still emerging, and the broad and exploratory nature of our research questions, we chose to undertake in-depth semi-structured interviews with a select range of experts from a range of backgrounds

and roles [334]. All had direct experience of working on projects relating to privacy-preserving computation, and occupied different strategic positions in the developing ecosystem. They included: researchers working across HE, SMPC and DP research; industry practitioners and designers with experience delivering practical applications of these technologies; as well policy experts with experience in PETs. We deliberately selected some experts whose careers and roles bridged between the domains of research, industry and policy, some having moved from one to the other over the course of a career, while others maintaining feet in multiple domains simultaneously. These participants can be seen as 'boundary workers', working between the boundaries of science and policy to facilitate the co-production of knowledge and innovation [335] and 'knowledge brokers' who facilitate connections between scientific and other audiences [336]. Including a variety of different roles also reflects the nature of these technologies as 'use-inspired basic research'[337] operating between 'basic' and 'applied' research paradigms [338]. This enabled us to not only understand how the knowledge surrounding these technologies is made in specific places (e.g. research labs, technology companies, government) but also 'how transactions occur between places' [339].

Because these technologies are still emerging, we inevitably could only draw from a small class of professionals, whose roles in the production of these technologies are to some extent ill-defined. As is typical with expert interviews, there was no comprehensive list of relevant experts to sample from; we therefore built a 'sample frame' based on publicly available materials from a wide variety of sources including research papers, industry and policymaking fora, and press [340], to identify potentially relevant experts, and also used snowball sampling.

As a result of the variety of roles and experiences of our 9 experts (see Table 6.1), we used a semi-structured interview format slightly tailored to four different roles (research, industry, policy, design). We invited participants to discuss their experiences, motivations, perceived opportunities and barriers, relating to this space. Open-ended questions allowed us to have relatively free reign to explore these issues

Participants	Description (& years experience in PETs)
P1[R]	Cryptography Researcher and an industrial PETs library developer (20-30y experience)
P2[R]	Cryptography Researcher at company specialising in privacy-preserving computation (5-10y experience)
P3[R]	Cryptography Researcher and an industrial PETs library developer (5-10y experience)
P4[P]	Law and Policy professional working on industrial adoption of privacy-preserving ML (4-6y experience)
P5[P]	Senior government adviser on technology, with strong interest in PETs and their applications (5-10y experience)
P6[I]	Security and Privacy Researcher working in executive role at large tech company (2-5y experience)
P7[I]	Data Scientist working on privacy at a 'big four' accounting firm (5-10y experience)
P8[I]	Researcher at consultancy specialising in privacy-preserving computation as a service (2-5y experience)
P9[D]	Designer at a tech and design agency specialising in ethical use of data & AI (2-5y experience)

Table 6.1: A summary table of the total participant sample from Research (R), Policy (P), Industry (I) and Design (D).

[341]. The interviews were conducted over video chat during Spring and Summer 2020. The 9 interviews varied in length between 35-75 minutes, with the average taking 55 minutes, producing 8.3 hours of audio recordings in total, which were transcribed. All parts of the study were approved by our institution's ethics review committee. We used thematic analysis [342] to identify key themes and ideas discussed by experts. Two researchers independently developed a set of codes based on close reading of disjoint subsets of the interview transcripts, using an open coding process. The two researchers then discussed and consolidated their codes to derive a common set, which was then applied by both researchers to all the interview transcripts, and memo notes were taken to record observations about the codes and their relation to one another [343]. A final round of discussion based on this data resulted in a set of themes and sub-themes, presented in the following section.

6.3 Findings

We divide the findings from our interviews into two main areas. The first addresses the technical challenges and opportunities around the adoption of privacy-preserving computation techniques — primarily concerning their transition from theoretical research into practical application. The second concerns the motivations and goals for deploying these techniques to address commercial and societal goals, and how the institutions that deploy them might explain and be held accountable for their use.

6.3.1 Technical challenges and opportunities

From Theory to Practice

Many participants brought up how advances in the theoretical grounding on which privacy-preserving computation techniques are based may not translate straightforwardly into specific real-world applications. This was acknowledged by both research scientists working on those foundations, and practitioners attempting to deploy the technologies in particular contexts. Many participants expressed confidence that those theoretical advances would translate into practice in time. P3, a researcher, argued that while it is *'early stages, from the business development perspective of homomorphic encryption'*, he was nevertheless *'confident that [the] technology is useful, practical'* (P3[R]).

It was generally accepted that much of the research had only reached the stage of proofs-of-concept rather than deployments; while *'doable in principle'*, *'there is a lot of work to do before that bigger picture potential can be realized'* (P1[R]). Similarly, P9[D] explained that as a designer she was *'trying to make them more widely understood within the design and tech community ... There's lots of research in academia at the moment, but not very many examples of them being used in practice'*.

The policy experts we interviewed were optimistic that the technology was already nearly ready for practical deployment. For P4[P] *'the technology has scaled to the point*

that ... it's definitely commercially deployable'. For P5[P], while practical deployment would require a series of *'reasonable engineering and architectural compromises'*, he was still optimistic that *'existing approaches to homomorphic encryption are tractable'*.

While both research scientists and policy experts were optimistic about the big picture, those trying to bridge theory and practice on the ground expressed frustration that much of the research was not directly relevant. In some cases, this was because the scientific work made simplifying assumptions that were rarely satisfied in real use cases. In the context of trying to apply differential privacy techniques to a project involving time series data, P7[I] admitted that he *'really struggled, you know, seeing the value in all those techniques that academia likes to talk about ... how am I going to use that with time series?'*.

Similarly, several participants pointed to the variety of messy underlying data and software issues that exist on the ground that hamper deployment. P7[I] explained that the initial challenges are around *'how do we list data assets, and manage access, at an enterprise scale?'*; while P8[I] spoke of clients with various custom systems and data formats, so *'while we're solving the security aspect of the communication between counterparties ... we still haven't fixed this engineering problem - it's not going away'*.

P2[R], a research scientist who had worked on both foundational theory and engineering, explained how applying techniques in practice involved moving carefully between theory and engineering:

> 'Engineers do need to learn a lot to deploy these kind of technologies and what makes the whole thing complicated is that they need to acquire a kind of knowledge that is not something that the university professor knows ... a lot of low level optimizations make a huge difference, and yeah, from theory to practice they need to somehow be invented.'

In P2's opinion, such work would not come from a *'linear transmission of knowledge'*, but rather through continuous iteration and *'course correction'* between theoreticians and engineers.

Interdisciplinarity and translation between roles

As well as theoreticians and engineers to bridge the gap between theory and practice, several participants discussed the need for people with different skills, backgrounds and motivations to work together. This included not only the combinations of different expertise involved in foundational privacy-preserving computation research such as math, statistics, and cryptography, but also specialists in specific application domains. As P2[R] noted, successful deployment depends on a *'component of multidisciplinarity'*:

> 'In my experience this is very hard to get the right people and crucially, with the right incentives in the same room ... You not only need data scientists, but also security engineers, mathematicians and then experts in the application domain.'

The differing motivations and cultures of these different communities was seen by some as as a problem as it leads to certain important problems being neglected. Commenting on the misalignment between incentives of academic researchers and industry, P3(R), who had worked in both sectors, lamented how research is *'driven by the need to get published; it favors more ... performance breakthroughs, functional breakthroughs'*, meanwhile, topics like usability are *'not so interesting for the basic core research community'*.

The need for an even broader range of disciplinary expertise and professional skills was articulated by P4[P], who described her role as *'to bridge the lexical gap between technologists, lawyers, and policymakers to defragment the current initiatives in PETs'*. Drawing from previous experience working on AI in government, where *'insulated development'* led by technologists failed to account for the *'constitutional implications'* of these technologies, she warned that *'the same could happen for PETs without this sort of ... interdisciplinary discourse'*.

Usability for Developers

A common theme among both researchers and industry practitioners was the complexity of applying privacy-preserving PETs from a software engineering perspective. They discussed a set of inherent challenges facing developers around flexibility, performance, and specifying appropriate parameters.

Several participants described how PPCTs, in particular homomorphic encryption, can be very *'brittle'* (P1[R]): small changes in parameters can result in drastic reductions in performance, security, or privacy guarantees. Such sensitivity can be hard for developers to anticipate and manage, especially as there are many different parameters to tune. This was contrasted against other PETs, like public key cryptoschemes, where there is one main parameter — key size — which has a fairly predictable relationship with security guarantees and computational overheads:

> 'I mean RSA, you have the bit length and that's pretty much it. These things [homomorphic encryption applications] you have a ton of decisions to make when it comes to how to instantiate it, and they have implications for both speed and for the actual function that you will need to compute.' (P1[R])

This results in problems for developers not just in the initial implementation of a privacy-preserving technique, but also as they inevitably need to update a system to *'evolve when you need to change various details ... performance and tractability can be so highly dependent on small details'* (P1[R]).

Participants articulated this as a trade off between approaches to building PPCTs that either work out-of-the-box but have poor and unpredictable performance, *or* that have reasonable performance, but require fine-tuning by engineers. For instance, while the invention of fully homomorphic encryption enables both addition and multiplication and therefore arbitrary computation, specific applications still need to be converted into those arithmetic operations and may incur great computational costs depending on how that is implemented. While it may be possible to 'come up with a system with

adequate performance for your application', this often requires having an application which is *'fully specified and well defined in mind, and you have a team of experts working for you'* (P1[R]).

Several participants spoke about the development of privacy-preserving computation software libraries for developers (see Section 2.2), often contrasting two approaches which reflect the trade-off articulated above: on the one hand, libraries which create an abstraction layer which obscures the underlying complexity; and on the other, libraries which expose all of that complexity so that the developers still need to create bespoke solutions for their application context. P1[R] noted that many developers expect a library to provide *'abstractions that are convenient'*; otherwise *'It's like telling people: OK, I'll give you transistors and you'll build from them ... people don't think this way and for good reason'*. P2[R] made a comparison to machine learning frameworks (e.g. Tensorflow and Scipy):

> They have a very nice abstraction layer that allows them to say "OK, here's my function over the reals: optimize it, under the hood"... We will have worried about implementing matrix multiplication super quickly over floating point numbers so that the data scientists can assume [it's] like doing math on their on their notebook, right? This level of abstraction doesn't exist yet [for privacy-preserving computation]

However, P2[R] cautioned against such an approach for privacy-preserving computation libraries, because it would preclude *'a lot of optimizations that come from understanding the underlying protocol'*. P1 echoed this, stating that:

> 'The only way that we know now of making the computation go reasonably fast is to use a lot of tricks and the developer needs to know about those tricks'

As a result, P2[R] felt that *'general purpose tools'* would inevitably fail to meet developer's performance expectations and thus give them a mistaken impression about the true potential of PETs.

While many participants were in favor of some form of standardisation via libraries, P8[I] explained that the prospects for a standard platform depends on where the technologies are being deployed to. In the context of SMPC, because smartphone operating system providers *'control the platform, they can decide ... this is how it's going to work'*; whereas P8[I]'s work involved deploying SMPC into a wide range of different clients' environments where *'we can't really dictate to them how they store their data'*; as a result the possibility for standardisation was small.

6.3.2 Motivating, explaining and governing privacy-preserving computation

Our participants also raised various insights relating to the motivations for adopting PPCTs, and challenges relating to explanation and accountability.

Motivations

Unsurprisingly, 'privacy' was often cited as the motivation for developing and deploying privacy-preserving computation technologies. However, some subtly different articulations and understandings of privacy emerged from our interviews, as well as some other motivations which went beyond privacy altogether.

Some very directly motivated the adoption of privacy-preserving computation by reference to the interests of individuals in privacy and the protection of their personal data: *'it's individual privacy - it's human rights'* (P6[I]). In comparison to the push for similar technologies in other markets, privacy-preserving computation was more a response to individual privacy:

> 'People do understand when their privacy is violated. So ... the push for these technologies is very different to the push the semiconductor industries have had ... So I give a lot of time on the examples that are user-centric' (P6[I]).

However, appeals to individual privacy were often mediated via other pressures. First, organisations deploying PETs may not have a direct relationship with those

individuals, but are instead concerned with third-parties affected through business-to-business relations:

> 'You have customers: these may be business to business customers, but that also extends to customers of customers and therefore it boils down to individuals' (P6[I])

Second, some cited the existence of privacy and data protection regulation as an incentive to provide and deploy PETs: *'because of GDPR* [the E.U. General Data Protection Regulation]*... all the regulatory environment is ... very favorable for providers'* (P7[I]). This regulatory pressure meant that investment in PPC could be accounted for in terms of corporate risk management: *'to have compliance at least formally speaking with GDPR ... it's really protecting assets of a company'* (P6[I]).[4]

Third, P9[D] argued that rather than just enabling existing data processing to be done in a more privacy-preserving way, these technologies could enable new insights which *'you might not have been able to gain before because of the sensitivities around the data that you are using'*. P4[P] highlighted a range of *'missed opportunities'* for privacy-preserving computation *'for a good purpose'*. These included cases such as the Boston Women's Workforce Council who *'used secure multiparty computation to confidentially analyze gender wage gaps without ... disclosing who the salary belonged to'*.[5] P5[P] noted the opportunities for government national security services to use HE techniques like private set intersection to identify suspects without combining certain databases in the clear, something that might not otherwise be undertaken due to the *'intrusiveness'* of sharing data of large numbers of innocent citizens between departments.[6]

While individual privacy was cited by all participants as an important motivator, it was often an indirect motivator, and in some cases perhaps insufficient on its own (e.g. without being coupled with new opportunities to extract value from data). Other participants articulated motivations for pursuing privacy-preserving computation which

[4] A sentiment echoed in [297]

[5] `https://thebwwc.org/mpc`

[6] A use case discussed in [344].

had nothing to do with individual privacy as such. For example, for some researchers (e.g. P1, P2), it was basic intellectual curiosity (*'somebody thinks of something that ... looks interesting to them'* (P1)). Other cases included where competing businesses would have a mutual interest in the output of some computation on their respective data, but would not otherwise share that data out of *'fear of losing a competitive edge'* (P6[I]). Intellectual property protection was also frequently cited as a key motivation for many business applications.

Privacy-preserving computation techniques were also seen by some as offering the possibility to navigate regulatory obligations and trade-offs in different ways. First, they have the potential to fulfill obligations to protect data in new, more 'technological' ways, offering *'technological safeguards that can't be easilly overridden'*, the kind of protection that *'paper safeguards, like contractual guarantees and policies, just can't provide'* (P4[P]). They were seen as especially promising in cases where different regulatory obligations might appear to be in conflict, as P4[P] explained:

> 'Anti money laundering regulations are very data maximalist; they want you to collect more data [to prevent] financial crimes. But in the meantime the GDPR is quite the opposite; it wants you to minimize data, ... and this really conflicts with the regime of AML. I think that PETs could actually cut through these legal conflicts and really provide a practical solution ... it's not actually transferring PII, but it still allows for banks to prepare for AML protocols'

Similarly, for P5[P], privacy-preserving methods had the capacity to change what is possible without sharing data and thereby shift the scales in legal balancing tests [345] that might otherwise make certain data analysis unlawful:

> ' UK law ... sets out a test for those of us in national security which is *necessity* and *proportionality*. So if you can shift the proportionality, then you're in a better position so you can avoid intruding, you can avoid privacy risk'.

In these ways, such techniques were envisioned by P4[P] and P5[P] as enabling organisations in the public and private sector to break free of what P4[P] called

'legal gridlocks' that currently are (or are perceived to) exist around data use and enable new kinds of analysis.[7]

Explanation

Our participants discussed various facets relating to *explaining* privacy-preserving computation, including *how* they go about explaining it to different audiences (and in some cases, why they don't even try).

The researchers described a variety of contexts in which they had had to explain underlying techniques and their strategies for doing so. For a general audience, P2's strategy was to explain simplified versions of protocols, such as simulating a secure multi-party computation for dating using playing cards (see [346]). While these were *'fun to explain'*, P2[R] was unsure about the effectiveness of such explanations:

> 'Then in the future, [the audience] will be like: "Oh yeah, multi-party computation, the thing with the cards." That doesn't mean that my explanation was effective... My feeling is that people tend to end up amused and satisfied.' (P2[R])

Such explanations were offered as a starting point to encourage people *'who are attracted by that kind of magic'* and would *'go into Wikipedia immediately after'* (P2[R]). However, P3[I] felt that there was a lack of accessible educational material: *'there is certainly not enough material and the classical crypto papers are essentially useless for someone who is not an encryption expert'*; they suggested that explanations of core concepts might be more effective if tackled as part of a standardisation process and included within libraries.

Several participants also cautioned that the kind of explanations offered (if any) need to be tailored to the audience. On the one hand, explanations could be too technical: *'If you start with equations ... you lose 99 percent of the audience right*

[7]While a legal analysis of potential conflicts between these two areas of law is beyond the scope of this work, we note that data protection and financial services regulators have, at least in the UK, affirmed their compatibility in general terms: see `https://www.fca.org.uk/news/statements/fca-and-ico-publish-joint-update-gdpr`.

away' (P7[I]). On the other hand, short intuitive explanations might be too simplistic for informing executive decisions:

> 'So one thing is getting people interested, and the other one is informing, like, executive decisions. I don't think they should be informed ... by two minute stories... I don't think decisions about encryption are made based on an intuitive understanding of crypto'

For P9[D], designers have a role to play in explaining privacy-preserving techniques through prototyping their use in specific contexts. This included explanations to end users, but also *'a different language to explain it to those designers as well'*. Previously, their design agency hadn't *'seen much demand for them on the industry side'*; however, that changed after publishing a blog post explaining visually how differential privacy could work in the context of a project on identifying inequalities in urban mobility:

> 'Each step of the randomized response process ... we had an image to go with it, so that you could see ... the noise that you are adding to data.
>
> Visually seeing it was really helpful for me as a designer and then tying it to sort of real life stories so that I could see how you wouldn't be able to re-identify someone. Imagining what that makes possible forces you to think about the qualities of that technique, what it now enables you to do'

When it came to explaining these systems to end-users, however, some participants questioned whether this was a worthwhile goal. P9[D] couldn't imagine *'many scenarios where it's necessary to explain what privacy-preserving techniques are being used to an end user who is trying to do something with their phone'*. P7[I] asked himself whether end-users understood these techniques, and answered: *'Well in general, not. Is it a problem? I'm not sure it's a problem'*. In such cases, it was seen as sufficient that end-users *'trust the provider of the solution that they do a good job'*(P7[I]).

