Advancements in Privacy Enhancing Technologies for Machine Learning

Adam James Hall

A thesis presented for the degree of

Doctor of Philosophy

Department of Computer Science

Edinburgh Napier University

Scotland

XX/XX/2020

Abstract

The field of privacy preserving machine learning is still in its infancy and has been growing in popularity since 2019. Privacy enhancing technologies within the context of machine learning are composed of a set of core techniques. These relate to cryptography, distributed computation- or federated learning- differential privacy, and methods for managing distributed identity. Furthermore, the notion of contextual integrity exists to quantify the appropriate flow of information.

The aim of this work is to advance a vision of a privacy compatible infrastructure, where web 3.0 exists as a decentralised infrastructure, enshrines the user's right to privacy and consent over information concerning them on the Internet.

This thesis contains a mix of experiments relating to privacy enhancing technologies in the context of machine learning. A number of privacy enhancing methods are advanced in these experiments, and a novel privacy preserving flow is created. This includes the establishment of an open-source framework for vertically distributed federated learning and the advancement of a novel privacy preserving machine learning framework which accommodates a core set of privacy enhancing technologies. Along with this, the work advances a novel means of describing privacy preserving information flows which extends the definition of contextual integrity.

This thesis establishes a range of contributions to the advancement of privacy enhancing technologies for privacy preserving machine learning. A case study is evaluated, and a novel, heterogeneous stack classifier is built which predicts the presence of insider threat and demonstrates the efficacy of machine learning in solving problems in this domain, given access to real data. It also draws conclusions about the applicability of federated learning in this use case. A novel framework is introduced that facilitates vertically distributed machine learning on data relating to the same subjects held on different hosts. Researchers can use this to achieve vertically federated learning in practice. The weaknesses in the security of the Split Neural

Networks technique are discussed, and appropriate defences were explored in detail. These defences harden SplitNN against inversion attacks. A novel distributed trust framework is established which facilitated peer-to-peer access control without the need for a third party. This puts forward a solution for fully privacy preserving access control while interacting with privacy preserving machine learning infrastructure. Finally, a novel framework for the implementation of structured transparency is given. This provides a cohesive way to manage information flows in the privacy preserving machine learning and analytics space, offering a well-stocked toolkit for the implementation of structured transparency which utilises the aforementioned technologies. This also exhibits homomorphically encrypted inference which fully hardens the SplitNN methodology against model inversion attacks.

The most significant finding in this work is the production of an information flow which combines; split neural networks, homomorphic encryption, zero-knowledge access control and elements of differential privacy. This flow facilitates homomorphic inference through split neural networks, advancing the state-of-the-art with regard to privacy preserving machine learning.

Dedication

I would like to thank Prof. Bill Buchanan and Dr. Nick Pitropakis for their support and infinite patience in the delivery of this project. I would also like to thank my parents Steve and Lesley Hall for their support and for laying the foundations which helped me to this stage of my life. I would like to extend my deepest gratitude to the OpenMined community for their collaboration and opportunities for personal development that were afforded to me as a contributor, dev lead and, finally, director. I would also like to thank Will Abramson for their companionship while exploring through this vast new space of technology.

Declaration

I, Adam James Hall, confirm that this thesis and the work presented in it are my own achievement.

1. Where I have consulted the published work of others this is always clearly attributed.
2. Where I have quoted from the work of others the source is always given. With the exception of such quotations, this thesis is entirely my own work.
3. I have acknowledged all main sources of help.
4. If my research follows on from previous work or is part of a larger collaborative research project I have made clear exactly what was done by others and what I have contributed myself.
5. I have read and understand the penalties associated with Academic Misconduct.
6. I also confirm that I have obtained informed consent from all people I have involved in the work of this thesis following the School's ethical guidelines.

NAME: ADAM JAMES HALL
DATE: WEDNESDAY, 24 MAY 2023
MATRICULATION NO: [redacted]

Acknowledgements

Contents

List of Figures

List of Tables

Glossary

API Application Programming Interface: A set of rules, protocols, and tools for building software applications. APIs specify how software components should interact and are used to enable the integration between different systems and devices. 62, 128, 131, 133, 140

ASCII American Standard Code for Information Interchange: A character encoding standard for electronic communication, representing text in computers and other devices. 30

Data Subject An individual who is the subject of personal data. Unlike the owner of the data, who may possess, control, or have legal rights over the information, a data subject is specifically the person whom the data is about. Data subjects are central to data protection and privacy legislation, such as the GDPR, which grants them specific rights regarding their personal data. These rights include access to their data, the right to rectify inaccuracies, and the right to be forgotten, among others. The distinction between a data subject and a data owner emphasizes the data subject's rights to control how their personal information is used, even if they do not "own" the data in a legal or financial sense. 24, 28, 33–35, 62, 63, 78, 95, 96, 101, 129, 135

DID Decentralized Identifiers: A new type of identifier that enables verifiable, self-sovereign digital identities. DID are fully under the control of the DID subject, independent from any centralized registry, identity provider, or certificate authority. 8, 70–73, 82, 117–120, 123–125, 135

DO Data Owner: An individual or entity responsible for the data within an organization. They have control over the collection, use, and distribution of the data and are accountable for its security and compliance with relevant laws and policies. In GDPR terms, this may be thought of as the data controller. 12, 13, 15, 24, 26, 35, 95–101, 116, 117, 119, 122, 128, 130, 131, 133, 135–140

DP Differential Privacy: A technique for sharing information about a dataset while protecting individual data. 7, 35, 45–53, 55, 62, 75, 95, 96, 102, 114, 115, 135, 139

DS Data Scientist: A professional who uses scientific methods, processes, algorithms, and systems to extract knowledge and insights from structured and unstructured data. Data Scientists apply machine learning techniques and statistical analysis to solve complex problems and make data-driven decisions. In GDPR terms, this may be thought of as the data processor. 12, 13, 24, 26, 27, 63, 94–101, 116, 119–124, 128, 130, 131, 133–140

DT Decision Tree: A decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements. 87, 90

ECDH Elliptic Curve Diffie-Hellman: A key agreement protocol that allows two parties, each having an elliptic curve public-private key pair, to establish a shared secret over an insecure channel. This shared secret can then be used to encrypt subsequent communications using a symmetric key cipher. ECDH is known for its strength and efficiency, particularly in environments where processing power, storage, and bandwidth are at a premium. 135

EU European Union: A political and economic union of 27 member states that are located primarily in Europe. 26, 27, 33, 34

FHE Fully Homomorphic Encryption: An encryption scheme that allows computation on ciphertexts, generating an encrypted result that, when decrypted, matches the result of operations performed on the plaintext. It enables complex data analysis and operations to be performed securely without access to the raw data. 56, 57, 60

FL Federated Learning: Trains algorithms across multiple decentralized devices without exchanging data samples. 38, 63, 64, 66, 74, 75, 95, 103, 104, 118, 120–122, 125

GAN Generative Adversarial Networks: AI algorithms for unsupervised learning via contesting neural networks. 55

GDPR General Data Protection Regulation: EU law on data protection and privacy. 33–35, 63

HE Homomorphic Encryption: Allows computations on ciphertexts, producing encrypted results that match operations performed on plaintext. 11, 15, 36, 56, 57, 60, 61, 82, 95, 127, 128, 134, 137, 138, 142–144

HFL Horizontally Federated Learning: Uses the same feature set across entities for collaborative model training. 62

HTTP Hypertext Transfer Protocol: An application-layer protocol used for transmitting hypermedia documents, such as HTML. It is the foundation of data communication for the World Wide Web, where hypertext documents include hyperlinks to other resources that the user can easily access, for example, by a mouse click or by tapping the screen. HTTP follows a client-server model where the web browser is the client and communicates with the web server that hosts the website. 33, 83

ID Identifier: A token or string of characters used to uniquely identify an entity within a specific context. 25, 86, 94, 96–98, 101

IID Independent and Identically Distributed: Variables with the same probability distribution and mutual independence. 64

IPFS InterPlanetary File System: A protocol and peer-to-peer network for storing and sharing data in a distributed file system. IPFS uses content-addressing to uniquely identify each file in a global namespace connecting all computing devices. It aims to make the web faster, safer, and more open. 71

LR Logistic Regression: A statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model. 87, 90

MIT Malicious Insider Threat: Refers to the risk posed by individuals within an organization who may harm the organization through malicious actions such as data theft, sabotage, or espionage. These insiders have authorized access, making their threats particularly challenging to detect and mitigate. 8, 12, 76–82, 84, 86, 89–92

ML Machine Learning: A branch of artificial intelligence that focuses on the development of algorithms and statistical models that enable computers to perform tasks without explicit instructions by relying on patterns and inference. 13, 29, 35, 37, 39, 40, 44, 69, 70, 74, 75, 77, 95, 102, 104, 117–119, 121, 123–126, 134

MNIST Modified National Institute of Standards and Technology database: A large database of handwritten digits for image processing systems training. 12, 13, 37, 94, 98, 100, 101, 103, 106–108, 111, 113, 115, 136

NBN Neural Bayesian Network: A probabilistic graphical model that combines principles of neural networks with Bayesian networks to model complex data distributions and relationships. It is used for predictive modeling and decision-making under uncertainty. 87, 89, 90

NHS National Health Service: The publicly funded healthcare system of the United Kingdom, responsible for providing the majority of healthcare services to residents. The NHS offers a wide range of services, including primary care, in-patient care, long-term healthcare, ophthalmology, and dentistry. 69, 70, 119, 120

NN Neural Network: Inspired by biological neural networks for function estimation. 40, 41, 44, 45, 66, 87, 90

NoPeekNN NoPeekNN is a novel approach in the field of privacy-preserving machine learning. It focuses on minimizing the amount of information that can be inferred about the input data from the intermediate representations within a neural network. This method aims to enhance the privacy of data while maintaining the utility of the model, making it particularly useful for scenarios where sensitive data is involved. 13, 38, 68, 103, 106–108, 110–115

NSA National Security Agency: A national-level intelligence agency of the United States Department of Defense, responsible for global monitoring, collection, and processing of information and data for foreign and domestic intelligence and counterintelligence purposes. 27

PETs Privacy Enhancing Technologies: A set of technologies that protect user privacy by minimizing personal data usage, maximizing data security, and empowering individuals with control over their personal information. 35, 62, 129, 130

PHE Partially Homomorphic Encryption: An encryption technique that permits certain types of computations to be carried out on ciphertexts and obtain an encrypted result which, when decrypted, matches the outcome of operations performed on the plaintext. Unlike FHE, PHE supports only a limited set of operations. 57

PPML Privacy Preserving Machine Learning: Focused on developing algorithms that protect users' privacy and data confidentiality. 26, 28, 35–37,

60, 66, 75, 93, 116, 117, 126, 141–145

PSI Private Set Intersection: Allows two parties to find dataset intersections without revealing unnecessary information. 37, 60–62, 95–98, 101

RF Random Forest: An ensemble learning method for classification, regression, and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. 87, 90

ROC Receiver Operating Characteristic curve: A graphical plot used to show the diagnostic ability of a binary classifier system as its discrimination threshold is varied. It is commonly used in machine learning to evaluate the tradeoff between true positive rates and false positive rates of models. 12, 76, 77, 87, 89–92, 141

RSA Rivest–Shamir–Adleman: Widely used public-key cryptosystem for secure data transmission. 57, 58, 72

SGD Stochastic Gradient Descent: Gradient Descent version using a subset of training data for updates. 44, 45, 52, 114

SMC Secure Multi-party Computation: Enables parties to compute a function over their inputs while keeping those inputs private. 11, 56, 57, 62, 95, 126

SOC Security Operations Center: A centralized unit that deals with security issues on an organizational and technical level. An SOC within a company or institution monitors and analyzes security posture on an ongoing basis, and is responsible for detecting, analyzing, and responding to cybersecurity incidents using a combination of technology solutions and a strong set of processes. 80, 92

SplitNN Split Neural Network: A neural network model divided across multiple devices for distributed learning. 11, 12, 36–38, 66–69, 94–97, 100, 101, 114, 127, 137, 141–143

SSI Self-Sovereign Identity: An approach to digital identity that gives individuals control over the storage and management of their personal data. 33, 37

ST Structured Transparency: A concept that refers to the organized and systematic presentation of information and operations within a system, ensuring clarity, accountability, and understanding for all stakeholders involved. 31–33, 36, 37, 129, 130, 136, 140

STUN Session Traversal Utilities for NAT: A protocol that assists devices behind a Network Address Translator (NAT) in discovering their public IP addresses and the types of NATs they are behind. This is crucial for facilitating peer-to-peer (P2P) communication and is commonly used in VoIP technologies. 132, 133

SVM Support Vector Machine: A supervised machine learning model that is used for classification and regression tasks. It works by finding the hyperplane that best divides a dataset into classes in the feature space. SVMs are known for their effectiveness in high-dimensional spaces and for cases where the number of dimensions exceeds the number of samples. 80, 87, 90

ToIP Trust over IP: A framework designed to establish trust across the internet using a layered approach to digital identity verification, including cryptographic assurance at its core. 33, 36, 37, 116, 143, 144

UDP User Datagram Protocol: A communication protocol used across the Internet that offers a minimal, connectionless, best-effort service for transmitting datagrams. It is suitable for purposes where error check-

ing and correction are either not necessary or are performed in the application, speeding up the process of transmission. 133

UK United Kingdom: A sovereign country located off the northwestern coast of mainland Europe. It comprises England, Scotland, Wales, and Northern Ireland, and is known for its historical and cultural influence worldwide. 27, 34

VC Verifiable Credentials: Digital documents that are tamper-evident and can be cryptographically verified. VC can be used to represent all of the same information that a physical credential could, but in a way that is more secure, more privacy-respecting, and more interoperable. 72, 117–119, 122, 123, 125

VFL VerticallyFederated Learning: Collaborates on model training without direct data sharing. 12, 36, 37, 62, 65, 95, 96, 99, 101, 102

WebRTC Web Real-Time Communication: An open-source project that provides web browsers and mobile applications with real-time communication via simple application programming interfaces (APIs). It supports video, voice, and generic data to be sent between peers, allowing developers to build powerful voice- and video-communication solutions. 128, 133

Chapter 1

Introduction

Machine learning has been an active area of research for decades. However, it was not until the 21st century it began to be truly popularised [1]. This was marked first by the introduction of more advanced algorithms in the early 2000s [2] [3]. In the early 2010s the introduction of platforms like Microsoft Azure in 2014, Amazon Web Services in 2015 and Google Cloud Platform in 2017 facilitated an advancement in the availability of cloud computing resources and completely transformed the magnitude of computing power available to researchers performing machine learning tasks. Simultaneously, the amount of available data doubled in the digital economy year on year from 2010, at 2 zettabytes, to 64.2 zettabytes at the end of the decade [4]. Like combining elements of fuel, heat and oxygen; data, compute power and algorithms facilitated an explosion of innovation which turned machine learning from an ember observed at the turn of the millennium to the sprawling and uncontrollable wildfire seen today.

While the field of artificial intelligence continues to expand at an exponential pace, a final frontier is yet to be explored. This frontier is a gateway to the most high value data which has been hitherto kept out of reach by red tape and the personal rights of Data Subjects and DOs. This frontier is the information contained in private data. Until this point, DSs have mastered many benign problems belonging to publicly accessible data, problems which

would threaten no harm to individuals if data found its way into the wrong hands. These are datasets like labelled handwritten digits, or items of clothing [5][6]. However, data which may be used to find new cures for diseases, diagnose illness and even anticipate the most well-hidden organisational threats are only accessible by those who have privileged access.

In many cases this is appropriate. Highly valuable patient data, for example, should not be shared even though this may lead to breakthroughs in the way that diseases are diagnosed, understood and prevented. Once data has been shared, it is easily copied and sold. This is known as the copy problem [7]. The copy problem lies at the root of many of the issues that data sharing represents. When data has been shared beyond the scope of its current controller, there are no technical constraints which may be put on the recipient, which would inhibit their ability to perform whatever operations they wish, including selling it on to the highest bidder. For this reason, the majority of medical data remains tightly under the control of medical care providers upon creation.

Another problem which exists is the bundling problem [7]. The bundling problem arises when some bits of data can only be shared with other bits of information. These accompanying bits may contain information relating to the veracity of the data element which is to be shared. However, in sharing these bundled bits, more may be revealed than is absolutely necessary. A prime example of unnecessary information loss is when a young adult enters a bar and is required to show their ID. The verifier of their age in this scenario only requires the binary attribute; whether this young adult is above the legal drinking age. However, the young adult must present their personal ID which may contain their address, their actual date of birth, their nationality, their full name etc. If the verifier is prejudiced, their nationality may be used against them in future. The verifier may also remember their name as well as the other attributes presented in their ID in order to identify them in some other record accessible to the verifier where the subject has been anonymised. In this case, the required inclusion of extra information unnecessary to the

question at hand would lead to unintended outcomes if used outside of the current context.

The copy problem and bundling problem are two examples which highlight the necessity of keeping data private. However, the benefits of sharing data are obvious. Sharing data exposes a greater surface area to researchers seeking to make technological breakthroughs which are catalysed by said data. This point is demonstrated by the author of this thesis in work produced in 2016 [8]. In this work, the author collaborated with the Paediatrics Department of the Southern General Hospital in Glasgow in order to build a dataset relating to the onset of type two diabetes in children. While this classifier was 78% accurate and could have improved patient outcomes through early diagnoses and warning of insulin resistance in children, expanding the dataset into different source hospitals was stymied by bureaucracy which protects patient privacy. This is just one example where patient privacy concerns took priority over the utilitarian value of improved patient outcomes.

However, at the turn of the recent decade, a new set of techniques began to gain traction. These techniques relate to the processing of data which facilitates the extraction of information while the data remains inaccessible to the processor and in the sole custody of the original DO. It is with these technologies that the thesis is primarily concerned with.

1.0.1 Current Issues in the Data Industry

The reasons why this work is important are three-fold. To begin with, there are ethics to consider around the mass hoarding and abuse of personal data by DSs. New EU data protection legislation also provides ample motivation with up to 4% of annual turnover resulting from non-compliance while processing the data of an EU citizen. Finally, there are the potential subject areas that PPML will open up to machine learning previously. Without any data being shared with researchers, there is now the capacity to train models on private data.

With the growth of the popularity of data science methods, a new and

highly valuable asset has entered the market: data. The tools used to process and model the inherent structure of data are now widely available, and the practices of unabated data collection, hoarding and wholesale have become commonplace [9].

Recent history provides a plethora of examples of where data has been used by DSs without the consent or knowledge of owners. In one case, Ex-NSA operatives abused access to data, using it to track ex-spouses [10]. This abuse also exists on a corporate and state level. It is common knowledge that Cambridge Analytica used people's private data without knowledge or consent to influence the US presidential elections [11]. The firm paid around 320,000 voters to take a short questionnaire about their personality and political inclinations. However, this questionnaire gave Cambridge Analytica the capacity to scrape the Facebook data of these individuals and all of their friends, regardless of whether these friends had consented to this. Once all of this data is harvested, this is correlated with other data sources, creating a substantial amount of information about these individuals. This data was leveraged to target specific individuals in swing states with tailored propaganda [11] in the Trump election campaign. Not only this, the firm propagated provocative false information through smaller subsidiary firms with no links to Cambridge Analytica (Figure 1.1).

It is also well-known that they played a similarly contentious role within the UK EU exit referendum [13]. The firm has denied involvement, however, Britanny Kaiser, a senior member of the organisation, has since disputed this narrative with email evidence supporting her case [14]. This same representative reports that they were involved in around 10 national elections per year including; Malaysia, Kenya, Lithuania, Iraq and Afghanistan. Kaiser describes using *weapons-grade* communications warfare techniques to influence public opinion which is illegal to practice in the UK. These revelations around the capacity for big data techniques to manufacture consent in liberal democracies are startling. While Cambridge Analytica has filed for bankruptcy, this has far from solved the issue of data-fueled mass manipulation. In fact, it

Figure 1.1: How Cambridge Analytica stole the data of 50 million people [12]

continues to spiral out of control into ever more sinister directions.

For example, in 2018, Facebook was linked to violence against 650,000 refugees fleeing Bangladesh in Myanmar. Hate speech and misinformation inciting violence were propagated en masse through the social media platform, a primary source of news for large swathes of the population (Figure 1.2) [15].

Should data science continue to be weaponized in this way, it could be used to incite society to commit atrocities against itself. It is also clear that big data practices in this form, where no rights or considerations are afforded to Data Subjects, represent an imminent threat to the integrity of international democracies and political sovereignty, paving the way to authoritarianism.

PPML allows for data-owners to keep hold of their data assets, never sharing them to the outside world, and never to be copied, shared or processed without their consent. This new phenomenon of targeted propaganda arguably could not exist without this unfettered access to people's data. Through the adoption and popularisation of PPML architectures proposed in this document, mass-scale aggregation without prior consent would be

Figure 1.2: Inciting Violence in Myanmar
[12]

impossible.

1.0.2 Modalities of Machine Learning

In this section, the core modalities of machine learning in practice will be discussed. These modalities are comprised of Supervised Learning, Unsupervised Learning and Reinforcement Learning.

1.0.2.1 Supervised Learning

Supervised learning is defined as the subset of ML problems where there is some human knowledge in the loop. The idea is that the model trains on some set of labelled data and memorises the structure of these labelled datasets in order to predict unseen future cases. Supervised learning is split into two subdomains; Regression and Classification. Regression models aim to predict some continuous or numerical label. This makes sense when predicting potential profits over a new business quarter or the price of land.

Classification problems on the other data-owner aim to sort unlabeled input data into classes or categories. For example, it may be necessary to predict whether a patient at a hospital has diabetes or to sort handwritten characters into their ASCII symbols. In this case, the ASCII symbols would be the categories.

1.0.2.2 Unsupervised Learning

Unsupervised learning takes place where data is unlabeled, meaning there is no human knowledge in the loop. The main goal of unsupervised learning is to uncover hidden patterns and group similar data points into clusters. The two sub-types of Unsupervised learning are; clustering and dimensionality reduction. For example, Employees may be clustered based on their Big Five personality traits in order to quickly identify groups of like-minded individuals or identify a profile which is associated with insider threats. With dimensionality reduction, the number of features that are fed to the classification algorithm are reduced. It may be necessary to use a feature reduction technique like principle component analysis to achieve this.

1.0.2.3 Reinforcement Learning

Reinforcement Learning is designed around the theory of mind in relation to how animals and humans learn. In Reinforcement Learning, an agent is embedded within an environment represented as a series of Markov states; where all knowledge needed for a given situation is present in the current state. The algorithm learns to make a series of actions which stem from these states in order to maximise their objective function. The agent iteratively makes decisions in their environment which relate to some reward function, and they associate this reward with given Markov states and the actions which may stem from this state. The agent continues to iterate until it converges on an optimal solution for the environment.

1.0.3 Contextual Integrity and Structured Transparency

An important factor in governing the application of machine learning to private data is discerning the appropriate use cases of privacy preserving technology. For the purposes of this work, the author has chosen to subscribe to the contextual integrity framework expounded by Helen Nissenbaum [16]. This framework is extended in recent work to apply specifically to privacy-enhancing technologies in information flows [7], this extension is called ST. The attributes described in the ST framework are used to justify design choices in the final chapter of this thesis, where a novel framework is built to facilitate ST in privacy preserving information flows.

Within the contextual integrity framework, appropriate privacy is evaluated within the parameters of their social context. These contexts are shaped by social norms and individuals may determine the presence of a privacy violation by comparing the information flow being analysed to the social expectations surrounding their context. As such, conceptions of privacy are based on ethical concerns which may evolve as the ethics of society evolve. The central thesis of contextual integrity is that privacy is neither a right to secrecy nor control of information, but the right to *appropriate* flow of information [17]

This has been applied in a variety of different domains which include social media, surveillance and healthcare [17]. Within contextual integrity, a key concept is the notion of an information flow. Information flows are defined as the flow of information from sender to recipient within a specific context of sphere [16]. This notion of information flows is applied to the greater societal sphere, which can be thought of as being composed of these atomic information flows. The central idea here is that if the privacy for individual information flows is solved, privacy for the society is solved. The contextual integrity framework contains key conditions relating to the appropriate flow of information [18].

Relevant Entities - are the agents relevant to the information flow. These

are composed of the sender; the one from whom the information flows, the recipient; the one to whom the information flows and the subject; about whom the information is concerned[18].

Information Type - represents the category of information being shared. In the context of the information flow built in the aforementioned healthcare example [8], the type of information is patient health data gathered by the hospital. In the later chapters, malicious insider threats will be examined. In this case, the information type would be logistical logs from an enterprise.

The Transmission Principle - represents the specific constraints (terms or conditions) regulating the flow of information from entity to entity prescribed by informational norms [18]. An example of a transmission principle would be confidentiality which would prohibit the recipient from sharing received information with agents and entities external to the flow.

ST extends this framework by adding key attributes specific to privacy preserving information flows. The idea is that all privacy preserving flows are always intended to share information. However, the information to be shared is purposefully limited to some selective portion of the original information- where the input data to the flow is to be processed in some way which preserves its confidentiality but allows for some prescriptive disclosure of an aspect of that original information. ST has a specific set of key attributes and defines a specific information flow taxonomy.

Under ST, information flows are broken into the following key types. Messaging flows represent cases where information is shared peer to peer from sender to receiver in such a way that the confidentiality and verification of sender and recipient are possible. This is exemplified in the case of traditional transport layer security where the message is kept confidential through encryption, and the sender and recipient are verified through their ownership of respective private keys [7]. Service provider flows involve some external entity which processes information before returning a result to a data owner or controller [7]. Finally, an aggregation flow is similar to a service provider flow, however in this case the aggregator consumes the re-

sult of the information flow after processing inputs. ST applies the following key attributes to this taxonomy.

Input Privacy - is the ability to process information while it remains hidden to the processor. Examples of preserving input privacy would be processing data while those data items remain encrypted.