Governance and Accountability

A final theme was around the challenges of *governance* and *accountability* of privacy-preserving computation. These topics often followed organically from discussions of

explanation; attempts to explain these systems were often made in the course of trying to *justify* their use to affected stakeholders, and justification is a key element of accountability [347]). But even if explanations don't lead to real understanding on an individual level, it might still be possible to justify them to the public. P5[P] put it this way:

> 'These technologies are extremely difficult to understand... Do they meaningfully address genuine privacy issues? Yes they do. Do they address public concern? That's not to do with the technologies *per se*, [but] how the technologies are explicated and made available. If you told the public: "As a result of using these technologies, we are able to limit the amount of your personal information that's shared, and are still able to offer you valuable services", they would be enthusiasts.'

Other participants expressed scepticism that the public would take such guarantees at face value. In the context of proposals for privacy-preserving facial recognition in border control, P6 asked:

> 'if someone publicises this new system ... just by saying: "and by the way the privacy of the information is very well handled because we use the state of the art cryptography", what does that mean to a citizen?'

Both P6 and P7 suggested that certifications and trust marks applied to services which use these techniques could enable individuals to seek out more trustworthy systems. However, expecting individuals to exercise meaningfully informed choices in relation to different services involving privacy-preserving computation was seen by some as adding to the burden of responsibility unhelpfully placed on individuals. P9 reflected on how *'constantly making decisions about data in the technology that we use is just not sustainable'*; instead, they suggested that *'collective consent models and other governance mechanisms ... that can make decisions on behalf of people'* might be a better approach. Similarly, P2[R] felt decisions about the technical details of the adoption of these technologies ought to be made by *'using experts or authorities'* who can act as *'proxies ... [who] understand their communities'*.

While most of our participants pointed to the positive potential of privacy-preserving computation techniques, a few were also concerned about the power imbalances they

might reinforce. When the stakeholders are individuals, they are *'by definition, the weaker party'*, and *'lack the resources ... to induce changes; every time we talk about privacy there is some asymmetry that is implied by it.'* (P1[R]). For P9[D], it is important to recognise the limitations of PPCs as they are just:

> 'a technical solution to protecting people's privacy ... you have to think about the wider system that they sit within and what other kind of power dynamics are in that system.' (P9[D])

6.4 Discussion

The findings from our interviews raised several important implications for the design and governance of privacy-preserving computation. They reveal how these techniques are being not only technically but also socio-technically constructed and constituted by a variety of actors, each pursuing overlapping and sometimes diverging agendas. Clearly, privacy-preserving computation techniques entail a variety of human-centric challenges which HCI research could seek to address. These challenges are multifaceted and will require diverse approaches — something that HCI as a methodologically diverse field is well-positioned to reflect. Furthermore, these challenges are inter-related: for instance, the way in with these technologies are translated from theory to practice may well affect how they can be explained and held accountable; while closer inspection of how 'privacy' and other motivations are unpacked might reconfigure what kinds of interdisciplinary collaborations are required in a particular context. Our aim in this section is to reflect on these, to understand both the design problems facing these techniques, and the challenges they raise in relation to the interests of a variety of users and wider society. This discussion is not intended as direct 'implications for design'; rather, we hope to draw attention to issues which require further research, as well as interdisciplinary discussion.

6.4.1 Whither the end user?

While our experts generally acknowledged the individuals whose personal data is being privately computed on as an important stakeholder group, few seemed to prioritise seeking their understanding and acceptance. This is in contrast to the small number of existing HCI studies that investigate 'user acceptability' of particular privacy-preserving computation techniques such as differential privacy [317, 319] and MPC [320]. User acceptability could and should be further examined in particular contexts; for instance, Colnago et al. [348] suggest further work is needed to explore whether such techniques embedded in Internet-of-Things privacy assistants might 'help mitigate people's reservations about data collection practices and reduce the chance they opt out'. There is clearly great scope for important research within this paradigm of user acceptability.

However, our experts spoke about privacy-preserving computation technologies more as tools enabling *organisations* to achieve a variety of goals (including managing privacy risks, but also protecting corporate assets and secrets), rather than as a means of directly serving users' interests. While user acceptance was not entirely disregarded, it did not appear to be a primary concern; even P9, a designer well-versed in user-centred design, doubted that people could or should be expected to understand and make decisions about privacy-preserving computation. Privacy may be important, and these techniques may have the potential to meaningfully embed it, but whether or not individuals understand and accept them seemed to be almost a secondary issue. In many of the use cases they mentioned, individuals whose data is being computed may not have any direct interaction with the system, nor any choice about whether to use it. In expressing such doubts, our interviewees might appear to be denying a sacrosanct tenet of HCI as a *human*-centred discipline. However, rather than denying the importance of user acceptance, we believe that these doubts should in fact point us

towards alternative human-centric approaches to the development of privacy-preserving computation, in addition to solely looking at end users as data subjects.

First, our findings point towards studying the needs of different kinds of end users; specifically, those developers and designers who attempt to apply foundational privacy-preserving computation techniques in real-world applications. This echoes recent calls to acknowledge that 'developers are users too', as Green and Smith argue in relation to crypto and security libraries [349]. Similarly, P9[D] pointed to the relative lack of awareness and understanding of these techniques among designers. As with the application of other complex methods in computer science, such as machine learning, it may be difficult for designers to use privacy-preserving computation techniques in design practice due to unfamiliarity with how they work and awareness of what they can achieve [350]. P9[D] made the case for technical specialists and designers to work together to translate these technologies into 'design material' which design practitioners can use to explore real use cases.

In addition to understanding developers, designers, and others as *users* of privacy-preserving computation techniques, studying them also allows us to explore how a human value like privacy shapes the construction of complex computational systems. This perspective accords with 'third wave' approaches to HCI which orient attention towards the ethical obligations and values of designers [291], and incorporate different disciplinary perspectives which examine how social and political dimensions are embedded and reflected in systems [351]. As such, rather than just considering whether end users or laypeople understand, trust, and accept privacy-preserving computation technologies, we might also benefit from considering the perspectives of the various people involved in constructing their technical, commercial and regulatory foundations. Assessing whether an innovative technology will be acceptable to users through lab and field studies may be valuable, but such approaches often neglect the ways in which such technologies are interpreted, shaped, and mutually constructed over time through their designers, users, and broader political, economic and regulatory forces

[328, 352]. As a result, it is equally important to consider the plurality of different actors and broader contexts through which values like privacy will be understood, traded-off, and embedded in these systems (or not).

6.4.2 The Limits of Abstraction

If, as suggested above, we are to consider the needs of developers and designers as users of underlying privacy-preserving computation techniques, then how might those needs be met? Many of the interviewees identified the need to create building blocks for privacy-preserving computation. In an ideal world, these building blocks would allow developers to *abstract away* the technical details and apply them to applications in different contexts. Creating such abstractions is fundamental to progress in computer science and programming; in Edsger Dijkstra's words, it is 'our only mental aid ... to organize and master complexity' [353]. However, many of the experts expressed uncertainties about the form such abstractions should take and the extent to which they could reasonably be made in the domains of privacy-preserving computation. Especially with homomorphic encryption, abstracting away the details of implementation could mean losing the ability to optimise performance through engineering 'tricks' (P1[R]).

Attempts to create tools for developers to enable them to integrate privacy-preserving computation techniques into their products may therefore need to grapple with this need to balance abstraction and engagement with the implementation details. Specific applications will always require some 'intimacy with the details' [354] that might otherwise be abstracted away. Some of our interviewees argued that the necessary education required for developers could potentially be integrated into standardised APIs. This suggests that broader adoption of privacy-preserving computation may benefit from work in HCI which considers APIs and libraries as 'first class design objects' [355, 356], with the goal of 'driving adoption of software components' [357]. This could involve (re)designing them around the typical ways programmers learn, e.g. on-the-fly, via information foraging, and trial and error [358, 359].

However, the nature and extent to which developers need to become intimate with the details, and how they might do so, will clearly depend on the particular technique in question. For instance, a DP library might implement a variety of noise sampling and injection techniques, but this is relatively simple compared to the much more complex mathematics and reasoning involved in deciding on and managing an appropriate privacy budget, which requires case-by-case human consideration. For SMPC, libraries might take care of some of the networking details, but leave difficult decisions regarding the protocol up to the developer. The nature and value of these standardised building blocks will therefore vary greatly between approaches.

Ultimately, the design and adoption of these privacy-preserving computation building blocks may need to reckon with the messy realities of underlying enterprise IT infrastructure, agile and iterative approaches to software development [324], and service-oriented architectures [325]. Given these practical considerations, the full complexity of these technologies might instead need to be mediated via a two-step process: general-purpose libraries which expose all of the complexity of a domain (e.g. homomorphic encryption) that enable specialist privacy engineers to create particular privacy-preserving computation components for common operations or use cases (e.g. private set intersection for contact discovery); those components could then be adapted and deployed with minimal configuration by non-specialist developers as microservices.

6.4.3 Privacy-Enhanced Technocracy

The way privacy-enhancing technologies are sometimes described can make them seem esoteric, exotic, and mysterious. For instance, in industry press they have been described as 'black magic'[8] and a 'holy grail'[9]. Such language suggests that their development is entirely in the hands of a small and specialised cabal of cryptographers and engineers, much like the early programmers who regarded themselves as 'high

[8] `https://dualitytech.com/tag/homomorphic-encryption/`
[9] `https://www.fastcompany.com/90314942/duality-homomorphic-encryption`

priests' of assembly code.[10] It is possible to imagine how, in these respects, they might end up sharing the same 'rampant hyperbole and political envisioning' [361] of a higher-profile cryptographic technology — blockchain.

While our interviewees avoided such language, and even criticised the perceived 'hype' around PETs, they did reflect the highly specialised knowledge required to make use of the underlying mathematics, and drew parallels with magic. P6[I] described feeling like 'Gandalf the wizard' upon telling people that computation on encrypted data was possible, while P1[R] described the need for engineering 'tricks' to optimise performance within reasonable levels. From this perspective, the technical work of applying privacy-preserving computation seems more like craft than science, which the guild of cryptographers and engineers are uniquely capable of performing [362].

However, the mystery of their inner workings could easily serve as an excuse for not making these systems accountable to affected stakeholders. When reflecting on the challenges laypeople face in trying to understand PPCTs on any meaningful level, both P5[P] and P7[I] expressed some doubts about the possibility that individuals could ever be expected to really understand *how* they work. However, without some form of explanation, and absent any other mechanism for meaningfully communicating their risks and opportunities, there is a risk that privacy-preserving computation becomes not just a *technical* but a *technocratic* solution imposed on populations without popular consent by grey eminences operating behind the scenes.

However, several of the experts did acknowledge the need for mechanisms of accountability and governance to developed as these technologies are rolled out. P6[I] and P7[I] suggested this could involve certification schemes. Similarly, while P1[R] and P9[D] were doubtful about individuals being able to meaningfully consent to these technologies, they proposed alternative forms of collective governance, where the interests of affected individuals could be represented by relevant representatives and experts who can make informed choices and demands on their behalf. These and other

[10] In the words of Rear Admiral Grace Hopper [360].

democratic mechanisms will need to be explored in order to counter a privacy-enhanced technocracy, and methods from HCI — such as participatory design [363], futures workshops [364], and other governance approaches — may have much to offer.

6.4.4 Secrets, Assets, Human Rights: Unpacking 'Privacy'

Our findings attest to the many varied interpretations and uses of the term privacy. As previous work has explored, and as discussed above, the notion of privacy in Privacy-Enhancing Technologies is often a narrow interpretation of what is a multi-faceted and contested concept [330–333]. This is certainly the case for the subset of privacy-preserving computation PETs studied here. They turn privacy into something mathematically formalisable, e.g. in terms of entropy in cryptographic approaches, or indistinguishability in statistical approaches, which can all be understood as variations of 'confidentiality', a pillar of the security triad [365]. This means that other ways of understanding privacy may be de-emphasised and de-prioritised.

There are continuities here with earlier PETs, such as de-identification techniques based on hashing personal data. Phillips argues that that these techniques embody privacy as protection 'from unwanted intrusion' [333]. However, they leave in place the ability of powerful observers to produce 'panoptic' knowledge which can be used to sort and discipline populations [366, 367]. Similarly, if we understand privacy as confidentiality, this can be engineered through architectures of data minimisation [368]; but this can lead to design choices which preclude alternative understandings of privacy (e.g. privacy-as-control), and hinder the exercise of related rights afforded by data protection law [369]. In our discussions with the experts, these alternative understandings of privacy were conspicuous by their absence.

Our findings also demonstrate that even while discourse around privacy-preserving computation restricts certain interpretations of privacy, it also stretches the meaning of privacy to incorporate unorthodox meanings, such as competitive secrecy, corporate asset protection, and government security. These are clearly significant and important

use cases for the technology, but they arguably bear only a family resemblance to privacy as it relates to individuals and society. Indeed, intellectual traditions which value privacy as an individual right and public good have often been associated with opposition to corporate and government secrecy; according to them, privacy should be reserved for the weak, while transparency should be an obligation required of the strong [370]. In referring to all of these things as 'privacy', privacy-preserving computation technologies may elide significant political tensions between them. This is not to deny that they may have a powerful role to play in supporting privacy as an individual right and as a public good [371, 372]; but this confluence of quite different values under one banner complicates the narrative around whose interests they serve.

As well as tending to address narrow and perhaps unorthodox conceptualisations of privacy, it is important to recognise that these technologies do not protect other important values and interests. If our aim is to build and shape systems encompassing multiple social goals, where privacy is just one such goal, then privacy-preserving computation techniques have to be considered in relation to the whole system and the social context. The danger is that the societal problems of data processing technologies — such as the ways they create distinctions and hierarchies that reinforce power, shape politics, or facilitate abuse — are sidelined, redefined, or collapsed under the banner of 'privacy', so that privacy-preserving computation techniques can be positioned as *the* solution (what Pinch and Bijker term 'closure by problem redefinition' [373]). This danger was alluded to in P6[I]'s example of privacy-preserving border control (where people are still ultimately at the mercy of a powerful state), and in P9[D]'s concern about considering the wider power dynamics in the context of deployment.

6.5 Conclusion

Like other technologies which have been touted as potentially revolutionary in recent years, the concrete impact of these privacy-preserving computation techniques remains

to be seen. New technologies often emerge in unexpected ways, at unpredictable times from niches of computer science: hypertext, Merkle trees, and neural networks were once confined to their respective research sub-fields before they became known more widely as the world wide web, blockchain, and 'AI' (in its latest guise of deep learning). Prior to their take-up in wider society, these specialised areas of research were conceived as laying the groundwork for purely technical pieces of invisible infrastructure, whose implications for human-computer interaction were remote and unclear.

However, we believe it is worth researchers studying such technologies prior to their widespread adoption. Whatever technical and institutional forms they take, the journey of privacy-preserving computation techniques from the annals of cryptography into production code will be shaped in substantial part by the approach they take to a variety of human and societal challenges. Indeed, these challenges directly implicate some fundamental concerns including: multifaceted (re)conceptualisations of the notion of 'the user'; helping people navigate and manage computational complexity and its consequences; exploring how values like privacy can be reflected in the systems we build; and examining how different political agendas, economic rationale, and user groups shape and are shaped by those systems. These concerns all cohere and overlap in the emerging space of privacy-preserving computation.

This chapter aimed to provide a preliminary and partial outline of those challenges, laying some of the groundwork for substantial further exploratory and in-depth work to be done. In addition to several recent studies which focused on people's understanding of these techniques and their willingness to disclose personal data in the presence of them, we have outlined a broader set of research questions prompted for HCI by PPCs. These include understanding specific application contexts; usability of PPC libraries and tools from a non-specialist developer's perspective; and understanding the explanation and governance challenges associated with these techniques. In conclusion, this chapter addressed our final sub-question *"How do key stakeholders perceive, understand and evaluate cryptographic primitives such as secure multiparty computation, homomorphic*

encryption and differential privacy?". This was answered through a series of insights, gained through our study, around challenges, perception and motivations for privacy preserving computation. In the next chapter, we bring together different elements of our research, refer back to the overarching research question and conclude this thesis.

7 Conclusions

Contents

This thesis introduced various state-of-the-art solutions for provably secure, understandable and deployable deep learning systems. We now provide a summary of the contributions of this thesis, draw overall conclusions of our work and suggest future research directions. We also discuss limitations and additional applications of our work.

7.1 Summary of Contributions

By employing confluence and convergence of privacy-preserving computation technologies, machine learning optimizations and advancements in system design, we designed ingredients for effective, efficient and equitable privacy-preserving deep learning systems, in this thesis. In the next few paragraphs, we describe our contributions. We discuss these contributions with reference to the overarching research question *"How can we design, build and evaluate effective, efficient and equitable privacy-preserving Deep Learning systems?"*, and the sub-questions.

First, we started by addressing our very first sub-question *"How can we design crypto-friendly protocols and architectures for secure deep learning?"*. We explored the challenge of ensuring the privacy between two parties efficiently without compromising accuracy. In particular, we considered a problem where two non-colluding parties are to jointly train a neural network model privately. We proposed a number of adaptations in the neural network training process to make it more crypto-friendly. Furthermore, we designed an efficient MPC protocol enabling secure computation over this adapted training process. We achieved accuracy levels similar to cleartext floating point training using our secure protocols for privacy preservation. We show that our method can provide significant speed-ups over the prior state of the art by taking a holistic approach and employing optimizations at both levels—cryptographic protocol and the ML training process.

Second, we extended the setting above to a large-scale decentralized training scenario, where most data is unlabeled in an attempt to answer sub-question *"Can secure deep learning approaches be decentralised and/or federated?"*. For this we extend our setting to a scenario with a large number of devices streaming unlabeled data. To train models securely in such highly decentralized setting, we proposed the concept of *silos* and combined federated learning. We employed knowledge distillation, in the form of model generations, to learn from unlabeled data. Finally, we used multi-party

computation for secure aggregation of the models. We showed that our approach results in a relative improvement of up to 54% in model accuracy, compared to training over just the labeled subset of the dataset.

Third, we focused on delivering reliable secure inference over private models and inputs. In particular this addresses the sub-question *"How can we ensure the integrity and security of the derived models?"*. Here the focus was on extending our approaches to ensure equitability and integrity alongside security. We proposed an efficient mechanism for certifying and validating a model to enforce non-functional constraints. We do this by designing a novel MPC-friendly, covertly secure hashing scheme. We show that we could benefit from the covert security guarantees in the current setting, where publicly provable defamation could act as a deterrent. This shall ensure integrity not just for privacy preserving frameworks, but machine learning models in general.