Output Privacy - refers to the inability to infer information about the inputs of a flow beyond the intended output of the flow. An example of a violation of output privacy would be to perform a membership inference attack on some model, using the model classification on some data sample to infer whether that data sample belongs to the data used to train the model.

Input Verification - is the ability to verify a set of input parameters to an information flow originating from trusted entities [7]. This may be performed through input traditional transport layer security, as is the case in receiving a web page through the HTTP protocol. It may also, in the case of a SSI/ToIP stack, be relevant to verifying specific attributes of parameters to an information flow through verifiable credentials.

Output Verification - refers to the ability to verify attributes relating to the correct processing of inputs in an information flow. This may be process auditing or the processing of data using a server hosting a trusted execution environment.

1.0.4 The General Data Protection Regulation

Like most industries in their infancy, data science has not historically been adequately regulated [19]. The lack of appropriate governance has contributed to an environment where the unsanitary handling of private data has led to numerous large-scale breaches [20].

Reacting to the numerous cases of abuse against privacy and consent, governments have begun to reshape regulations around the acquisition, storage and use of personal data. In 2018 the EU brought GDPR into force. This legislation was designed to give Data Subjects new rights regarding their personal data, with an emphasis on holding organisations to account

for their use of the private information of citizens in research. Organisations thus found guilty of breaching GDPR can be fined up to €20,000,000 or 4% of their annual revenue, whichever is greater. These reforms are a first, decisive step towards bringing data practices under some level of governance.

GDPR expands the rights of individuals to privacy and consent, but this new legislation further galvanised protections over user data which still stifle innovation, inhibiting the ability of researchers to learn from information contained in private data. GDPR set a strong precedent where Data Subject rights are prioritised over the potential utility that these models are capable of facilitating, for example, the aforementioned case in the health sector where predictive models have the capacity to detect and predict diseases. This remains an ongoing tension amongst legislators who must manage this trade-off [19].

While the application of principles in GDPR has the capacity to improve common practices around data use with respect to Data Subjects, applying GDPR remains a challenge for enterprises. In 2022, six years after GDPR was introduced, 95% of US companies are using error-prone, manual processes for GDPR compliance[21]. In addition to this, despite the formidable power to impose a fine of up to 4% of a company's revenues, GDPR is rarely enforced adequately by authorities. In 2020 Brave Browser produced a report which detailed how Europe's governments are failing GDPR [22]. In this study, it was shown that only six out of 28 EU member states employ more than 10 tech specialists. Half of the GDPR enforcement agencies had budgets that were deemed to be inadequate. While the UK had the largest enforcement agency, only 3% of the staff were focused on tech privacy problems. This resulted in Brave filing a formal complaint against 27 EU member states for failing to adequately implement GDPR. Two years later, an article was published in Wired which detailed how GDPR is failing to bring big tech to account [23]. While GDPR has thus far been difficult to enforce, privacy-enhancing technologies offer an exciting opportunity for technologists trying to ensure GDPR compliance in the next generation of big tech applications.

In Chapter 5, Article 44, GDPR stipulates a general principle for the transfer of data to third-party organisations [24]. This is solved in the case of federated analytics, where statistical estimators and learning algorithms are sent to data locations rather than data being brought to the researcher performing analysis [25][26]. This facilitates compliance with GDPR in this respect. No data changes location, it's analysed at the location of the data holding processor.

In the case of Chapter 3, article 17, GDPR asserts the Data Subject's right to erasure [24]. This is also known as the 'right to be forgotten'. DP techniques allow for information to be extracted from datasets while ensuring that information from individual members of the dataset cannot be identified in the aggregate result [27][28][29] [30]. In the case of machine learning, this guarantees the rights of individuals to be forgotten by providing a mathematical proof that they were never remembered in the first place.

For the reasons stated above, PETs offer some relief to organisations trying to adapt their practices to GDPR, however, they are far from a panacea. In the case of processing, chapter 2, article 5 states that data may only be "collected for specified, explicit and legitimate purposes and not further processed in a manner that is incompatible with those purposes" [24]. In the case of data science, it is common practice for datasets to be collected with the intention of use in future research, however, the specifics of this research are unknown to the DO at the time of collection. This challenge remains even in the context of PPML as researchers, by law, must gather consent from each subject in a data set before performing their specific queries or answering their specific research questions. This problem is not mitigated by the introduction of PETs for ML.

1.1 Aim and objectives

It is the aim of this work to break down barriers to innovation concerning private data by advancing the notion of PPML. The intention of this break-

through technology is to facilitate the extraction of information from these private data sources by allowing for this analysis to take place while the researcher is blind to the underlying datasets. In order to achieve the aim identified in this thesis the following research objectives are proposed:

- **Objective I** - The identification and exploration of a domain where the adoption of these techniques will allow for greater models to be built through access to real-world data where access would traditionally be blocked.
- **Objective II** - To advance the state-of-the-art with respect to one sub-field of the PPML space; VFL through the use of SplitNN. This objective can be split into three sub-objectives:
  - **Objective II.I** - The creation of a novel, open-source implementation of SplitNN.
  - **Objective II.II** - The identification of practical defences against model inversion attacks against SplitNN models.
  - **Objective II.III** - The introduction of s novel methodology: HE inference using SplitNN.
- **Objective III** - The introduction of the ToIP stack as a mechanism for providing peer-to-peer trust and access control when running distributed PPML infrastructures.
- **Objective IV** - The advancement of the notion of ST as a means of rationalising the protections placed on information flows during PPML.

1.2 Contributions and Novelty

The main contributions of this thesis are:

- Insider threat detection was identified and explored as a promising area for improvements to be made to through PPML. A heterogeneous stack

classifier for predicting the presence of insider threats was developed to demonstrate the promise of this domain, should private real world data be made accessible through privacy enhancing technologies. [31]

- The creation of a novel, open source framework for facilitating vertically federated learning with SplitNNs[32][33].
- The investigation and presentation of a set of mitigation techniques against model inversion in the case of SplitNNs [34].
- A novel implementation of a SplitNN and homomorphic encryption hybrid; capable of completely protecting activation signals passed through a SplitNN at inference time [35].
- The implementation of the SSI/ToIP stack for implementing peer-to-peer trust in a distributed PPML framework. [36]
- The introduction and presentation of Syft; a novel framework for the implementation of ST [35]

1.2.1 PyVertical

Chapter 4 extends the proposal of [37] regarding the use of SplitNNs and PSI in Vertical Federated Learning. The PySyft library for PPML is used[38] to train a VFL ML algorithm on data distributed across the premises of one or multiple data owners. This work is released as an open-source framework, PyVertical. At the time of writing , this is the first open-source framework to perform machine learning on vertically distributed datasets using SplitNNs[1]. This method is verified on a two-party, vertically-partitioned MNIST dataset. This work presents a dual-headed scenario, where data from two separate data owners (who hold different parts of the data samples) and a data scientist (who, in this case, holds data labels) are securely aligned and combined for model training. However, this work could be extended to multiple data owners using the same principle described here.

[1]Code is available at PyVertical: https://github.com/OpenMined/PyVertical

The work outlined is an adaptation of a paper accepted for publication in the Workshop on Distributed and Private Machine Learning at the Ninth International Conference on Learning Representations (ICLR) [33].

1.2.2 Practical Defences Against Model Inversion

Chapter 5 defines a threat model for SplitNNs in which training and inference data are stolen in an FL system. The practical limitations on attack efficacy is examined, such as the amount of data available to an attacker and their prior knowledge of the target model. In particular, NoPeekNN is extended [39], a method for limiting information leakage in SplitNNs. NoPeekNN's defensive utility Is assessed in the outlined attack setting. Additionally, a simple method for protecting user data is introduced, consisting of random noise added to the intermediate data representation of SplitNN. This approach is also compared to NoPeekNN[2]

Chapter 2

Literature Review

2.1 Machine Learning

ML is the name given to a discipline of artificial intelligence and carries many definitions. While the objective meaning of ML is notoriously difficult to define [40], it is generally understood as the design and development of algorithms which learn from data in order to make predictions or decisions based on that data [41]. While the derivatives of the modern algorithms commonly used in ML have existed for decades [42], the increasing availability of computational resources has allowed ML to become the new state-of-the-art with regard to performing predictions on high dimensional data [43]. This burgeoning field is capable of solving complex problems across a pervasive range of application domains, including finance [44], healthcare [45] [46] and Information Security [47][31].

While ML can be split into the categories of supervised, unsupervised and reinforcement learning, these categories share a series of meta-components. All ML algorithms can be described in terms of the model being trained, $\hat{f}(x)$, the objective function is maximised or minimised, $L(y, \hat{f}(x))$, and the optimisation algorithm, $\mathcal{O}$, which optimises $\hat{f}(x)$ with respect to $L(y, \hat{f}(x))$ [48]. The following sections will deal with these meta-components.

2.1.1 Learning Models

In the context of ML, a model, $\hat{f}(x)$, is a mathematical representation of knowledge which may be used to make predictions given a set of independent variables. Fundamentally it is the goal of the ML process to produce a representation of $\hat{f}(x)$ which can be used to make the most accurate predictions which generalise well to future cases [49]. In the case of NNs the model is represented with matrices which hold optimised perceptron weights and biases [42]. In the case of support vector machines, it is the hyperplane which separates different classes of data points[48]. For the purposes of this thesis, one learning model will be focused upon: NNs. In the following chapters, the implementation of 'split' NNs for vertically distributed learning will be given.

2.1.1.1 Neural Network Classifiers

NN classifiers were designed to mimic the structure of the human brain. The human brain can be described as a mesh of interconnected neurons, making and breaking connections during the process of learning. While NNs are significantly less complex than real brains, they follow the same approximate architecture. NNs are a thatched matrix of interconnected 'perceptrons' built-in layers. Each layer takes in a set of signals which are connected to each perceptron by a weight value. When the weighted inputs are combined with an activation signal and fed into a perceptron, the perceptron will add these signals together with a bias value. If the total value of the signal plus bias is greater than the threshold of the perceptron, the perceptron will activate and pass its own signal forward to the next layer of perceptrons. The signal is fed forward in this way until a final layer is reached and this will form a prediction.

When learning takes place the output of the final layer is compared to true values corresponding to the input data. This comparison is performed using an objective function. The error value produced in by the equation in

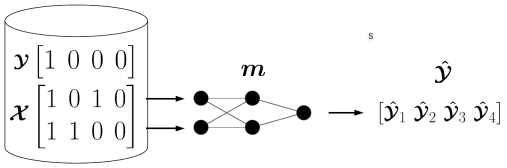


Figure 2.1: A simple binary prediction

$$E = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}$$

Figure 2.2: A simple error formula (Mean Squared Error)

Figure 2.2 is then used to back-propagate gradients to the weights and biases of each perceptron layer by layer in order to minimise the objective function as seen in Figure 2.3. These gradients represent the direction each parameter needs to 'step' in order to improve the model. The gradients are multiplied by the learning rate of the model in order to be adjusted. Once adjustments have been made, the model repeats until the model converges on optimal parameter values. This is an example of an optimisation algorithm.

NNs have become increasingly widely used in recent years due to their optimal performance in a variety of different domains. These include computer vision, natural language processing and speech recognition. In addition, examples can be found of NN classifiers performing optimally when applied to conventional tabular data. In Figure 2.1, a data set is comprised of matrices X and y. Each row in X represents an instance of data to be learned. The y set represents the labels of data instances whereas the X matrix represents the attributes of these instances. X values are fed through the model, m, to create a prediction, $\hat{y}$. Arranging data instances into a matrix allows for the entire batch to be computed in one operation [50].

2.1.2 Objective Functions

An objective function, $L(y, \hat{f}(x))$, which may also be referred to as a cost or loss function, measures the extent to which a predicted output differs from true data [51]. This gives an objective measure of the efficacy of a

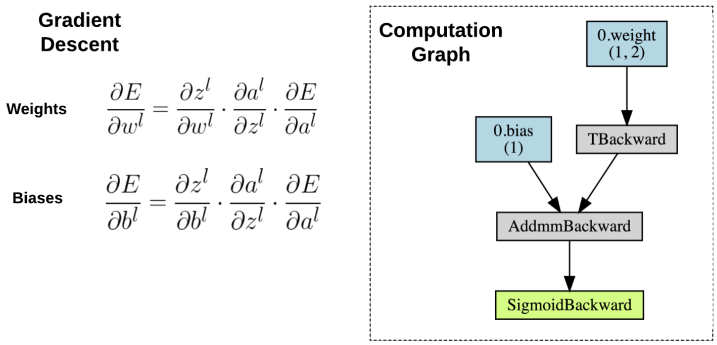


Figure 2.3: Back-propagation calculus and simple computation graph

model against given data points. Objective functions are generally used to evaluate models during training whereas other statistical methods are used to evaluate models once training has been completed. The loss computed with $L(y, \hat{f}(x))$ during training is used as a reference point when deriving the optimal values of $\hat{f}(x)$ when updating the model. There are a number of different objective functions which may be used to evaluate the outputs of a model.

Objective functions are used to measure the extent to which the predictions of a model miss the true values, or targets, which correspond to the model inputs. The output of an objective function provides key feedback to the optimisation algorithm used to adjust model parameters. Two key types of objective functions are regression problems, which are used when a numeric or continuous value is the object of prediction, and classification problems, where a particular class of label is being predicted from the input values.

2.1.2.1 Regression problems

Regression problems can be described as a set of problems where the dependent variable being predicted is defined by a continuous variable. Regression problems are central to economic statistics [52] although they can be applied to any problem where the target being predicted is numerical. The first use of the term *regression* was Sir Francis Galton in anthropological studies in the

late 19th century [53]. There are a number of objective functions which may be used to compute loss in regression problems. These are expressed below where predictions $\hat{y}$ and actual values y are D dimensional vectors with y_i existing on the i^{th} row of y.

Mean Absolute Error (MAE) represents the average magnitude of difference between predictions and actual values regardless of whether the predictions are larger or smaller than the truth value. MAE is resistant to outliers existing in the training data. This is represented in Equation 2.1 [54][55].

$$\sum_{i=1}^{D} |y_i - \hat{y}_i| \tag{2.1}$$

Mean Squared Error (MSE) is similar to MSA, however, it measures the mean squared difference between a predicted value and the actual value. Thus, it is the overall difference between the predicted and truth value, regardless of the size of differences individually or whether these differences are positive or negative. However, unlike MAE, MSE is easily affected by outliers and a range of target values. [54][55]. MSE is described in Equation 2.2.

$$\sum_{i=1}^{D} (y_i - \hat{y}_i)^2 \tag{2.2}$$

Huber Loss is an objective function used for regression problems which combines MSE and MSA [56]. Huber Loss is defined in Equation 2.3. For values of $|y - \hat{y}|$ which are smaller in magnitude, MSE is used to compute the loss. For larger values, MAE is used. The threshold which dictates the use of either method is dictated by δ.

$$L_\delta = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & if\ |(y - \hat{y})| < \delta \\ \delta((y - \hat{y}) - \frac{1}{2}\delta) & otherwise \end{cases} cal \tag{2.3}$$

Other related models include supervised learning, unsupervised learning and reinforcement learning.

2.1.2.2 Optimisation Algorithms

Optimisation algorithms represent the third and final piece in the ML ontology. Optimisation algorithms are used to update the parameters of a learning model to optimise the objective function. There are a number of different optimisation algorithms which are used in practice.

2.1.2.3 Gradient Descent (GD)

GD is an extremely common iterative optimisation algorithm. It is commonly used with linear or logistic regression and NNs. The parameters of the model are used with the computation graph associated with the objective function in order to differentiate gradients for model parameters which establish the direction of travel the optimiser must step in order to improve performance w.r.t the objective function. A learning rate is combined with the gradients in order to create the size parameters that will 'step' in order to improve said performance. The gradient and learn rate are then applied to the model parameters in order to update them. This process repeats iteratively before converging on the optimal model. The key idea here is that by iteratively updating the model parameters the model will create an increasingly better fit to the data. In equation 2.4, θ_j is the j^{th} parameter of the learning model. α represents the learning rate which has been applied. m is the number of elements in the dataset. $h_\theta(x^i$ is the function which predicts the output for the i^th training example. y^i is the actual output of the model. x^i_j represents the j^{th} feature of the i^{th} training example.

$$\theta_j := \theta_j - \alpha(\frac{1}{m}) \sum_{i=1}^{m} (h_\theta(x^i) - y^i) x^i_j \tag{2.4}$$

2.1.2.4 Stochastic Gradient Descent (SGD)

SGD is a variation on GD which introduces an element of randomness to the model optimisation process. SGD chooses a random subset of the training examples. This is useful when dealing with larger sets by reducing the

computational load required by limiting training to these random subsets. SGD is also useful as it is capable of escaping local optima in the search space for optimal model parameters. This is a major advantage that it has over GD. In the training of NNs, this is particularly useful as it avoids overfitting. However, SGD becomes much more sensitive to the learning rate as gradients are generated from these random subsets. The formula for SGD is the same as GD, but subsets of the data are chosen.

2.1.2.5 Adaptive Gradient Algorithm (AdaGrad)

AdaGrad adjusts the learning rate for each model parameter based on the observed gradients from previous training rounds. Parameters which are frequently updated are allocated a learning rate which is smaller in magnitude. This lets AdaGrad tailor the learning rate to the requirements of each individual parameter. A major advantage of AdaGrad is that it needs far less manual tuning than both GD and SGD. This is particularly useful in situations where the data has infrequent updates. However, this also means that the learning rate of the optimisation algorithm tends to decay over multiple rounds. The formula for AdaGrad is given in Equation 2.5 where θ represents the model being tuned, α is the learning rate, r is the variable which keeps track of the sum of element-wise operations over time and g is the current gradient w.r.t the parameters of θ.

$$\theta := \theta - \frac{\alpha}{\sqrt{r + \epsilon}} g \tag{2.5}$$

2.2 Differential Privacy (DP)

DP has been established as a strong standard for evaluating privacy [27][28][29] [30]. This technique originally grew out of privacy preserving statistical databases. This is based on the notion of *privacy through peturbation* [57] which was originally brought into the cryptographic sphere as sum queries [58]. However, where privacy preserving statistical databases add noise to re-

sultant queries to protect the contents of the database, a differentially private function would perturb its true outputs somewhat to make its inputs ambiguous. DP is thus the measure of the *sensitivity* of a function *f*. Sensitivity is formally defined as the maximum amount, over the domain of *f*, that any single argument to f can change the output. This gives a formal guarantee to the privacy of a function, ε. As ε diminishes in size, privacy becomes stronger [59]. However, in mechanisms where DP is created through the addition sof some degree of noise, lower values of ε can mean reduced accuracy [60]. The use of DP allows researchers and objective, formulaic metrics to use when optimising the injection of *noise* parameter of their DP functions.

2.2.1 Information, Entropy and Kullback-Leibler Divergence

Information and entropy were originally popularised by Claude Shannon in 1948 [61]. These concepts are core to understanding data science. In Shannon's paper, he discusses information in terms of using the minimum possible bits used to convey a message. Entropy can be thought of in the same terms. Shannon defined information in terms of bits and. The number of bits of information given by any probability is given by *I(p)* in Equation 2.2.1 and the entropy of an entire probability distribution is given by $H(P)$. In Shannon's theory, to transmit one bit of information to a recipient means to reduce the uncertainty of the message by two.

$$
\begin{aligned}
I(p) &= -log_2(p) \\
H(P) &= -\sum_i p_i log_2(p_i)
\end{aligned}
$$

$$
\begin{aligned}
I(0.5) &= -\log_2(0.5) = 1, \\
H(P) &= -2 \times 0.5 \log_2(0.5) = 1.
\end{aligned}
\tag{2.6}
$$

Suppose it is necessary to communicate a message about the weather and the possibilities of sun and rain are equally likely (Figure 2.4, Equation 2.6).

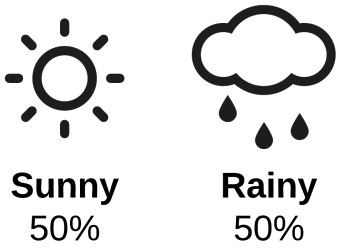


Figure 2.4: Information in communications

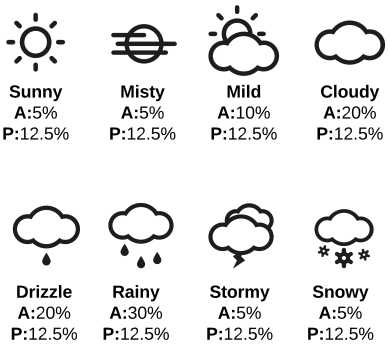


Figure 2.5: Information in communications

In this case, if it were to be forecast that it is sunny, the *uncertainty* of the weather forecast would be reduced by a factor of two. This is equivalent to one bit of information being transmitted. As both probabilities are equally likely, the average information or uncertainty of the distribution is also 1. The weather station may use any number of bits to communicate this information, but the maximum use-able information which would be transmitted in any one broadcast is limited to one.

$$H(P,Q) = -\sum p_i log_2(q_i) \tag{2.7}$$

$$D_{KL}(P||Q) = H(P,Q) - H(P) \tag{2.8}$$

$$D_{KL}(P||Q) = -\sum_{1=0}^{n} p(x_i) log\left(\frac{q(x_i)}{q(x_i)}\right) \tag{2.9}$$

$$H(P) = -8 \times 0.125 \log_2(0.125) = 3 \tag{2.10}$$

$$\begin{aligned} H(P,Q) &= -4 \times 0.125 \log_2(0.05) - 2 \times 0.125 \log_2(0.2) \\ &\quad - 0.125 \log_2(0.1) - 0.125 \log_2(0.3) \\ &= 4 \times 0.54 + 2 \times 0.29 + 0.41 + 0.21 \\ &= 3.36 \end{aligned} \tag{2.11}$$

$$D_{KL}P||Q = 3.36 - 3 = 0.36 \tag{2.12}$$

Consider a more nuanced example where there are eight potential weather categories Figure 2.5. If it is assumed that the probability distribution among all-weather categories is equal, a probability distribution is given, P. However, this may differ from the probability of the actual weather distribution, A. This measurement difference in probability distribution is called *cross-entropy* [62]. The formula for cross-entropy is shown in Equation 2.7. If the cross-entropy of P and Q is equal to the entropy of P, then this means that the two distributions are equal. However, if the cross-entropy of P and Q exceeds the entropy of P, the probability distributions are said to diverge. The amount that the cross-entropy and entropy differ is called the Kullback-Leibler Divergence (Equation 2.8) [62]. This can also be formulated where p and q are functions operating on x (Equation 2.9). The divergence in the predicted and actual weather forecast is in Equations 2.10, 2.11 and 2.12. This is often used in machine learning as a measure for loss [62]. However, this vocabulary is also used when discussing DP.

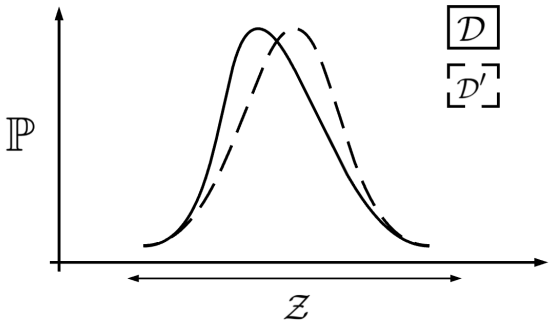


Figure 2.6: $\mathcal{D}$ and $\mathcal{D}'$ divergence

2.2.2 Differential Privacy Definitions

An algorithm is differentially private if, when performing some analysis task on an arbitrary set of data, the removal or addition of any one data instance creates an indistinguishable result from another instance in the dataset [63] [59]. This condition is formally defined for a function ($\mathcal{F}$), analysing two neighbouring datasets with a Hamming distance (the number of locations where $\mathcal{D}$ and $\mathcal{D}'$ differ) of 1 (Equation 2.13) and any output ($\mathcal{S}$) if Equation 2 holds [59]. In Equation 2.14, the privacy protection is formally defined by ε [63] [59]. This definition accommodates multiplicative differences in the distribution of $\mathcal{S}$ between $\mathcal{D}$ and $\mathcal{D}'$ and is referred to as (ε, 0) DP. A graphical representation of a (ε, 0) probability distribution between $\mathcal{D}$ and $\mathcal{D}'$ can be seen in the Figure 2.6. The probability distributions diverge slightly. However, there is no point where they do not overlap. The probability of the result having been produced by $\mathcal{D}$ is higher as it tends to negative values of $\mathcal{Z}$, and as $\mathcal{Z}$ gets larger, the probability density shifts to $\mathcal{D}'$. This means that there are some queries which are less private than others. However, there are no queries which could not be explained as the opposing set.

$$|\mathcal{D} \cap \mathcal{D}'| = 1 \tag{2.13}$$

$$\mathbb{P}[\mathcal{F}(\mathcal{D}) \in \mathcal{S}] \leq e^{\varepsilon}\mathbb{P}[\mathcal{F}(\mathcal{D}') \in \mathcal{S}] \tag{2.14}$$

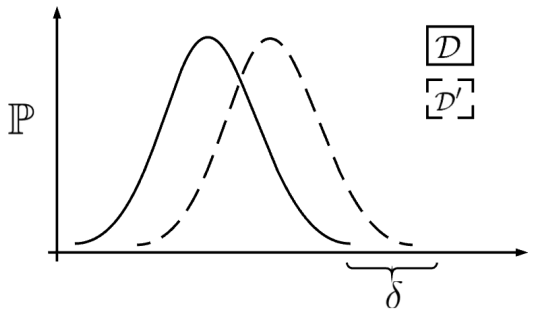


Figure 2.7: $\mathcal{D}$ and $\mathcal{D}'$ divergence with δ

However, $\mathcal{F}$ may not always defined so that the tails of $\mathcal{F}(\mathcal{D})$ and $\mathcal{F}(\mathcal{D}')$ neatly line up (Figure 2.7). It may be the case that there are a minority of queries which give information around unique instances in the dataset. In these cases, the privacy of a function $\mathcal{F}$ with respect to (ε, δ) is defined. This allows for an additive difference between $\mathcal{F}(\mathcal{D})$ and $\mathcal{F}(\mathcal{D}')$, δ. This is shown in Equation 2.15.

$$\mathbb{P}[\mathcal{F}(\mathcal{D}) \in \mathcal{S}] \leq e^{\varepsilon}\mathbb{P}[\mathcal{F}(\mathcal{D}') \in \mathcal{S}] + \delta \tag{2.15}$$

One property of DP is *composability*. This translates into the extent to which someone can compute the overall privacy loss associated with multiple DP queries. DP can also be defined in terms of a privacy loss random variable; C, where aux is the auxiliary input. The privacy loss of any given output is measured $(\hat{y})$ in the range of possible values of $\hat{y}$. C is the random variable produced when evaluating the privacy loss of any given outcome. This is illustrated in Equation 2.16 [27] [64].

$$C(\hat{y}; \mathcal{M}, aux, \mathcal{D}, \mathcal{D}') \triangleq log\frac{\mathbb{P}\left[\mathcal{M}(aux, d) = \hat{y}\right]}{\mathbb{P}\left[\mathcal{M}(aux, d') = \hat{y}\right]} \tag{2.16}$$

2.2.3 Renyi Differential Privacy (RDP)

A relaxation of the definition of privacy is Renyi DP (α, ε). This generalisation uses Renyi divergences between $f(D)$ and $f(D')$; α. Renyi divergence is defined in Equation2.17 [65]. For the endpoints of the interval $(1, \infty)$. The

Renyi divergence is defined through continuity. $D_1(P \mid\mid Q)$ is set to be $lim_{alpha \to 1} D_\alpha(P \mid\mid Q)$. This is verified as equal to Kullback-Leiber divergence (Equation 2.18). This is also known as relative entropy [65].

$$D_\alpha(P||Q)) \triangleq \frac{1}{\alpha - 1} log E_{x \sim Q}[\left(\frac{P(x)}{Q(x))}\right)^\alpha \tag{2.17}$$

$$D_1(P \mid\mid Q) = E_{x \sim P} log \frac{P(x)}{Q(x)} \tag{2.18}$$

Renyi Divergence with $\alpha = \infty$ and DP are thus related. The randomised mechanism f is differentially private (ε) if satisfies the equation only if its distribution over any two inputs D and D' satisfies the following (Equation 2.19). This is also true for a finite value of α (Equation 2.20).

$$D_\infty(f(D) \mid\mid f(D)) \leq \varepsilon \tag{2.19}$$

$$D_\alpha(f(D) \mid\mid f(D)) \leq \varepsilon \tag{2.20}$$

A randomised mechanism $f : D \to R$ has (α, ε) if for any adjacent $D, D' \in \mathcal{D}$, Equation 2.21 holds [63].

$$D_\alpha(f(D)||f(D')) = \frac{1}{\alpha - 1} log E_{\theta \sim f(D')}[\left(\frac{f(D)(\theta)}{f(D')(\theta)}\right)^\alpha] \leq \varepsilon \tag{2.21}$$

RDP is credited as being cleaner than conventional DP when comparing algorithms over compositions [66]. RDP does not need to specify (ε, δ) for each instance being evaluated. This allows this mechanism to elegantly handle compositions of heterogeneous mechanisms.