Finally, we focused on the sub-question *"How do key stakeholders perceive, understand and evaluate cryptographic primitives such as secure multiparty computation, homomorphic encryption and differential privacy?"*. This was to provide a broader perspective to our methods, their acceptability and applicability. We studied the usability of our protocols and the privacy-preserving computation technologies at large in practice. We interviewed a number of key stakeholders, including practitioners, policy makers, and other experts. We identified a variety of design and governance challenges in the development of privacy-preserving computation. These challenges will need to be addressed for widespread adoption of privacy-preserving machine learning technologies. We have thus addressed our overarching research question *"How can we design, build and evaluate effective, efficient and equitable privacy-preserving Deep Learning systems?"* by addressing and evaluating each of four constituent sub-questions.

7.2 Limitations

The work does have certain limitations. First, our technical contributions do not address issues around output privacy. Even if inputs are secure and private, any public output can potentially compromise privacy. For example, should the model be made public upon private training, it would still be susceptible to white-box access attacks, such as reconstruction attacks [55, 56]. Similarly, if only the inference is made public, the models would be susceptible to black-box access attacks, such as model inversion attacks [58] and membership inference attacks [59]. Our protocols do not account for attacks as a result of contaminated data [254]. While beyond the scope of this thesis, there are several ways to mitigate such attacks, e.g. by combining our protocols with strategies, such as employing differential privacy alongside MPC amongst other ideas.

Second, our current protocols provide either semi-honest security (Chapter 3) or covert security (Chapter 5). While we explain our threat model in detail, sometimes it might be insufficient, e.g. when the parties involved are not cooperative and honest. As a result, one might need to address this question in greater depth and explore security for malicious adversaries.

Third, MPC protocols operate within an algebraic number field and therefore are suited for fixed-point computation. While it is possible to perform approximate floating-point computation [374], this is slow and computationally expensive. Ordinarily, DNN training involves extensive floating-point computations. To design crypto-friendly protocols, we adapted specialized machine learning frameworks for quantized training. These quantized training procedures made use of fixed-point computation. For these procedures, in our evaluations, we saw only marginal loss in accuracy. However quantization might affect the applicability of our protocols for sufficiently different DNN architectures.

In terms of compute and communication performance, we did improve by more than an order of magnitude. This makes the MPC-based secure training, evaluation and

certified predictions practical for an array of architectures. While such architectures are capable of serving a rich variety of applications, certain applications tend to be dependent on very large or complex DNN architectures. For such applications, our protocols (and secure protocols in general) might still be intractable, and there is scope for further improvement.

More generally, privacy-preserving computation formalises privacy as confidentiality, but this is not the only way to understand privacy. These methods may still enable 'panoptic' surveillance, which, in turn, can reinforce problematic power relations in our digitally enabled society.

7.3 Future Directions

Privacy-Preserving DNN Training. Our two party protocol for DNN training is envisaged for security against semi-honest adversaries. However, in the real world, the parties cannot always be trusted to stick to the protocol. We use a mixed protocol switching between Boolean shares and arithmetic shares for efficiency. However, it is possible to extend our ideas around matrix-vector multiplication and non-linear operations to a setting secure against covert or malicious adversaries. In particular, either the entire computation could be reformulated into a single maliciously secure GC evaluation protocol (e.g. authenticated garbling [375]) or a maliciously secure secret sharing scheme (e.g. SPDZ [376]) could be used. Alternatively, maliciously secure conversion protocols–from Boolean to arithmetic shares and vice versa, such as daBits [377], could be employed. The efficiency of these schemes is a matter of further exploration.

Our experiments on our framework for training over unlabeled data focused on identical architectures for all the global models. Nonetheless, our approach supports different model architectures for each silo. Ensembles involving diverse architectures have been shown to be efficient. We leave for future research the question of how

our approach performs with diverse model architectures. Moreover, we motivate the idea of silos and benefits of using multiple models in this work, but the theoretical analysis of these guarantees could be explored in greater detail. Also, there is a potential to explore privacy-preserving inference approaches for secure knowledge distillation and ensuring model privacy. Moreover, while we address the problem of model contamination in our work around certified predictions, safeguarding training data against contamination [378] remains a substantial challenge.

Our work on privacy-preserving DNN training represents a significant move towards decentralised AI systems, which in effect has a cascading effect not only on privacy preservation but more fundamental questions such as agency, negotiability and legibility around human data interaction [379]. Among the challenges to re-decentralization laid down by Crowcroft et al. [379], we attempt to address integrity and incentives in Chapters 5 and 6. However there's scope for future work to address issues around identity, availability, trade, power, long-term control etc. The Oxford Martin programme on *Ethical Web And Data Architectures*[1] aims to address some of these challenges within AI and privacy through re-designing digital infrastructures. Furthermore, combining our decentralized privacy-preserving approaches with user-controlled virtual containers (such as SoLiD [380], to keep users' data private while at rest) is an important way forward to achieve greater privacy and autonomy.

Certified Predictions. We propose a covertly secure approach for certified predictions. The approach could be extended to a maliciously secure variant with optimized parameters, as we motivate in Chapter 5. This would be of value in those circumstances when the interaction involves only few queries and defamation is not enough of a deterrent against cheating. There is also a gap to be closed between constructions and lower bounds if we allow large commitments. The requirements on prng are slightly inconvenient and a deterministic way to find vectors for construction 3 (Chapter 5), like in construction 4, would be useful. More speculatively, one could investigate if

[1] `https://www.oxfordmartin.ox.ac.uk/ethical-web-and-data-architectures`

the commitment size could be reduced to $O(1)$ while maintaining half an AND gate per bit cost. More broadly, the idea around certified predictions has a rich scope. As we generalize the idea into MPC-friendly commitments, this has potential to be used for applications other than certified predictions—for example, securely binding an entity to a commitment they made at an earlier occasion.

Design and Governance Challenges for PPC. Understanding and evaluating the perspective of stakeholders is important in when realizing technological advances such as PETs. Simultaneously identifying and evaluating challenges accelerates development and enhances usability. While we develop a deep understanding of the challenges, most of these are not trivial to mitigate. There is a need to understand and address the gaps across different stakeholders. For example, one of the research experts in the study indicated that they tried to explain MPC to the people ultimately using the technology, but other participants indicated that this might not actually be needed or useful. Similarly, while researchers are confident that the technology is quite ready, at least, for a segment of practical applications, some early adopters and practitioners felt otherwise. Moreover, there are challenges around design, standardization and regulation that became apparent in the study. A deeper analysis into strategies to mitigate these challenges on a technological front as well as at the level of business processes would be crucial in determining the success of privacy-preserving computation. In particular, involving stakeholders in the design decisions and conducting a follow-up study focused on addressing the most pertinent challenges would seem to be important.

7.4 Epilogue

AI systems have gained immense traction, and for good reason. They have gone above and beyond the human ability to ingest, digest and analyze data. While an excellent opportunity, they quickly translate into a privacy threat. Does that mean AI is itself a threat? Does that mean we keep our data to ourselves? This cannot be

entirely true. As Craig Venter once said *"Privacy with medical information is a fallacy. If everyone's information is out there, it's part of the collective."* He explained how data, for example medical information, should not be kept private. Sharing medical information immediately and without filters could ultimately save a lot more lives.

Does this mean we, as a society, shall live with this conundrum between utility and privacy? The privacy-utility trade-off, as a result of existing legal requirements and data owners' comfort, tilts the balance in the favor of limiting access to such data. But that need not necessarily be the case. Society needs tools and methods to soften the trade-off and safeguard privacy while still keeping the data useful. In this thesis, we propose methods to drive this process. Taking an example from our work, training a neural network model for data shared between two parties, ordinarily, would need them to reveal their data to one another. But with convergence of modern privacy preserving computation technologies and privacy aware machine learning models, we developed an efficient privacy preserving solution. We show the significant merit in designing customized cryptographic protocols for computation, adapting machine learning processes into their crypto-friendly versions and tying everything together through an efficient system design.

Of course, as we scale and explore such approaches further, one needs to think about logistics and practical deployability. This includes richer compute resources and communication bandwidth at the edge to accommodate the decentralized architectures. Combining our approaches with user-controlled virtual containers, where user data is kept private at rest, could be a future worth exploring. Such an approach could deliver on enhanced privacy, safer data movement and greater autonomy to the users. This can truly enhance and complement the promise of privacy preserving computation for machine learning.

Is privacy the only challenge? Regrettably not. AI systems learn from data. Decisions are made based on data. Data, which is captured from the real-world. Data can be well-structured, comprehensive and yet biased, and so also could the

systems that learn from it. To protect the promise of modern AI, it is important to focus on equitability alongside privacy. For instance, COMPAS, a criminal risk assessment system used by US courts, is based on biased data. It is used to assess the likelihood of a defender for recidivism. An investigation into the system [10], revealed that black offenders are twice as likely as white offenders to be labeled a higher risk but not actually re-offend. While there has been active research into tools for debiasing AI models [281], enforcing such methods poses a challenge. In our work, we describe protocols for enforcing fairness, explainability and equitability at large in AI models while ensuring privacy.

Although our methods and privacy preserving computation, at large, offer great promise, the final challenge lies in the recognition by society and policymakers of their potential, developers making effective design choices, and generally, accelerating their uptake. Thus far, there has been some encouraging support for these technologies. For instance, *Promoting Digital Privacy Technologies Act (S.224)*[2] resolves to encourage research and support for privacy-preserving computation. But their full potential is yet to be realised. In this thesis, we also studied and highlighted motivations, explanations, governance and design challenges for privacy preserving computation. It is our conviction that this would help in understanding and mitigating challenges around real-world adoption of PPC alongside the development of the technology itself. It remains to be seen how these technologies could shape the future of AI systems in governance, healthcare, finance and other critical sectors. We are optimistic that they offer great promise.

Finally, it is our hope that this work acts as a catalyst to motivate future research and provides a base upon which to build effective, efficient and equitable privacy-preserving machine learning.

[2] `https://www.congress.gov/bill/117th-congress/senate-bill/224`

Appendices

A
Simulation and Proofs

A.1 Execution in the Ideal World

Next, we present in detail the ideal world execution of a function $g(x, y)$. The ideal world is parameterized by the party corrupted by adversary $\mathcal{A}$, which we denote by $\mathsf{C} \in \{\mathsf{P}_1, \mathsf{P}_2, \perp\}$ ($\perp$ is just a value different from P_1 and P_2 to represent that all parties are honest) and, as mentioned above, two probabilities $p_{\mathtt{exec}}, p_{\mathtt{commit}}$. Let us remark that $\mathcal{A}$ has an auxiliary input, and that all parties are initialized with the same value on their security parameter tape (including the trusted party), but we leave both of these aspects implicit for clarity.

An unusual aspect of this ideal world is the presence of an "observation of the environment" which happens after the commitment has been made but before the computation. The idea being that a party has committed to an input if they are unable to make it depend on something they learnt between commitment and computation. We assume that this observation is drawn from some distribution $\mathcal{E}$ and that the distribution can be sampled from by a polynomial time algorithm. This latter assumption is to stop the environment from encoding, say, collisions of zero for a secure hash function.

1. P_1 receives input. Party P_1 receives its prescribed input x.

2. Commitment of P_1's input. At this stage P_1 sends to the trusted party the input that it intends to use in a subsequent computation, which we denote by w. If $\mathtt{C} \neq \mathsf{P}_1$, then $w = x$, and otherwise $\mathcal{A}$ sets w to be an arbitrary valid input value in a way that might depend on x.

3. The environment is revealed The value e is sampled from the distribution $\mathcal{E}$ Party P_1 is given the value of e

4. P_2 receives input and parties send inputs. Party P_2 receives its prescribed input y. Next, P_1 and P_2 send to the trusted party their inputs to be used in the computation, denoted a, b, respectively.

- If $\mathtt{C} = \mathsf{P}_1$, then $\mathcal{A}$ sets

  $a \in \{w, \mathtt{abort}, \mathtt{corrupted}, \mathtt{cheat\_exec}, \mathtt{cheat\_commit}\}$ for P_1, and otherwise P_1 sets $a = w$.

- If $\mathtt{C} = \mathsf{P}_2$ then $\mathcal{A}$ sets b for P_2, otherwise P_2 sets $b = y$.

5. Early abort & blatant cheating. $\mathcal{A}$ is given the opportunity to have $\mathtt{C}$ abort or announce that it is corrupted. This results in updating either a (if $\mathtt{C} = \mathsf{P}_1$) or b (if $\mathtt{C} = \mathsf{P}_2$) to $\mathtt{abort}$ or $\mathtt{corrupted}$. If that is the case, a (resp. b) is forwarded to P_2 (resp. P_1) and the trusted party halts.

6. Attempted cheat option. If $a = \mathtt{cheat\_exec}$, then the trusted party tosses a coin $X = \mathtt{Ber}(p_{\mathtt{exec}})$, where $p_{\mathtt{exec}}$ is the probability of P_1 getting caught cheating at this stage, and

- If $X = 1$ then the trusted party sends $\mathtt{corrupted}$ to both P_1 and P_2.

- If $X = 0$ then the trusted party sends `undetected` to P_1, along with y (P_2's input). Following this, $\mathcal{A}$ gets to choose P_2's output of the protocol, and sends it to the trusted party.

The ideal execution ends at this point if $a =$ `cheat_exec`.

7. Attempted break commitment option. If $a =$ `cheat_commit` then the trusted party requests from P_1 (a) a probability q and (b) a new value w' for w. The trusted party then sets $p = q$, if $w = w'$, and $p = \max(q, p_{\texttt{commit}})$ otherwise, where $p_{\texttt{commit}}$ is the probability of P_1 getting caught cheating at this stage. (Note that this simply allows the adversary to choose an arbitrary probability of getting caught when cheating to rewrite w with the same value again). Then, the trusted party (i) tosses a coin $Y = \texttt{Ber}(p)$, (ii) rewrites w to take value w', (iii) runs $g_1(w, b)$ with the updated w, and (iv) gives $\mathcal{A}$ the opportunity to abort P_1. Next,

- if $Y = 1$ then the trusted party sends `corrupted` to both P_1 and P_2 and halts, and
- if $Y = 0$ then the trusted party sends `undetected` to P_1.

Let us remark that giving $\mathcal{A}$ the opportunity to abort upon observing the output in in step 7. is allowed just to simplify the presentation of out protocol, and that an extension where $\mathcal{A}$ does not receive an output when caught cheating is easy to achieve by just adding a round of interaction to our protocol. In that extra round P_2 enables P_1 to ungarble their output after verifying the commitment resulting from the secure computation.

8. Trusted party gives out outputs. The trusted party evaluates $g(w, b)$, and gives $\mathcal{A}$ the chance to abort the execution. Otherwise it gives their designated output to P_2, at which point $\mathcal{A}$ is allowed to either abort the execution, or let the honest party receive their output.

Outputs. The honest party outputs what they received in the final step, and $\mathcal{A}$ outputs an arbitrary (probabilistic) polynomial-time computable function of C's input, any auxiliary input, and its view during the execution.

A.1.1 Definitions

The following simulation security definition deviates from most such definitions in that we allow the adversary in the ideal model to be logically omniscient, whereas it is standard to restrict the simulator to polynomial time computations. The polynomial time assumption is important in the context of zero-knowledge proofs and for certain systems of composability. However standard bit commitment is impossible in the universal composability model [381], thus we must settle for weaker composition guarantees here. We hope it is clear that the ideal setting here is information theoretically secure. Thus even a logically omniscient adversary cannot possibly learn things that it should not in the ideal model. The simulator could be made computable at the expense of slightly complicating the proof, but as this is unnecessary and also not standard we prefer to keep the proof simple.

Denote by $\texttt{IDEAL}^{p_{\texttt{exec}},p_{\texttt{commit}},\mathcal{E}}_{g,\mathcal{S}(z),i}(x,y,\lambda)$ the environment variable and the outputs of the honest parties and adversary in an execution in the ideal world above, and let $\texttt{REAL}^{p_{\texttt{exec}},p_{\texttt{commit}},\mathcal{E}}_{g,\mathcal{A}(z),i}(x,y,\lambda)$ denote the environment variable and the outputs of the honest parties and the adversary in a real execution of a protocol π.

Definition A.1.1. *Let g and p be as above. A protocol π* securely computes *g* with committed first input in the presence of a malicious P1 or a covert P2 with *p*-deterrent *if for every non-uniform probabilistic polynomial time adversary $\mathcal{A}$ for the real model, there exists a definable adversary $\mathcal{S}$ for the ideal model such that for each $i \in \{1,2\}$:*

$$\left\{\mathit{IDEAL}^{p_{exec},p_{commit},\mathcal{E}}_{g,\mathcal{S}(z),i}(x,y,\lambda)\right\}_{x,y,z\in\{0,1\}^*,\lambda\in\mathbb{N}} \equiv^c \left\{\mathit{REAL}^{p_{exec},p_{commit},\mathcal{E}}_{g,\mathcal{A}(z),i}(x,y,\lambda)\right\}_{x,y\in\{0,1\}^*,\lambda\in\mathbb{N}}$$

In PVC security it is important that a fail-stop adversary is not labelled a cheat (at least in most contexts including ours) for that we say that:

Definition A.1.2. *A protocol π is* non-halting detection accurate *if for every fail-stop adversary $\mathcal{A}$ controlling party P1, the probability of an honest P2 outputting* `corrupted` *is negligible.*

In order to have PVC security in place of the covert security we require that their be some algorithm `Blame`. When applied to the view of an honest party that has outputted `corrupted`, it must return a proof of that corruption. The proof is verified by another algorithm `Judgement`, which will output $\mathsf{cheated}$ if and only if it is a genuine proof. These ideas are formalized as follows.

Given an algorithm `Commit` and a protocol $\mathcal{P}$ let the commitment protocol formed by them consist of P1 running `Commit` and sending the result to P2, P1 then receiving $e \leftarrow \mathcal{E}$ and then both P1 and P2 engaging in $\mathcal{P}$ and taking the output from that protocol as the output.

Definition A.1.3. *A quadruple* $(\mathit{Commit}, \mathcal{P}, \mathit{Blame}, \mathit{Judgement})$ securely computes g with committed first input in the presence of a malicious P1 or a covert P2 with p-deterrent and public verifiability *if the following hold:*

1. *(Simulatability with p-deterrent:) The commitment protocol formed from* `Commit` *and $\mathcal{P}$ securely computes g with committed first input in the presence of a malicious P1 or a covert P2 with p-deterrent and is non-halting detection accurate.*

2. *(Accountability:) For every PPT adversary $\mathcal{A}$ controlling P1 and interacting with an honest P2,*

$$\mathbb{P}(\mathsf{P}_2 \text{ outputs } \mathit{corrupted} \wedge \mathit{Judgement}(\mathit{Blame}(\mathit{View}(\mathsf{P}_2)))) \neq \mathsf{cheated})$$

   is negligible.

3. *(Defamation-Free:) For every PPT adversary $\mathcal{A}$ controlling P2 and interacting with an honest P1,*

$$\mathbb{P}(\mathcal{A} \text{ outputs } \wedge \texttt{Judgement}(Cert) = \mathsf{cheated})$$

is negligible.

A.2 Proofs

Theorem 5.5.9. *The quadruple* $(\texttt{Commit}, \mathcal{P}, \texttt{Blame}, \texttt{Judgement})$ *securely computes g with committed first input in the presence of a malicious* `P1` *or a covert* `P2` *with $p/2$-deterrent and public verifiability.*

If pvccommit *and* check *are used as given in the previous section then we can replace the $\mathcal{P}$ with the protocol in Fig. 5.5 and still have the same security guarantee.*

Furthermore, if in either case it can be guaranteed that `P1` *is honest in running the commitment algorithm in Fig. 5.3, then the deterrent factor improves from $p/2$ to p.*

Proof. **Simulating P1** First we consider simulatability in the case where P1 is corrupted. Given a non-uniform probabilistic polynomial time adversary $\mathcal{A}$, we construct $\mathcal{S}$ as follows.

First note that $\mathcal{S}$ can uniformly randomly choose a randomness tape which will be used for all of its black box runs of $\mathcal{A}$, this reduces the task to the case where $\mathcal{A}$ is a deterministic adversary.

By running $\mathcal{A}$ the simulator is given the commitment c that $\mathcal{A}$ chooses to use (which may or may not be generated by applying `Commit` to some w). Now $\mathcal{S}$ can look at how $\mathcal{A}$ would respond to every possible environment variable e (this is fine because it is a mathematical function which need not be computable). If, in response to e, $\mathcal{A}$ does any of early abort, blatant cheat or cheat during the execution the outcome is independent of what commitment was made so it would not matter what $\mathcal{S}$ commits

to in the ideal world. The other possibility is that $\mathcal{A}$ does none of those things and submits some x' as their input alongside supposed randomness r'.

The underlying protocol Π allows input extraction in polynomial time (as is used in the proofs of security for that protocol by Wang [268]) thus in all of these other cases $\mathcal{S}$ can extract which x' will be used in response to each e. For each one $\mathcal{S}$ can then compute $a = \mathsf{assert}(x', \mathsf{sk}, r'; i, \mathsf{pk})$ for every $i \in \mathcal{I}$ and check for what fraction of the i we have $\mathsf{check}(c, a, \mathsf{pk}) = \mathsf{cheated}$.

For those e that result in being caught with probability at least $p/2$ it would not matter what $\mathcal{S}$ committed to as it will be able to attempt to cheat the commitment to change the input to x' and get caught with the correct probability. After which it will receive $g_1(x', y)$, add it to the simulated view, and proceed according to what $\mathcal{A}$ would do next. Aborting if and only if $\mathcal{A}$ chooses to abort. P2 will then receive `corrupted` with the correct probability.

Those e that result in less than $p/2$ probability of being caught, it will matter that $\mathcal{S}$ commited to the value of x' that $\mathcal{A}$ wants to use. Thus for the commitment phase $\mathcal{S}$ will commit with the trusted party to the value $\tilde{x}$ that is most likely to be used as x' (with respect to the randomness of e). If the adversary uses $x' = \tilde{x}$ then $\mathcal{S}$ will now be able to tell the trusted party it wants to use that value and it wants to get caught with the correct probability.

The remaining possibility, that e results in $\mathcal{A}$ using an input $x' \neq \tilde{x}$ and r' which has a probability less than $p/2$ of resulting in P1 being caught, would be a serious problem for $\mathcal{S}$. We claim however that this can happen with only negligible probability.

Suppose to the contrary that some non-negligible fraction of the weight of $\mathcal{E}$ resulted in these bad x', r'. Then as each must individually have weight at most that assigned to $\tilde{x}$ we can construct a polynomial time algorithm as follows.

Sample $e \leftarrow \mathcal{E}$ and extract the input of $\mathcal{A}$ for this e, compute $a = \mathsf{assert}(x', \mathsf{sk}, r'; i, \mathsf{pk})$ for all $i \in \mathcal{I}$, check to see if less than a $p/2$ fraction of the is would result in $\mathsf{check}(c, a, \mathsf{pk}) = \mathsf{cheated}$. Repeat this process until two distinct such values of (x', r')

have been found with this property. As the fraction of e that result in finding such an x' other than the most common one is non-negligible, this algorithm runs in expected polynomial time.

However, this can (by putting a polynomial time upper bound on the run time and failing if it reaches it) be converted into a PPT algorithm which contradicts the general binding property of the PVC commitment scheme. Thus proving the claim.

This addresses simulating the correct distribution between e, the output of P2 and messages explicitly sent in our protocol to P1. The messages sent to P1 in the secure computation black box are dealt with by the simulator for Π as given by Wang [268].

Extending this to the optimized integrity check case is straight forward, everything is the same except $\mathcal{S}$ must produce a fake hash-commitment for $\mathcal{A}$ to sign. This can be done by hashing randomness due to the hiding property of the commitment scheme this would not break indistinguishability. Further whilst the signed version should be given to P2 in the real world it does not form part of P2's output so we need not worry about coordinating with that.

If the commitment had been produced honestly, then $\mathcal{A}$ must have some x, r that it committed to that it knows of. This together with any value of x', r' that collides with the resulting commitment c less than some fraction p of the time will break general binding. Thus the same simulator as above with this extra observation gives the stronger security.

Simulating P2 Simulating the other side is much easier. P2 receives a commitment c to some x, however due to the hiding property of the commitment $\mathcal{S}$ can get away with providing $\mathcal{A}$ with a commitment to some arbitrary value, say 0. The simulator can now have $\mathcal{A}$ interact with a copy of P1 (multiple times) in order to extract their input y and index i. It can then give this input to the trusted party to find the correct value of $g_2(x, y)$. It can then add the result of $g_2(x, y)$ to the simulation of the adversaries view using the simulator for Π.

With the optimization the only change is that rather than giving the value of assert for the given i, a pair $(i, c[i])$ is signed and added to the simulated view.

Correctness If both parties are honest then Π will correctly output $(g_1, (x, y), g_2(x, y))$ and the result of assert. As assert is computed correctly on the honest inputs the check with the commitment from the first step will return valid. And thus P2 will accept the output and the parties will have successfully computed g.

Accountability For accountability, we need to show that a cheating P_1 gets caught *publicly*, in the sense that if P_2 claims that P_1 cheated then there's a proof accompanying that claim except with negligible probability. Note that if P_1 cheats inside the secure computation with Π, this follows from the PVC properties of Π, and the definition of Blame and Judgemenet in terms of $\mathtt{Blame}_{\mathtt{exec}}$ and $\mathtt{Judgement}_{\mathtt{exec}}$. If P_1 cheats in that the input to Π differs from the committed value then $\mathtt{Judgement}_{\mathtt{commit}}(\mathtt{Blame}_{\mathtt{commit}}(\cdot))$ will output cheated after verifying that $\mathtt{Blame}_{\mathtt{commit}}(\cdot)$ constitutes a valid signature of the fact that the commitment c and evaluation of H at i do not match. This happen with all but negligible probability due to the properties of the cryptographic signature, and the correctness of the PVC commitment scheme, i.e. different values for indexed hashes necessarily come from different inputs.

Defamation Freeness Defamation freeness states that proofs of cheating cannot be forged. This holds for proofs outputted by Π by the fact that it satisfies PVC security, and it holds for proofs generated by $\mathtt{Blame}_{\mathtt{exec}}$ due to the properties of the cryptographic signature scheme.

□

Proposition 1. *Given any non-trivially collision bounded family of indexed hash functions $\{H_k\}_{k\in K}$ with H_k given by the (polynomial size) circuit C_k with n-bit main input, and m-bit output. With all but negligible probability over the generation of $k = G(\lambda)$, the circuit C_k must have at least $\lceil (n-m)/2 \rceil$ nonlinear gates.*

Proof. We give a polynomial time algorithm which, given a circuit, C, that implements a function from $\{0,1\}^a \times \{0,1\}^n$ to $\{0,1\}^m$ and contains fewer than $\lceil (n-m)/2 \rceil$ nonlinear gates, returns a non-zero input $x \in \{0,1\}^n$ such that $C(s,x) = C(s,0)$ for all $s \in \{0,1\}^a$. As this algorithm finds a 1-collision with certainty if the circuit is small enough, for H_k to be secure that must happen with negligible probability in λ. And the result is immediate.

Consider the wires of C that are either outputs of the circuit or inputs to nonlinear gates. The hypotheses imply that there are $< n$ of such wires. Wire j must contain the XOR of a linear (i.e. parity) function f_j of the input with an affine function of the key and nonlinear gate outputs (either of which could be trivial).

The conditions $f_j(x) = f_j(0)$ form a collection of $< n$ linear constraints in n variables. Since $x = 0$ is obviously a solution of this under-determined system, then it must have also a nontrivial solution, which can be found efficiently. □

Proposition 2. *Let $\{h_k\}_{k \in K}$ be a collision resistant family of hash functions with h_k given by the circuit C_k with n-bit input and m-bit output. With all but negligible probability with respect to the generation of $k = G(\lambda)$, the circuit C_k must have at least $n - m$ nonlinear gates.*

Proof. The idea of the proof is similar to that of Proposition 1. We give a polynomial time algorithm which given a circuit C from $\{0,1\}^n$ to $\{0,1\}^m$ with fewer than $n - m$ nonlinear gates returns a collision in that circuit. Thus to have collision resistance the circuit can be that small only with negligible probability.

Let d be the number of nonlinear gates in C, we will assume WLOG that they are AND gates. We will derive $d + m$ affine conditions on x which determine $C(x) = C(0)$. As affine systems are efficiently solved, collision resistance requires that there be at most one solution ($x = 0$) to this system. For this to happen $d + m \geq n$ must hold, which proves the statement.

Thus it remains to derive the aforementioned $d + m$ affine conditions. We need to fix the output of each AND gate using only one affine condition on x. This can be achieved as follows. Consider each AND gate in (a totalisation of the partial) order from input to output, i.e. a topological ordering of the circuit. For each AND in that sequence, if the first input can be set to 0 with an affine condition then add that condition to the set and move on to the next gate. Otherwise, the first input is already determined so we to need only add an affine condition that fixes the second input. Either way that is only one condition per gate. In summary, by adding one condition for each of the output wires we can determine their values, so we are done. □

Proposition 3. *Suppose that $\{H_k\}_{k\in K}$ is a non-trivially collision bounded and hiding family of hash functions. Let H_k be given by C_k with an n-bit main input and d nonlinear gates, then with all but negligible probability, $d \geq n/5$. Further if $|\mathcal{I}| = 1$, then $d \geq n/3$.*

Proof. For a indexed hash function circuit C with n-bit main input and d nonlinear gates we explain how to do each of the following in polynomial time in the size of C:

- Transform C into another circuit $\tilde{C}$ with d non-linear gates.
- Simulate the output of $\tilde{C}(i, r, x)$, given the output of $C(i, r, x)$ and i.
- Derive a collision in $\tilde{C}$ from a collision in C

Finally we will show that $\tilde{C}_k$ has output length at most $3d_k$ with all but negligible probability. It follows that $\{\tilde{C}_k\}_{k\in K}$ is non-trivially collision bounded and the result follows from Propositions 1 and 2.

Throughout this proof L with a subscript will denote a linear function.

Let C be a circuit with input (i, r, x) where x is the n-bit input, $r \in \mathcal{R}$ is the randomness and i is an index. All the following computations can be done in polynomial time we will avoid repeating this fact for each one.

Note that $C(i, r, x)$ can be rewritten as $L_1(i, r, x, g(L_2(i, r, x)))$ for some nonlinear function g where L_2 has a $2d$ bit output and g has a d bit output and is implemented with d non-linear gates.

Considering L_2 as a linear function of $\mathcal{R}$ we can find its kernel T which has codimension at most $2d$. Compute representations of $\pi_{T^\perp}(r)$ and $\pi_T(r)$, represented in a basis of $T^\perp$ and a basis of T, call them r_1 and r_2 respectively. Thus r_1 has length at most $2d$, and $L_2(i, r, x)$ is equal to some $L_3(i, r_1, x)$. We can thus write $C(i, r, x)$ as

$$L_4(i) + L_5(r_2) + L_6(r_1, x, g(L_3(i, r_1, x))) \tag{A.1}$$

We now define $\tilde{C}(i, r_1, x)$ to be a representation of

$$\pi_{(\mathrm{Im}L_5)^\perp}(C(i, r, x) - L_4(i)) = L_7(x, r_1, g(L_3(i, r_1, x))) \tag{A.2}$$

in a basis of $\mathrm{Im}L_7$.

As $\tilde{C}(i, r, x)$ is a linear function of $C(i, r, x)$ and i we can write it with d nonlinear gates.

As $\tilde{C}(i, r_1, x)$ is a known linear function of i and $C(i, r, x)$ so simulating it is trivial.

We abuse notation and use r for the function that recovers r from a derived r_1 and r_2. Suppose that $\tilde{C}(i, r_1, x) = \tilde{C}(i, r_1', x')$ and $x \neq x'$. Then we can compute $C(i, r(r_1, 0), x) - C(i, r(r_1', 0), x')$ which by the definition of $\tilde{C}$ will be in $\mathrm{Im}L_5$, we can then choose r_2 such that

$$C(i, r(r_1, 0), x) - C(i, r(r_1', 0), x') = L_5(r_2) \tag{A.3}$$

which combined with Equation A.1 yields

$$C(i, r(r_1, 0), x) = C(i, r(r_1', r_2), x') \tag{A.4}$$

Finally, recall that $\tilde{C}(i, r_1, x)$ is a representation of $L_7(x, r_1, g(L_3(i, r_1, x)))$ and that $\tilde{C}$ has full rank. Thus we can write each $\tilde{C}_k$ as

$$L_8^k(x) + L_9^k(r_1, g(L_3(i, r_1, x))) \tag{A.5}$$

As $(r_1, g(L_3(i, r_1, x)))$ is at most $3d$ bits long the rank of L_9^k must be at most $3d$. If $\mathrm{Im}L_8^k$ is contained in $\mathrm{Im}L_9^k$ then the length of the output of $\tilde{C}_k$ is at most $3d$. Otherwise, $\pi_{(\mathrm{Im}L_9^k)^\perp}\tilde{C}_k(i, r, x)$ is a non-trivial linear function of x, however this latter possibility must occur with negligible probability otherwise $\{\tilde{C}_k\}_{k\in K}$, and thus $\{C_k\}_{k\in K}$, is not hiding. □

Theorem 5.10.1. *Construction 4 is $b/|\mathcal{I}|$-collision bounded.*

Proof. By Theorem 5.4.9 it suffices to show that for two distinct inputs x, x' at most b values of i will result in $d_4(i, x) = d_4(i, x')$. Consider two distinct inputs x, x', and assume WLOG that they differ in the first b field elements. Let y and y' be the first b field elements from x and x' respectively.

It suffices to show that only b values of i will result in $d_4(i, y) = d_4(i, y')$. Start by expanding out the difference.

$$\begin{aligned} d_4(i, y) - d_4(i, y') = \sum_{j=1}^{b/2} \Big(& i^{2j-1}(y_{2j} - y'_{2j}) + \\ & i^{2j}(y_{2j-1} - y'_{2j-1}) + \\ & y_{2j-1}y_{2j} - y'_{2j-1}y'_{2j} \Big) \end{aligned}$$

Let s be the function on the natural numbers that swaps $2j$ and $2j-1$ for all j. The difference is given by the inner product of $(1, i, i^2, ..., i^b)$ with a vector $v(y, y')$ in $\mathbb{F}^{b+1}$ with zeroth entry being

$$\sum_{j=1}^{b/2} y_{2j-1}y_{2j} - y'_{2j-1}y'_{2j} \tag{A.6}$$

and jth entry for $j > 0$ being $y_{s(j)} - y'_{s(j)}$.

We have $d_4(i, y) = d_4(i, y')$ only if $(1, i, ..., i^b)$ is perpendicular to $v(y, y')$. But as any $b+1$ vectors of the form $(1, i, ..., i^b)$ form a Vandermonde matrix and thus are linearly independent, it follows that at most b of them could be perpendicular to any fixed $v(y, y')$. □

References

[1] Rosalind W Picard et al. "Author Response: Wrist Sensor Reveals Sympathetic Hyperactivity and Hypoventilation before Probable SUDEP." In: *Neurology* 90.15 (Apr. 2018), p. 713. pmid: 29632115.

[2] Ognjen Rudovic et al. "Personalized Machine Learning for Robot Perception of Affect and Engagement in Autism Therapy". In: *Science Robotics* 3.19 (June 2018), eaao6760. URL: http://robotics.sciencemag.org/lookup/doi/10.1126/scirobotics.aao6760.

[3] Andong Zhan et al. "Using Smartphones and Machine Learning to Quantify Parkinson Disease Severity". In: *JAMA Neurology* 75.7 (July 2018), p. 876. URL: http://archneur.jamanetwork.com/article.aspx?doi=10.1001/jamaneurol.2018.0809.

[4] Suchi Saria. "Individualized Sepsis Treatment Using Reinforcement Learning". In: *Nature Medicine* 24.11 (Nov. 2018), pp. 1641–1642. URL: http://www.nature.com/articles/s41591-018-0253-x.

[5] JB Heaton, Nicholas G Polson, and Jan Hendrik Witte. "Deep learning in finance". In: *arXiv preprint arXiv:1602.06561* (2016).

[6] Gökcen Eraslan et al. "Deep learning: new computational modelling techniques for genomics". In: *Nature Reviews Genetics* 20.7 (2019), pp. 389–403.

[7] Austin Tate. *Privacy – Examples of Misuse of Private Data*. 2014. URL: https://blog.inf.ed.ac.uk/atate/2014/02/24/examples-of-misuse-of-private-data%20[Accessed%20:%202018-11-22].

[8] Carole Cadwalladr and Emma Graham-Harrison. *How Cambridge Analytica Turned Facebook 'Likes' into a Lucrative Political Tool*. 2018. URL: https://www.theguardian.com/technology/2018/mar/17/facebook-cambridge-analytica-kogan-data-algorithm%20[Accessed:%202018-12-01].

[9] Andy Crabtree et al. "Building Accountability into the Internet of Things: The IoT Databox Model". In: *Journal of Reliable Intelligent Environments* 4.1 (Apr. 2018), pp. 39–55. URL: http://link.springer.com/10.1007/s40860-018-0054-5.

[10] Julia Angwin et al. "Machine Bias There's Software Used across the Country to Predict Future Criminals. And It's Biased against Blacks". In: *ProPublica* (May 2016). URL: https://criminaljusticesection.wordpress.com/2016/05/23/machine-bias-theres-software-used-across-the-country-to-predict-future-criminals-and-its-biased-against-blacks/.