2.2.4 Differentially Private Deep-Learning (DPDL)

DP has also been applied to deep-learning [67][68]. During the gradient descent process, model parameters (θ) are updated over multiple epochs

(t) in order to model the target function present in the training data. This process is called stochastic gradient descent (SGD). Using the chain rule in differential calculus, the SGD algorithm computes the gradients of each parameter which are necessary to minimise the error of the classifier w.r.t the training data (x, y). The SGD algorithm takes a *step* in the direction of this gradient, determined by the learn-rate (η), for each parameter in order to minimise the loss (L) function (Equation 2.22). This can also be done more efficiently with data as batches, this is known as large-batch SGD. (Equation 2.23)

$$\theta_{t+1} = \theta_t - \eta_t \nabla_{\theta_t} L(\theta_t, x_t, y_t) \tag{2.22}$$

$$\theta_{t+1} = \theta_t - \frac{\eta_t}{B} \sum_{i=1}^{B} \nabla_{\theta_t} L(\theta_t, x_{(t,i)}, y_{(t,i)}) \tag{2.23}$$

$$[x]_C = x / max(1, || \ x \ ||_2 \ / C) \tag{2.24}$$

$$\theta_{t+1} = \theta_t - \frac{\eta_t}{B} \sum_{i=1}^{B} [\nabla_{\theta_t} L(\theta_t, x_{(t,i)}, y_{(t,i)})]_C + N(0, \sigma^2 C^2 I) \tag{2.25}$$

A Gaussian mechanism was proposed in recent work [64] to create DP SGD. Here, an upper bound on the l_2-norm of gradients at each epoch is used to create a clipping parameter (C) (Equation 2.24). A noise multiplier is added, this is the ratio between the clipping parameter and the standard deviation of noise applied at each update in each epoch (Equation 2.25). This is described in the algorithmic form in Figure 2.8 and illustrated in Figure 2.9. This work has been implemented in PyTorch, allowing researchers to take advantage of these techniques in the form of differentially private optimisers. ß

Figure 2.8: Differentially Private Backpropagation Algorithm
[64]

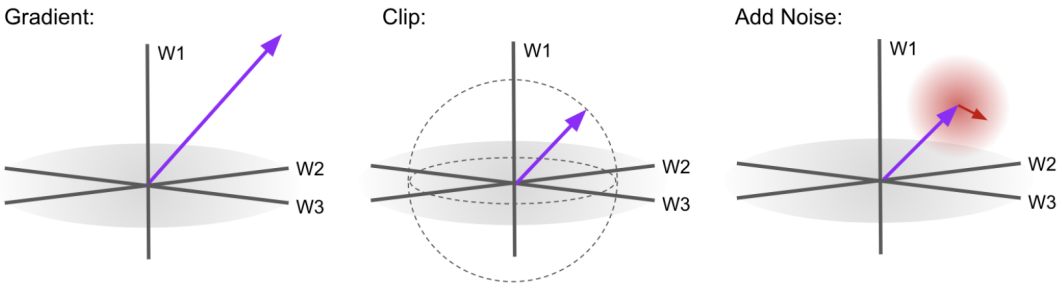


Figure 2.9: Differentially Private Backpropagation Algorithm Visualisation
[63]

2.2.5 Private Aggregation of Teacher Ensembles (PATE)

In this approach, private data (X, Y) is broken down into n subsets (X_n, Y_n) which are trained on n algorithms. These trained models (m_n) are aggregated into a *teacher* meta-classifier (tM). Unlabeled public data (X) can be labelled using tM to create a dataset $(X, \hat{Y})$. This newly labelled public data is used to train a *student* classifier (sM) [27] The sM is then able to be shared as a differentially private classifier. This architecture is shown in Figure 2.10.

When $\hat{y}$ votes are combined, this can cause issues in even distributions of M, or in cases where the teacher votes have tied. Let $\mathcal{R}$ be the range of classes that $\hat{y}$ can hold. The label count for any given $\hat{y} \in [\mathcal{R}]$ for input $(\overrightarrow{x})$ is the number of teachers, tm_n which have chosen $\hat{y}$. This is summarised in equation 2.26. To further introduce ambiguity to the labelling process and to settle ties in the case of even votes, a small amount of noise is added to n. This is achieved through a Laplacian mechanism (Equation 2.2.5).

$$n_{\hat{y}}(\overrightarrow{x}) = \mid \{i : i \in [n], f_i(\overrightarrow{x}) = \hat{y}\} \mid \tag{2.26}$$

$$f(x) = \underset{\hat{y}}{argmax} \left\{ n_{\hat{y}}(\overrightarrow{x}) + Lap\left(\frac{1}{\gamma}\right) \right\} \tag{2.27}$$

A student classifier is then to be trained on unlabeled public data which is not sensitive and has been labelled using the discussed aggregation mech-

anism. In this mechanism, the privacy loss only relates to the queries being performed on *tM*. Thus the privacy of this approach can also be strengthened by choosing a random subset of teacher votes at the labelling of each instance [27]. After training sM on $(X, \hat{Y})$ the privacy of the original (X, Y) values is preserved even if both the architecture and parameters of sM are reverse-engineered.

PATE requires the presence of an unlabeled dataset in order to function. However, should there be no public set available, a set of plausible instance values $(\hat{X})$ could be generated through a GAN (Generative Adversarial Networks) approach.

2.2.6 PATE-G

Where conventional PATE is trained on a fully-labelled public dataset, it can also be trained on a partially labelled set through GANs [69]. When applied to PATE, this methodology is referred to as PATE-G. GANs are trained in a semi-supervised fashion, with two models, a *generator* and a *discriminator*. The generator creates sample data taken from a Gaussian distribution of the actual (X, y) set [27]. The discriminator is then trained to tell the difference between actual X, Y values and $(\hat{X}, \hat{Y})$ ones. The cost function of the generator is to minimise the number of $(\hat{x}, \hat{y})$ instances which are correctly classified as $(\hat{x}, \hat{y})$. The cost function of the discriminator is to maximise the number of $(\hat{x}, \hat{y})$ instances it classifies correctly. Both models compute loss and adjust their parameters at the same time. Once learning is complete in this iteration, the generator creates a new distribution and proceeds to identify the next set of approximations. This is applied to PATE-G where the privacy of the labelled public data set is strengthened by only labelling a small subset using the teacher model [27]. A generator produces a Gaussian distribution of the labelled $(X, \hat{Y})$ values and fills the blanks in the unlabeled instances of the public set. The discriminator is trained to discern the Gaussian labels from those labelled by *tM*.

2.3 Cryptography

In this section, cryptographic techniques for the computation of ciphertext are discussed. These allow for analytics and machine learning techniques to be performed on data while it remains encrypted. This allows information flows to be designed which hide the true values of input data while facilitating the consumption of plaintext outputs. In this section, HE and SMC.

2.3.1 Secure Multi-party Computation

SMC provides the capacity to do FHE computations in a distributed setting [70]. This is based upon the notion of secret-sharing [71]. This is when the secret number x is split into a set of shares, $X'[n]$, where the sum of all members of $X'[n]$ is equal to x. The idea is that each element of $X'[n]$ may be distributed to n parties. So long as these parties perform the same operations on their element of $X'[n]$, the sum of this set will still be equal to x, should x have these same operations performed [72][73]. This can be observed in Figure 2.11. It may be observed the computation on the left split into three shares on the right. Both compute a sum that is equal.

Bridging the gap between SMC frameworks and distributed deep learning frameworks has been identified as an important avenue for future work in recent research [74]. The SMCVector was recently accomplished by the OpenMined community, making SMC computation available to PyTorch [72]. This is useful for computing NNs and has been made use of by the secureNN application [75]. While this may have been the case at the time the paper was published, this is being improved upon rapidly with PySyft now able to perform with full SMC capacities.

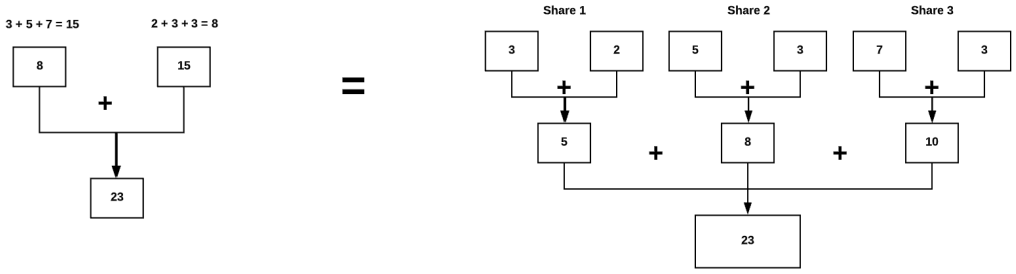


Figure 2.11: SMC Example

2.3.2 Homomorphic Encryption

HE allows cryptographers to perform arbitrary mathematical operations on a ciphertext without first decrypting it [76]. The resulting product of this calculation can be decrypted to reveal the answer to the said calculation in plaintext. Traditionally data scientists must be trusted with sensitive training data in plaintext in order to train models. However, the functionality provided by HE schemes stops this plaintext exposure from being a necessity. HE can be subcategorised into two groups; partially homomorphic (PHE) and fully homomorphic (FHE). PHE schemes allow for only some logical and mathematical operations to be performed on training data whereas FHE schemes allow for all operations to be performed.

2.3.2.1 Rivest Shamir Adleman (RSA)

The original RSA paper was written in 1978[77]. While this was never originally intended for homomorphic purposes, it allows for multiplicative homomorphic operations. This makes RSA only *partially* homomorphic. Using this scheme, a sender can encrypt their plain-text m_1 and m_2 by raising these to the power of e and calculating the modulo N on this output. In this circumstance, the e and the N are both publicly known. This turns m_1 and m_2 into c_1 and c_2 respectively. it is proven that the multiplicatively homomorphic properties of RSA in Equation 2.28.

$$c_1 = m_1^e \pmod N; c_2 = m_2^e \pmod N$$

$$\begin{aligned} c_1 \times c_2 &= m_1^e \times m_2^e && \pmod N \\ &= (m_1 \times m_2)^e && \pmod N \end{aligned} \tag{2.28}$$

Unlike other schemes [76], the multiplication operation does not incur any noise or inaccuracy overhead through homomorphic operations. RSA can also be used to perform a divide operation on encrypted messages. This is observed in Equation 2.29.

$$\begin{aligned} c_1 \div c_2 &= m_1^e \div m_2^e && \pmod N \\ &= m_1^e \times m_2^{-e} && \pmod N \\ &= (\frac{m_1}{m_2})^e && \pmod N \end{aligned} \tag{2.29}$$

2.3.2.2 Paillier

Invented in 1999, the Paillier is a probabilistic asymmetric cryptographic scheme [78]. The advantage of the Paillier system lies in its decisional composite residuosity assumption. This claims that given composite integers n and z, it is difficult for an attacker to figure out which value of y satisfies the following condition posed in Equation 2.30.

$$z \equiv y^n \pmod{n^2} \tag{2.30}$$

The public and private keys are chosen in Paillier from two large prime numbers p and q, where $p = q$. $n = pq$ is computed with $\lambda = \phi(n)$ where $\phi(n) = (p-1)(q-1)$ and $\mu = \phi(n)^{-1} \pmod n$.
Our encryption key and decryption key are (n, g) and (λ, μ), respectively. The variable, g is a random integer value of $g \in \mathbb{Z}_{n^2}^*$. $Z_{n^2}^*$ is defined as the set of integers between 1 and n^2 that are relatively prime to n^2.

For the first element of the public key, a positive integer less than n must

be chosen. Then a random positive integer r is chosen which is less than co-prime to n. The ciphertext is given by Equation 2.31. In order for c to be decrypted, it is required that $c < n^2$. c is decrypted to m with Equation 2.32.

$$c = g^m \times r^n \pmod{n^2} \tag{2.31}$$

$$\begin{aligned} m = L(c^\lambda \pmod{n^2}) \cdot \mu \pmod{n} \\ \text{Where: } L(x) = \frac{x-1}{n} \end{aligned} \tag{2.32}$$

The Decisional Composite Residuosity Assumption (DCRA) is defined such that the complexity of processing n^{th} residue classes has intractable computational complexity. However, this acts as the trapdoor function for the Paillier scheme. A residue class is defined as a set of integers that are congruent modulo n for any positive integer n. A number z is defined as a n^{th} residue modulo n^2 if a number exists where $y \in^*_{n^2}$ such that Equation 2.30.

$$z = y^n \pmod{n^2} \tag{2.33}$$

Distinguishing n-th residues from non n-th residues is computationally complex as it cannot be established polynomially. Since inverting the encryption equation of the Paillier scheme is the composite residuosity class problem, Paillier ensures semantic security [78].

Paillier demonstrates the homomorphism from $(^*_{n^2}, \times)$ to $(^*_{n^2}, +)$ via a lemma. Let the n-th residuosity class of w with respect to g be denoted as $\|w\|$:

$$\forall w_1, w_2 \in Z^*_{n^2} \quad \|w_1 w_2\| = \|w_1\|_g + \|w_2\| \pmod{n} \tag{2.34}$$

This lemma allows the following homomorphic properties:

- The product of c_1 and c_2 is equal to the sum of m_1 and m_2
- The product of c_1 and g^{m_2} is equal to the sum of m_1 and m_2

- c_1 raised to the power of m_1 is equal to the product of m_1 and m_2

2.3.2.3 Gentry

Gentry's HE scheme is a form of lattice-based cryptography. As more than calculations are performed on ciphertexts encrypted under Gentry's scheme, more noise is added to the resultant output [76]. This limits the number of operations which can be performed before the accuracy of the plain-text mapping is reduced. However, this is counteracted by the *bootstrapping* mechanism, which allows for the plain text to be re-encrypted homomorphically in order to refresh the noise in the signal. While this is still unfeasible due to big-O complexity, it has been the foundational work in FHE and the basis of many works since in cryptography.

2.3.2.4 Homomorphic Encryption for Machine Learning

Historically, linear regression models have also been computed homomorphically [79]. However, with some HE schemes, it is possible to do encrypted machine learning. Researchers recently developed a system where a model can be used to predict while encrypted on a cloud server. This can be observed in Figure 2.12 [80]. Other recent work includes predicting all required data types under an FHE scheme. This is still at a high computational cost but is still promising for the field of PPML [81]. The SEAL library was recently implemented, a library which facilitates computation on encrypted data. This has been made accessible to Python through PySeal [82].

2.3.3 Private Set Intersection

PSI [83, 84, 85, 86] is a multi-party computation cryptographic technique which allows two parties, where each holds a set of elements, to compute the intersection of these elements, without revealing anything to the other party except for the elements in the intersection. Different PSI protocols have been proposed [87, 88, 89, 90] and employed for scenarios such as private contact

Figure 2.12: Cloud-based HE Deep Learning
[80]

discovery [91] and also privacy-preserving contact tracing [37].

In this work, a PSI implementation based on a Diffie-Hellman key exchange is employed that uses Bloom filters compression to reduce the communication complexity [37]. This protocol works with two parties computing the intersection between their sets. However, the chosen PSI framework can be replaced with an alternative implementation, for instance, to compute directly the intersection of datasets coming from more than two parties [92].

Other core methods of using HE will be covered in future chapters.

2.4 Federated Learning

The centralisation of resources in machine learning information flows forces a Pareto efficiency trade-off between data utility and privacy. When participating in these flows, data owners cannot know that their data has not been sold, retained for far longer than intended or otherwise used for purposes outside the understood context-relative informational norms [16]. A key risk factor here is the data copy problem. Data once copied, even by well-

meaning actors, cannot ever be guaranteed to have been destroyed, as has become painfully clear by the takedown of the DukeMTMC dataset ([93]). The controversy surrounding the unintended usage of the public dataset was raised when the faces of ordinary people were used without permission to serve questionable ends. This resulted in the data being revoked, breaking the social context of the Data Subject's participation in that flow ([94]). Copies of the data and derivative datasets were still freely available on the internet and academic papers are still being published. This is such a pervasive issue that websites exist for tracking the leakage and misuse of people's facial data for inappropriate, extra-contextual use cases ([95]).

Name	Date	Org	Base	PSI	VFL	HFL	DP	HE	SMPC	ZkAC	OPC
PySyft	Jul17	OpnMnd	TH,+	3rd	Y	Y	3rd	3rd	Y	3rd	Y
TFF	Sep18	Google	Any	N	N	Y	3rd	N	3rd	N	N
FATE	Sep18	WeBank	TH,+	N	N	Y	3rd	N	Y	N	N
LEAF	Dec18	CMU	TF	N	N	Y	3rd	N	N	N	N
eggroll	Jul19	WeBank	TF,+	N	N	Y	3rd	N	N	N	N
PaddleFL	Sep19	Baidu	PD	Y	Y	Y	Y	N	Y	N	N
FLSim	Nov19	iQua	TH	N	N	Y	N	N	N	N	N
DT-FL	Nov19	BIL Lab	TH	N	N	Y	N	N	N	Y	N
Clara	Dec19	NVIDIA	TF	N	N	Y	Y	N	N	N	N
DP&FL	Feb20	SherpaAI	TF,SKL	N	N	Y	Y	N	N	N	N
IBMFL	Jun20	IBM	KS+	N	N	Y	Y	N	N	N	N
FLeet	Jun20	EPFL	TF?	N	?	?	Y	N	N	N	N
IFed	Jun20	WuhanU	CS	N	?	?	Y	N	N	N	N
FedML	Jul20	FedML	TH	Y	Y	N	N	N	N	N	N
Flower	Jul20	Cmbrdge	Any	N	N	N	Y	N	N	N	N

Table 2.1: Federated Learning Systems Feature Support Comparison
We extend the work of [96] to compare PETs feature support in existing federated learning architectures. Org: Organisation associated with the tool. Base: core machine learning technology. PSI: does the framework provide an API for PSI computation? VFL: does the framework provide an API for some form of Vertically Federated Machine Learning? HFL: does the framework provide an API for some form of Horizontally Federated Machine Learning? DP: does the framework provide an API for some form of Differential Privacy? HE: does the framework provide an API for some form of Homomorphic Encryption? SMPC: does the framework provide an API for some form of SMC? ZkAC: does the framework provide an API for some form of Zero-knowledge Access Control? OPC: Does the framework provide an object-level RPC? TFF - [97]), FATE - [98], LEAF - [99]), eggroll - [100], PaddleFL - [101], FLSim - [102], DT-FL - [103]. Clara - [104], DP%FL - [105], IBMFL - [106]), DT-FL - [103], FLeet - [107], IFed - [108], FedML - [109], Flower - [110]

The increasing prevalence of organised criminal groups online remains

a risk, forcing centralised DSs to respect informational norms and securely store the user's data. Adequate maintenance of confidentiality, integrity, and availability of data represents an increasing liability for processors. Between 2015 and 2020, cybersecurity spending worldwide almost doubled from $75.6 billion to $124 ([111]). The recent General Data Protection Regulation (GDPR) legislation ([112]) requires explicit consent from Data Subjects to allow the processing of their data for any purpose. GDPR and similar laws represent a formidable bureaucratic overhead to researchers who process data concerning EU citizens. This issue is further compounded by a lack of trust between Data Subjects and DSs. Under these circumstances, research on private data is blocked due to privacy ramifications- without ever contributing to an information flow's contextually driven purpose or values. If these obstacles are removed, research goes ahead with potentially disastrous social and political consequences for the Data Subjects and their community.

2.4.1 Centralised Learning

The incumbent learning technique in data industry is centralised learning. This is where data, researcher and algorithm all occupy the same host at the time of learning. This is far more efficient than FL approaches when it comes to network overhead. The reason for this is that, unlike in FL approaches, no network communications are required. This can also lead to more efficient use of computation resources as FL necessitates multiple different data sources be processed on different machines.

However, centralised learning offers no privacy to data owners. The researcher must first pull data to their device in order to train their model locally. As a result, the researcher is able to read their data values in plaintext. There is nothing to stop data being copied and sold on, no way to prove that data has been deleted once processed and no defences against data being bundled with other data sources in order to reveal aspects about the subjects which they did not intend when they submitted their data [7].

Figure 2.13: Federated Learning Protocol Diagram
[43]

2.4.2 Horizontal Federated Learning

Horizontal is the new standard in privacy-preserving ML, with a significant body of work going into large-batch synchronous training [25]. While there exist other open-source frameworks for distributed deep learning, such as SINGA [26], very few are as well integrated or have the same capacity for methodology aggregations. This FL methodology copies one atomic model and distributes it to multiple hosts with data. This facilitates training on a large corpus of federated data [113] [114]. This is observed in Figure 2.13. In the worst case scenario it has been shown in the convex case with IID data that in the worst case scenario, the global model is no worse one trained on a single host. These hosts then train models and aggregate these model updates to update the final model [115]. The researcher never sees the data directly, only aggregates model gradients at the end [116]. However, this requires high network bandwidth and is vulnerable to invalidated input attacks [117]; where an attacker might want to create a bias toward a particular classification type for a particular set of input data. Network architecture may be observed in Figure 2.14 [114].

Many imaginings of the way that FL will be applied in industry is to store

Figure 2.14: Federated Learning Network Diagram
[114]

personal data on phones. This phone can then be plugged into a wall and connected to a Wi-fi outlet. Models are sent down to people's phones and then trained upon [113]. The remote Android attestation mechanism is used in recent work [113], this helps to protect against poisoning attacks [118] by verifying the identity of hosts. Another recent work, SafetyNets, proposes a mechanism where the correct processing of private data is verified through mathematical proof of completion [119].

2.4.3 Vertical Federated Learning

VFL is the concept of collaboratively training a model on a dataset where data features are split amongst multiple parties [120]. For example, different healthcare organizations may have different data for the same patient. Considering the sensitivity of the data, these two organizations cannot simply merge their information without violating that person's privacy. For this reason, a machine learning model should be trained collaboratively, and data should be kept on the corresponding premises. Machine learning algorithms for vertically partitioned data is not a new concept, and many studies for new models and algorithms have been proposed in this area [121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132]. Existing open-source VFL frameworks include FedML [133], which implements multi-party linear models [131].

2.4.3.1 Split Neural Networks

Split learning is a concept of training a model that is split into segments held by different parties or on different devices. A neural network model trained this way is called a Split Neural Network, or SplitNN. In SplitNN, each model segment transforms its input data into an intermediate data representation (as the output of a hidden layer of a classic neural network). This intermediate data is transmitted to the next segment until the training or inference process is completed. During backpropagation, the gradient is also propagated across different segments. Compared to data-centric FL, split learning can also be useful to reduce the computational burden on data owners, who in many real-world scenarios may have limited computational resources [134, 135].

Split Neural Networks (SplitNNs) were first discussed as a PPML technique in late 2018 [74]. The training of a neural network (NN) is *spli* across multiple hosts. Each segment in the chain is a self-contained NN that feeds into the segment in front. The host with the training data has the beginning segment of the network and the end segment. Intermediate segments of the chain are held by participating hosts.