[11] Jon Kleinberg and Jon. "Inherent Trade-Offs in Algorithmic Fairness". In: *Abstracts of the 2018 ACM International Conference on Measurement and Modeling of Computer Systems - SIGMETRICS '18*. New York, New York, USA: ACM Press, 2018, pp. 40–40. URL: http://dl.acm.org/citation.cfm?doid=3219617.3219634.

[12] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. "Explainable AI: A review of machine learning interpretability methods". In: *Entropy* 23.1 (2021), p. 18.

[13] Dominique Machuletz, Stefan Laube, and Rainer Böhme. "Webcam Covering as Planned Behavior". In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 2018, pp. 1–13.

[14] Alfred J Menezes et al. *Handbook of Applied Cryptography*. CRC press, 1996.

[15] Gentry and Craig. "A Fully Homomorphic Encryption Scheme". In: (2009). URL: https://dl.acm.org/citation.cfm?id=1834954.

[16] Moni Naor, Benny Pinkas, and Reuban Sumner. "Privacy Preserving Auctions and Mechanism Design". In: *Proceedings of the 1st ACM Conference on Electronic Commerce - EC '99*. New York, New York, USA: ACM Press, 1999, pp. 129–139. URL: http://portal.acm.org/citation.cfm?doid=336992.337028.

[17] Cynthia Dwork. "Differential Privacy: A Survey of Results". In: *International Conference on Theory and Applications of Models of Computation*. Springer. 2008, pp. 1–19.

[18] Payman Mohassel and Yupeng Zhang. "SecureML: A System for Scalable Privacy-Preserving Machine Learning". In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, May 2017, pp. 19–38. URL: http://ieeexplore.ieee.org/document/7958569/.

[19] Payman Mohassel and Peter Rindal. "ABY 3: A Mixed Protocol Framework for Machine Learning". In: *Proceedings of the 2018 ACM Conference on Computer and Communications Security*. ACM. 2018, pp. 35–52.

[20] Sameer Wagh, Divya Gupta, and Nishanth Chandran. "SecureNN: 3-Party Secure Computation for Neural Network Training". In: *Proceedings on Privacy Enhancing Technologies* 1 (2019), p. 24.

[21] Ehsan Hesamifard et al. "Privacy-Preserving Machine Learning as a Service". In: *Proceedings on Privacy Enhancing Technologies* 2018.3 (), pp. 123–142. URL: https://petsymposium.org/2018/files/papers/issue3/popets-2018-0024.pdf.

[22] Florian Bourse et al. "Fast Homomorphic Evaluation of Deep Discretized Neural Networks". In: *Annual International Cryptology Conference*. Springer. 2018, pp. 483–512.

[23] Nathan Dowlin et al. "CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy". In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning*. Vol. 48. New York, NY, USA: JMLR, 2016, pp. 201–210. URL: http://sealcrypto.codeplex.com.

[24] Amartya Sanyal et al. "TAPAS: Tricks to Accelerate (Encrypted) Prediction as a Service". In: *International Conference on Machine Learning*. 2018, pp. 4497–4506.

[25] Brendan McMahan et al. "Communication-Efficient Learning of Deep Networks from Decentralized Data". In: *Artificial Intelligence and Statistics*. PMLR. 2017, pp. 1273–1282.

[26] Nitin Agrawal et al. "QUOTIENT: Two-Party Secure Neural Network Training and Prediction". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 2019, pp. 1231–1247.

[27] Meiyin Wu and Li Chen. "Image recognition based on deep learning". In: *2015 Chinese Automation Congress (CAC)*. IEEE. 2015, pp. 542–546.

[28] Li Deng and John C Platt. "Ensemble deep learning for speech recognition". In: *Fifteenth annual conference of the international speech communication association*. 2014.

[29] Geoffrey E Hinton. "Learning multiple layers of representation". In: *Trends in cognitive sciences* 11.10 (2007), pp. 428–434.

[30] Jakub Konečn et al. "Federated Learning: Strategies for Improving Communication Efficiency". In: (2016). URL: `https://arxiv.org/pdf/1610.05492.pdf`.

[31] Shi-Cho Cha et al. "Privacy enhancing technologies in the Internet of Things: Perspectives and challenges". In: *IEEE Internet of Things Journal* 6.2 (2018), pp. 2159–2187.

[32] Nesrine Kaaniche, Maryline Laurent, and Sana Belguith. "Privacy enhancing technologies for solving the privacy-personalization paradox: Taxonomy and survey". In: *Journal of Network and Computer Applications* (2020), p. 102807.

[33] Nitin Agrawal et al. "MPC-Friendly Commitments for Publicly Verifiable Covert Security". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 2021.

[34] Nitin Agrawal et al. "Exploring Design and Governance Challenges in the Development of Privacy-Preserving Computation". In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 2021, pp. 1–13.

[35] Xavier Glorot and Yoshua Bengio. "Understanding the Difficulty of Training Deep Feedforward Neural Networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. 2013. URL: `http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf`.

[36] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. "A Fast Learning Algorithm for Deep Belief Nets". In: *Neural Computation* 18.7 (July 2006), pp. 1527–1554. pmid: `16764513`. URL: `http://www.ncbi.nlm.nih.gov/pubmed/16764513%20http://www.mitpressjournals.org/doi/10.1162/neco.2006.18.7.1527`.

[37] Pascal Vincent et al. "Extracting and Composing Robust Features with Denoising Autoencoders". In: *Proceedings of the 25th International Conference on Machine Learning*. ACM, 2008. URL: `http://machinelearning.org/archive/icml2008/papers/592.pdf`.

[38] Trevor Hastie, Robert Tibshirani, and J. H. (Jerome H.) Friedman. *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*, p. 745.

[39] K.S. Narendra and K. Parthasarathy. "Identification and Control of Dynamical Systems Using Neural Networks". In: *IEEE Transactions on Neural Networks* 1.1 (Mar. 1990), pp. 4–27. URL: http://ieeexplore.ieee.org/document/80202/.

[40] Yoshua Bengio et al. "Greedy Layer-Wise Training of Deep Networks". In: *Advances in Neural Information Processing Systems*. 2007, pp. 153–160. URL: http://www.iro.umontreal.ca/~lisa/pointeurs/BengioNips2006All.pdf.

[41] Yoshua Bengio. "Learning Deep Architectures for AI". In: *Machine Learning* 2.1 (2009), pp. 1–127. URL: http://www.iro.umontreal.ca/~bengioy/papers/ftml_book.pdf.

[42] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[43] Jakub Konecny et al. *Federated Optimization: Distributed Machine Learning for on-Device Intelligence*. 2016. arXiv: 1610.02527.

[44] H Brendan McMahan et al. "Federated learning of deep networks using model averaging". In: *arXiv preprint arXiv:1602.05629* (2016).

[45] Virginia Smith et al. "Federated multi-task learning". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017, pp. 4427–4437.

[46] Yue Zhao et al. "Federated learning with non-iid data". In: *arXiv preprint arXiv:1806.00582* (2018).

[47] Keith Bonawitz et al. "Practical Secure Aggregation for Privacy-Preserving Machine Learning". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security - CCS '17*. New York, New York, USA: ACM Press, 2017, pp. 1175–1191. URL: http://dl.acm.org/citation.cfm?doid=3133956.3133982.

[48] Robin C Geyer, Tassilo Klein, and Moin Nabi. "Differentially private federated learning: A client level perspective". In: *arXiv preprint arXiv:1712.07557* (2017).

[49] Tyler Hunt et al. "Chiron: Privacy-Preserving Machine Learning as a Service". In: *undefined* (2018). URL: https://www.semanticscholar.org/paper/Chiron%3A-Privacy-preserving-Machine-Learning-as-a-Hunt-Song/b3f2a11d45757e675be123d55ec0eb192bcca990.

[50] Dan Bogdanov et al. "Privacy-Preserving Statistical Data Analysis on Federated Databases". In: Springer, Cham, 2014, pp. 30–55. URL: http://link.springer.com/10.1007/978-3-319-06749-0_3.

[51] Hassan Takabi, James B. D. Joshi, and Gail-Joon Ahn. "SecureCloud: Towards a Comprehensive Security Framework for Cloud Computing Environments". In: *2010 IEEE 34th Annual Computer Software and Applications Conference Workshops*. IEEE, July 2010, pp. 393–398. URL: http://ieeexplore.ieee.org/document/5615236/.

[52] Deyan Chen and Hong Zhao. "Data Security and Privacy Protection Issues in Cloud Computing". In: *2012 International Conference on Computer Science and Electronics Engineering*. IEEE, Mar. 2012, pp. 647–651. URL: http://ieeexplore.ieee.org/document/6187862/.

[53] Andrew C Yao. "Protocols for Secure Computations". In: *23rd Annual Symposium on Foundations of Computer Science (Sfcs 1982)*. IEEE. 1982, pp. 160–164.

[54] Corinna Cortes and Vladimir Vapnik. "Support-Vector Networks". In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. URL: http://link.springer.com/10.1007/BF00994018.

[55] Jianjiang Feng and A K Jain. "Fingerprint Reconstruction: From Minutiae to Phase". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.2 (Feb. 2011), pp. 209–223. URL: http://ieeexplore.ieee.org/document/5432222/.

[56] Mohammad Al-Rubaie and J. Morris Chang. "Reconstruction Attacks against Mobile-Based Continuous Authentication Systems in the Cloud". In: *IEEE Transactions on Information Forensics and Security* 11.12 (Dec. 2016), pp. 2648–2663. URL: http://ieeexplore.ieee.org/document/7523420/.

[57] Naveed Akhtar and Ajmal Mian. *Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey*. URL: https://arxiv.org/pdf/1801.00553.pdf.

[58] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. "Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 2015, pp. 1322–1333.

[59] Reza Shokri et al. "Membership Inference Attacks against Machine Learning Models". In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, May 2017, pp. 3–18. URL: http://ieeexplore.ieee.org/document/7958568/.

[60] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. "An Empirical Exploration of Recurrent Network Architectures". In: *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. 2015. URL: http://proceedings.mlr.press/v37/jozefowicz15.pdf?utm_campaign=Revue%20newsletter&utm_medium=Newsletter&utm_source=revue.

[61] Yu Cheng et al. "Risk prediction with electronic health records: A deep learning approach". In: *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM. 2016, pp. 432–440.

[62] Jie-Zhi Cheng et al. "Computer-aided diagnosis with deep learning architecture: applications to breast lesions in US images and pulmonary nodules in CT scans". In: *Scientific reports* 6.1 (2016), pp. 1–13.

[63] Andre Esteva et al. "Dermatologist-level classification of skin cancer with deep neural networks". In: *nature* 542.7639 (2017), pp. 115–118.

[64] Zachary C Lipton et al. "Learning to diagnose with LSTM recurrent neural networks". In: *arXiv preprint arXiv:1511.03677* (2015).

[65] Edward Choi et al. "Doctor ai: Predicting clinical events via recurrent neural networks". In: *Machine learning for healthcare conference*. PMLR. 2016, pp. 301–318.

[66] Rasool Fakoor et al. "Using deep learning to enhance cancer diagnosis and classification". In: *Proceedings of the international conference on machine learning*. Vol. 28. ACM, New York, USA. 2013.

[67] Jian Zhou and Olga G Troyanskaya. "Predicting effects of noncoding variants with deep learning–based sequence model". In: *Nature methods* 12.10 (2015), pp. 931–934.

[68] Qiang Song, Anqi Liu, and Steve Y Yang. "Stock portfolio selection using learning-to-rank algorithms with news sentiment". In: *Neurocomputing* 264 (2017), pp. 20–28.

[69] Jiequn Han, Arnulf Jentzen, and E Weinan. "Solving high-dimensional partial differential equations using deep learning". In: *Proceedings of the National Academy of Sciences* 115.34 (2018), pp. 8505–8510.

[70] Ebberth L Paula et al. "Deep learning anomaly detection as support fraud investigation in brazilian exports and anti-money laundering". In: *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE. 2016, pp. 954–960.

[71] C Srinivasa Kumar et al. "An Adaptive Deep Learning Model to Forecast Crimes". In: *Proceedings of Integrated Intelligence Enable Networks and Computing*. Springer, 2021, pp. 459–470.

[72] Patricia Craja, Alisa Kim, and Stefan Lessmann. "Deep learning for detecting financial statement fraud". In: *Decision Support Systems* 139 (2020), p. 113421.

[73] Ivan Damgård and Mads Jurik. "A Generalisation, a Simpli.Cation and Some Applications of Paillier's Probabilistic Public-Key System". In: Springer, Berlin, Heidelberg, 2001, pp. 119–136. URL: `http://link.springer.com/10.1007/3-540-44586-2_9`.

[74] R. L. Rivest, A. Shamir, and L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". In: *Communications of the ACM* 21.2 (Feb. 1978), pp. 120–126. URL: `http://portal.acm.org/citation.cfm?doid=359340.359342`.

[75] David Naccache and Jacques Stern. "A New Public Key Cryptosystem Based on Higher Residues". In: *Proceedings of the 5th ACM Conference on Computer and Communications Security - CCS '98*. New York, New York, USA: ACM Press, 1998, pp. 59–66. URL: `http://portal.acm.org/citation.cfm?doid=288090.288106`.

[76] Taher ElGamal. "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms". In: *Advances in Cryptology*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1984, pp. 10–18. URL: `http://link.springer.com/10.1007/3-540-39568-7_2`.

[77] Josh Benaloh. "Dense probabilistic encryption". In: *Proceedings of the workshop on selected areas of cryptography*. 1994, pp. 120–128.

[78] Tatsuaki Okamoto and Shigenori Uchiyama. "A New Public-Key Cryptosystem as Secure as Factoring". In: Springer, Berlin, Heidelberg, 1998, pp. 308–318. URL: http://link.springer.com/10.1007/BFb0054135.

[79] Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. "Multi-Bit Cryptosystems Based on Lattice Problems". In: *Public Key Cryptography – PKC 2007*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 315–329. URL: http://link.springer.com/10.1007/978-3-540-71677-8_21.

[80] Shafi Goldwasser and Silvio Micali. "Probabilistic Encryption & How to Play Mental Poker Keeping Secret All Partial Information". In: *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing - STOC '82*. New York, New York, USA: ACM Press, 1982, pp. 365–377. URL: http://portal.acm.org/citation.cfm?doid=800070.802212.

[81] Pascal Paillier. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes". In: *Advances in Cryptology — EUROCRYPT '99*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 223–238. URL: http://link.springer.com/10.1007/3-540-48910-X_16.

[82] Peter L. Montgomery and Peter L. Montgomery. "A Survey of Modern Integer Factorization Algorithms". In: *CWI QUARTERLY* 7 (1994), pp. 337–366. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.2831.

[83] R L Rivest, L Adleman, and M L Dertouzos. "On Data Banks and Privacy Homomorphisms". In: *Foundations of Secure Computation, Academia Press* (1978), pp. 169–179.

[84] Burt Kaliski. "Quadratic Residuosity Problem". In: *Encyclopedia of Cryptography and Security*. Springer US, 2005, pp. 493–493. URL: http://www.springerlink.com/index/10.1007/0-387-23483-7_336.

[85] Tibor Jager. "The Generic Composite Residuosity Problem". In: *Black-Box Models of Computation in Cryptology*. Wiesbaden: Vieweg+Teubner Verlag, 2012, pp. 49–56. URL: http://www.springerlink.com/index/10.1007/978-3-8348-1990-1_5.

[86] Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. "GAZELLE: A Low Latency Framework for Secure Neural Network Inference". In: *USENIX Security Symposium*. Baltimore, MD, USA, 2018. URL: https://www.semanticscholar.org/paper/GAZELLE%3A-A-Low-Latency-Framework-for-Secure-Neural-Juvekar-Vaikuntanathan/deed54256b13054855838a4df00f0b87ad438556.

[87] Steven D. Galbraith and Steven D. "Elliptic Curve Paillier Schemes". In: *Journal of Cryptology* 15.2 (Jan. 2002), pp. 129–138. URL: http://link.springer.com/10.1007/s00145-001-0015-6.

[88] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. "Evaluating 2-DNF Formulas on Ciphertexts". In: Springer, Berlin, Heidelberg, 2005, pp. 325–341. URL: http://link.springer.com/10.1007/978-3-540-30576-7_18.

[89] Michael R Fellows and Neal Koblitz. "Combinatorial Cryptosystems Galore!" In: *Contemporary Mathematics* 168 (1994), p. 51. URL: https://pdfs.semanticscholar.org/2f73/36787a993b65c9e0aae886e4b82f660be2d6.pdf.

[90] Le Van Ly. "Polly Two : A New Algebraic Polynomial-Based Public-Key Scheme". In: *Applicable Algebra in Engineering, Communication and Computing* 17.3-4 (Aug. 2006), pp. 267–283. URL: http://link.springer.com/10.1007/s00200-006-0010-0.

[91] Françoise Levy-dit-Vehel et al. "A Survey on Polly Cracker Systems". In: *Gröbner Bases, Coding, and Cryptography*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 285–305. URL: http://link.springer.com/10.1007/978-3-540-93806-4_16.

[92] Tomas Sander, Adam Young, and Moti Yung. "Foundations of Computer Science : Proceedings of the Annual Symposium, New York, 1999." In: *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 1999, p. 800. URL: https://dl.acm.org/citation.cfm?id=796534.

[93] Carlos Aguilar Melchor, Philippe Gaborit, and Javier Herranz. "Additively Homomorphic Encryption with D-Operand Multiplications". In: Springer, Berlin, Heidelberg, 2010, pp. 138–154. URL: http://link.springer.com/10.1007/978-3-642-14623-7_8.

[94] Yuval Ishai and Anat Paskin. "Evaluating Branching Programs on Encrypted Data". In: *Theory of Cryptography*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 575–594. URL: http://link.springer.com/10.1007/978-3-540-70936-7_31.

[95] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. "Public-Key Cryptosystems from Lattice Reduction Problems". In: Springer, Berlin, Heidelberg, 1997, pp. 112–131. URL: http://link.springer.com/10.1007/BFb0052231.

[96] Marten van Dijk et al. "Fully Homomorphic Encryption over the Integers". In: Springer, Berlin, Heidelberg, 2010, pp. 24–43. URL: http://link.springer.com/10.1007/978-3-642-13190-5_2.

[97] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. "Scale-Invariant Fully Homomorphic Encryption over the Integers". In: Springer, Berlin, Heidelberg, 2014, pp. 311–328. URL: http://link.springer.com/10.1007/978-3-642-54631-0_18.