In its initial design, this was between a centralised model (*Bob*) and a set of k hosts with data (*Alice*). The initial intention was that this would be either built as a centralised set-up, where image snapshots of model parameters are sent backwards and forward to the server as more data-owning hosts train. These two modalities are shown in Figure 2.15.

This is shown to be completely identical to training a single atomic model and to have outstanding advantages over FL in terms of computation and bandwidth requirements [43]. The SplitNN is a new and largely untested technique for preserving privacy, but there is a great capacity for these to be reconfigured and experimented with. It is even potential for these to be trained on both vertically and horizontally aligned data [74]. This is not possible with FL, the current standard for PPML. Some potential ontologies

Figure 2.15: SplitNN; centralised and peer-to-peer mode
[43]

Figure 2.16: SplitNN Performance Comparison
[74]

proposed as future work are shown in Figure 2.17. A performance comparison of to and Large Batch Stochastic Gradient Descent (LBSGD) iis observed in Figure 2.16

However, what is not addressed by this work is the potential for data leakage when Bob receives forward passes from Alice. In the centralised methodology, Bob has knowledge of the model at the time of training as well as the gradients being fed back. This allows Bob to maintain knowledge of the model parameters as Alice and Bob update over-training. Given Bob knows the model parameters at each training epoch, as the model behaves deterministically, this allows Bob to reverse engineer the input signal, which creates the weights, particularly in cases where the output dimensionality is vast. This problem of invertibility has been partially addressed in recent work, which attempts to use distance correlation to optimise parameters [136]. However, it is yet to be asserted that in order to reduce data leakage, it

Figure 2.17: SplitNN Different Configurations
[74]

must be assured that the recipient of the model segment activations *must not* have knowledge of the model which produces the activations.

2.4.4 Model Inversion Attacks and Defences in Split Neural Networks

Model inversion is a class of attack which attempts to recreate data fed through a predictive model, either at inference or training time [137, 138, 139, 140]. It has been shown that model inversion attacks work better on earlier hidden layers of a neural network due to the increased structural similarity to input data [141]. This makes SplitNNs a prime target for model inversion attacks.

NoPeekNN is a method for limiting data reconstruction in SplitNNs by minimising the distance correlation between the input data and the intermediate tensors during model training [39]. NoPeekNN optimises the model by a weighted combination of the task's loss and a distance correlation loss, which measures the similarity between the input data and the intermediate data. NoPeekNN's loss weighting is governed by a hyperparameter $\alpha \in [0, \infty]$. While NoPeekNN was shown to reduce an autoencoder's ability to reconstruct input data, it has not been applied to an adversarial model inversion attack.

Abuadbba et al. [142] apply noise to the intermediate tensors in a SplitNN to defend against model inversion attacks on one-dimensional ECG data. This defence is framed as a differential privacy mechanism [143]. However, in that work, the addition of noise greatly impacts the model's accuracy for even modest epsilon values (98.9% to roughly 90% at $\epsilon = 10$). A similar method, *Shredder*, was introduced by [144], which adaptively generates a noise mask to minimise mutual information between input and intermediate data.

2.5 Identity

ML is revolutionising how data is dealt with. This is catalysed by hallmark innovations such as AlphaGo [145]. Attention has turned to attractive domains, such as healthcare [146], self-driving cars and smart city planning [147]. Ernst and Young estimate that National Health Service (NHS) data is worth £9.6 Billion a year [148]. While this burgeoning application of data science has scope to benefit society, there are also emerging trust issues. The data-sets required to train these models are often highly sensitive, either containing personal data - such as data protected under the GDPR in the EU [149] - or including business-critical information. Additionally, developing and understanding ML models is often a highly specialised skill. This generally means that two or more separate parties must collaborate to train an ML model. One side might have the expertise to develop a useful model, and the other around the data which they want to train the model, to solve a business problem.

2.5.1 Trust and the Data Industry

Trust is a complicated concept that is both domain and context-specific. Trust is directional and asymmetric, reflecting that between two parties, the trust is independent for each party [150]. Generally, trust can be defined as the willingness of one party to give control over something to another party,

based on the belief that they will act in the interest of the former party [151]. In economic terms, it is often thought of as a calculation of risk, with the understanding that risk can never be fully eliminated, just mitigated through mutual trust between parties [152]. The issue of trust is ever-present in the healthcare industry. Healthcare institutions collect vast amounts of personal medical information from patients in the process of their duties. This information can in turn be used to train an ML model. This could benefit society by enhancing the ability of clinicians to diagnose certain diseases.

DeepMind brought the debate around providing access to highly sensitive, and public, data-sets to private companies into the public sphere when they collaborated with the Royal Free London NHS Trust in 2015. This work outlined 'Streams', an application for the early detection of kidney failure [153]. However, the project raised concerns surrounding privacy and trust. DeepMind received patient records from the Trust under a legal contract dictating how this data could be used. Later this was criticised as being vague and found to be illegal by the Information Commissioner's Office [154]. Furthermore, DeepMind did not apply for regulatory approval through the research authorisation process to the Health Research Authority - a necessary step if they were to do any ML on the data. The team working on Streams has now joined Google, raising further concerns about the linkage of personal health data with Google's other records [155].

While there was significant push back against the DeepMind/Royal Free collaboration, this has not prevented other research collaborations. This includes the automated analysis of retinal images [156] and the segmentation of neck and head tumour volumes [157]. In both these scenarios, the appropriate authorisation from the Health Research Authority was obtained, and the usage of the data transferred was clearly defined and tightly constrained.

2.5.2 Decentralised Identifiers

DIDs are tools which can be used to manage trust in a distributed or privacy-preserving environment. DIDs represent a new type of digital identifier

currently being standardised in a World Wide Web Consortium (W3C) working group [158]. A DID persistently identifies a single entity that can self-authenticate as being in control of the identifier. This is different from other identifiers which rely on a trusted third party to attest to their control of an identifier. DIDs are typically stored on a decentralised storage system such as a distributed ledger, so, unlike other identifiers such as an email address, DIDs are under the sole control of the identity owner

Any specific DID scheme that implements the DID specification must be resolvable to its respective document using the DID method defined by the scheme. Many different implementations of the DID specification exist which utilise different storage solutions. These include Ethereum, Sovrin, Bitcoin and IPFS; each with their own DID method for resolving DIDs specific to their system [159].

The goal of the DID specification is thus to ensure *interoperability* across these different DID schemes such that, it is possible to understand how to resolve and interpret a DID no matter where the specific implementation originates from. However, not all DIDs need to be stored on a ledger; in fact, there are situations where doing so could compromise the privacy of an individual and breach data protection laws, such as with GDPR. Peer DIDs are one such implementation of the DID specification that does not require a storage system, and in this implementation DIDs and DID documents are generated by entities who then share them when establishing peer-to-peer connections. Each peer stores and maintains a record of the other peer's DID and DID Document [160].

2.5.3 DID Communication (DIDComm)

DIDComm [161] is an asynchronous encrypted communication protocol that has been developed as part of the Hyperledger Aries project [162]. The protocol uses information within the DID Document, particularly the parties' public key and their endpoint for receiving messages, to send information with verifiable authenticity and integrity. The DIDComm protocol has now

moved into a standards track run by the decentralised Identity Foundation [163].

As defined in Algorithm 1, Alice first encrypts and signs a plaintext message for Bob. She then sends the signature and encrypted message to Bob's endpoint. Once the transaction has been received Bob can verify the integrity of the message, decrypt it and read the plaintext. All the information required for this interaction is contained within Bob and Alice's DID Documents. Examples of public-key encryption protocols include ElGamal [164], RSA [77] and elliptic curve based [165]. Using this protocol both Bob and Alice are able to communicate securely and privately, over independent channels and verify the authenticity and integrity of the messages they receive.

2.5.4 Verifiable Credentials

The VC Data Model specification became a W3C recommended standard in November 2019 [166]. It defines a data model for a verifiable set of tamper-proof claims that are used by three roles; *Issuer*, *Holder* and *Verifier* as it can be seen in Figure 2.18. A verifiable data registry, typically a distributed ledger, is used to store the credential schemes, the DIDs, and DID documents of Issuers.

Algorithm 1 DID Communication Between Alice and Bob

1: Alice has a private key sk_a and a DID Document for Bob containing an endpoint ($endpoint_{bob}$) and a public key (pk_b).
2: Bob has a private key (sk_b) and a DID Document for Alice containing her public key (pk_a).
3: Alice encrypts plaintext message (m) using pk_b and creates an encrypted message (e_b).
4: Alice signs e_b using her private key (sk_a) and creates a signature (σ).
5: Alice sends (e_b, σ) to $endpoint_{bob}$.
6: Bob receives the message from Alice at $endpoint_{bob}$.
7: Bob verifies σ using Alice's public key pk_a
8: **if** Verify$(\sigma, e_b, pk_a) = 1$ **then**
9: Bob decrypts e_b using sk_b.
10: Bob reads the plaintext message (m) sent by Alice

When issuing a credential, an Issuer creates a signature on a set of at-

Figure 2.18: Verifiable Credential Roles
[166]

tributes for a given schema using a private key associated with their public DID through a DID document. The specification defines three signature schemes which are valid to use when issuing credentials; JSON Web Signatures [167], Linked Data Signatures [168] and Camenisch-Lysyanskaya (CL) Signatures [169]. This paper focuses on the Hyperledger stack, which uses CL Signatures. In these signatures, a blinded link secret (a large private number contributed by the entity receiving the credential) is included in the attributes of the credential. This enables the credential to be tied to a particular entity without the Issuer needing to know the secret value.

When verifying the proof of a credential from a Holder, the Verifier needs to confirm a number of aspects:

1. The DID of the Issuer can be resolved on the public ledger to a DID document. This document should contain a public key which can be used to verify the integrity of the credential.

2. The entity presenting the credential knows the secret that was blindly signed by the Issuer. The Holder creates a zero-knowledge proof attesting to this.

3. The issuing DID has the authority to issue this kind of credential. The signature alone only proves integrity, but if the Verifier accepts credentials from any Issuers, it would be easy to obtain fraudulent credentials. In a production system at scale, this might be done through a registry,

supported by a governance framework; a legal document outlining the operating parameters of the ecosystem [170].

4. The Issuer has not revoked the presented credential. This is done by checking that the hash of the credential is not present within a revocation registry (a cryptographic accumulator [171]) stored on the public ledger.

5. Finally, the Verifier needs to check that the attributes in the valid credential meet the criteria for authorisation in the system. An often-used example is that the attribute in a valid passport credential is over a certain age.

All the communications between either the Issuer and the Holder, or the Holder and the Verifier are done peer-to-peer using DIDComm. It is important to note that the Issuer and the Verifier never need to communicate.

2.5.5 Federated Learning

In a centralised ML scenario, data is sent to the Researcher, instead in an FL setup the model is sent to each data participant. The FL method has many variations, such as *Vanilla*, *Trusted Model Aggregator*, and *Secure Multi-party Aggregation* [172]. However, at a high level, the Researcher copies one atomic model and distributes it to multiple hosts who have the data. The hosts train their respective models and then send the trained models back to the Researcher. This technique facilitates training on a large corpus of federated data [113] [114]. These hosts then train models and aggregate these model updates into the final model. In the case of Vanilla FL, this is the extent of the protocol. However, this can be extend this with a secure aggregator, a middle man in between the Researcher and the hosts, which averages participant models before they reach the Researcher. To further improve security, this can be extended using Secure Multiparty Aggregation to average models whilst they have been encrypted into multiple shares [72]. The Researcher thus never sees the data directly, and only aggregates model gradients at the

end [116]. However, this requires high network bandwidth and is vulnerable to invalidated input attacks [118], where an attacker might want to create a bias toward a particular classification type for a particular set of input data.

2.6 Conclusions

This chapter has given a background in the literature relating to the core technologies from which the contributions in the later sections are derived. Initially, the components of ML were discussed. DP was covered with a background on entropy and some differentially private techniques. The cryptography methods used in later chapters were discussed as well as some of the state-of-the-art with respect to FL. Finally, identity was dealt with in the context of PPML. In the chapters to follow, novel applications of these building blocks will be explained and demonstrated with experiments.

Chapter 3

Insider Threat Detection

This chapter explores insider threats that continue to present a major challenge for the information security community. Despite constant research taking place in this area, a substantial gap still exists between the requirements of this community and the solutions that are currently available. In this case study, the approach used by contemporary researchers in this area is employed. Synthetic data is analysed, and a model is built in order to prove the feasibility of solving this problem [31].

Specifically, this chapter uses the CERT dataset r4.2 along with a series of machine learning classifiers to predict the occurrence of a particular MIT scenario - the uploading of sensitive information to Wikileaks before leaving the organization. These algorithms are aggregated into a meta-classifier which has stronger predictive performance than its constituent models. It also defines a methodology for performing pre-processing on organizational log data into daily user summaries for classification and is used to train multiple classifiers. Boosting is also applied to optimise classifier accuracy. Overall the models are evaluated through analysis of their associated confusion matrix and Receiver Operating Characteristic (ROC) curve, and the best-performing classifiers are aggregated into an ensemble classifier. This meta-classifier has an accuracy of **96.2%** with an area under the ROC curve of **0.988**.

MIT is defined as someone who is motivated to adversely impact the

mission of an organization with respect to the confidentiality, integrity or availability of information using the privileges associated with their role [173]. Insider attack makes up a considerable portion of the cyber-threat landscape, with around 40% of organizations labelling the vector as the most damaging faced [174], and malicious insiders and hackers make up 47% of data breaches [175]. MIT is also the most costly per record to resolve ($155.6 per leaked record [175]).

Despite the frequent occurrence of MIT attacks, detection and mitigation remain problematic. In 2018, 90% of companies considered themselves vulnerable to insider threats [176]. A further 38% of companies admit that their detection and prevention capabilities are inadequate [174], which demonstrates a substantial gap between the current advancements in MIT detection and growing security requirements. Given the availability of computational resources, using ML techniques to solve problems of larger complexity is increasingly a viable option.

As a field, data-driven approaches to detecting MIT are increasing, but front-line attempts still report more effective models than where machine learning has been applied [177]. These initial attempts have often used a Graph-Based approach [178] and fuzzy logic-based anomaly detection approaches [179].

This paper presents a new methodology for processing organizational log data into a format for classifying whether particular individuals belong to a particular threat archetype on a daily basis. It then outlines the training of multiple learning algorithms in order to classify this threat scenario while experimenting with boosted and non-boosted learning methods. The best-performing algorithms are aggregated using a probability vote in order to create a model which has the largest area under the ROC curve of all the developed models.

The contributions of this work can be summarized as:

- A methodology for splitting MIT into subcategories to improve predictive performance when compared to previous prediction approaches,

which largely treat insider threat as a single category. MIT can take a number of forms, this can complicate prediction for these techniques. The present work identifies a model by which individual threat archetypes are detected through supervised learning algorithms.

- Investigation of boosting when optimizing the performance of classification algorithms in the field of application.
- Demonstrates an approach for aggregating high-performance classifiers into an optimal meta-classifier in the MIT domain.

3.1 Privacy Preserving Machine Learning Considerations

In this case, it was proven that the approach outlined in the previous paragraph was effective on synthetic data. However, the fact this was applied to synthetic data was a major caveat to the study as a whole. It is not hard to imagine that data produced in a lab environment, generated on the basis of a set of preconceived rules presumed to be important by the experiment designers would be highly different in practice from real-world data.

While it would benefit the field of research to release real-world data into the corpus of data available for studying this domain, there are clear unintended side effects that this would cause.

To begin with, to do so would be a flagrant violation of the privacy of corporate employees who would be the Data Subjects in question. This would likely affect the job satisfaction of those involved as their daily routines are exposed to anyone with an internet connection and an interest in MITs.

Secondly, organisations controlling data in this domain would be unlikely to expose data in this way as it may give an unfair advantage to competing organisations. These competitors may use the information pertaining to the behavioural patterns of employees to glean vital information about the logistics of the data controller's business operations and any inefficiencies or

weaknesses that may exist.

Unabated access to logistical information of the data controller's organisation may open avenues to potential attacks on employee-controlled devices. High-level executives may be identified and individually targeted on their way to the workplace or returning home from work. If working from home, their IP address may be analysed to reveal their home location. Allowing their home infrastructure there to be targeted by grabbing hashes of home wifi passwords and cracking hashes. On a more macro level, competitors may be able to analyse weaknesses in data controller logistical operations.

MITs are sparse and occur more than once every seldom in singular organisations. This makes datasets gathered by singular organisations, particularly smaller businesses, inherently inappropriate from machine learning approaches which require a larger number of positive instances to generalise well. However, if it were possible to create a federation of analogous enterprises unified for the purpose of building an effective global model for predicting MIT, this would allow for a far greater data source to be utilised in order to build a far more effective global model which may be shared amongst participants in the federation. The exposure of subject and logistical information would be mitigated by not sharing any data controlled by any one organisation.

3.2 MITs

The most popular approach for dealing with MITs remains unsupervised, anomaly-detection-based approaches [180]. One unsupervised approach - and which was applied to the CERT dataset r6.2 - uses a deep neural network to establish a baseline of normal behaviour for each user for each day and compare it to new days. When anomaly scores are organized into employee percentiles for each day, almost every malicious employee is placed well above the 95th percentile for high anomaly scores [181]. This shows great potential in having the ability to quickly create a shortlist of high-

risk individuals. However, these systems often lack the capacity to identify positive or negative cases of insider threat. They also give no indication as to the nature of the anomaly, whether it could be a malicious event or it could just be caused by employees breaking their usual work habits in innocuous ways. The task then falls on the security operations centre (SOC) to classify the nature of the anomaly. This can be laborious and time-consuming.

These drawbacks can be mitigated when using classification techniques to identify insider threats by training specific models to identify particular attack scenarios. Despite this potential of breaking MIT into smaller categories for prediction, this approach has been rarely identified in the literature. Instances of classifiers being used in research to predict MIT tend to treat MIT as a single class of problem. In addition, data sources used for MIT classification can vary. Most of the data sets used in related classification problems use log information from the systems that individuals have accessed. This can be further sub-categorised into synthetic, non-synthetic and mixed log data. However, recent research also shows data being analyzed from a plethora of different sources.

One example of an approach that breaks the convention of predicting MIT from logs uses psycho-physiological signal data to train SVMs. This is taken from electroencephalography and electrocardiogram sensors which are placed on a small group of participating individuals who either performed an intentional MIT activity or were benign. This data is then used to train SVMs to classify instances with an average accuracy of 86% [182]. While this approach was able to perform with reasonable accuracy, it is difficult to say whether this would be true in a non-staged MIT environment. In addition, the sample size was only 10, and this would have to be tested on a much larger population sub-sample in order for the resultant classifier to be credible. Finally, even if the equipment were available to feasibly perform this kind of analysis, there may be further obstacles when acquiring consent from employees undergoing full-time analysis at work.

Another example that appears in the research uses both classification

and clustering techniques on real-world organizational data [183]. This two-pronged approach attempts to predict which employees in the organization are likely to quit using classification while also using an unsupervised approach to detect which users may be insider threats. The classifier had an accuracy of 73.4% when detecting quitters. The unsupervised approach for detecting insider threats was effective in that all insider threat cases had an anomaly measure above the median score. However, this tended to be the norm among the scores of benign individuals, also casting doubt on the effectiveness of this approach for predicting insider threat as a single class.

One final, single-pronged approach creates a classifier using the CERT synthetic dataset r6.2 to predict insider threat. Here, researchers compare the performance of traditional machine learning algorithms against their long, short-term memory recurrent neural network. This classifier achieved an accuracy of 93.85%, outperforming the next most accurate algorithm by around 5% [180]. This accuracy was achieved by thoroughly pre-possessing the initial log data. Firstly, events are standardized and aggregated into a format around the behaviours and attributes of individuals. Features are then extracted for the training phase and testing phases respectively.

Our methodology takes into consideration the other approaches of MIT detection proposed in the literature, but it expands upon this by introducing boosting, stacked classifiers and the use of behavioural archetypes to narrow down the scope of prediction.

3.3 Methodology

Following the prevalence of previous approaches in predicting insider threats in the CERT synthetic dataset ecosystem [180] [181], this source was chosen to perform the experiments. However, instead of choosing r6.2 with only one instance of each threat, r4.2 was chosen. Unlike r6.2, r4.2 was created as a *dense needle* data set. This contains a large number of positive cases of each threat scenario. This is an ideal classification problem as there is a vast

wealth of positive cases that predictive models can learn the structure of. With more true cases to learn from, the resultant classifier is likely to be far more adept at correctly identifying true future cases.

Dataset Version 4.2 contains around 20GB of employee activity logs for 1000 employees over 17 months. Three different attack scenarios have been simulated, each allocated 30 malicious employees from the 1000-employee pool. These are described as follows:

1. User who DID not previously use removable drives or work after hours starts logging in after hours, using a removable drive, and uploading data to *wikileaks.org*. Leaves the organization shortly thereafter.

2. User begins surfing job websites and soliciting employment from a competitor. Before leaving the company, they use a thumb drive (at markedly higher rates than their previous activity) to steal data.

3. System administrator becomes disgruntled. HE downloads a key logger and uses a thumb drive to transfer it to his supervisor's machine. The next day, HE uses the collected key logs to log in as his supervisor and send out an alarming mass email, causing panic in the organization. HE leaves the organization immediately.

Despite previous research [180], creating a general-purpose model for predicting MIT regardless of the MIT scenario category was not the focus of this work. As the three scenarios provided suggest, MIT can take many forms. Not all scenarios will carry a similar signature. Any model which is applied to MIT as a blanket solution will need to be vague enough not to rule out MIT cases which are distinct in nature but also well-fitted enough to actually catch cases of MIT in their particular scenarios. One model per scenario ensures that even small nuances in each case are learned, whereas a more generalized model may become less accurate when classifying threats across a wider spectrum. If this is successful, the dependent variable can be turned into a categorical variable to predict each distinct scenario. However,

this categorical approach may not be compatible with some kinds of learning algorithms and may also create unnecessary noise for algorithms to deal with. A potentially better approach would be to train a separate model to detect each separate threat scenario. This would allow each model to be as well fitted to its specific scenario as possible.

Data set r4.2 is originally composed of the following sets of logs:

1. Employee Login/Logoff event logs.
2. Device Connect/ Disconnect event logs.
3. Employee HTTP event logs.
4. Monthly Record of Employees.
5. Psychometric Profiles of Employees.
6. File Accesses.

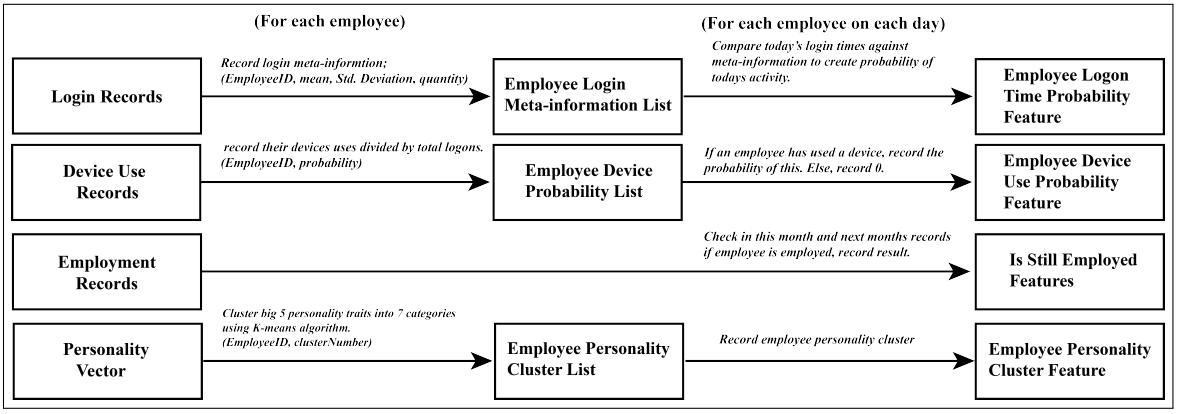


Figure 3.1: Architecture of data pre-processor

3.3.1 Data Aggregation and Feature Selection

In order to decide which elements should be aggregated to generate features, threat scenario one is further analyzed. Three key traits of the scenario are identified:

1. Users who are not usually device users suddenly start using a device;
2. Users who usually only log in during work hours start logging on after work hours; and
3. Users leave the organization shortly after the incident.

Data is aggregated from the logs to give a summary of each individual in the organization with respect to the information identified. This is done in daily time intervals as this provides a good balance between time-frame granularity and the computational complexity of training. The approach is outlined in Figure 3.1. With this approach, the aggregated training data will take the form of a daily summary of each employee's activity. As only employees who have used a device are capable of performing this MIT scenario, only these employees have been included in the training data. This reduces the number of employees from 1000 to 266. In addition, in order to reduce training time, only data from the month of July has been added to the set. This month was chosen because it had the highest incidence of MIT. After aggregating data for each employee who was active during the month of July, there is a base training set of 7,260 instances where only 18 are positive threat cases. In order to reduce this set imbalance, a spread sub-sample of negative cases is taken. This reduces the negative-to-positive ratio to only 15-to-1, leaving a training set size of only 288 instances. The elements of this training data are the features selected during the pre-processing phase.

The first trait of scenario one is an abnormality of device usage. The probability of a device being used is calculated for each employee using the information in the device connection logs. Each employee's probability of using a device ($P(D)$) is derived using the Formula in Equation 3.1. U is the number of devices connected associated with a user, and T is the total number of connected events in the log. This probability for each employee is stored in a list. If an employee uses a device on any day, the probability of this happening is recorded as a feature. If there is no device usage, this is recorded as 0.