[98] N. P. Smart and F. Vercauteren. "Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes". In: Springer, Berlin, Heidelberg, 2010, pp. 420–443. URL: http://link.springer.com/10.1007/978-3-642-13013-7_25.

[99] Nitesh Aggarwal, CP Gupta, and Iti Sharma. "Fully Homomorphic Symmetric Scheme without Bootstrapping". In: *Proceedings of 2014 International Conference on Cloud Computing and Internet of Things*. IEEE, Dec. 2014, pp. 14–17. URL: http://ieeexplore.ieee.org/document/7062497/.

[100] Y. Govinda Ramaiah and G Vijaya Kumari. "Efficient Public Key Homomorphic Encryption over Integer Plaintexts". In: *2012 International Conference on Information Security and Intelligent Control*. IEEE, Aug. 2012, pp. 123–128. URL: http://ieeexplore.ieee.org/document/6449723/.

[101] Zvika Brakerski and Vinod Vaikuntanathan. "Fully Homomorphic Encryption from Ring-Lwe and Security for Key Dependent Messages". In: Springer, Berlin, Heidelberg, 2011, pp. 505–524. URL: http://link.springer.com/10.1007/978-3-642-22792-9_29.

[102] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) Fully Homomorphic Encryption without Bootstrapping". In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference on - ITCS '12*. New York, New York, USA: ACM Press, 2012, pp. 309–325. URL: http://dl.acm.org/citation.cfm?doid=2090236.2090262.

[103] Junfeng Fan and Frederik Vercauteren. "Somewhat Practical Fully Homomorphic Encryption". In: *undefined* (2012). URL: https://www.semanticscholar.org/paper/Somewhat-Practical-Fully-Homomorphic-Encryption-Fan-Vercauteren/531f8e756ea280f093138788ee896b3fa8ca085a.

[104] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. "NTRU: A Ring-Based Public Key Cryptosystem". In: Springer, Berlin, Heidelberg, 1998, pp. 267–288. URL: http://link.springer.com/10.1007/BFb0054868.

[105] Subhash Khot. "Hardness of Approximating the Shortest Vector Problem in Lattices". In: *Journal of the ACM (JACM)* 52.5 (2005), pp. 789–808.

[106] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. "On-the-Fly Multiparty Computation on the Cloud via Multikey Fully Homomorphic Encryption". In: *Proceedings of the 44th Symposium on Theory of Computing - STOC '12*. New York, New York, USA: ACM Press, 2012, p. 1219. URL: http://dl.acm.org/citation.cfm?doid=2213977.2214086.

[107] Joppe W. Bos et al. "Improved Security for a Ring-Based Fully Homomorphic Encryption Scheme". In: Springer, Berlin, Heidelberg, Dec. 2013, pp. 45–64. URL: http://link.springer.com/10.1007/978-3-642-45239-0_4.

[108] Yarkin Doröz and Berk Sunar. "Flattening NTRU for Evaluation Key Free Homomorphic Encryption". In: *undefined* (2016). URL: https://www.semanticscholar.org/paper/Flattening-NTRU-for-Evaluation-Key-Free-Homomorphic-Dor%C3%B6z-Sunar/b58dc9c0f88ef58f332f83f2486083e237adce4b.

[109] Bingxin Liu and Huapeng Wu. "Efficient Architecture and Implementation for NTRUEncrypt System". In: *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, Aug. 2015, pp. 1–4. URL: http://ieeexplore.ieee.org/document/7282143/.

[110] Yarkın Doröz et al. "Accelerating LTV Based Homomorphic Encryption in Reconfigurable Hardware". In: Springer, Berlin, Heidelberg, 2015, pp. 185–204. URL: http://link.springer.com/10.1007/978-3-662-48324-4_10.

[111] Kurt Rohloff and David Bruce Cousins. "A Scalable Implementation of Fully Homomorphic Encryption Built on NTRU". In: 2014, pp. 221–234. URL: http://link.springer.com/10.1007/978-3-662-44774-1_18.

[112] N P Smart and F Vercauteren. *Fully Homomorphic SIMD Operations*. URL: https://eprint.iacr.org/2011/133.pdf.

[113] Shai Halevi and Victor Shoup. "Algorithms in HElib". In: Springer, Berlin, Heidelberg, 2014, pp. 554–571. URL: http://link.springer.com/10.1007/978-3-662-44371-2_31.

[114] Shai Halevi and Victor Shoup. "Bootstrapping for HElib". In: Springer, Berlin, Heidelberg, 2015, pp. 641–670. URL: http://link.springer.com/10.1007/978-3-662-46800-5_25.

[115] Kim Laine. *Simple Encrypted Arithmetic Library 2.3.1*. URL: https://github.com/Microsoft/GSL/..

[116] Léo Ducas and Daniele Micciancio. "FHEW: Bootstrapping Homomorphic Encryption in Less than a Second". In: Springer, Berlin, Heidelberg, 2015, pp. 617–640. URL: http://link.springer.com/10.1007/978-3-662-46800-5_24.

[117] Craig Gentry, Amit Sahai, and Brent Waters. *Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based*. 2013. URL: https://eprint.iacr.org/2013/340.pdf.

[118] Ilaria Chillotti et al. *Faster Fully Homomorphic Encryption: Bootstrapping in Less than 0.1 Seconds*. URL: https://eprint.iacr.org/2016/870.pdf.

[119] Ilaria Chillotti et al. *Improving TFHE: Faster Packed Homomorphic Operations and Efficient Circuit Bootstrapping*. URL: https://eprint.iacr.org/2017/430.pdf.

[120] Wei Dai, Yarkın Doröz, and Berk Sunar. "Accelerating SWHE Based PIRs Using GPUs". In: 2015, pp. 160–171. URL: http://link.springer.com/10.1007/978-3-662-48051-9_12.

[121] Yarkın Doröz et al. *Toward Practical Homomorphic Evaluation of Block Ciphers Using Prince*. URL: https://eprint.iacr.org/2014/233.pdf.

[122] Jung Hee Cheon et al. "Homomorphic Encryption for Arithmetic of Approximate Numbers". In: Springer, Cham, 2017, pp. 409–437. URL: http://link.springer.com/10.1007/978-3-319-70694-8_15.

[123] Michael O Rabin. *How to Exchange Secrets with Oblivious Transfer*. 1981. URL: https://www.iacr.org/museum/rabin-obt/obtrans-eprint187.pdf.

[124] Shimon Even, Oded Goldreich, and Abraham Lempel. "A Randomized Protocol for Signing Contracts". In: *Communications of the ACM* 28.6 (June 1985), pp. 637–647. URL: http://portal.acm.org/citation.cfm?doid=3812.3818.

[125] O. Goldreich, S. Micali, and A. Wigderson. "How to Play ANY Mental Game". In: *Proceedings of the Nineteenth Annual ACM Conference on Theory of Computing - STOC '87*. New York, New York, USA: ACM Press, 1987, pp. 218–229. URL: http://portal.acm.org/citation.cfm?doid=28395.28420.

[126] Moni Naor and Benny Pinkas. "Computationally Secure Oblivious Transfer". In: *Journal of Cryptology* 18.1 (Jan. 2005), pp. 1–35. URL: http://link.springer.com/10.1007/s00145-004-0102-6.

[127] Dan Boneh. "The Decision Diffie-Hellman Problem". In: Springer, Berlin, Heidelberg, 1998, pp. 48–63. URL: http://link.springer.com/10.1007/BFb0054851.

[128] Yuval Ishai et al. "Extending Oblivious Transfers Efficiently". In: Springer, Berlin, Heidelberg, 2003, pp. 145–161. URL: http://link.springer.com/10.1007/978-3-540-45146-4_9.

[129] Gilad Asharov et al. "More Efficient Oblivious Transfer and Extensions for Faster Secure Computation". In: *Proceedings of the 2013 ACM Conference on Computer & Communications Security*. ACM. 2013, pp. 535–548.

[130] Xiao Wang, Alex J Malozemoff, and Jonathan Katz. *EMP-Toolkit: Efficient MultiParty Computation Toolkit*. 2016. URL: https://github.com/emp-toolkit.

[131] Andrew Chi-Chih Yao. "How to Generate and Exchange Secrets". In: *Foundations of Computer Science, 1986., 27th Annual Symposium On*. IEEE. 1986, pp. 162–167.

[132] D. Beaver, S. Micali, and P. Rogaway. "The Round Complexity of Secure Protocols". In: *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing - STOC '90*. New York, New York, USA: ACM Press, 1990, pp. 503–513. URL: http://portal.acm.org/citation.cfm?doid=100216.100287.

[133] M. Bellare et al. "Efficient Garbling from a Fixed-Key Blockcipher". In: *2013 IEEE Symposium on Security and Privacy*. IEEE, May 2013, pp. 478–492. URL: http://ieeexplore.ieee.org/document/6547128/.

[134] Yehuda Lindell and Benny Pinkas. "A Proof of Security of Yao's Protocol for Two-Party Computation". In: *Journal of Cryptology* 22.2 (Apr. 2009), pp. 161–188. URL: http://link.springer.com/10.1007/s00145-008-9036-8.

[135] Vladimir Kolesnikov and Thomas Schneider. "Improved Garbled Circuit: Free XOR Gates and Applications". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2008, pp. 486–498.

[136] Dahila Malkhi et al. "Fairplay—a Secure Two-Party Computation System". In: *SSYM'04 Proceedings of the 13th Conference on USENIX Security Symposium*. San Diego, CA, 2004, pp. 20–20. URL: https://dl.acm.org/citation.cfm?id=1251395.

[137] Wilko Henecka et al. "TASTY". In: *Proceedings of the 17th ACM Conference on Computer and Communications Security - CCS '10*. New York, New York, USA: ACM Press, 2010, p. 451. URL: http://portal.acm.org/citation.cfm?doid=1866307.1866358.

[138] Yan Huang et al. "Faster Secure Two-Party Computation Using Garbled Circuits". In: *USENIX Security Symposium* (2011). URL: https://www.semanticscholar.org/paper/Faster-Secure-Two-Party-Computation-Using-Garbled-Huang-Evans/15509fdb7fc7bf065fcdf776b38cf3d72d10c113.

[139] Benjamin Kreuter, Abhi Shelat, and Chih-Hao Shen. "Towards Billion-Gate Secure Computation with Malicious Adversaries". In: *undefined* (2012). URL: https://www.semanticscholar.org/paper/Towards-Billion-Gate-Secure-Computation-with-Kreuter-Shelat/7e9260f06dcf392040bb844b0a97a6ed35720860.

[140] Adi Shamir and Adi. "How to Share a Secret". In: *Communications of the ACM* 22.11 (Nov. 1979), pp. 612–613. URL: http://portal.acm.org/citation.cfm?doid=359168.359176.

[141] G. R. Blakley. "Safeguarding Cryptographic Keys". In: *Managing Requirements Knowledge, International Workshop on(AFIPS)*. New York: IEEE Computer Society, Dec. 1979, p. 313. URL: https://www.computer.org/csdl/proceedings/afips/1979/5087/00/50870313-abs.html.

[142] Michael Ben-Or, Avi Wigderson, and Avi Wigderson. "Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation". In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing - STOC '88*. New York, New York, USA: ACM Press, 1988, pp. 1–10. URL: http://portal.acm.org/citation.cfm?doid=62212.62213.

[143] David Chaum, Claude Crépeau, and Ivan Damgård. "Multiparty Unconditionally Secure Protocols". In: *Advances in Cryptology - CRYPTO '87*. Ed. by Carl Pomerance. Santa Barbara: Springer, 1988, pp. 462–462. URL: http://link.springer.com/10.1007/3-540-48184-2_43.

[144] Dan Bogdanov. "Sharemind: Programmable Secure Computations with Practical Applications". University of Tartu, Jan. 2013. URL: http://dspace.ut.ee/handle/10062/29041.

[145] Daniel Demmler, Thomas Schneider, and Michael Zohner. "ABY - a Framework for Efficient Mixed-Protocol Secure Two-Party Computation". In: *NDSS*. The Internet Society, 2015.

[146] Ivan Damgård et al. *Multiparty Computation from Somewhat Homomorphic Encryption*. URL: https://eprint.iacr.org/2011/535.pdf.

[147] Ivan Damgård et al. "Practical Covertly Secure MPC for Dishonest Majority-or: Breaking the SPDZ Limits". In: *LNCS* 8134 (2013), pp. 1–18. URL: http://dx..

[148] Samee Zahur and David Evans. *Obliv-c: A Language for Extensible Data-Oblivious Computation*. URL: https://pdfs.semanticscholar.org/2d25/81b990fd8b2df020cea5a6392b15f771bf0a.pdf.

[149] Peter Rindal and Mike Rosulek. *Faster Malicious 2-Party Secure Computation with Online/OOine Dual Execution **. 2016. URL: https://eprint.iacr.org/2016/632.pdf.

[150] Xiao Wang, Samuel Ranellucci, and Jonathan Katz. "Authenticated Garbling and Efficient Maliciously Secure Two-Party Computation". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2017, pp. 21–37.

[151] Xiao Wang, Samuel Ranellucci, and Jonathan Katz. "Global-Scale Secure Multiparty Computation". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2017, pp. 39–56.

[152] Marcel Keller, Valerio Pastro, and Dragos Rotaru. "Overdrive: Making SPDZ Great Again". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2018, pp. 158–189.

[153] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. URL: `www.tensorflow.org.`.

[154] Berry Schoenmakers. "MPyC—Python Package for Secure Multiparty Computation". In: *Workshop on the Theory and Practice of MPC. https://github. Com/Lschoe/Mpyc*. 2018.

[155] Dorothy E Denning, Peter J Denning, and Mayer D Schwartz. *The Tracker: A Threat to Statistical Database Security*. URL: `http://www.dbis.informatik.hu-berlin.de/fileadmin/lectures/SS2011/VL_Privacy/Tracker1.pdf`.

[156] Irit Dinur and Kobbi Nissim. "Revealing Information While Preserving Privacy". In: *Proceedings of the Twenty-Second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems - PODS '03*. New York, New York, USA: ACM Press, 2003, pp. 202–210. URL: `http://portal.acm.org/citation.cfm?doid=773153.773173`.

[157] Cynthia Dwork et al. "Our Data, Ourselves: Privacy via Distributed Noise Generation". In: Springer, Berlin, Heidelberg, 2006, pp. 486–503. URL: `http://link.springer.com/10.1007/11761679_29`.

[158] Cynthia Dwork et al. "Calibrating Noise to Sensitivity in Private Data Analysis". In: Springer, Berlin, Heidelberg, 2006, pp. 265–284. URL: `http://link.springer.com/10.1007/11681878_14`.

[159] Frank McSherry and Kunal Talwar. "Mechanism Design via Differential Privacy". In: *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*. IEEE, Oct. 2007, pp. 94–103. URL: `http://ieeexplore.ieee.org/document/4389483/`.

[160] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. "Smooth Sensitivity and Sampling in Private Data Analysis". In: *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing - STOC '07*. New York, New York, USA: ACM Press, 2007, p. 75. URL: `http://portal.acm.org/citation.cfm?doid=1250790.1250803`.

[161] Victor Costan and Srinivas Devadas. "Intel SGX Explained". In: *undefined* (2016). URL: `https://www.semanticscholar.org/paper/Intel-SGX-Explained-Costan-Devadas/a42e086f2382d518a0213879050e344539c2bcfa`.

[162] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. "Deep Sparse Rectifier Neural Networks". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings. 2011, pp. 315–323.

[163] Y. Lecun. "THE MNIST Database of Handwritten Digits". In: *http://yann.lecun.com/exdb/mnist/* (). URL: https://ci.nii.ac.jp/naid/10027939599/.

[164] Y. Lecun et al. "Gradient-Based Learning Applied to Document Recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. URL: http://ieeexplore.ieee.org/document/726791/.

[165] Matthieu Courbariaux et al. "Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1". In: (Feb. 2016). URL: http://arxiv.org/abs/1602.02830.

[166] Bita Darvish Rouhani, M. Sadegh Riazi, and Farinaz Koushanfar. "DeepSecure: Scalable Provably-Secure Deep Learning". In: *Proceedings of the 55th Annual Design Automation Conference on - DAC '18*. New York, New York, USA: ACM Press, 2018, pp. 1–6. URL: http://dl.acm.org/citation.cfm?doid=3195970.3196023.

[167] Jian Liu et al. *Oblivious Neural Network Predictions via Minionn Transformations*. ACM, 2017.

[168] M Sadegh Riazi et al. "Chameleon: A Hybrid Secure Computation Framework for Machine Learning Applications". In: *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*. 2018, pp. 707–721.

[169] Reza Shokri and Vitaly Shmatikov. "Privacy-Preserving Deep Learning". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15*. New York, New York, USA: ACM Press, 2015, pp. 1310–1321. URL: http://dl.acm.org/citation.cfm?doid=2810103.2813687.

[170] Le Trieu Phong et al. *Privacy-Preserving Deep Learning via Additively Homomorphic Encryption*. URL: https://eprint.iacr.org/2017/715.pdf.

[171] Yoshinori Aono et al. "Privacy-Preserving Logistic Regression with Distributed Data Sources via Homomorphic Encryption". In: *IEICE Transactions* 99 (D(8) 2016), pp. 2079–2089. URL: https://www.semanticscholar.org/paper/Privacy-Preserving-Logistic-Regression-with-Data-Aono-Hayashi/7f59aab601a18f7fc9659e7e8dc7562fdbac0ca6.

[172] Yoshinori Aono et al. "Scalable and Secure Logistic Regression via Homomorphic Encryption". In: *Proceedings of the Sixth ACM on Conference on Data and Application Security and Privacy - CODASPY '16*. New York, New York, USA: ACM Press, 2016, pp. 142–144. URL: http://dl.acm.org/citation.cfm?doid=2857705.2857731.

[173] Le Trieu Phong and Tran Thi Phuong. *Privacy-Preserving Deep Learning for Any Activation Function*. URL: https://arxiv.org/pdf/1809.03272.pdf.

[174] Li Wan et al. “Regularization of Neural Networks Using Dropconnect”. In: *IN ICML* (2013). URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.649.6992.

[175] Payman Mohassel and Peter Rindal. “ABY3: A Mixed Protocol Framework for Machine Learning”. In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security - CCS ’18*. New York, New York, USA: ACM Press, 2018, pp. 35–52. URL: http://dl.acm.org/citation.cfm?doid=3243734.3243760.

[176] Sameer Wagh, Divya Gupta, and Nishanth Chandran. *SecureNN: Efficient and Private Neural Network Training*. URL: https://eprint.iacr.org/2018/442.pdf.

[177] Donald Beaver. “Efficient Multiparty Protocols Using Circuit Randomization”. In: *Advances in Cryptology — CRYPTO ’91*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 420–432. URL: http://link.springer.com/10.1007/3-540-46766-1_34.