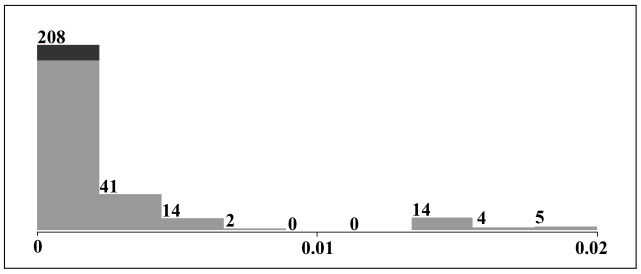


Figure 3.2: Histogram showing the distribution of threat cases (shown in dark grey) among the probabilities of employees using a device

$$P(D) = \frac{U}{T} \qquad Z = \frac{\bar{X} - \mu}{\left(\frac{\sigma}{\sqrt{n}}\right)} \tag{3.1}$$

The distribution of threat cases with regard to the probability of an employee using a device is shown in Figure 3.2.

The second signature of scenario one is an abnormality in logon times. In order to turn this into a feature, a probability distribution is generated for each employee's login times. This is achieved for each employee through the following steps; All of the logon times for each individual employee are compiled into a list. The mean (μ) and standard deviation (σ) and the number of logon time measurements (n) are recorded for each employee. These can then be used with each new logon time on each new day ($\bar{X}$) for each employee to create Z-scores (Z). This equation is shown in Equation 3.1. These Z-scores can be plotted onto a normal distribution curve using a Z-table.

This gives the probability that any employee will log in at any particular time with respect to their personal habits. The distribution of threat cases with regard to the probability of an employee logging on at that time is shown in Figure 3.

The third identified trait of this threat scenario is employees leaving the organization shortly after the incident. Employee records are supplied in the data set. If the employee isn't in the records for any particular month, this means that the employee is no longer employed in the organization. In order to add this as a feature to the training data, when a new day instance

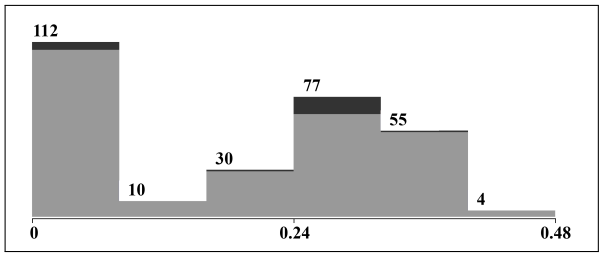


Figure 3.3: Histogram showing the distribution of threat cases (shown in dark grey) among probability of an employee logging on at at particular time

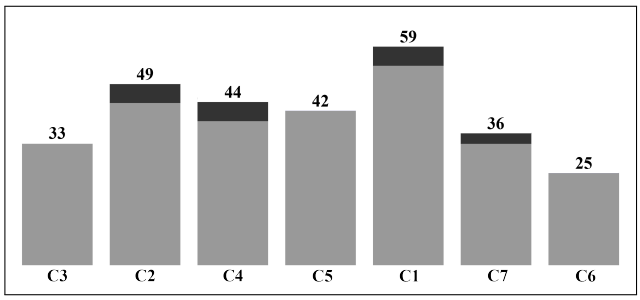


Figure 3.4: Histogram showing the distribution of threat cases (shown in dark grey) among psychometric clusters

is being aggregated, the employee logs for the current month and the next month are searched for that user ID. The positive or negative results of this search are then recorded as features for that user on that day. In the data, all of the positive cases for this scenario were not employed in the next month.

While the psychometric information was not directly included in the threat scenario traits, this is still added as a feature in the training data in order to test for significance. Originally, this takes the form of a vector denoting the employee's score on the 'Big 5 Personality traits' indexes. These vectors were clustered into seven categories using a simple k-means algorithm. The personality cluster for each employee was then recorded into a list to be referenced when generating the training data. This can be observed in Figure 4, true MIT cases only appear in four of the seven psychometric categories.

Finally, labelled threat cases are taken from the answers section of the data set version 4.2. The format of each instance in the set is as follows: I)Employee log on time probability *(continuous)*; II)Device Connect probability *(continuous)*; III)Is the employee employed this month? *(boolean)*; IV)Is the employee employed next month? *(boolean)*; and V)Psychometric cluster of

employee *(categorical)*.

3.3.2 Model Building

In order to build models on this training data, the Weka toolkit was utilized [184]. Using this toolkit, multiple learning algorithms were trained on the data. Each algorithm was then compared against a boosted version of the same algorithm. Boosting is where classifiers are trained iteratively - at each iteration the incorrectly classified instances are amplified in the training data in order to, ideally, improve performance. After boosting, the resultant accuracy and the area under the Receiver Operating Characteristic (ROC) curve were compared to evaluate model performances. These include:

1. Neural Network (NN)
2. Naïve Bayesian Network (NBN)
3. Support Vector Machine (SVM)
4. Random Forest (RF)
5. Decision Tree (DT)
6. Logistic Regression (LR)

The best performers are identified using the ROC and accuracy values shown in tables 3.3 and 3.4. These algorithms are:

1. Neural Network
2. Boosted Naïve Bayesian Network
3. Boosted Support Vector Machine
4. Random Forest
5. Logistic Regression

Figure 3.5: Graph demonstrating performance approximation of models A and B.

[185]

The aforementioned algorithms are aggregated into a single *metalearner* using probability vote. By combining algorithms in this way, the strength of separate models to approximate a greater area under the is leveraged [185]. In Figure 3.5, the shaded area outside of both of the functions is approximated by combining methods A and B. Similarly, the area between the ROCs of the five identified models is approximated. This is achieved by combining the classifiers using probability vote, allocating the vote weight based on the probability of each classifier being correct. This learner missed only correctly classified 14 out of 18 true cases and 262 out of 270 false cases.

In Table 3.2, the performance of the meta-learner is compared against the next best-performing classifier, a boosted naive Bayesian network. In Figure 3.6,the curve of the meta-learner may be observed and in Table 3.1 is the confusion matrix of the classifier.

Table 3.1: Metalearner Confusion Matrix

	Predicted False	Predicted True
Actual False	262	8
Actual True	4	14

3.4 Evaluation

The effectiveness of the work described above will now be critically evaluated on the basis of pre-processing approach, approach to classification and meta-

Table 3.2: Metalearner Performance Comparison

	MetaLearner	Boosted NBN	Difference
Accuracy	96.2%	97.2%	-1%
Area Under ROC	0.988	0.980	0.08%

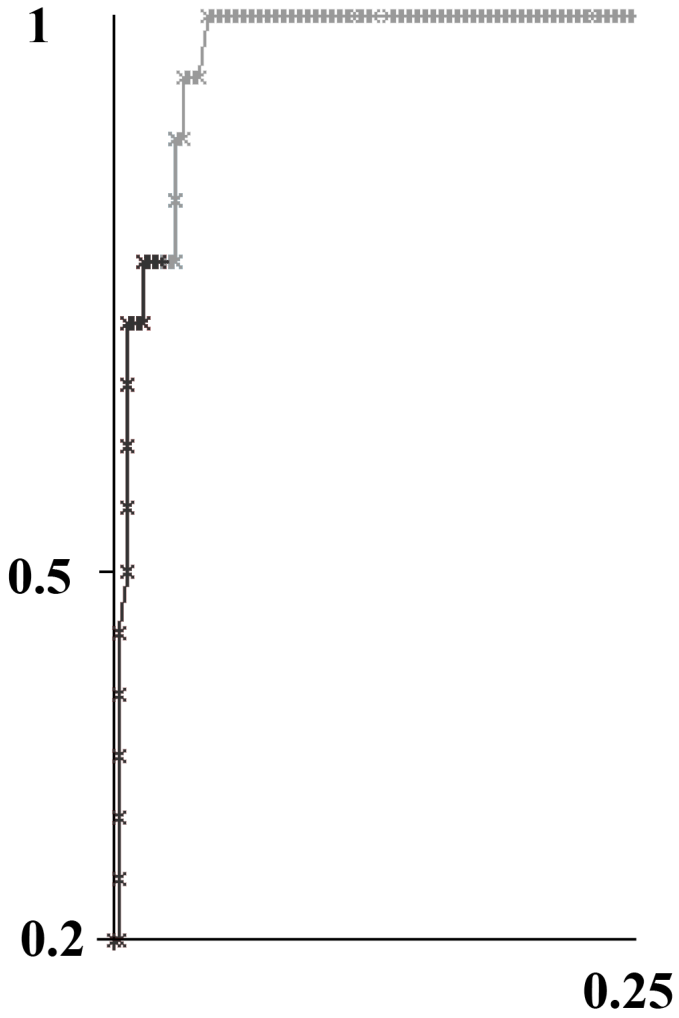


Figure 3.6: ROC curve showing predictive performance of the meta-learner. MIT cases are in dark grey, and non-MIT cases are in light grey.

learner performance. Results are shown in tables 3.3 and 3.4. Models are validated using 10-fold cross-validation [186]. The results of boosting are mixed. Some are improved by the boosting approach, and some do not perform as well. The best-performing classifiers are selected to be aggregated into a heterogeneous stack classifier.

Table 3.3: Classifier Accuracy's before boosting

Classifier	NN	NBN	SVM	RF	DT	LR
Accuracy	95.8%	91.3%	95.4%	97.5%	96.1%	96.5%
Area under ROC	0.974	0.954	0.872	0.982	0.915	0.983

Table 3.4: Classifier Accuracies after boosting

Classifier	NN	NBN	SVM	RF	DT	LR
Accuracy	95.8%	97.2%	97.2%	97.2%	97.2%	96.8%
Area under ROC	0.952	0.980	0.980	0.888	0.932	0.802

3.4.1 Pre-processing

During the course of this work, one archetypal threat scenario was analyzed, and a data pre-processing approach was developed to optimize instance data for predicting this scenario. While the original data set was large and complex, this was significantly distilled in order to create instances out of just five data points. The histograms representing the instance attributes in figures 3.2, 3.3 and 3.4 show a discernible split between the MIT and non-MIT cases. The quality of these features is demonstrated by the performance metrics of the algorithms that are trained upon them. Each classifier that trained upon the data had an accuracy of at least 95%, which is highly accurate. The only common factor present in training the classifiers that produced these performance metrics was the data created during the pre-processing phase.

3.4.2 MIT Category Granularity vs. Workload Trade-off

Despite the quality of training data, the proposed approach addresses only one threat archetype at the present time. This work may be extended by developing alternative models for separate categories. Using this approach, similar pre-processing will have to be performed for each MIT scenario. This represents a greater pre-processing workload than approaches that treat MIT as a single category. By extension, this is also true for any learning algorithms that will be trained. Separate classifiers will have to be trained for separate archetypes.

In order to combat the increase in workload associated with the finer granularity in the MIT category, different levels of archetypal nuance can be established in order to optimize the number of archetypes to the associated pre-processing and training workload. In the training data, there were only three archetypes. This is likely to be different in real-world scenarios. In real data, there is likely to be an entire spectrum of insider threat scenarios. This spectrum, however, could be split into a number of subcategories, where scenarios in each category carry similarities. The more times that MIT has been divided into subcategories.

The more work that will be involved in the pre-processing, the more training data sets will need to be developed, and subsequently, more models will need to be trained on these separate sets. Having said this, on the basis of the results observed in this study, it can be hypothesized that the greater the nuance in MIT category, the more accurate the learner is likely to be.

3.4.3 Metalearner Performance

While the accuracy of prediction is not improved through the aggregation of high-performance models into a probability vote meta-learner, the Area under the ROC is the greatest of all that have been observed. This shows that, with respect to this metric, the meta-learner is greater than each of its constituent classifiers. While the overall accuracy suffers in this approach,

this is only due to the fact that a small number of negative cases have been classified as true. The large area under the ROC shows that more true cases have been identified correctly than any other model. The cost of incorrectly identifying an insider as MIT is the cost of a SOC analyst verifying that this insider has not, in fact, performed one of the threat scenarios. The cost of incorrectly identifying a true insider case as negative is the cost of the data breach. Depending on the sensitivity and quantity of records that are leaked, this could cost millions.

3.5 Discussion

In this work, a new methodology for pre-processing insider threats to optimize classification results based on insider threat categories is established. The resultant data set produced using the established methodology is used to train a series of classifiers which all outperform the predictive performance of previous strategies identified in the research. The most performant of these models is aggregated into a meta-learner algorithm using probability vote. Probability vote was the best-performing vote aggregation method after testing majority vote and weighted vote. This produces a model with a ROC curve containing a greater area underneath than any of the other models that were explored in this work. This indicates the suitability of this approach for improving overall classifier performance.

On the basis of the results identified, this work could be further expanded by tailoring instance data to the other two scenarios present in data set r4.2. A general model could also be developed on this data in order to test the hypothesis that instance data tailored to each scenario creates more performant classifiers than one generalized classifier. In addition, these features could possibly be extended using a genetic algorithm approach, which may produce features of higher quality. Finally, real-life data could be used to train future models relating to red-team simulated scenarios. This would allow the effectiveness of this approach to be tested in the wild, further

validating its applicability to this problem domain.

3.6 Conclusions

The conclusions drawn in this study could be reinforced given access to real data. At present, the meta-classifier generated in this example is only proven to ring true for the synthetic data provided. However, given the introduction of PPML approaches, a federation of real-world data owners could be established without exposing these organisations to the risk of attacks made possible through the exposure of said logistical information.

Furthermore, a larger corpus of data could be established using if multiple data-owning organisations collaborated in a federation aiming to solve the problem of insider threat collaboratively. This would facilitate a marge larger set of positive instances of insider threat than any individual organisation could achieve.

Chapter 4

PyVertical: A Vertically Federated Learning Framework for Multi-headed SplitNN

4.1 Introduction

In this chapter, PyVertical (a vertically federated learning framework for multi-headed SplitNN) is introduced. This framework supports vertically federated learning through the use of SplitNNs. The proposed framework allows a DS to train neural networks on data features vertically partitioned across multiple owners while keeping raw data on an owner's device. To link entities shared across different datasets' partitions, it uses Private Set Intersection on IDs associated with data points. To demonstrate the validity of the proposed framework, the training of a simple dual-headed SplitNN for an MNIST classification task, with data samples vertically distributed across two and a is presented.

With ubiquitous data collection, individuals are constantly generating diverse swathes of data, including location, health, and financial information. These data streams are often collected by separate entities and are sufficient

for high-utility use cases. A common challenge faced by DSs is utilising data isolated in silos to train machine ML algorithms. When this data is commercially sensitive, personal or otherwise under strict legal protection, it cannot be simply merged with data controlled by another party. To ensure data privacy is not compromised during the training or inference process, several privacy-preserving ML techniques, such as FL [187, 188, 189, 190, 38], focus on training ML models on distributed datasets by keeping data in the custody of its corresponding holder. FL typically splits data horizontally. This is where datasets are distributed across multiple owners that have the same features and represent different Data Subjects [191]. However, it is common in real-world scenarios to find datasets which are vertically distributed [192], i.e. different features of the same Data Subject are distributed across multiple DOs. For example, specialists or general hospitals may hold different parts of a patient's medical data.

To address the issue of learning from vertically distributed data, SplitNNs are used to first map data into an abstract, shareable representation. This allows information from multiple sources to be combined for learning without exposing raw data. This is combined with PSI to identify and link data points belonging to the same data samples shared among parties. This process facilitates VFL for non-linear functions.

4.1.1 Choosing Split Neural Networks: Advantages and Justifications

SplitNNs provide numerous advantages over other privacy-enhancing techniques in given use cases. There are, however, many techniques for implementing federated learning. The alternatives to SplitNN include SMC , HE [193] and DP [194]. SplitNN has a marked advantage over methods which involve encrypted computation as SplitNN is able to compute data in plaintext form. This means that the drastic computational overhead involved in computing HE calculations may be avoided. Similarly, SMC requires multi-

ple hosts working in tandem to complete computations. This is not required in SplitNN. DP may be used to shield the original values of data before being aggregated to a DS machine for processing. However, this method forces a choice between the privacy of original values and the accuracy of the DP values post-distortion. This may be particularly problematic in healthcare. While DP may be used in conjunction with processing SplitNNs, this is not a hard requirement so data may be processed privately while not adversely influencing the veracity of data.

4.2 Framework Description

We introduce PyVertical, a framework written in Python for VFL using SplitNNs and PSI. PyVertical is built upon the privacy-preserving deep learning library PySyft [38] to provide security features and mechanisms for model training, such as pointers to data, without exposing private information.

A set of data features are distributed across one or more DOs. A full dataset split vertically across the features is referred to as a *vertical dataset*. Each of the DOs takes part in the model training alongside a DS who orchestrates the process. The DS could also be a DO itself, holding features or data labels. The data features in the vertical datasets may intersect. Each data point is associated with a unique ID based on the data point's subject, the format of which is agreed upon by the DOs (e.g. legal names, email addresses, ID card numbers). The DOs use PSI to agree on a shared set of data IDs (process described in Section 4.2.1); each DO discards non-shared data from their datasets and sorts their datasets by ID, such that element n of each vertical dataset corresponds to the same Data Subject.

In the framework, the DS is able to define a SplitNN model and send the corresponding model segments to the DOs. Each DO's model segment maps their data samples to an abstract representation with k_i neurons. The output from each model segment (which would correspond to a hidden layer of a complete classic neural network) is sent to the DS and concatenated to form

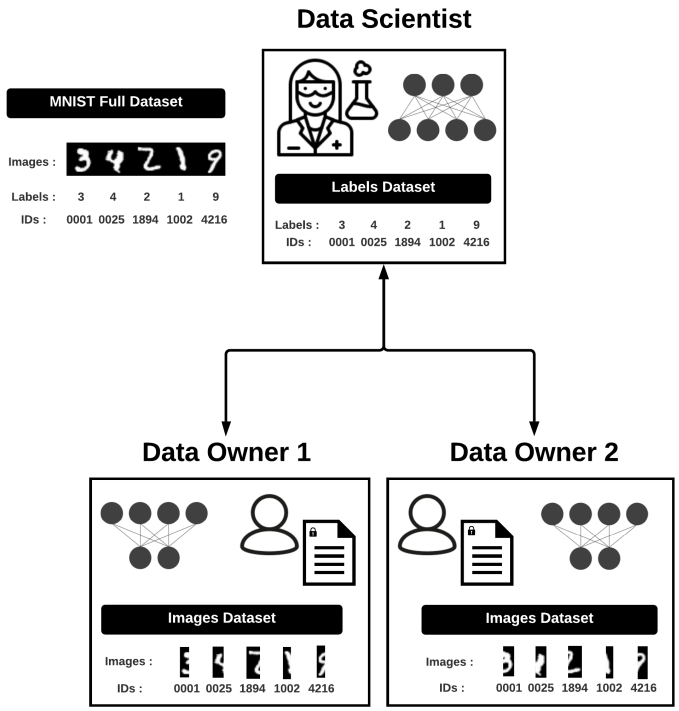


Figure 4.1: Parties and datasets in the conducted experiment. DS holds a part of the SplitNN and the labels dataset. DO hold their images datasets and parts of the SplitNN

a $\sum_i k_i$ length vector. The DS also defines a model segment corresponding to the final part of the SplitNN. This segment remains on the DS's premises and maps the concatenated, intermediate data (i.e. the output from DOs' model segments) into a shape relevant to the task. During model training, the DS calculates batch loss and updates their model segment's weights. The DS then sends the final gradient to the DOs, each of whom updates their own model segment by completing backpropagation independently. It is assumed that all parties are honest-but-curious actors. Figure 4.1 demonstrates model inference under this framework for the experiment outlined in Section 4.5.

4.2.1 Data Resolution Protocol

We use a PSI Python library [195] to identify intersections between data points in two datasets based on unique IDs. In this work, consider a setting where the DS has access to ground truth labels. For all three parties (two DOs + one DS) to agree on data points shared among all datasets, the protocol

works as follows.

To begin with, the DS runs the PSI protocol independently with each DO. The intersection of IDs between the DS and each DO is revealed to the DS. The DS computes the global intersection from the two previously independently computed intersections and communicates the global intersection to the DOs. In this setting, the DOs do not communicate and are not aware of each other's identity in any regard. In practice, this is an ideal feature of the protocol as having knowledge of a group's or individual's participation in a training process can reveal sensitive commercial and personal information in and of itself. Moreover, as the single IDs intersection lists are only revealed to the DS, there is no risk for the DOs to learn which information the other DOs own. Each of the DOs learns only the information necessary for training or inference.

4.3 PyVertical Protocol

Figure 4.2 describes the PyVertical protocol applied to the MNIST dataset for a single DO. The dual-headed PSI data linkage process is presented in Figure 4.3. Note that, in this illustration, there is only one DS; the duplicated icon is just to illustrate in more detail how the DS runs a single PSI with each DO separately and that this could be done in parallel.

4.4 Experimental Setup

The DO model segment maps 392-length input into a 64-length intermediate vector with a ReLU activation, which is an abstract encoding of the data. The DS controls a separate neural network that takes as input a 128-length vector (concatenated DO outputs) and transforms it into a softmax-activated, 10-class vector representing the possible digits in the dataset. The DS's model has a 500-length hidden layer with a ReLU activation. All layers are fully connected. A learning rate of 0.01 is used for the DO models and 0.1 for the

Figure 4.2: VFL proof-of-concept implementation
[37]

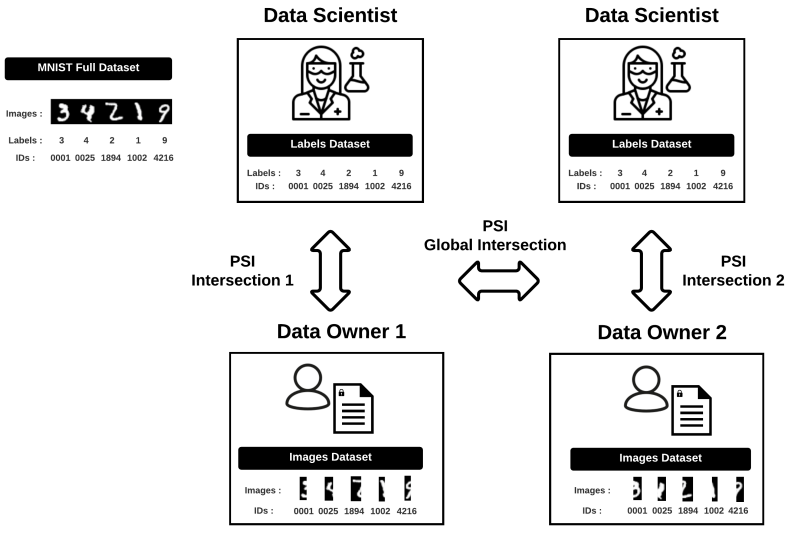


Figure 4.3: i) DS computes the intersection with DO 1. ii) DS computes the intersection with DO 2. iii) DS computes the global intersection.

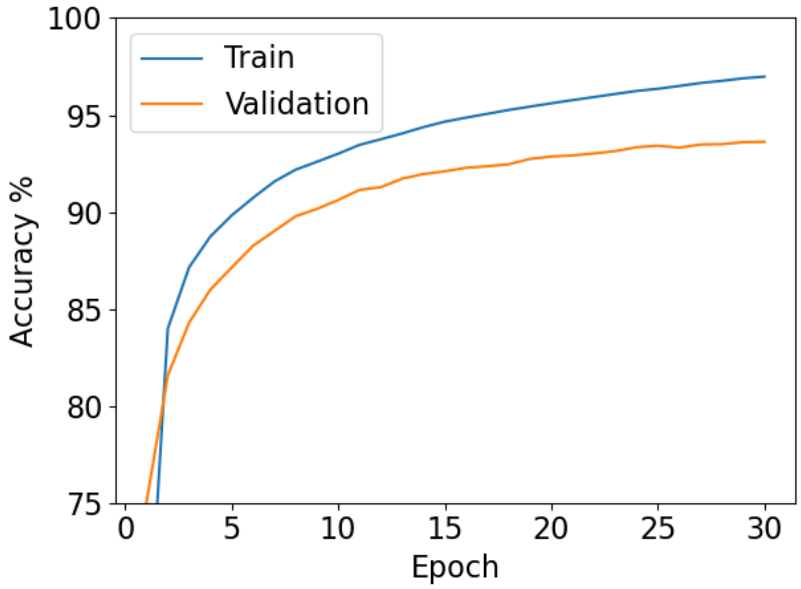


Figure 4.4: Train and validation accuracy for an unoptimised dual-headed SplitNN on vertically-partitioned MNIST.

DS model. Data are grouped into batches of size 128. Only the first 20,000 training images of MNIST are used, and the model is trained for 30 epochs.

4.5 Evaluation

To verify the validity of this framework, a dual-headed SplitNN is trainedw on a vertically-partitioned version of the MNIST dataset. Vertically distributed data is generated by splitting the images in MNIST into left and right halves, providing a dataset of each half to different DOs. The DS defines and sends an identical, multi-layered neural network to each of the DOs that takes 392-length vectors as input (flattened representation of 28x14 pixel images). The DS also defines and keeps on its premises the second part of the neural network, which outputs a softmax layer for classification. The DS can access the ground truth labels and calculate the loss for each data batch. The DS controls the training process and hyperparameters.

The objective of this experiment is to demonstrate that the proposed framework allows vertically partitioned learning. This specific experiment

should be considered a proof-of-concept, thus not highly optimised for the specific task. Nevertheless, the results of the experiment are reported in Figure 4.4.

4.6 Discussion

This work was developed and distributed as an open-source project. It is the hope that PyVertical can serve as a useful tool for researching neural networks-based VFL. PSI was found as an appropriate and useful method for resolving Data Subjects across datasets; many datasets and domains already collect unique IDs, such as usernames or national identifiers for medical data, making this method widely applicable. Finally, a dual-headed model was successfully trained on a vertically-partitioned MNIST dataset, demonstrating that the proposed framework and method work in principle.

The experiment performed in this work assumes that all the parties involved (DOs and DSs) act honestly. To develop a truly scalable, robust VFL system, additional precautions should be taken into account: identity management, validation of adherence to PSI protocol, and a method agreeing on data ID schema, to name a few.