[178] Solon Barocas and Andrew D. Selbst. “Big Data’s Disparate Impact”. In: *California Law Review* 104 (2016), pp. 671–732. URL: https://www.ssrn.com/abstract=2477899.

[179] Toon Calders and Indrė Žliobaitė. “Why Unbiased Computational Processes Can Lead to Discriminative Decision Procedures”. In: Springer, Berlin, Heidelberg, 2013, pp. 43–57. URL: http://link.springer.com/10.1007/978-3-642-30487-3_3.

[180] Wim Schreurs et al. “Cogitas, Ergo Sum. The Role of Data Protection Law and Non-Discrimination Law in Group Profiling in the Private Sector”. In: *Profiling the European Citizen*. Dordrecht: Springer Netherlands, 2008, pp. 241–270. URL: http://link.springer.com/10.1007/978-1-4020-6914-7_13.

[181] Cynthia Dwork et al. “Fairness through Awareness”. In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference on - ITCS ’12*. New York, New York, USA: ACM Press, 2012, pp. 214–226. URL: http://dl.acm.org/citation.cfm?doid=2090236.2090255.

[182] Moritz Hardt Google, Eric Price, and Nathan Srebro. “Equality of Opportunity in Supervised Learning”. In: *Advances in Neural Information Processing Systems*. Red Hook, NY, 2016, pp. 3315–3323. URL: https://papers.nips.cc/paper/6374-equality-of-opportunity-in-supervised-learning.pdf.

[183] Matt J. Kusner et al. “Counterfactual Fairness”. In: *Advances in Neural Information Processing Systems*. Long Beach, CA, USA, 2017. URL: https://www.semanticscholar.org/paper/Counterfactual-Fairness-Kusner-Loftus/515ccd058edbbc588b4e98926897af5b1969b559.

[184] Muhammad Bilal Zafar et al. “Fairness beyond Disparate Treatment & Disparate Impact: Learning Classification without Disparate Mistreatment”. In: *Proceedings of the 26th International Conference on World Wide Web*. 2017, pp. 1171–1180.

[185] William Dieterich, Christina Mendoza, and MS Tim Brennan. *COMPAS Risk Scales: Demonstrating Accuracy Equity and Predictive Parity*. Northpointe, 2016. URL: http://go.volarisgroup.com/rs/430-MBX-989/images/ProPublica_Commentary_Final_070616.pdf.

[186] Richard Berk et al. "Fairness in Criminal Justice Risk Assessments". In: *Sociological Methods & Research* (July 2018), p. 004912411878253. URL: http://journals.sagepub.com/doi/10.1177/0049124118782533.

[187] Alexandra Chouldechova. "Fair Prediction with Disparate Impact: A Study of Bias in Recidivism Prediction Instruments". In: *Big Data* 5.2 (June 2017), pp. 153–163. URL: http://www.liebertpub.com/doi/10.1089/big.2016.0047.

[188] Michael Veale and Reuben Binns. "Fairer Machine Learning in the Real World: Mitigating Discrimination without Collecting Sensitive Data". In: *Original Research Article* (). URL: https://us.sagepub.com/en-us/nam/open-access-.

[189] Sara Hajian and Josep Domingo-Ferrer. "A Study on the Impact of Data Anonymization on Anti-Discrimination". In: *2012 IEEE 12th International Conference on Data Mining Workshops*. IEEE, Dec. 2012, pp. 352–359. URL: http://ieeexplore.ieee.org/document/6406462/.

[190] Sara Hajian. "Simultaneous Discrimination Prevention and Privacy Protection in Data Publishing and Mining". Universitat Rovira, June 2013. URL: http://arxiv.org/abs/1306.6805.

[191] Indrė Žliobaitė and Bart Custers. "Using Sensitive Personal Data May Be Necessary for Avoiding Discrimination in Data-Driven Decision Models". In: *Artificial Intelligence and Law* 24.2 (June 2016), pp. 183–201. URL: http://link.springer.com/10.1007/s10506-016-9182-5.

[192] Valeria Nikolaenko et al. "Privacy-Preserving Ridge Regression on Hundreds of Millions of Records". In: *2013 IEEE Symposium on Security and Privacy*. IEEE. 2013, pp. 334–348.

[193] Adrià Gascón et al. "Privacy-Preserving Distributed Linear Regression on High-Dimensional Data". In: *Proceedings on Privacy Enhancing Technologies* 2017.4 (2017), pp. 345–364.

[194] Valeria Nikolaenko et al. "Privacy-Preserving Matrix Factorization". In: *ACM Conference on Computer and Communications Security*. ACM, 2013, pp. 801–812.

[195] Niki Kilbertus et al. "Blind Justice: Fairness with Encrypted Sensitive Attributes". In: *International Conference on Machine Learning*. 2018, pp. 2635–2644.

[196] Phillipp Schoppmann, Adrià Gascón, and Borja Balle. "Private Nearest Neighbors Classification in Federated Databases". In: *IACR Cryptology ePrint Archive* 2018 (2018), p. 289.

[197] Ebrahim M Songhori et al. "Compacting Privacy-Preserving k-Nearest Neighbor Search Using Logic Synthesis". In: *2015 52nd ACM/EDAC/IEEE Design Automation Conference*. IEEE. 2015, pp. 1–6.

[198] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: 1502.03167.

[199] Johan Bjorck et al. *Understanding Batch Normalization*. 2018. arXiv: 1806.02375.

[200] Tim Salimans and Durk P Kingma. "Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks". In: *Advances in Neural Information Processing Systems*. 2016, pp. 901–909.

[201] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. *Layer Normalization*. 2016. arXiv: 1607.06450.

[202] John Duchi, Elad Hazan, and Yoram Singer. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". In: *Journal of Machine Learning Research* 12 (Jul 2011), pp. 2121–2159.

[203] Diederik P Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. arXiv: 1412.6980.

[204] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. "On the Convergence of Adam and Beyond". In: *International Conference on Learning Representations*. 2018.

[205] Google. *TensorFlow Lite*. 2018. URL: https://www.tensorflow.org/mobile/tflite.

[206] Benoit Jacob et al. "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference". In: *CVPR*. IEEE Computer Society, 2018, pp. 2704–2713.

[207] Suyog Gupta et al. "Deep Learning with Limited Numerical Precision". In: *International Conference on Machine Learning*. 2015, pp. 1737–1746.

[208] Daisuke Miyashita, Edward H Lee, and Boris Murmann. *Convolutional Neural Networks Using Logarithmic Data Representation*. 2016. arXiv: 1603.01025.

[209] Urs Köster et al. "Flexpoint: An Adaptive Numerical Format for Efficient Training of Deep Neural Networks". In: *Advances in Neural Information Processing Systems*. 2017, pp. 1742–1752.

[210] Shuang Wu et al. "Training and Inference with Integers in Deep Neural Networks". In: *International Conference on Machine Learning*. New Orleans, 2018. URL: https://www.semanticscholar.org/paper/Training-and-Inference-with-Integers-in-Deep-Neural-Wu-Li/acdf151b8efc2c6b05662d69f27531afc557dc85.

[211] Lu Hou, Ruiliang Zhang, and James T Kwok. "Analysis of Quantized Models". In: *International Conference on Learning Representations*. 2019.

[212] Dan Bogdanov, Sven Laur, and Jan Willemson. "Sharemind: A Framework for Fast Privacy-Preserving Computations". In: *European Symposium on Research in Computer Security*. Springer. 2008, pp. 192–206.

[213] Oded Goldreich, Silvio Micali, and Avi Wigderson. "How to Play Any Mental Game or A Completeness Theorem for Protocols with Honest Majority". In: *STOC*. ACM, 1987, pp. 218–229.

[214] Kartik Nayak et al. "GraphSC: Parallel Secure Computation Made Easy". In: *2015 IEEE Symposium on Security and Privacy*. IEEE. 2015, pp. 377–394.

[215] Henry S. Warren Jr. *Hacker's Delight, Second Edition*. Pearson Education, 2013. URL: http://www.hackersdelight.org/.

[216] Minje Kim and Paris Smaragdis. *Bitwise Neural Networks*. 2016. arXiv: 1601.06071.

[217] Itay Hubara et al. "Binarized Neural Networks". In: *Advances in Neural Information Processing Systems*. 2016, pp. 4107–4115.

[218] Mohammad Rastegari et al. "Xnor-Net: Imagenet Classification Using Binary Convolutional Neural Networks". In: *European Conference on Computer Vision*. Springer. 2016, pp. 525–542.

[219] Dan Alistarh et al. "QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding". In: *Advances in Neural Information Processing Systems*. 2017, pp. 1709–1720.

[220] Hao Li et al. "Training Quantized Nets: A Deeper Understanding". In: *Advances in Neural Information Processing Systems*. 2017, pp. 5811–5821.

[221] Wei Wen et al. "Terngrad: Ternary Gradients to Reduce Communication in Distributed Deep Learning". In: *Advances in Neural Information Processing Systems*. 2017, pp. 1509–1519.

[222] Jeremy Bernstein et al. "SIGNSGD: Compressed Optimisation for Non-Convex Problems". In: *International Conference on Machine Learning*. 2018, pp. 559–568.

[223] Christopher De Sa et al. *High-Accuracy Low-Precision Training*. 2018. arXiv: 1803.03383.

[224] M Sadegh Riazi et al. "{XONN}: XNOR-Based Oblivious Deep Neural Network Inference". In: *28th {USENIX} Security Symposium ({USENIX} Security 19)*. 2019, pp. 1501–1518.

[225] Niv Gilboa. "Two Party RSA Key Generation". In: *Annual International Cryptology Conference*. Springer. 1999, pp. 116–129.

[226] Martin Abadi et al. "Tensorflow: A System for Large-Scale Machine Learning". In: *12th USENIX Symposium on Operating Systems Design and Implementation*. 2016, pp. 265–283.

[227] Niv Gilboa. "Two party RSA key generation". In: *Annual International Cryptology Conference*. Springer. 1999, pp. 116–129.

[228] J. Ross Quinlan. "Simplifying Decision Trees". In: *International journal of man-machine studies* 27.3 (1987), pp. 221–234.

[229] Angel Cruz-Roa et al. "Automatic Detection of Invasive Ductal Carcinoma in Whole Slide Images with Convolutional Neural Networks". In: *Medical Imaging 2014: Digital Pathology*. Vol. 9041. International Society for Optics and Photonics. 2014, p. 904103.

[230] Mohammad Malekzadeh et al. *Mobile Sensor Data Anonymization*. 2018. arXiv: 1810.11546.

[231] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. "The HAM10000 Dataset, a Large Collection of Multi-Source Dermatoscopic Images of Common Pigmented Skin Lesions". In: *Scientific Data* 5 (Aug. 2018), p. 180161. URL: http://www.nature.com/articles/sdata2018161.

[232] Dua Dheeru and Efi Karra Taniskidou. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml.

[233] Daliang Li and Junpu Wang. *Fedmd: Heterogenous Federated Learning via Model Distillation*. 2019. arXiv: 1910.03581.

[234] Tribhuvanesh Orekondy et al. *Gradient-Leaks: Understanding and Controlling Deanonymization in Federated Learning*. 2018. arXiv: 1805.05838.

[235] Wenqi Wei et al. *A Framework for Evaluating Gradient Leakage Attacks in Federated Learning*. 2020. arXiv: 2004.10397.

[236] Sangchul Hahn and Heeyoul Choi. *Self-Knowledge Distillation in Natural Language Processing*. 2019. arXiv: 1908.01851.

[237] Andy Liaw, Matthew Wiener, et al. "Classification and Regression by randomForest". In: *R news* 2.3 (2002), pp. 18–22.

[238] Tianqi Chen and Carlos Guestrin. "Xgboost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*. 2016, pp. 785–794.

[239] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. 2015. arXiv: 1503.02531.

[240] Cristian Bucilu, Rich Caruana, and Alexandru Niculescu-Mizil. "Model Compression". In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2006, pp. 535–541.

[241] Jimmy Ba and Rich Caruana. "Do Deep Nets Really Need to Be Deep?" In: *Advances in Neural Information Processing Systems*. 2014, pp. 2654–2662.

[242] Antti Tarvainen and Harri Valpola. "Mean Teachers Are Better Role Models: Weight-Averaged Consistency Targets Improve Semi-Supervised Deep Learning Results". In: *Advances in Neural Information Processing Systems*. 2017, pp. 1195–1204.

[243] Ying Zhang et al. "Deep Mutual Learning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4320–4328.

[244] Tommaso Furlanello et al. *Born Again Neural Networks*. 2018. arXiv: 1805.04770.

[245] Chenglin Yang et al. *Knowledge Distillation in Generations: More Tolerant Teachers Educate Better Students*. 2018. arXiv: 1805.05551.

[246] Nicolas Papernot et al. *Scalable Private Learning with Pate*. 2018. arXiv: 1802.08908.

[247] Slawomir Goryczka and Li Xiong. "A Comprehensive Comparison of Multiparty Secure Additions with Differential Privacy". In: *IEEE transactions on dependable and secure computing* 14.5 (2015), pp. 463–477.

[248] Stacey Truex et al. “A Hybrid Approach to Privacy-Preserving Federated Learning”. In: *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*. 2019, pp. 1–11.

[249] Hongyan Chang et al. *Cronus: Robust and Heterogeneous Collaborative Learning with Black-Box Knowledge Transfer*. 2019. arXiv: 1912.11279.

[250] Mohassel Payman and Zhang Yupeng. “Secureml: A System for Scalable Privacy-Preserving Machine Learning”. In: *Security and Privacy (SP), 2017 IEEE Symposium On*. 2017.

[251] Keith Bonawitz et al. “Federated Learning with Autotuned Communication-Efficient Secure Aggregation”. In: *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. IEEE. 2019, pp. 1222–1226.

[252] Theo Ryffel et al. *A Generic Framework for Privacy Preserving Deep Learning*. 2018. arXiv: 1811.04017.

[253] Piotr Bojanowski et al. “Enriching Word Vectors with Subword Information”. In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146.

[254] Jamie Hayes and Olga Ohrimenko. “Contamination Attacks and Mitigation in Multi-Party Machine Learning”. In: *Advances in Neural Information Processing Systems* 31 (2018), pp. 6604–6615.

[255] Shahar Segal et al. *Fairness in the Eyes of the Data: Certifying Machine-Learning Models*. 2020. arXiv: 2009.01534.

[256] Gilad Asharov and Claudio Orlandi. “Calling out Cheaters: Covert Security with Public Verifiability”. In: *ASIACRYPT*. Vol. 7658. Lecture Notes in Computer Science. Springer, 2012, pp. 681–698.

[257] Carsten Baum. “On garbling schemes with and without privacy”. In: *International Conference on Security and Cryptography for Networks*. Springer. 2016, pp. 468–485.

[258] Jonathan Katz, Alex J Malozemoff, and Xiao Wang. “Efficiently Enforcing Input Validity in Secure Two-party Computation.” In: ().

[259] Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. “Sufficient Conditions for Collision-Resistant Hashing”. In: *Theory of Cryptography*. Ed. by Joe Kilian. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 445–456.

[260] Yonatan Aumann and Yehuda Lindell. “Security against Covert Adversaries: Efficient Protocols for Realistic Adversaries”. In: *Journal of Cryptology. The Journal of the International Association for Cryptologic Research* 23.2 (2010), pp. 281–343.

[261] Vipul Goyal, Payman Mohassel, and Adam D. Smith. “Efficient Two Party and Multi Party Computation against Covert Adversaries”. In: *EUROCRYPT*. Vol. 4965. Lecture Notes in Computer Science. Springer, 2008, pp. 289–306.

[262] Ivan Damgård, Martin Geisler, and Jesper Buus Nielsen. “From Passive to Covert Security at Low Cost”. In: *TCC*. Vol. 5978. Lecture Notes in Computer Science. Springer, 2010, pp. 128–145.

[263] Yehuda Lindell. "Fast Cut-and-Choose Based Protocols for Malicious and Covert Adversaries". In: *CRYPTO (2)*. Vol. 8043. Lecture Notes in Computer Science. Springer, 2013, pp. 1–17.

[264] Itai Dinur et al. "Optimized Interpolation Attacks on LowMC". In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2015, pp. 535–560.

[265] Martin R Albrecht et al. "Ciphers for MPC and FHE". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2015, pp. 430–454.

[266] O.S Rothaus. "On "Bent" Functions". In: *Journal of Combinatorial Theory, Series A* 20.3 (1976), pp. 300–305. URL: https://www.sciencedirect.com/science/article/pii/0097316576900248.

[267] Krzysztof Pietrzak. "Cryptography from Learning Parity with Noise". In: Jan. 2012, pp. 99–114.

[268] Cheng Hong Jonathan Katz Vladimir Kolesnikov Wen-jie Lu Xiao Wang. "Covert Security with Public Verifiability: Faster, Leaner, and Simpler". In: *EuroCrypt*. 2019.

[269] Cheng Hong et al. "Covert Security with Public Verifiability: Faster, Leaner, and Simpler". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2019, pp. 97–121.

[270] Guido Bertoni et al. "On the Indifferentiability of the Sponge Construction". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2008, pp. 181–197.

[271] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. "Mitigating Unwanted Biases with Adversarial Learning". In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. ACM. 2018, pp. 335–340.

[272] Thomas Davenport and Ravi Kalakota. "The Potential for Artificial Intelligence in Healthcare". In: *Future healthcare journal* 6.2 (2019), p. 94.

[273] Mariusz Bojarski et al. *Explaining How a Deep Neural Network Trained with End-to-End Learning Steers a Car*. 2017. arXiv: 1704.07911.

[274] RE Putra, AI Nurhidayat, and AY Wicaksono. "Implementation of Neural Network to Determine the New College Students". In: *IOP Conference Series: Materials Science and Engineering*. Vol. 288. 1. IOP Publishing. 2018, p. 012121.

[275] Zachary C Lipton. "The Mythos of Model Interpretability: In Machine Learning, the Concept of Interpretability Is Both Important and Slippery." In: *Queue* 16.3 (2018), pp. 31–57.

[276] Dario Amodei et al. *Concrete Problems in AI Safety*. 2016. arXiv: 1606.06565.

[277] Ziad Obermeyer et al. "Dissecting Racial Bias in an Algorithm Used to Manage the Health of Populations". In: *Science* 366.6464 (2019), pp. 447–453.

[278] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "" Why Should i Trust You?" Explaining the Predictions of Any Classifier". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016, pp. 1135–1144.

[279] Sina Mohseni et al. *Practical Solutions for Machine Learning Safety in Autonomous Vehicles*. 2019. arXiv: 1912.09630.