4.7 Conclusions

This work investigates a symmetric SplitNN model: we assume that each DO holds an identical model segment and that data points are split equally between DOs. Future work should investigate the impact of imbalanced vertical datasets [122] and the resulting difficulties from the asymmetric model segment convergence due to the use of different-sized models and learning rates.

Finally, an example of a training process with two DOs and a DS holding labels is given. While the proposed framework can support more parties in

principle, future work may aim to investigate how to apply the process to massively multi-headed VFL tasks. Additionally, this may be extended with a plan to research and integrate other privacy-preserving ML techniques into the workflow, such as decentralised identities [196, 36] and DP [194, 143, 34], to further enhance privacy guarantees.

Chapter 5

Model Inversion Attacks on Split Neural Networks and their Defences

5.1 Introduction

This chapter describes a threat model under which a split network-based FL system is susceptible to a model inversion attack by a malicious computational server. It demonstrates that the attack can be successfully performed with limited knowledge of the data distribution by the attacker. A simple additive noise method is applied to defend against model inversion, finding that the method can significantly reduce attack efficacy at an acceptable accuracy trade-off on MNIST. Furthermore, it is shown that NoPeekNN, an existing defensive method, protects different information from exposure, suggesting that a combined defence is necessary to fully protect private user data [34].

Training deep learning models has typically required the centralised collection of large datasets, which threatens users' privacy by handing control of sensitive data to untrusted third parties. A recently developed method to protect privacy is FL, which allows users to maintain control over their

data by collaboratively training models on their own devices [197, 188, 190]. FL makes it easier to develop models in domains such as medicine, where legal requirements impose constraints on the sharing of personal data [198, 199]. Split learning is a variant of FL in which models are split across parties. Comparatively, split learning reduces the computational cost to the data owner at the expense of increasing data communication between them [135].

However, maintaining control of data during training does not solve all privacy issues. As in any computational system, neural networks are susceptible to various attacks. Recent work has demonstrated that methods to extract data [137, 200, 201] and model parameters [202] from a model, or corrupt the model performance [203] are possible. The large amounts of data required to train deep networks, and their propensity to memorisation, pose a serious privacy risk to users. Several tools for collaborative, private learning are being actively developed [38, 133]. While having the potential to democratise deep learning, those tools could also present new opportunities for bad actors to attack ML models at a large scale. Therefore, it is vital to understand the practical limitations of possible attacks and defences before such systems become widely adopted.

5.2 Threat model

In this thesis, an honest-but-curious computation server is considered and an arbitrary number of data owners who run the correct computations during training and inference. At least one party attempts to steal input data from other parties using a model inversion attack. The attack process is as follows: 1) The attackers collect a dataset of inputs (raw data) and intermediate data produced by the first model segment. 2) They train an attack model to convert the intermediate data back into raw input data. 3) They collect intermediate data produced by some data owners and run it through the trained attack model to reconstruct the raw input data. This attack is considered a “black box" since the internal parameters of the data owner model segment are not

used in the attack. It is assumed that the model training process has been orchestrated by a third party and that there is only one computational server.

This thesis considers the susceptibility of inference-time data to model inversion; it does not investigate the efficacy of the attack on data collected during training. The susceptibility of split learning to other attack types (e.g. membership inference [200], Sybil attacks [204, 205]) is out of the scope of this work. Moreover,"white box" model inversion attacks, which extract training data memorised by the model segments, is not investigated.

In a split learning training scheme, a data owner controls a part of a model which maps input data into an intermediate data representation. A computational server controls a second part of the model, which maps the intermediate data into the desired output. As a single data owner may not have a large enough dataset to sufficiently train a model, multiple data owners may be employed to run federated training. In this scenario, each data owner has a copy of the same model segment. The complete model may be utilised for inference by actors not involved in the training process, in which case each model user must also hold a copy of the model segment.

The attack can be achieved by different parties under different settings:

1. **A computational server colluding with a data owner**. The data owner provides a dataset to train the attack model and the model segment. The computational server has access to intermediate data produced by any data owner on which the attack is run.

2. **A computational server acting alone**. The computational server obtains access to a data owner's model segment through fraud (e.g. acting as a data owner to gain access to the system), coercion or theft and must construct its own dataset.

3. **A data owner acting alone**. The data owner trains an attack model on their own data. The data owner intercepts intermediate data sent to the computational server by another data owner.

It is possible that a computational server stores intermediate data provided to it during training and inference; hence, a successful attack involving the computational server compromises also historical uses of the model, including data owners who contributed to only a single round of training. As the distribution of intermediate data generated during each round of training differs, the attack may not work for early-round training.

5.3 Noise Defence and Experiments

This work introduces a *noise defence* in which additive Laplacian noise is applied to the intermediate data representation on the data owners' side before sending it to the computational server. Noise is drawn from a Laplace distribution parameterized by location μ and scale b each time the model is used, so the data owner's model is no longer a one-to-one function. This obscures the data communicated between model segments and makes it harder for the attacker to learn the mapping from the intermediate representation to the input data. The noise is added to intermediate data only after the model is trained (as an alternative, it could also be applied during training). The noise defence can be applied unilaterally by the data holder. This is useful in settings where a data holder does not trust the computational server.

NoPeekNN is qualitatively compared to the noise defence for defence against model inversion. Quantitatively, the distance correlation between original and reconstructed images is compared, where a more significant correlation implies a better reconstruction. The utility of NoPeekNN and the noise defence introduced in this work, both independently and as combined are also investigated.

Classifiers are trained in combination with NoPeekNN with loss weightings α equal to 0.1, 0.5, 1.0, and the noise defence mechanism with noise scales b equal to 0.1, 0.5, 1.0 ($\mu = 0$ for all). In all experiments, a simple convolutional neural network is trained on MNIST [206]. An an attack model with a similar size and architecture to the classifiers is trained on a dataset

disjoint from any data used during the training phase. Furthermore,it is explored how the number of data points available to train attack models and knowledge of the data structure may impact the attack's success. To validate the impact of prior knowledge, reconstruction of MNIST images using an attack model trained on EMNIST is attempted [5], which is similar to MNIST but instead contains images of handwritten characters (a-z).

5.4 Training Details

The MNIST dataset is pre-split into a training set, containing 60,000 images, and a test set, containing 10,000 images. The first 40,000 images of the training set are used to train the classifier, images 40,000-45,000 to train the attacker and images 45,000-50,000 to validate the attacker. For each experiment, a simple convolutional neural network is trained for 10 epochs with a batch size of 32. Larger batch sizes are limited by the computational complexity of calculating distance correlations. The Adam optimizer is used with a learning rate of 0.001 for both model segments. In the attack setting outlined in this work, attackers have access to the classifier model; therefore, the usage of an attack model with an appropriate architecture for the target model is expected. The attack models are trained with the same hyperparameters. As the attack models worked "out-the-box", a dedicated adversary with unlimited access to a target model could achieve greater attack efficacy.

5.5 In-Training Noise Defence

Applying noise to intermediate data during the training process allows the model to adapt to the randomness and produce models with higher utility. Algorithm 2 describes the training noise defence in conjunction with NoPeekNN.

Algorithm 2 NoPeekNN with Noise Defence

laplacian noise scale b
NoPeekNN weight α
Data owner model f_1 with weights θ_1
Computational server model f_2 with weights θ_2
Learning rates λ_1, λ_2
for $epoch \leftarrow 1, 2, \ldots, N$ **do**
  for $inputs, targets \leftarrow dataset$ **do**
    $intermediate \leftarrow f_1(inputs)$
    $noise \sim L(0, b)$
    $intermediate \leftarrow intermediate + noise$
    $outputs \leftarrow f_2(intermediate)$
    $\theta_1 \leftarrow \theta_1 + \lambda_1 \frac{\partial}{\partial \theta_1} \alpha \mathcal{L}_{dcor}(inputs, intermediate) + \mathcal{L}_{task}(outputs, targets)$
    $\theta_2 \leftarrow \theta_2 + \lambda_2 \frac{\partial}{\partial \theta_2} \mathcal{L}_{task}(outputs, targets)$

5.5.1 Attack Constraints

Figures 5.1, 5.2 and 5.3 plot reconstructions of data extracted from an MNIST classifier by an attack model trained on different numbers of data points. The plots represent the reconstruction of images extracted from a classifier with no defence mechanisms applied. The figures show one example for each class of the MNIST dataset. Columns 1 and 3 are real data points; columns 2 and 4 are reconstructions. At 500 images (Figure 5.2), the class of each data point can be inferred from the reconstruction, except for the examples of classes 4 and 8. However, the intricacies of each specific data point have not been captured in the reconstruction. The reconstructions of the attack model trained on 1250 images (Figure 5.3) contain many of the data points intricacies. In some tasks, merely knowing the class of the data exposes user privacy (commonly the case in many medical tasks, such as cancer detection). In other cases (e.g. facial keypoint detection), it is necessary to reconstruct the intricacies of input data to expose private user information.

However, in this setting, the attackers own and control the model segments. There are therefore no time constraints imposed on the attackers to develop the attack model, as to perform the attack they do not have to interact with a live system, which may expose their actions. Consequently, the

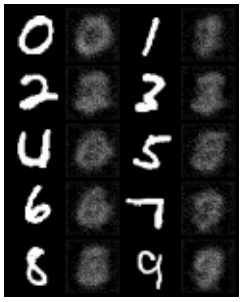


Figure 5.1: Data reconstructions from an attacker trained on 100 images.

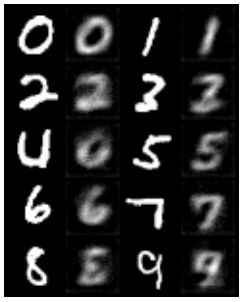


Figure 5.2: Data reconstructions from an attacker trained on 500 images.

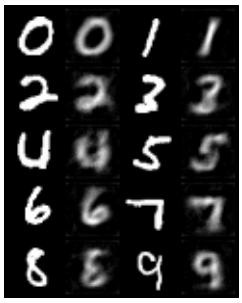


Figure 5.3: Data reconstructions from an attacker trained on 1250 images.

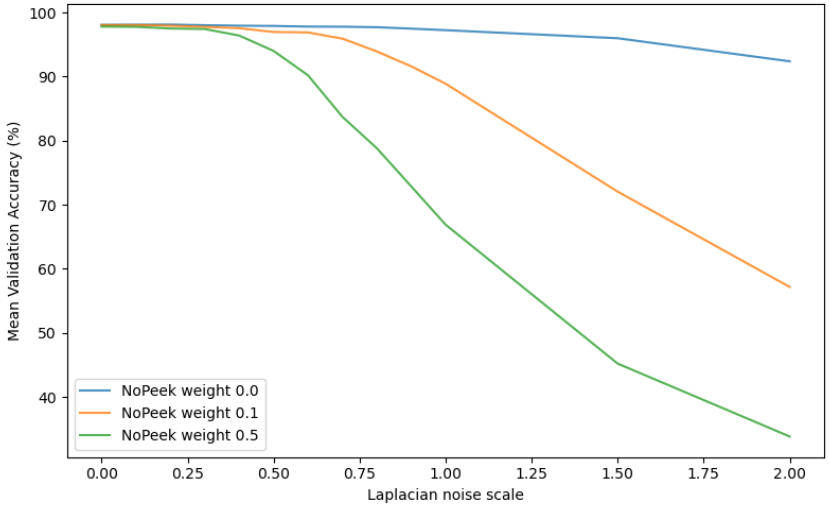


Figure 5.4: Accuracies on a validation dataset by classifiers as a function of the scale of laplacian noise added to intermediate data after training.

poor attack performance achieved at small dataset sizes impedes an attacker but does not stop them.

5.5.2 Defences

Figure 5.4 plots the accuracy of a classifier as a function of the noise scale for noise added after training and NoPeekNN weight. As NoPeekNN weight increases, the trade-off between noise and accuracy increases. This is likely because a greater emphasis on NoPeekNN during training produces tighter intermediate data distributions, so the same amount of noise is more abnormal and therefore destructive to the computational server model segment's capacity to model intermediate data into class probabilities. Even at high NoPeekNN's weighting loss, there is a reasonable accuracy trade-off for noise scales which destroy reconstruction capability (≈ 0.5), which suggests that a hybrid defence would be feasible.

Table 5.1 shows the accuracy of classifiers on a validation dataset and the mean distance correlation between input and intermediate data. A greater NoPeekNN weight typically reduces distance correlation; this is an expected result as NoPeekNN explicitly optimises for distance correlation. Interestingly, there is a correlation between training noise scale and distance corre-

lation, suggesting that additive noise may cancel out some of NoPeekNN's work.

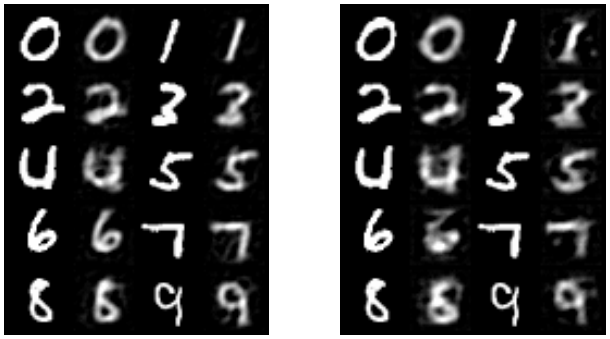
Noise scale	NoPeekNN weight	Accuracy (%)	Distance Correlation
0.0	0.0	97.92 ± 0.00	0.843 ± 0.002
0.0	0.2	97.74 ± 0.00	0.529 ± 0.003
0.0	0.5	97.32 ± 0.00	0.435 ± 0.003
0.1	0.0	97.94 ± 0.039	0.842 ± 0.002
0.1	0.2	97.534 ± 0.047	0.487 ± 0.003
0.1	0.5	97.292 ± 0.037	0.432 ± 0.003
0.2	0.0	97.738 ± 0.050	0.850 ± 0.002
0.2	0.2	97.398 ± 0.086	0.499 ± 0.004
0.2	0.5	97.178 ± 0.086	0.436 ± 0.003
0.5	0.0	97.668 ± 0.076	0.843 ± 0.002
0.5	0.2	97.256 ± 0.075	0.521 ± 0.003
0.5	0.5	97.25 ± 0.094	0.458 ± 0.003

Table 5.1: Validation accuracy and distance correlation between input data and intermediate tensor of classifiers using NoPeekNN and training noise defences. Validation is run 10 times for each model. Every instance of a model which is run is given an independent random seed and the average accuracy and distance correlation is recorded.

5.6 Results

Both the proposed noise defence mechanism and NoPeekNN do not significantly impact the classification model's accuracy. However, NoPeekNN reduces the distance correlation, as expected, as it is explicitly optimised for that.

Figure 5.5(b) plots reconstructions of MNIST images made by an attack model trained on 5,000 EMNIST images. The reconstructions are more blurred and noisy than those made by an attack model trained on MNIST (Figure 5.5(a)), but they are still good enough to infer the class of the original images and carry some intricacies of the data points. This demonstrates that it is sufficient for an attacker to know only vague details about the



(a) (b)

Figure 5.5: (**a**) MNIST data reconstructions from an attacker trained on 5,000 MNIST images. (**b**) MNIST data reconstructions from an attacker trained on 5,000 EMNIST images.

dataset (“greyscale", “simple", “handdrawn"). In the threat model outlined in Section 5.2, the computational server could attack the data owner models without a colluding data owner. One defensive measure is to have proper and considered control of model information during training and inference: if the computational server does not need to know what the model is for, that information should be withheld. Robust identity checks could be carried out on potential data owners to ensure the computational server cannot infiltrate the system, such as the use of cryptographic identifiers proposed in [196, 36].

Figure 5.6 plots model inversion reconstructions of a member of each class in the dataset, applied to classifiers with varying levels of noise (Figure 5.6(a)) or NoPeekNN weighting (Figure 5.6(b)). Qualitatively, the reconstructions are completely destroyed at a noise scale of 1.0 (columns 5 and 10). A noise scale of 0.5 (columns 4 and 9) obscures some of the more pronounced features of the input images while retaining the broad structure. Lower noise scales allow an almost complete reconstruction. Conversely, reconstructions from NoPeekNN adopt a more generic version of the ground-truth class, but the reconstructions are more coherent. For example, slants in the digits are removed (as in classes 1, 3 and 5) as NoPeekNN weight increases. This qualitative analysis suggests that a combination of NoPeekNN and noise defence will produce a more robust defence. Which defensive technique should be prioritised depends on the model’s task. For example, a model

(a) (b)

Figure 5.6: (**a**) Model inversion attack on an MNIST classifier using the *noise defence* mechanism. Left-to-right: true image, reconstructions on models with (0, 0.1, 0.5, 1.0) noise scale. (**b**) Model inversion attack on an MNIST classifier using NoPeekNN defence. Left-to-right: true image, reconstruction on models with (0, 0.1, 0.5) NoPeekNN weighting.

which detects the presence of disease would prioritise class obfuscation ("Disease" or "No Disease") using NoPeekNN. On the other hand, for facial recognition, individual user privacy is more critical, so noise should be applied to prohibit clean reconstructions.

The noise defence introduced in this work provided sufficient privacy/utility trade-offs for the MNIST dataset. While simple, there are numerous tasks with a data complexity similar to MNIST, which may benefit from split learning, such as signature recognition. The privacy/utility trade-off could be improved by post-training the computational server's model segment with noise applied while keeping the data owner's model segment fixed.

However, this approach removes a data owner's unilateral ability to defend against model inversion.

A Laplacian noise distribution was used in this work due to its relation to DP; other noise distributions should have a similar defensive effect and may even provide a better privacy/utility trade-off.

It has been demonstrated that a user's data is susceptible to exposure by an adversary under a SplitNN training setting. This happens even under limited knowledge of the task being solved. Consequently, any split learning framework where the computation server has access to the data owner model could be compromised. A simple method for destroying data reconstructions by additive noise has been introduced and it has been shown that it protects different information about the input data from exposure than NoPeekNN.

5.6.1 Discussion

As the noise defence and NoPeekNN are introduced at the intersection between model segments, they do not protect against white box model inversion, which extracts training data memorised by a data owner's model segment. A training-time differentially private mechanism, such as DP SGD or PATE [207, 208, 209], could offer protection against those attacks. Additionally, we will quantitatively investigate the privacy-utility trade-off, as explored by [210]. Furthermore, the compatibility of defensive measures to protect against several possible attacks should be investigated. In that regard, the noise defence may be formulated as a local DP mechanism on the input dataset to the computational server's model part, which may confer additional defensive capability against other attacks, for example, membership inference. The more complex the dataset, the more difficult for an attack model to learn the data reconstruction mapping. The effectiveness of the noise defence in combination with NoPeekNN should be explored on more complex data sources.

5.7 Conclusion

The work presented above shows the application of noise and distance correlation maximisation techniques as a method to protect activation signals passed between hosts from membership inference attacks.

This work only considered the defensive utility of additive noise to model inversion attacks on the intermediate data. However, it could be considered a local DP process, which confers some protection to the computational server model segment against model inversion and membership inference attacks, among others. Additionally, this work only investigates the efficacy of model inversion attacks on a 2-dimensional image dataset, MNIST. MNIST is a very simple, low-resolution greyscale dataset; therefore data reconstruction is relatively easy to perform. Future work should investigate the utility of the noise defence, in combination with NoPeekNN, on more complex datasets.

Chapter 6

A Distributed Trust Framework for Privacy Preserving Machine Learning

6.1 Introduction

This chapter introduces the ToIP stack as a mechanism for providing peer-to-peer trust and access control when running distributed PPML infrastructures. This is achieved through the utilisation of the ToIP stack [36].

When training a machine learning model, it is standard procedure for the researcher to have full knowledge of both the data and the model. However, this engenders a lack of trust between DOs and DSs. DOs are justifiably reluctant to relinquish control of private information to third parties. Privacy-preserving techniques distribute computation in order to ensure that data remains in the control of the owner while learning takes place. However, architectures distributed amongst multiple agents introduce an entirely new set of security and trust complications, including data poisoning and model theft. This chapter outlines a distributed infrastructure which can be used to facilitate peer-to-peer trust between entities, collaboratively performing a privacy-preserving workflow. The outlined prototype enables the initialisation of industry gatekeepers and governance bodies as credential issuers

under a certain application domain. Before participating in the distributed learning workflow, malicious actors must first negotiate valid credentials from these gatekeepers. It details a proof of concept using Hyperledger Aries, DIDs and VCs to establish a distributed trust architecture during a PPML experiment. Specifically, secure and authenticated DID communication channels are utilised in order to facilitate a federated learning workflow related to mental health care data.

ML is a powerful tool for extrapolating knowledge from complex data sets. However, it can also represent several security risks concerning the data involved and how that model will be deployed [211]. An organisation providing ML capabilities needs data to train, test and validate its algorithm. However, DOs tend to be wary of sharing data with third-party processors [212]. This is due to the fact that once data is supplied, it is almost impossible to ensure that it will be used solely for the purposes which were originally intended. This lack of trust between DOs and processors is currently an impediment to the advances which can be achieved through the utilisation of big data techniques. This is particularly evident with private medical data, where competent clinical decision support systems can augment clinician-to-patient time efficiencies [8, 213]. In order to overcome this obstacle, new distributed and privacy-preserving ML infrastructures have been developed where the data no longer needs to be shared or even known to the Researcher in order to be learned upon [72].

In a distributed environment of agents, establishing trust between these agents is crucial. Privacy-preserving methodologies are only successful if all parties participate in earnest. If a malicious is introduced, they may send a Trojan model which, instead of training, could store a carbon copy of the private data. Conversely, if a malicious actor is introduced in place of a DO, they may be able to steal a copy of the model or poison it with bad data. In cases of model poisoning, malicious data is used to train a model in order to introduce a bias which supports some malicious motive. Once

poisoned, maliciously trained models can be challenging to detect. The bias introduced by the malicious data has already been diffused into the model parameters. Once this has occurred, it is a non-trivial task to de-parallelise this information.

If one cannot ensure trust between agents participating in a Federated Learning (FL) workflow, it opens the workflow up to malicious agents who may subvert its integrity through the exploitation of resources such as the data or the ML model used. In this chapter, it is shown how recent advances in digital identity technology can be utilised to define a trust framework for specific application domains. This is applied to FL in a healthcare scenario and reduces the risk of malicious agents subverting the FL ML workflow. Specifically, the paper leverages: DIDs [158]; VCs [166]; and DID Communication, [161]. Together, these allow entities to establish a secure, asynchronous digital connection between themselves. Trust is established across these connections through the mutual authentication of digitally signed attestations from trusted entities. The authentication mechanisms in this paper can be applied to any data collection, data processing or regulatory workflow and are not limited to solely the healthcare domain.

6.2 Implementation Overview

Our work performs a basic FL example between Hyperledger Aries agents to validate whether distributed ML could take place over the DIDComm transport protocol. A number of Docker containers representing entities in a healthcare trust model were developed, creating a simple ecosystem of learning participants and trust providers (Figure 6.1). The technical specifications of the system are as follows: 2.0GHZ dual-core Intel Core i5 CPU, with 8GB RAM and 256GB SSD. For each Docker container running a Hyperledger Aries agent,the open-source Python Aries Cloud Agent developed by the Government of British Columbia is used [214]. Hospital containers are initialised with the private data that is used to train the model.

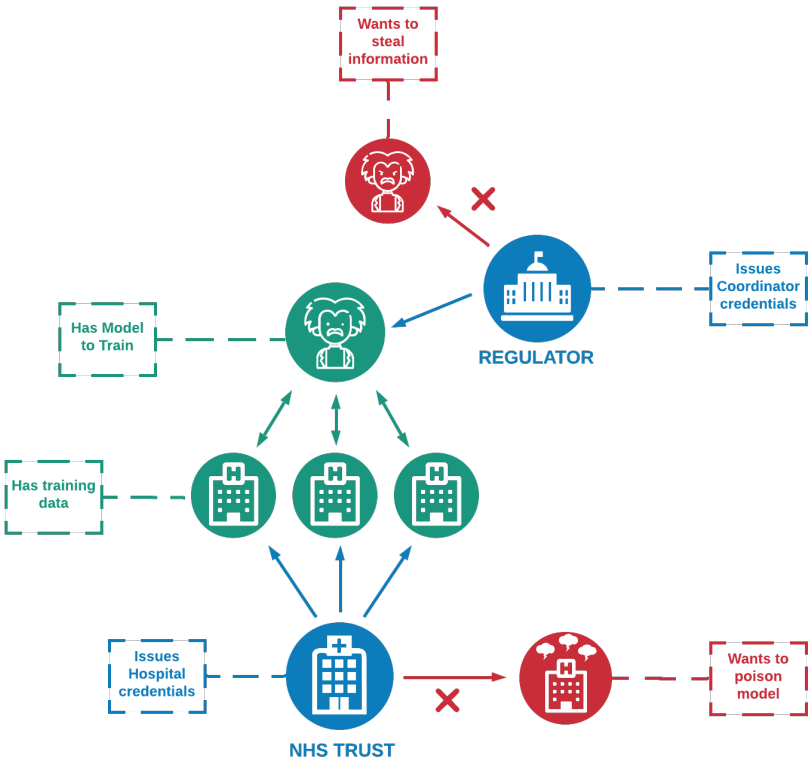


Figure 6.1: ML Healthcare Trust Model

6.2.1 Establishing Trust

We define a domain-specific trust architecture using VCs issued by trusted parties for a healthcare use case. This includes the following agent types: a) NHS Trust (Hospital Credential Issuer); b) Regulator (Researcher Credential Issuer); c) Hospital (DO); and d) Researcher (DS).

This is used to facilitate the authorisation of DOs (verifiable Hospitals) and DSs. A DS who would like to train a model is given credentials by an industry watchdog, who in a real-world scenario could audit the model and research purpose. In the United Kingdom, for example, the Health Research Authority is well placed to fulfil this role. Meanwhile, Hospitals in possession of private health data are issued with credentials by an NHS authority enabling them to prove they are real hospitals. The credential schema and the DIDs of credential Issuers are written to a public ledger — the development ledger provided by the government of British Columbia is

used [215] for this work.