[280] David Madras et al. *Learning Adversarially Fair and Transferable Representations*. 2018. arXiv: 1802.06309.

[281] L Elisa Celis et al. "Classification with Fairness Constraints: A Meta-Algorithm with Provable Guarantees". In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*. 2019, pp. 319–328.

[282] Zecheng He, Tianwei Zhang, and Ruby B Lee. *VerIDeep: Verifying Integrity of Deep Neural Networks through Sensitive-Sample Fingerprinting*. 2018. arXiv: 1808.03277.

[283] Zahra Ghodsi, Tianyu Gu, and Siddharth Garg. "Safetynets: Verifiable Execution of Deep Neural Networks on an Untrusted Cloud". In: *Advances in Neural Information Processing Systems*. 2017, pp. 4672–4681.

[284] Martin Albrecht et al. "MiMC: Efficient Encryption and Cryptographic Hashing with Minimal Multiplicative Complexity". In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2016, pp. 191–219.

[285] Abigail Sellen et al. "Reflecting Human Values in the Digital Age". In: *Communications of the ACM* 52.3 (2009), pp. 58–66.

[286] Rebecca Balebako et al. "The Privacy and Security Behaviors of Smartphone App Developers". In: (2014).

[287] Alma Whitten and J Doug Tygar. "Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0." In: *USENIX Security Symposium*. Vol. 348. 1999, pp. 169–184.

[288] CPOmagazine.Com. *What's Old Is New Again: Examining Privacy Enhancing Technologies*. Mar. 2020. URL: https://www2.deloitte.com/nl/nl/pages/risk/articles/is-2020-finally-the-year-of-the-pets.html.

[289] Deloitte.com. *Is 2020 Finally the Year of the PETs (Privacy Enhancing Technologies)?* Mar. 2020. URL: https://www2.deloitte.com/nl/nl/pages/risk/articles/is-2020-finally-the-year-of-the-pets.html.

[290] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. "Rappor: Randomized Aggregatable Privacy-Preserving Ordinal Response". In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. 2014, pp. 1054–1067.

[291] Daniel Fallman. "The New Good: Exploring the Potential of Philosophy of Technology to Contribute to Human-Computer Interaction". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2011, pp. 1051–1060.

[292] Yasemin Acar, Sascha Fahl, and Michelle L Mazurek. "You Are Not Your Developer, Either: A Research Agenda for Usable Security and Privacy Research beyond End Users". In: *2016 IEEE Cybersecurity Development (SecDev)*. IEEE. 2016, pp. 3–8.

[293] Nelly EJ Oudshoorn and Trevor Pinch. *How Users Matter: The Co-Construction of Users and Technologies*. MIT press, 2003.

[294] UN Privacy Preserving Techniques Task Team. *UN Handbook on Privacy-Preserving Computation Techniques*. http://publications.officialstatistics.org/handbooks/privacy-preserving-techniques-handbook/UN%20Handbook%20for%20Privacy-Preserving%20Techniques.pdf, 2020.

[295] The Royal Society. *Protecting Privacy in Practice: The Current Use, Development and Limits of Privacy Enhancing Technologies in Data Analysis*. The Royal Society, 2019.

[296] *Microsoft SEAL (Release 3.5)*. Apr. 2020. URL: `https://github.com/Microsoft/SEAL`.

[297] Rosario Cammarota et al. *Trustworthy AI Inference Systems: An Industry Research View*. 2020. arXiv: `2008.04449`.

[298] Blase Ur et al. "Smart, Useful, Scary, Creepy: Perceptions of Online Behavioral Advertising". In: *Proceedings of the Eighth Symposium on Usable Privacy and Security*. ACM. 2012, p. 4.

[299] Jennifer King. ""How Come i'm Allowing Strangers to Go through My Phone?"—Smartphones and Privacy Expectations". In: *Symposium on Usable Privacy and Security (SOUPS)*. 2013.

[300] Irina Shklovski et al. "Leakiness and Creepiness in App Space: Perceptions of Privacy and Mobile App Use". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 2014, pp. 2347–2356.

[301] Max Van Kleek et al. "Better the Devil You Know: Exposing the Data Sharing Practices of Smartphone Apps". In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM. 2017, pp. 5208–5220.

[302] Ponnurangam Kumaraguru and Lorrie Faith Cranor. "Privacy Indexes: A Survey of Westin's Studies". In: (2005).

[303] Jialiu Lin et al. "Expectation and Purpose: Understanding Users' Mental Models of Mobile App Privacy through Crowdsourcing". In: *Proceedings of Conference on Ubiquitous Computing*. ACM. 2012, pp. 501–510.

[304] Rebecca Balebako et al. "Little Brothers Watching You: Raising Awareness of Data Leaks on Smartphones". In: *Proceedings of the Symposium on Usable Privacy and Security*. ACM. 2013, p. 12.

[305] Ruogu Kang et al. ""My Data Just Goes Everywhere:" User Mental Models of the Internet and Implications for Privacy and Security". In: *Proceedings of Symposium on Usable Privacy and Security*. 2015, pp. 39–52.

[306] Pedro Giovanni Leon et al. “What Matters to Users?: Factors That Affect Users’ Willingness to Share Information with Online Advertisers”. In: *Proceedings of Symposium on Usable Privacy and Security*. ACM. 2013, pp. 1–7.

[307] Alessandro Acquisti, Laura Brandimarte, and George Loewenstein. “Privacy and Human Behavior in the Age of Information”. In: *Science* 347.6221 (2015), pp. 509–514.

[308] Lemi Baruh, Ekin Secinti, and Zeynep Cemalcilar. “Online Privacy Concerns and Privacy Management: A Meta-Analytical Review”. In: *Journal of Communication* 67.1 (2017), pp. 26–53.

[309] Pedro Leon et al. “Why Johnny Can’t Opt out: A Usability Evaluation of Tools to Limit Online Behavioral Advertising”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 2012, pp. 589–598.

[310] Jialiu Lin et al. “Modeling Users’ Mobile App Privacy Preferences: Restoring Usability in a Sea of Permission Settings”. In: *Symposium on Usable Privacy and Security*. 2014, pp. 199–212.

[311] Florian Schaub et al. “A Design Space for Effective Privacy Notices”. In: *Proceedings of the Symposium on Usable Privacy and Security*. 2015, pp. 1–17.

[312] Rebecca Balebako et al. “The Impact of Timing on the Salience of Smartphone App Privacy Notices”. In: *Proceedings of the ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*. ACM. 2015, pp. 63–74.

[313] Bin Liu et al. “Follow My Recommendations: A Personalized Assistant for Mobile App Permissions”. In: *Proceedings of the Symposium on Usable Privacy and Security*. 2016.

[314] Amir Herzberg. “Why Johnny Can’t Surf (Safely)? Attacks and Defenses for Web Users”. In: *computers & security* 28.1-2 (2009), pp. 63–71.

[315] Lorrie Faith Cranor, Manjula Arjula, and Praveen Guduru. “Use of a P3P User Agent by Early Adopters”. In: *Proceedings of the 2002 ACM Workshop on Privacy in the Electronic Society*. 2002, pp. 1–10.

[316] Jeremy Clark, Paul C Van Oorschot, and Carlisle Adams. “Usability of Anonymous Web Browsing: An Examination of Tor Interfaces and Deployability”. In: *Proceedings of the 3rd Symposium on Usable Privacy and Security*. 2007, pp. 41–51.

[317] Brooke Bullek et al. “Towards Understanding Differential Privacy: When Do People Trust Randomized Response Technique?” In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2017, pp. 3833–3837.

[318] Stanley L Warner. “Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias”. In: *Journal of the American Statistical Association* 60.309 (1965), pp. 63–69.

[319] Aiping Xiong et al. *Towards Effective Differential Privacy Communication for Users’ Data Sharing Decision and Comprehension*. 2020. arXiv: 2003.13922.

[320] Lucy Qin et al. “From Usability to Secure Computing and Back Again”. In: *Fifteenth Symposium on Usable Privacy and Security ({SOUPS} 2019)*. 2019.

[321] Peter Leo Gorski et al. “Listen to Developers! A Participatory Design Study on Security Warnings for Cryptographic APIs”. In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 2020, pp. 1–13.

[322] Hala Assal and Sonia Chiasson. “’Think Secure from the Beginning’ a Survey with Software Developers”. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 2019, pp. 1–13.

[323] Steven M Furnell et al. “An Integrated View of Human, Organizational, and Technological Challenges of IT Security Management”. In: *Information Management & Computer Security* (2009).

[324] Seda Gurses and Joris Van Hoboken. “Privacy after the Agile Turn”. In: ().

[325] Blagovesta Kostova, Seda Gürses, and Carmela Troncoso. *Privacy Engineering Meets Software Engineering. On the Challenges of Engineering Privacy ByDesign*. 2020. arXiv: 2007.08613.

[326] Irit Hadar et al. “Privacy by Designers: Software Developers’ Privacy Mindset”. In: *Empirical Software Engineering* 23.1 (2018), pp. 259–289.

[327] Phillip Rogaway. “The Moral Character of Cryptographic Work.” In: *IACR Cryptol. ePrint Arch.* 2015 (2015), p. 1162.

[328] Sheila Jasanoff and Sang-Hyun Kim. “Containing the Atom: Sociotechnical Imaginaries and Nuclear Power in the United States and South Korea”. In: *Minerva* 47.2 (2009), p. 119.

[329] S Harrison, D Tatar, and P Sengers. “The Three Paradigms of HCI. Alt”. In: *CHI’07*. 2007.

[330] Herman T Tavani and James H Moor. “Privacy Protection, Control of Information, and Privacy-Enhancing Technologies”. In: *ACM Sigcas Computers and Society* 31.1 (2001), pp. 6–11.

[331] Seda Gürses and Bettina Berendt. “PETs in the Surveillance Society: A Critical Review of the Potentials and Limitations of the Privacy as Confidentiality Paradigm”. In: *Data Protection in a Profiled World*. Springer, 2010, pp. 301–321.

[332] Felix Stalder. “The Failure of Privacy Enhancing Technologies (PETs) and the Voiding of Privacy”. In: *Sociological Research Online* 7.2 (2002), pp. 25–39.

[333] David J Phillips. “Privacy Policy and PETs: The Influence of Policy Regimes on the Development and Social Implications of Privacy Enhancing Technologies”. In: *New Media & Society* 6.6 (2004), pp. 691–706.

[334] Alexander Bogner, Beate Littig, and Wolfgang Menz. *Interviewing Experts*. Springer, 2009.

[335] Robert Hoppe. “Scientific Advice and Public Policy: Expert Advisers’ and Policymakers’ Discourses on Boundary Work”. In: *Poiesis & Praxis* 6.3-4 (2009), pp. 235–263.

[336] Morgan Meyer. "The Rise of the Knowledge Broker". In: *Science communication* 32.1 (2010), pp. 118–127.

[337] Antti Oulasvirta and Kasper Hornbæk. "Hci Research as Problem-Solving". In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 2016, pp. 4956–4967.

[338] Donald E Stokes. *Pasteur's Quadrant: Basic Science and Technological Innovation*. Brookings Institution Press, 2011.

[339] Steven Shapin. "Placing the View from Nowhere: Historical and Sociological Problems in the Location of Science". In: *Transactions of the Institute of British Geographers* 23.1 (1998), pp. 5–12.

[340] Kenneth Goldstein. "Getting in the Door: Sampling and Completing Elite Interviews". In: *PS: Political Science and Politics* 35.4 (2002), pp. 669–672.

[341] Edward C Page, Bill Jenkins, William Ieuan Jenkins, et al. *Policy Bureaucracy: Government with a Cast of Thousands*. Oxford University Press on Demand, 2005.

[342] Virginia Braun and Victoria Clarke. "Using Thematic Analysis in Psychology". In: *Qualitative research in psychology* 3.2 (2006), pp. 77–101.

[343] Lora Bex Lempert. "Asking Questions of the Data: Memo Writing in the Grounded". In: *The Sage handbook of grounded theory* (2007), pp. 245–264.

[344] Emiliano De Cristofaro and Gene Tsudik. "Practical Private Set Intersection Protocols with Linear Complexity". In: *International Conference on Financial Cryptography and Data Security*. Springer. 2010, pp. 143–159.

[345] Ian Brown and Douwe Korff. "Terrorism and the Proportionality of Internet Surveillance". In: *European Journal of Criminology* 6.2 (2009), pp. 119–134.

[346] Antonio Marcedone, Zikai Wen, and Elaine Shi. "Secure Dating with Four or Fewer Cards." In: *IACR Cryptol. ePrint Arch.* 2015 (2015), p. 1031.

[347] Mark Bovens, Thomas Schillemans, and Robert E Goodin. "Public Accountability". In: *The Oxford handbook of public accountability* 1.1 (2014), pp. 1–22.

[348] Jessica Colnago et al. "Informing the Design of a Personalized Privacy Assistant for the Internet of Things". In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 2020, pp. 1–13.

[349] Matthew Green and Matthew Smith. "Developers Are Users Too: Designing Crypto and Security APIs That Busy Engineers and Sysadmins Can Use Securely". In: (2015).

[350] Graham Dove et al. "UX Design Innovation: Challenges for Working with Machine Learning as a Design Material". In: *Proceedings of the 2017 Chi Conference on Human Factors in Computing Systems*. 2017, pp. 278–288.

[351] Jeffrey Bardzell and Shaowen Bardzell. "Humanistic Hci". In: *Synthesis Lectures on Human-Centered Informatics* 8.4 (2015), pp. 1–185.

[352] Donald MacKenzie and Judy Wajcman. *The Social Shaping of Technology*. Open university press, 1999.

[353] Edsger W Dijkstra. "Selected Writings on Computing-a Personal Perspective. Texts and Monographs in Computer Science". In: *Springer, doi* 10 (1982), pp. 978–1.

[354] Friedrich Steimann. "Fatal Abstraction". In: *Proceedings of the 2018 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*. 2018, pp. 125–130.

[355] Brad A Myers and Jeffrey Stylos. "Improving API Usability". In: *Communications of the ACM* 59.6 (2016), pp. 62–69.

[356] Minhaz Zibran. "What Makes APIs Difficult to Use". In: *International Journal of Computer Science and Network Security (IJCSNS)* 8.4 (2008), pp. 255–261.

[357] Adam Mechtley. "API as Curriculum: Designing High-Level API Affordances as Instructional Scaffolds". In: *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. 2020, pp. 1–9.

[358] Joseph Lawrance et al. "How Programmers Debug, Revisited: An Information Foraging Theory Perspective". In: *IEEE Transactions on Software Engineering* 39.2 (2010), pp. 197–215.

[359] Caitlin Kelleher and Michelle Ichinco. "Towards a Model of API Learning". In: *2019 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE. 2019, pp. 163–168.

[360] Kathleen Broome Williams. *Grace Hopper: Admiral of the Cyber Sea*. Naval Institute Press, 2012.

[361] Chris Elsden et al. "Making Sense of Blockchain Applications: A Typology for HCI". In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 2018, pp. 1–14.

[362] Maarten Roy Prak. *Craft Guilds in the Early Modern Low Countries: Work, Power and Representation*. Ashgate Publishing, Ltd., 2006.

[363] Pelle Ehn. "Work-Oriented Design of Computer Artifacts". Arbetslivscentrum, 1988.

[364] Robert Jungk and Norbert Müllert. *Future Workshops: How to Create Desirable Futures*. Inst. for Social Inventions, 1987.

[365] Gurpreet Dhillon and James Backhouse. "Current Directions in IS Security Research: Towards Socio-Organizational Perspectives". In: *Information systems journal* 11.2 (2001), pp. 127–153.

[366] Oscar H Gandy Jr. *The Panoptic Sort: A Political Economy of Personal Information. Critical Studies in Communication and in the Cultural Industries*. ERIC, 1993.

[367] Julie E Cohen. *Configuring the Networked Self: Law, Code, and the Play of Everyday Practice*. Yale University Press, 2012.

[368] Sarah Spiekermann and Lorrie Faith Cranor. "Engineering Privacy". In: *IEEE Transactions on software engineering* 35.1 (2008), pp. 67–82.

[369] Michael Veale, Reuben Binns, and Jef Ausloos. "When Data Protection by Design and Data Subject Rights Clash". In: *International Data Privacy Law* 8.2 (2018), pp. 105–123.

[370] Paul De Hert and Serge Gutwirth. "Privacy, Data Protection and Law Enforcement. Opacity of the Individual and Transparency of Power". In: *Privacy and the criminal law* (2006), pp. 61–104.

[371] Zbigniew Kwecka et al. ""I Am Spartacus": Privacy Enhancing Technologies, Collaborative Obfuscation and Privacy as a Public Good". In: *Artificial intelligence and law* 22.2 (2014), pp. 113–139.

[372] Joshua AT Fairfield and Christoph Engel. "Privacy as a Public Good". In: *Duke LJ* 65 (2015), p. 385.

[373] Trevor J Pinch and Wiebe E Bijker. "The Social Construction of Facts and Artefacts: Or How the Sociology of Science and the Sociology of Technology Might Benefit Each Other". In: *Social studies of science* 14.3 (1984), pp. 399–441.

[374] Valerie Chen, Valerio Pastro, and Mariana Raykova. "Secure computation for machine learning with SPDZ". In: *arXiv preprint arXiv:1901.00329* (2019).

[375] Xinlong Wang, Mingyu You, and Chunhua Shen. *Adversarial Generation of Training Examples for Vehicle License Plate Recognition*. July 2017. arXiv: 1707.03124. URL: http://arxiv.org/abs/1707.03124.

[376] Marcel Keller, Emmanuela Orsini, and Peter Scholl. "MASCOT: Faster Malicious Arithmetic Secure Computation with Oblivious Transfer". In: *ACM Conference on Computer and Communications Security*. ACM, 2016, pp. 830–842.

[377] Dragos Rotaru and Tim Wood. "Marbled circuits: Mixing arithmetic and boolean circuits with active security". In: *International Conference on Cryptology in India*. Springer. 2019, pp. 227–249.

[378] Wenting Zheng et al. "Cerebro: A platform for multi-party cryptographic collaborative learning". In: *30th {USENIX} Security Symposium ({USENIX} Security 21)*. 2021.

[379] Jon Crowcroft, Gareth Tyson, and Richard Mortier. *Redhouse gases–a manifesto for redecentralization*. 2020.

[380] Andrei Vlad Sambra et al. "Solid: a platform for decentralized social applications based on linked data". In: *MIT CSAIL & Qatar Computing Research Institute, Tech. Rep.* (2016).

[381] Ran Canetti and Marc Fischlin. "Universally Composable Commitments". In: *Advances in Cryptology — CRYPTO 2001*. Ed. by Joe Kilian. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 19–40.