The system is established following the steps described in Algorithm 3. Once both the DS and the *Hospital* agents have been authenticated, the communication of the model parameters for FL can take place across this secure, trusted channel.

Algorithm 3 Establishing Trusted Connections

DS agent exchanges DIDs with the *Regulator* agent to establish a DIDComm channel.
Regulator offers an *Audited DS* credential over this channel.
DS accepts and stores the credential in their wallet.
for each *Hospital* agent **do**
    Initiate DID Exchange with *NHS Trust* agent to establish DIDComm channel.
    NHS Trust offers *Verified Hospital* credentials over DIDComm.
    Hospital accepts and stores the credential.
for each *Hospital* agent **do**
    Hospital initiates DID Exchange with *DS* to establish DIDComm channel.
    DS requests proof of *Verified Hospital* credential issued and signed by the *NHS Trust*.
    Hospital generates a valid proof from their *Verified Hospital* credential and responds to the *DS*.
    DS verifies the proof by first checking the DID against the known DID they have stored for the *NHS Trust*, then *resolving* the DID to locate the keys and verify the signature.
    if *Hospital* can prove they have a valid *Verified Hospital* credential **then**
        DS adds the connection identifier to their list of *Trusted Connections*.
    Hospital requests proof of *Audited DS* credential from the *DS*.
    DS uses *Audited DS* credential to generate a valid proof and responds.
    Hospital verifies the proof by checking the signature and DID of the Issuer.
    if *DS* produces a valid proof of *Audited DS* **then**
        Hospital saves the connection identifier as a trusted connection.

6.2.2 Vanilla Federated Learning

This chapter implements Federated Learning in its most basic form; where plain-text models are moved sequentially between agents. The DS entity begins with a model and a data set to validate the initial performance. The

model is trained using sample public mental health data which is pre-processed into useable training data. It is the intention of this work to demonstrate that privacy-preserving ML workflows can be facilitated using this trust framework. Thus, the content of learning is not the focus of this work. We also provide performance results relating to the accuracy and resource requirements of the system. the chosen workflow is referred to as Vanilla FL, and this is seen in Algorithm 4. In order to implement Vanilla FL, the original data set was split into four partitions, three training sets and one validation set.

This amalgamation of Aries and FL allow some of the existing limitations caused by a lack of trust among training participants to be mitigated . Specifically, these were: 1) Malicious data being provided by a false Hospital to spoil model accuracy on future cases, and 2) Malicious models being sent to Hospitals to later compromise them to leak information around training data values.

Algorithm 4 Vanilla Federated Learning

1: *DS* has validation data and a *model*, *Hospitals* have *training data*.
2: **while** *Hospitals* have unseen *training data* **do**
3: *DS* benchmarks *model* performance against *validation data* and sends *model* to the next *Hospital*.
4: This *Hospital* trains the *model* with their data and then sends the resulting *model* back to the *DS*.
5: *DS* benchmarks the final *model* against *validation* data.

6.3 Threat Model

Since no data changes hands, FL is more private than traditional, centralised ML [120, 216]. However, some issues still exist with this approach. Vanilla FL is vulnerable to model stealing by ML data contributors who can store a copy of the Researcher's model after training it. In cases where the model represents private intellectual property (IP), this setup is not ideal. On the other hand, with the knowledge of the model before and after training on

each private data set, the DS could infer the data used to train the model at each iteration [200]. Model inversion attacks [217, 137] are also possible where given carefully crafted input features and an infinite number of queries to the model, the DS could reverse engineer training values.

Vanilla FL is also potentially vulnerable to model poisoning and Trojan-backdoor attacks [118, 218, 219]. If DOs are malicious, it is possible to replace the original model with a malicious one and then send it to the DS. This malicious model could contain some backdoors, where the model will behave normally and react maliciously only to given trigger inputs. Unlike data poisoning attacks, model poisoning attacks remain hidden. They are more successful and easier to execute. Even if only one participant is malicious, the model's output will behave maliciously according to the injected poison. For the attacker to succeed, there is no need to access the training of the model; it is enough to retrain the original model with the new poisoned data.

For the mitigation of the attacks mentioned above, this system implements a domain-specific trust framework using VCs. In this way, only verified participants get issued with credentials that they use to prove they are a trusted member of the learning process to the other entity across a secure channel. This framework does not prevent the types of attacks discussed from occurring, but by modelling trust, it does reduce the risk that they will happen. Malicious entities could thus be checked on registration or removed from the trust infrastructure on bad behaviour.

Another threat to consider is the possibility of the agent servers getting compromised. Either the trusted Issuers could get compromised, and issue credentials to entities that are malicious, or entities with valid credentials within the system could become corrupted. Both scenarios lead to a malicious participant having control of a valid, verifiable credential for the system. This type of attack is a threat; however, it is outside the scope of this work. Standard cybersecurity procedures should be in place within these systems that make successful security breaches unlikely. OWASP provides guidelines and secure practices to mitigate these *traditional* cybersecurity threats [220].

The defensive mechanisms are not limited to these and can be expanded using Intrusion Detection and Prevention Systems (IDPS) [221].

6.4 Discussion

To evaluate the prototype, malicious agents were created to attempt to take part in the ML process by connecting to one of the trusted Hyperledger Aries agents. Any agent without the appropriate credentials, either a Verified Hospital or Audited DS credential, was unable to form authenticated channels with the trusted parties (Figure 6.1). These connections and requests to initiate learning or contribute to training the model were rejected. Unauthorised entities were able to create self-signed credentials, but these credentials were rejected. This is because they had not been signed by an authorised and trusted authority whose DID was known by the entity requesting the proof.

The mechanism of using credentials to form mutually verifiable connections proves useful for ensuring only trusted entities can participate in a distributed ML environment. It is noted that this method is generic and can be adapted to the needs of any domain and context. VCs enable ecosystems to specify meaning in a way that digital agents participating within that ecosystem can understand. These are increasingly used increasingly to define flexible, domain-specific trust. The scenario created in this work was used to highlight the potential of this combination. For these trust architectures to fit their intended ecosystems equitably, it is imperative to involve all key stakeholders in their design.

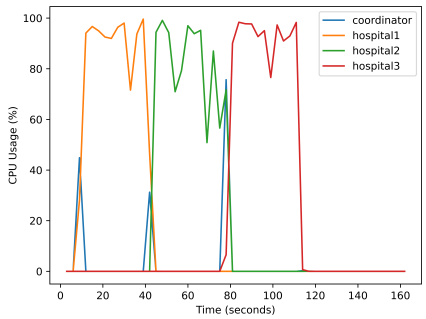
This work is focused on the application of a DID-based infrastructure in a Federated Learning scenario. It is assumed that there is a pre-defined, governance-oriented trust model implemented such that key stakeholders have a DID written to an integrity-assured ledger. The discovery of appropriate glsDIDs, and willing participants, either valid Researchers-Coordinators or Hospitals, related to a specific ecosystem is out-of-scope of the thesis. This chapter focuses on exploring how peer DID connections, once formed,

facilitate participation in the established ecosystem. A further system can be developed for the secure distribution of the DIDs between the agents that are willing to participate.

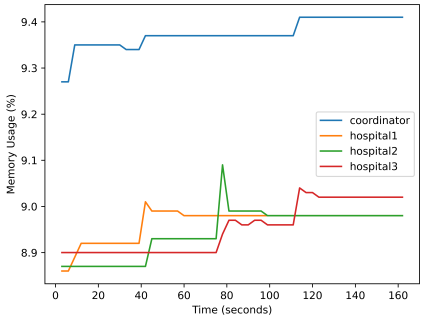
Furthermore, performance metrics for each host were recorded during the running of the workflow. In Figure 6.2 a),the CPU usage of each agent involved in the learning workflow may be observed. The CPU usage of the DS raises each time it sends the model to the Hospitals, and CPU usage of the Hospitals raises when they train the model with their private data. This result is consistent with what is expected given Algorithm 4 runs successfully. The memory and network bandwidths follow a similar pattern, as can be seen in Figure 6.2 b), Figure 6.2 c) and Figure 6.2 d). The main difference is that since the DS is averaging and validating each model against the training dataset every time, in turn, the memory and network bandwidth raises over time. From these results, it may be concluded that running federated learning in this way is compute-heavy on the side of the Hospitals but more bandwidth and slightly more memory intensive on the side of the DS.

The aim of this research is to demonstrate that a decentralised trust framework could be used to perform a privacy-preserving workflow. In this work a dummy model was trained on some basic example data. The intention here is merely to demonstrate that this is possible using the trust framework described. We give the confusion matrix of the model tested on the DS's validation data after each federated training batch. This demonstrates that the model was successfully adjusted at each stage of training upon the federated mental health dataset. The model develops a bias toward false positives and tends to get fewer true negatives as each batch continues. However, this may be due to the distribution of each data batch. Other than this, the learning over each batch tends to maximise true positives. This can be observed in Table 6.1.

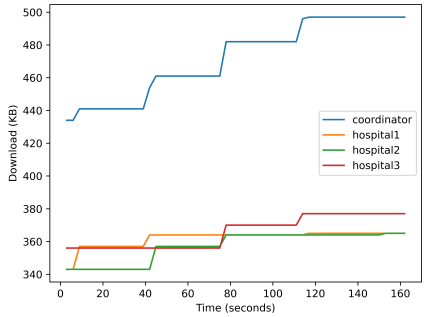
This chapter combines two fields of research, privacy-preserving ML and decentralised identity. Both have similar visions for a more trusted citizen-



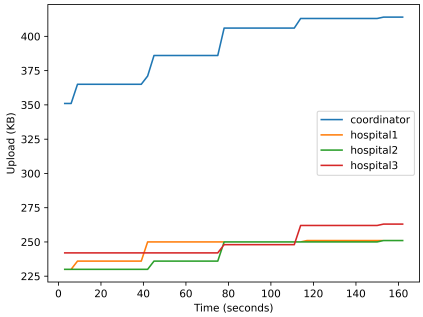
(a) CPU Usage (%) during workflow



(b) Memory Usage (%) during workflow



(c) Network Input (kB) during workflow



(d) Network Output (kB) during workflow

Figure 6.2: CPU, Memory Usage and Network Utilization of Docker container Agents during workflow

Table 6.1: Classifier's Accuracy Over Batches

Batch	**0**	**1**	**2**	**3**
True Positives	0	109	120	134
False Positives	0	30	37	41
True Negatives	114	84	77	73
False Negatives	144	35	24	10

focused and privacy-respecting society. In this research, it is shown how developing a trust framework based on DIDs and VCs for ML scenarios that involve sensitive data can enable increased trust between parties while also reducing the liability of organisations with data.

It is possible to use these secure channels to obtain a digitally signed contract for training or to manage pointer communications on remote data. While Vanilla FL is vulnerable to attacks as described in Section 6.3, the

purpose of this work was to develop a proof of concept showing that domain-specific trust can be achieved over the same communication channels used for distributed ML. Future work includes integrating the Aries communication protocols, which enables the trust model demonstrated here, into an existing framework for facilitating distributed learning, such as PyGrid, the networking component of OpenMined [72]. This will allow others to apply the trust framework to a far wider range of privacy-preserving workflows. Additionally it will allow trust to be enforced, mitigating model inversion attacks using differentially private training mechanisms [222]. Multiple techniques can be implemented for training a differentially private model; such as PyVacy [223] and LATENT [224]. To minimise the threat of model stealing and training data inference, SMC [225] can be leveraged to split data and model parameters into shares. SMC allows both gradients and parameters to be computed and updated in a decentralised fashion while encrypted. In this case, custody of each data item is split into shares to be held by relevant participating entities.

In the experiments, the Hyperledger Aries messaging functionality is utilised to convert the ML model into text and to be able to send it to the participating entities. Future work will focus on expanding the messaging functionality with a separate structure for ML communication. Furthermore, this may be extended to evaluate the type of trust that can be placed in these messages, exploring the Message Trust Context object suggested in a Hyperledger Aries RFC [226].

6.5 Conclusions

In this chapter, the issue of trust within the data industry is addressed. This radically decentralised trust infrastructure allows individuals to organise themselves and collaboratively learn from one another without any central authority figure. This breaks new ground by combining PPML techniques with a decentralised trust architecture.

Chapter 7

A Universal Framework for Structured Transparency

7.1 Introduction

This chapter outlines HE and SplitNN hybrid model (Syft 0.5). This allows for activation signals to be completely protected from membership inference at model inference time by computing the latter portion of the model on entirely HE activation signals. This enforces greater privacy preservation than the techniques outlined in Chapter 5.

Syft is a general-purpose framework that combines a core group of privacy-enhancing technologies which facilitate a universal set of structured transparency systems. This framework is demonstrated through the design and implementation of a novel privacy-preserving inference information flow where HE activation signals are passed through a SplitNN at inference time. It is shown that splitting the model further up the computation chain significantly reduces the computation time of inference and the payload size of activation signals at the cost of model secrecy. The proposed flow is evaluated with respect to its provision of the core structural transparency principles.

7.2 Framework Description

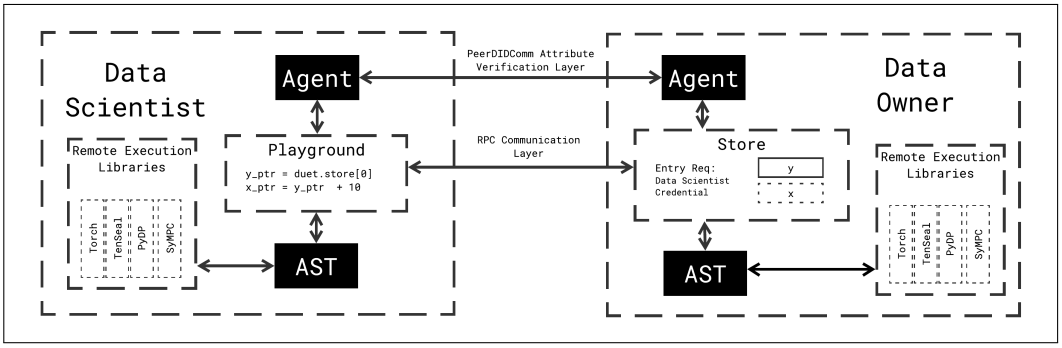


Figure 7.1: Duet Architecture Diagram

Syft allows DSs to perform compute data they don't own and cannot see. This is achieved through the separation of concerns between where data resides and where code is executed. Syft allows for remote computation on any object or library that has been mapped to the Abstract Syntax Tree (AST). From a user perspective, remote variables are operated locally as network pointers. Operations executed on these network pointers are encapsulated as a remote procedure call performed on the remote machine. Agents can be referenced to resolve the verifiable attributes of credentials or data and models in the store without exposing a persistent identity- facilitating zero-knowledge access management as outlined in [103].

After a WebRTC connection is established between peers, privacy is guaranteed through the remote execution model in which you compute operations on data without revealing it or the underlying permission scheme. Even if data is shared, it can still be encrypted using the secure multi-party computation solution, SyMPC, or HE through TenSEAL. Intermediary computation and results are held in a store controlled by the DO. The DO is able to approve or deny requests to access the data in the store by either manual approval (by hand) or automatic approval via a policy.

As an API design, Syft aims to provide a nearly transparent layer around supported frameworks and libraries so that DSs don't need to learn new

tools to develop their algorithms in a privacy-friendly fashion. Users should expect their existing code to run with minimal change and find writing new code within Syft intuitive. Syft significantly lowers the entry-level of access to cryptography, differential privacy and distributed computing for researchers and practitioners.

7.3 Structured Transparency

ST is an extension of the contextual integrity framework ([17]). Contextual integrity views privacy in terms of information flowing between actors in roles governed by context-relative informational norms. Nissembaum's central thesis is that people feel their privacy is violated when these informational norms are broken, for example when unanticipated information is learnt by unexpected actors ([17]). The framework leaves room for positive changes to these informational norms, suggesting that if entrenched contextual integrity is breached by new systems then they should be evaluated against whether these changes contribute to the purpose and values of the context. Contextual integrity emphasises the importance of privacy while challenging older definitions of privacy as either control or access to information [227, 228]. Instead, privacy is presented as the appropriate flow of information relative to a context, hence including control and access within a broader social framing ([16, 17]).

ST extends contextual integrity's definition of an information flow by introducing a more nuanced set of properties of the flow. These properties take into account new possibilities for structuring the context and relative informational norms around an information flow. PETs implemented in Syft and presented in this thesis, explicitly seek to facilitate these information flows. Information flows from a sender to a recipient, and the information pertains to a Data Subject, who could possibly not be the sender. The context within which this information flow is embedded prescribes the roles, activities, norms and purposes for which information flows take meaning and are

justified against. ST proposes additional definitions such as for input and output privacy, input and output verification and flow governance ([7]).

We seek to structure the transparency of an information flow by combining novel PETs for computation [229], encryption ([76, 230]) and authentication ([231, 232]). Such techniques can give confidence to the sender, recipients and subjects of an information flow that the information is integrity assured, verifiably from a sender and only visible to the set of actors and roles defined in the flow structure. Specifically, concerning machine learning use cases, the information contributed to a flow shouldn't be revealed or leaked at any step, only derivatives of this information aggregated to extract features without compromising the underlying information.

7.3.1 Architecture Concepts

Data Owner / Data Scientist - The DO is the Duet session creator, the party that has implicit control over the store and permissions. The DS is the general terminology for the peer node which connects to a DO and has tightly controlled limitations on access to objects in the store. There are multi-party configurations of peers which provide more complex scenarios that won't be detailed here.

Node & Client - In Syft the smallest unit of secure communication and computation is a Node. These Nodes are capable of running a variety of services. By registering new message types and code that executes on receipt of those messages; services augment the capability of a Node with additional functionality. To ensure validity, all messages are signed by their sender and include a verification key. Signing is done with a PyNaCl a Python Libsodium wrapper using 256-bit Ed25519. When a message arrives, it is verified, and then the delivery address is checked. Nodes are designed to be aware of other Nodes and are able to forward messages they receive to other known addresses. Once a message arrives at its destination, it is delivered to the appropriate service for execution.

Nodes have the concept of a root verification key and all services can

opt to allow execution by messages signed by root clients only. In the event of an error, exceptions can be transmitted back to the sender, defaulting to UnknownPrivateException and allowing for careful exposure of real exceptions when safe and relevant. Nodes can optionally contain a key-value store, which can be backed with memory or disk.

To provide a public API interface for users to communicate with a Node, a Client which provides a handle to easily invoke remote functions is used to view metadata about a remote Node. While there are two Nodes in the Duet Architecture (Figure 7.1), there is asymmetry with the DO side having a Store and both Clients having a handle to the DO Node. In this sense, all transfer of data is handled by requests to the DO node and approvals by the DO root client to those requests. Additionally the DO's Client never explicitly sends data or execution requests to the DS Node. This configuration provides a streamlined workflow in scenarios where one side has private data they wish to host.

While this is one possible topology, due to the flexible design, other architectures of two or more parties are possible over any size or shape of the network. By simply adding additional services to a Node, entirely new networked functionality can be constructed. This mechanism is how PyGrid is able to augment Syft to create Cloud and on-prem worker distribution and data storage.

The Store - Duet shares data through the serialization of objects as Protobuf messages. These messages are wrapped in a Storable Object (SO) interface that handles metadata and interaction with a Store. The Store may be located either in memory or on disk. In Duet, the Store is located within the Python process of the DO. However, because all the store interaction is done through serialized communication, nothing prevents the Store from existing on a completely separate process or network node. The Store hosts all the intermediate data created in the process of remote execution. Networked Pointers require the concept of a Distributed Garbage Collection (DGC) mechanism, which is responsible for tracking and freeing objects from the

Store when certain conditions are met.

Abstract Syntax Tree - The Abstract Syntax Tree (AST) is responsible for resolving the correct remote procedure call paths. The AST is a tree that maps any Python module or object to its desired remote behaviour, handling chained pointer generation and message creation on pointer method invocation. This one-to-one mirror with the existing Python module system makes it trivial to support nearly any third-party Python library with Syft.

A node in the AST can represent anything that a normal Python package can provide. Each node contains references to the actual module, class or callable object and, at the leaves of this tree, all attributes have a return type which determines the synthesised pointer class that is generated when they are invoked remotely. When the caller invokes a remote node via either a handle to the remote systems AST tree or an existing network pointer, a new network pointer is immediately created of the expected return type. A message is then dispatched to the remote system where the AST is used to locate the original reference and execute it; placing the real result into the Store. Under this model, only functionality within Python which explicitly allows listed and loaded during startup or later via an explicit loading command can be executed remotely.

Communication Protocol - Under the Duet architecture, Syft's network code only supports peer-to-peer connections between two nodes. To connect two peers, by default, Duet initialises an outgoing connection to a STUN server operated by OpenMined. While OpenMined is the default STUN server, this server is open source and an instance may be operated by anyone. This allows for the users of this system to run the infrastructure entirely independently. STUN is a technology commonly used by video conferencing software which is designed to allow applications to traverse NAT and firewalls. It works by establishing an outgoing connection first to allow any subsequent external traffic on the same port to be routed to the application which made the outgoing request. The STUN server brokers both connections by allowing each side to establish an outgoing connection first.

Once paired, the individual peers establish a connection to each other using WebRTC. This connection uses the DataChannel API from WebRTC over UDP, using DTLS as an integrity mechanism. After the connection, no further traffic is sent to the STUN server. Additionally, the source code and instructions to run a copy of this service is available, and connection to a self-run STUN service only requires adding a single URL parameter to the Duet initialisation function.

Pointers - Pointers are dynamic objects created by the AST to provide proxy control of remote objects. Pointers are usually generated from the known return type of a computation. Pointers map the functionality of a remote object to a local object by wrapping and handling the communication layer. Pointers are also responsible for garbage-collecting objects which are no longer reachable. The DS can create remote objects on the DO's side through messages that are sent to the DO. The implementation of the garbage collection (GC) process relies heavily on the underlying Python GC implementation. When the local pointer, which resides on the DS side, goes out of scope, a special Protobuf message is sent to the store. The sole purpose of this message is to remove the actual object that the DS pointer was referring to. This mechanism assumes that an object created by a DS would not be referred by another user, but it could easily be extended to use a reference counting mechanism where multiple nodes can hold a pointer to the same object.

7.3.2 Libraries

Syft becomes truly powerful when remote computation is extended with the functionality provided by other high-performance computing libraries or privacy-enhancing frameworks. These tools can be aggregated to support the core components of structurally transparent information flows.

PyTorch - is the initial support of high-performance computing on tensors, and the majority of the PyTorch API can be used in Syft. An important note is that the current architecture is not dependent on PyTorch, and the roadmap

includes support for other tensor libraries like Numpy, Tensorflow, JAX and scikit-learn.

Opacus - enables PyTorch models to be trained with respect to privacy with minimal changes in the codebase and infrastructure. It can be used to train PyTorch models ([233]).

TenSEAL - a library that enables machine learning frameworks to work on encrypted data, using HE schemes implemented in SEALCrypto[234][235].

SyMPC - SyMPC is a relatively young secure multi-party computation library developed in-house. It can not be used as a standalone piece of software since it highly relies on the communication primitives that can be found in Syft. Because of this, you would need to install SyMPC alongside Syft if you want to use any of the implemented functionalities. Since SyMPC is still at the beginning of the journey, it has some basic arithmetic operations between secretly shared values and secret/public values. For computing the correct result for simple multiplication and matrix multiplication there are employed some triples (presented in [236]. These primitives are generated by a Trusted Third Party and in this case, it is presumed to be the node that orchestrates the entire computation. A part of the design decisions and implementation details are taken from the Facebook Research Project CrypTen [237]. SyMPC aims to offer the possibility to train/run inference on opaque data using a range of different protocols depending on each DS's use case. As an implicit goal, the library should provide a simple way to implement new protocols, regardless of the underlying ML framework that you use.

AriesExchanger - Aries agents facilitate the construction, storage, presentation and verification of credentials, which may be used anywhere in the Hyperledger ecosystem. At its core, the AriesExchanger allows actors in an information flow (Sender or recipient) to verify attributes about the other based on attestations made by trusted authorities, determined by the context. Aries agents facilitate the zero-knowledge presentation of group membership as described by [232]. The Aries agent interface is accessed through

the AriesExchanger class. From a Syft perspective, AriesExchanger allows DOs to only initiate connections with DSs who have credentials to work with their data as described in ([103]). Similarly, DSs can verify whether remote datasets held by DOs comply with certain scheme requirements as attested to by the appropriate authority. Governance systems are defined and then implemented through the definition of credential schemes on a distributed ledger. This infrastructure allows for governance systems to be constructed and enforced without the need for a central governing authority. All trust verification is performed peer-to-peer using the Peer DID Communications (PeerDIDComm) protocol ([238]).

PyDP - The application of statistical is a Python wrapper for Google's Differential Privacy project. The library provides a set of ϵ-differentially private algorithms. These can be used to produce aggregate statistics over numeric data sets containing private or sensitive information. With PyDP you can control the privacy guarantee and accuracy of your model written in Python. PyDP is being extended to provide DP training of conventional data science algorithms such as Bayesian networks and decision trees with scikit-learn.

Syfertext - is a library which provides secure plaintext pre-processing and secure pipeline deployment for natural language processing in Syft([239]).

openmined_psi - a Private Set Intersection protocol based on ECDH, Bloom Filters, and Golomb Compressed Sets as described in [37].

7.4 Encrypted Split-Inference with Structural Transparency Guarantees

Our work is demonstrated through Duet, a 2-party scenario using the Syft framework. One party with ownership over data provides limited access to a model owned by another actor in order to conduct inference. An information flow is defined wherein a Data Subject receives the result of a prediction performed on their private data using another entity's private model. These

private information assets owned by the DO and DS respectively must interact to produce a prediction to be consumed by the DO. In a societal context, this may be governed under the same norms as a user consuming inference as a service. This context is used to evaluate the ST guarantees of the encrypted inference flow in Figure 7.4. In the actual experiment, MNIST ([240]) images are used as data and a deep convolutional neural network as the model.

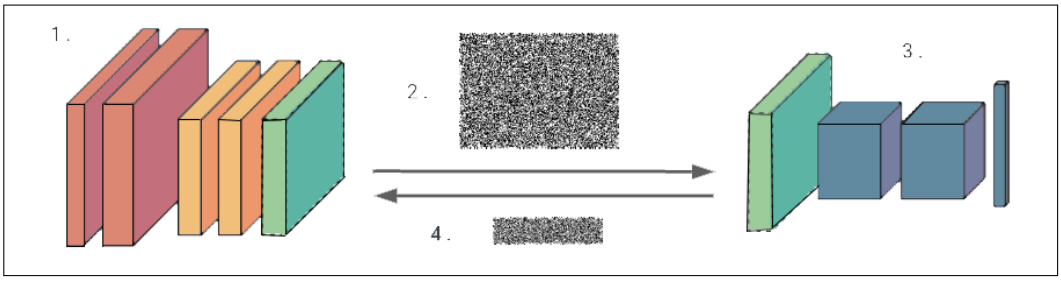


Figure 7.2: Private Inference Flow showing encrypted activation signals being sent away and processed by a remote server. 1. DO segment, 2. Encrypted Activation Signal 3. DS segment 4. Encrypted Output

Governance - Before this information flow may begin, a context is establish between actors and the norms that flow from this context. This is performed in step 1 of Figure 7.4 where DO and DS verify attributes about one another. In Figure 7.3, the DS proves they are a DS using the verifiable credential they received from authority. Similarly, the DO could declare their role as a DO suitable to participate. The social context of this flow is defined through the presentation of their relevant attributes under CL Signatures [232] through their Agent.

Input Verification - This section refers to steps two and three in Figure 7.4, where DO and DS load their model and data. The parameters are composed of a model and some data, both of which are private. In this case, the DO is also the consumer of the inference so their inpujts are of no consequence to the threat model. However, it is important to ascertain that the does in fact have the model we're interested in. This is possible by storing the model object in an Agent like a verifiable credential. Similarly, this can be done with

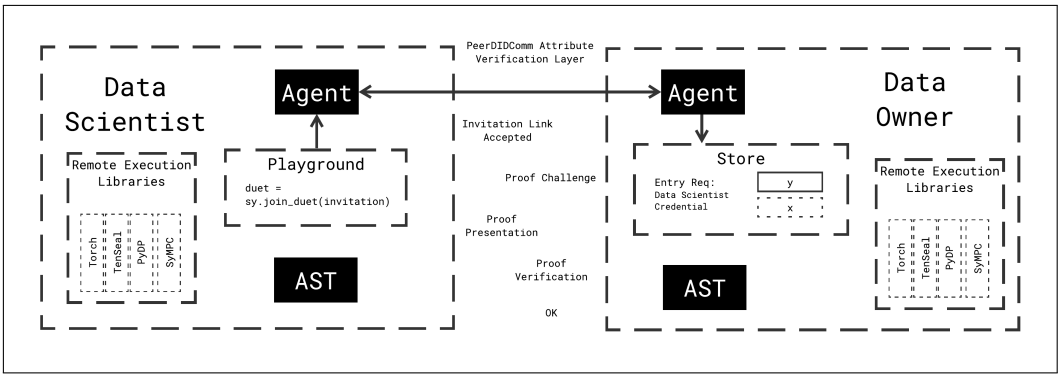


Figure 7.3: Aries Agent Verification flow

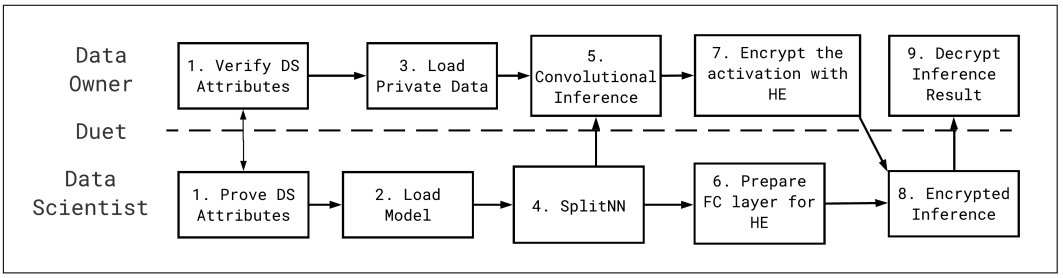


Figure 7.4: Private Inference Flow

datasets to define verifiable properties like completeness, provenance and scheme conformity.

Input Privacy - This section refers to steps four to eight in Figure 7.4 and is described visually in Figure 7.2 which describes the private inference flow. In order to maintain the privacy of the DOs's data and the DS's model, the DOs could encrypt their data and send it to the DS for private inference using CKKS HE ([241, 230]) [242]. However, high dimensional data incurs a significant computation overhead during encrypted computation and increased computation depth necessitates a larger polynomial modulus- increasing ciphertext size and computation requirements. Alternatively, the DS may opt to only share a portion of their model with DOs and execute a SplitNN training flow [135, 243]. Data remains with the DOs at inference time, and only the activation signal is shared. However, statistical information still persists in activation signals, despite representing less information than the original data. The information contained in activation signals can be used by

Forward Step	Processor	Modulus (bits)	File Size	Time Taken
Input Data	DO	plaintext	3.47KB	
Conv1	DO	plaintext	10.97 KB	
Conv2	DO	plaintext	2.4 KB	27ms
Encrypt Signal	DO	140	269.86 KB	11 ms
FC1	DS	140	205.06 KB	
Sq. Activation	DS	140	139.6 KB	
FC2	DS	140	68.81 KB	4.17s

Table 7.1: Experiment 1, where the DO only receives the convolutional layers. The split layer at FC2 is represented by the bar. As more layers need to be processed, a higher modulus is used. Tests were performed using 4 cores Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz. Forward Step describes the layer of processing the data has just emerged from. The Processor is the actor currently performing the computation. File Size is the size of the signal as it emerged from this layer of processing. Time taken gives the time that has passed since the last time is taken. HE parameters; polynomial degree is 8192, the coefficient modulus is 140 bits, there is a security level of 128 bits, and the scale is 2^{26}

.

Forward Step	Processor	Modulus (bits)	File Size	Time Taken
Input Data	DO	plaintext	3.47KB	
Conv1	DO	plaintext	10.97 KB	
Conv2	DO	plaintext	2.4 KB	
FC1	DO	plaintext	529 B	
Sq. Activation	DO	plaintext	529 B	1ms
Encrypt Signal	DO	88	139.62 KB	4.6 ms
FC2	DS	88	68.76 KB	97ms

Table 7.2: Experiment 2, where the FC1 layer is also sent to the DO. The split layer is represented by the bar. As fewer layers need to be processed, a lower modulus is used. Tests were performed using 4 cores Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz. Forward Step describes the layer of processing the data has just emerged from. The Processor is the actor currently performing the computation. File Size is the size of the signal as it emerged from this layer of processing. Time taken gives the time that has passed since the last time is taken. CKKS parameters; polynomial degree is 8192, the coefficient modulus is 88 bits, there is a security level of 128 bits, and the scale is 2^{26}

.

a malicious DS to exploit information leakage [39]. The Shredder technique ([144]) may also be used to apply noise to activation signals before transit, however, this does not provide as strong input privacy as encryption.

A hybrid approach is proposed which offers a trade-off between computational complexity and model privacy. The is allowed to compute a portion of inference in plaintext and the latter portion as ciphertext with the DS. The more model layers that are shared with the DOs, the less computation depth is needed, and the modulus coefficient is minimised. This results in transmitted ciphertext size and computation times which are dramatically reduced from 269.86KB to 139.6KB and 4.17s to 97ms respectively (Appendix Tables 7.1 and 7.2). However, sharing too many layers exposes the model to theft.

Output Privacy - is achieved here with respect to the output recipient in step 9 of Figure 7.4 where the DOs decrypts the inference result locally. However, the real issue here is not that the output of the flow may reveal the DOs's inputs, the DOs is the consumer of the output. It's that over enough inferences, the DOs may be able to infer membership of elements in the training set or perform model inversion attacks on the model ([200, 138, 139]). To protect the privacy of the information in the model, the model may be trained using the Opacus library and the PrivacyEngine utility it provides for differentially private stochastic gradient descent ([233]). This allows the DS to fit the model to the general curve of the dataset rather than over-fitting. This obstructs an attacker's ability to leverage the data in that data model ([207]). This DP training step is considered step zero and is not included in this inference flow. However, it is implemented in the experiment source code.

Output Verification - In this flow output verification may relate to the removal of any bias in the model or data. The DOs is the only stakeholder in the output, and so can be trusted with their own data, however,the veracity of the DS model is an important consideration. At the moment, the DOs relies on the credibility of the authority that attested DS credentials in

step 1 of Figure 7.4 where DOs and DS exchange credentials. This isn't fine-grained trust. However, with Aries infrastructure, schemes can be defined and deployed which may track the veracity and performance of models for presentation to inference consumers fit the requirements of any regulation for model governance use cases.

7.5 Conclusions

Contextual integrity and ST of information flows provide the theoretical framework for privacy at scale. While privacy has traditionally been viewed as a permission system problem alone, Syft delivers privacy at all levels; by leveraging multiple cryptographic protocols and distributed algorithms, it enables new ways to structure information flows. The work being demonstrated is one such novel flow, containing ST guarantees in a 2-party setting, or Duet. Duet provides a research-friendly API for a DO to privately expose their data, while a DS can access or manipulate the data on the owner's side through a zero-knowledge access control mechanism. This framework is designed to lower the barrier between research and privacy-preserving mechanisms so that scientific progress can be made on data that is currently inaccessible or tightly controlled. In future work, Syft will integrate further with PyGrid ([244]) to provide a user interface for organisational policy controls, cloud orchestration, dashboards and dataset management; to allow research institutions to utilise Syft within their existing research data policy frameworks.

Chapter 8

Conclusion

This thesis establishes a range of contributions to the advancement of privacy-enhancing technologies for PPML. The literature gave a background in the core technologies applied in this thesis. This ranged from machine learning, differential privacy, cryptography, federated learning and decentralised identity technologies. In Chapter 3, a case study was evaluated, and a novel, heterogeneous stack classifier was built which predicted the presence of insider threat. The area under the ROC curve was over 0.98, demonstrating the efficacy of machine learning in solving problems in this domain given access to real data. It also drew conclusions about the applicability of federated learning in this use case. In chapter four, a novel framework was introduced that facilitated vertically distributed machine learning on data relating to the same subjects held on different hosts. This exhibits a novel framework that researchers can use to achieve vertically federated learning in practice. In chapter five, the weaknesses in the security of the SplitNN technique were discussed, and appropriate defences were explored in detail. This hardened SplitNN against these attacks With chapter six, a novel distributed trust framework was established which facilitated peer-to-peer access control without the need for a third party. This puts forward a solution for fully privacy-preserving access control while interacting with PPML infrastructure. Finally, chapter seven described a novel framework for the implementation of structured transparency. This provides a cohesive way

to manage information flows in the PPML and analytics space, offering a well-stocked toolkit for the implementation of structured transparency. This also discussed HE inference which fully hardens the SplitNN methodology against model inversion attacks.

8.1 Delivery against objectives

The following defines the delivery against the objectives:

- **Objective I** - The identification of domains where the adoption of these techniques will allow for greater models to be built through access to real-world data where access would traditionally be blocked. In chapter four, the domain of malicious insider threats was explored. It was demonstrated that a highly effective model could be built given a large enough dataset. However, the dataset used was synthetic. This is due to the inherent drawbacks of organisations sharing their logistical information in open domains. The competitive nature of these organisations and the low individual instances of malicious insider threats in individual organisations make the introduction of federated datasets where organisations can collaboratively build effective classifiers without releasing their individual organisational-level information highly appealing. Here is an example where the introduction of federating learning would be highly beneficial.

- **Objective II** - To advance the state-of-the-art with respect to one subfield of the PPML space; vertically distributed learning through the use of SplitNN. This objective is split into three sub-objectives:

  - **Objective II.I** - The creation of a novel implementation of SplitNN. Chapter four introduced PyVertical, which is the first open-source framework for the implementation of SplitNNs. This framework contains novel features- the capacity to build multi-headed SplitNNs and the ability to align data of individual organisations that share

subjects in common for vertically distributed learning on individual subjects. This demonstrates the achievement of this sub-objective.

  - **Objective II.II** - The identification of practical defences against model inversion attacks against SplitNN models. In chapter five, model inversion attacks were demonstrated against the activation signals which are transferred between parties over a network. Two practical defences were evaluated; noising data and maximising distance correlation. In each case, the scale of protection was increased over thresholds, and the resultant accuracy changes in the model were measured. This provides examples of how inversion attacks may be mitigated.
  - **Objective II.III** - The introduction of s novel methodology: HE inference using SplitNN. In chapter seven this HE inference was demonstrated. This facilitated the processing of inference on neural networks where activation signals passed along the network was encrypted. This encryption completely defused the possibility of a malicious adversary inverting the information contained in the activation signal in order to learn information about the input data.

- **Objective III** - The ToIP stack as a mechanism for providing peer-to-peer trust and access control when running distributed PPML infrastructures. In chapter six, a distributed trust framework was implemented which facilitated peer-to-peer access control between a diffuse set of parties. This allowed for access control to be enforced without the need for a third-party authority- leaking information about the participants of the information flow.

- **Objective IV** - The advancement of the notion of Structured Transparency as a means of rationalising the protections placed on information flows during PPML. A technological framework was applied in

order to facilitate the theoretical framework of Structured Transparency. This provides a wide gamut of privacy-enhancing technologies which allow for the intelligent design of information flows as relating to the contextual integrity framework exposed by Helen Nissenbaum. This allows for the application of these privacy-enhancing technologies to be justified in terms of an informed conception of privacy.

8.2 Future work

There are a number of ways that the work contained in this thesis can be further expanded. To begin with, the heterogenous stack classifier approach could be applied to a data federation composed of real organisations looking to solve the problem of malicious insider threats. This would be a novel demonstration of privacy-enhancing technologies providing access to private data within a highly problematic domain where the majority of security incidents take place. This method could be applied across more domains than just malicious insider threats. This could range from healthcare, where hospitals could each possess a dataset and collaborate with patient advocacy groups in order to solve medical problems. Additionally, pyvertical could be further expanded with a series of further information flow examples with data held in even more locations and the defence techniques outlined in this thesis applied; noising, distance correlation maximisation and HE processing of inference. This could also be further developed by looking into a particular differential privacy approach for protecting the output privacy of data in these flows. The peer-to-peer architecture for PPML was demonstrated, however, this could be applied to other information flows than vanilla federated learning and homomorphic split inference. For example, this could applied in the context of a series of horizontally federated learning environments where this structure could be applied. The notion of verifiable credentials should be applied to data. This would create a new entity in the ToIP landscape; verifiable data. Verifiable data would allow verified

properties of the data to be advertised, for example, meta-characteristics like veracity, completeness and lawful ownership. It could be the prerequisite that a data owner in an information flow must demonstrate these characteristics before being accepted into a distributed learning example. This is an entirely new frontier which is necessitated by the mechanics of PPML, as actual data cannot be seen by the researcher and thus cannot be verified through conventional means. Finally, the Syft project can continue to be elaborated upon and improved, providing greater and greater applicability to privacy-preserving, structurally transparent information flows.

Bibliography

[1] Mahbubul Alam et al. "Survey on deep neural networks in speech and vision systems". In: *Neurocomputing* 417 (2020), pp. 302–321.

[2] Léon Bottou. "Large-scale machine learning with stochastic gradient descent". In: *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*. Springer. 2010, pp. 177–186.

[3] Rich Caruana and Alexandru Niculescu-Mizil. "An empirical comparison of supervised learning algorithms". In: *Proceedings of the 23rd international conference on Machine learning*. 2006, pp. 161–168.

[4] Blend Berisha, Endrit Mëziu, and Isak Shabani. "Big data analytics in Cloud computing: an overview". In: *Journal of Cloud Computing* 11.1 (2022), p. 24.

[5] Gregory Cohen et al. "EMNIST: Extending MNIST to handwritten letters". In: *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2017, pp. 2921–2926.

[6] Han Xiao, Kashif Rasul, and Roland Vollgraf. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms". In: *arXiv preprint arXiv:1708.07747* (2017).

[7] Andrew Trask et al. *Beyond Privacy Trade-offs with Structured Transparency*. 2020. arXiv: `2012.08347 [cs.CR]`.

[8] Adam James Hall, Amir Hussain, and M Guftar Shaikh. "Predicting insulin resistance in children using a machine-learning-based clinical decision support system". In: *International Conference on Brain Inspired Cognitive Systems*. Springer. 2016, pp. 274–283.

[9] Sylvia E Peacock. "How web tracking changes user agency in the age of Big Data: The used user". In: *Big Data & Society* 1.2 (2014), p. 205395171456422. ISSN: 2053-9517. DOI: 10.1177/2053951714564228. URL: http://journals.sagepub.com/doi/10.1177/2053951714564228.

[10] Abbott Adam. "NSA analysts 'wilfully violated' surveillance systems, agency admits". In: *The Guardian* (2013). URL: https://www.theguardian.com/world/2013/aug/24/nsa-analysts-abused-surveillance-systems.

[11] Roberto J González. "Hacking the citizenry?" In: *Anthropology today* 33.3 (2017), pp. 9–12.

[12] 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 (visited on 03/12/2023).

[13] Carole Cadwalladr. "The great British Brexit robbery: how our democracy was hijacked". In: *The Observer* (2017), pp. 1–13. ISSN: 0261-3077. DOI: ar000057q[pii]. URL: https://www.theguardian.com/technology/2017/may/07/the-great-british-brexit-robbery-hijacked-democracy?utm%7B%5C_%7Dsource=esp%7B%5C&%7Dutm%7B%5C_%7Dmedium=Email%7B%5C&%7Dutm%7B%5C_%7Dcampaign=GU+Today+USA+-+Collections+2017%7B%5C&%7Dutm%7B%5C_%7Dterm=224936%7B%5C&%7Dsubid=20620601%7B%5C&%7DCMP=GT%7B%5C_%7DUS%7B%5C_%7Dcollection.

[14] Paul Lewis. "Former Cambridge Analytica exec says she wants lies to stop". In: *The Guardian* (2013). URL: https://www.theguardian.com/

uk-news/2018/mar/23/former-cambridge-analytica-executive-brittany-kaiser-wants-to-stop-lies.

[15] Safi Michael. "Revealed: Facebook hate speech exploded in Myanmar during Rohingya crisis". In: *The Guardian* (2018). URL: https://www.theguardian.com/world/2018/apr/03/revealed-facebook-hate-speech-exploded-in-myanmar-during-rohingya-crisis.

[16] Helen Nissenbaum. "Privacy as contextual integrity". In: *Wash. L. Rev.* 79 (2004), p. 119.

[17] Helen Nissenbaum. *Privacy in context: Technology, policy, and the integrity of social life*. Stanford University Press, 2009.

[18] Adam Barth et al. "Privacy and contextual integrity: Framework and applications". In: *2006 IEEE symposium on security and privacy (S&P'06)*. IEEE. 2006, 15–pp.

[19] Richard Cumbley and Peter Church. "Is Big Data creepy?" In: *Computer Law and Security Review* 29.5 (2013), pp. 601–609. ISSN: 02673649. DOI: 10.1016/j.clsr.2013.07.007. arXiv: /dl.acm.org/citation.cfm?id=2017212.2017217 [http:]. URL: http://dx.doi.org/10.1016/j.clsr.2013.07.007.

[20] Trend Micro. "Follow the Data : Analyzing Breaches by Industry". In: *Trend micro analysis of privacy rights ...* (2015). URL: https://documents.trendmicro.com/assets/wp/wp-analyzing-breaches-by-industry.pdf.

[21] *Most U.S. Companies Still Not Prepared for GDPR or CCPA Compliance*. https://www.dataversity.net/most-u-s-companies-still-not-prepared-for-gdpr-or-ccpa-compliance/. Accessed: 05-03-23.

[22] *Europe's governments are failing the GDPR*. https://brave.com/static-assets/files/Brave-2020-DPA-Report.pdf. Accessed: 05-03-23.

[23] *How GDPR Is Failing*. https://www.wired.co.uk/article/gdpr-2022. Accessed: 05-03-23.

[24] European Commission. *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance)*. 2016. URL: https://eur-lex.europa.eu/eli/reg/2016/679/oj.

[25] Priya Goyal et al. "Accurate, large minibatch sgd: Training imagenet in 1 hour". In: *arXiv preprint arXiv:1706.02677* (2017).

[26] Beng Chin Ooi et al. "SINGA: A distributed deep learning platform". In: *Proceedings of the 23rd ACM international conference on Multimedia*. ACM. 2015, pp. 685–688.

[27] Nicolas Papernot et al. "Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data". In: 2015 (2016), pp. 1–16. arXiv: 1610.05755. URL: http://arxiv.org/abs/1610.05755.

[28] Cynthia Dwork et al. "Calibrating noise to sensitivity in private data analysis". In: *Journal of Privacy and Confidentiality* 7.3 (2016), pp. 17–51.

[29] Cynthia Dwork et al. "Our data, ourselves: Privacy via distributed noise generation". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2006, pp. 486–503.

[30] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. "Boosting and differential privacy". In: *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE. 2010, pp. 51–60.

[31] Adam James Hall et al. "Predicting Malicious Insider Threat Scenarios Using Organizational Data and a Heterogeneous Stack-Classifier". In: ().

[32] OpenMined. *PyVertical*. 2020. URL: https://github.com/OpenMined/PyVertical.

[33] Daniele Romanini et al. “PyVertical: A Vertical Federated Learning Framework for Multi-headed SplitNN”. In: *arXiv preprint arXiv:2104.00489* (2021).

[34] Tom Titcombe et al. “Practical Defences Against Model Inversion Attacks for Split Neural Networks”. In: *arXiv preprint arXiv:2104.05743* (2021).

[35] Adam James Hall et al. “Syft 0.5: A platform for universally deployable structured transparency”. In: *arXiv preprint arXiv:2104.12385* (2021).

[36] Will Abramson et al. “A Distributed Trust Framework for Privacy-Preserving Machine Learning”. In: *Lecture Notes in Computer Science* (2020), pp. 205–220. ISSN: 1611-3349. DOI: 10.1007/978-3-030-58986-8_14. URL: http://dx.doi.org/10.1007/978-3-030-58986-8_14.

[37] Nick Angelou et al. “Asymmetric Private Set Intersection with Applications to Contact Tracing and Private Vertical Federated Machine Learning”. In: *arXiv preprint arXiv:2011.09350* (2020).

[38] Theo Ryffel et al. “A generic framework for privacy preserving deep learning”. In: *arXiv preprint arXiv:1811.04017* (2018).

[39] Praneeth Vepakomma et al. “Reducing leakage in distributed deep learning for sensitive health data”. In: *arXiv preprint arXiv:1812.00564* (2019).

[40] Simon Hegelich. “Defining Machine Learning”. In: *Digital Phenotyping and Mobile Sensing: New Developments in Psychoinformatics*. Springer, 2022, pp. 455–460.

[41] Tom M Mitchell and Tom M Mitchell. *Machine learning*. Vol. 1. 9. McGraw-hill New York, 1997.

[42] K. Steinbuch and U. A. W. Piske. “Learning matrices and their applications”. In: *IEEE Transactions on Electronic Computers* EC-12.6 (1963), pp. 846–862. DOI: 10.1109/PGEC.1963.263588.

[43] Otkrist Gupta and Ramesh Raskar. "Distributed learning of deep neural network over multiple agents". In: *Journal of Network and Computer Applications* 116.April (2018), pp. 1–8. ISSN: 10958592. DOI: 10.1016/j.jnca.2018.05.003. arXiv: 1810.06060. URL: https://doi.org/10.1016/j.jnca.2018.05.003.

[44] Ning Chen, Bernardete Ribeiro, and An Chen. "Financial credit risk assessment: a recent review". In: *Artificial Intelligence Review* 45.1 (2016), pp. 1–23. ISSN: 15737462. DOI: 10.1007/s10462-015-9434-x.

[45] A. Hall, A. Hussain, and M. Shaikh. "Predicting Insulin Resistance in Children Using a Machine- Learning-Based Clinical Decision Support System". In: *International Conference on Brain Inspired Cognitive Systems* 1 (2016), pp. 58–67. DOI: 10.1007/978-3-319-49685-6.

[46] Enrico Pellegrini et al. "Machine learning of neuroimaging for assisted diagnosis of cognitive impairment and dementia: A systematic review". In: *Alzheimer's and Dementia: Diagnosis, Assessment and Disease Monitoring* 10 (2018), pp. 519–535. ISSN: 23528729. DOI: 10.1016/j.dadm.2018.07.004. arXiv: 1804.01961.

[47] Vitaly Ford and Ambareen Siraj. "Applications of Machine Learning in Cyber Security". In: *CAINE - International Conference on Computer Applications in Industry and Engineering* October 2014 (2014). ISSN: 15306992 15306984. DOI: 10.1021/acs.nanolett.5b03871. URL: https://www.researchgate.net/publication/283083699%7B%5C_%7DApplications%7B%5C_%7Dof%7B%5C_%7DMachine%7B%5C_%7DLearning%7B%5C_%7Din%7B%5C_%7DCyber%7B%5C_%7DSecurity.

[48] Trevor Hastie et al. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer, 2009.

[49] Simon Parsons. "Introduction to Machine Learning, Second Editon by Ethem Alpaydin, MIT Press, 584 pp., ISBN 978-0-262-01243-0". In: *The Knowledge Engineering Review* 25.3 (2010), pp. 353–353.

[50] M Jordan, J Kleinberg, and B Schölkopf. *Bishop, Christopher M. (2006). Pattern Recognition and Machine Learning*. ISBN: 9780387310732.

[51] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[52] Alan O Sykes. "An introduction to regression analysis". In: (1993).

[53] Michael A Golberg and Hokwon A Cho. *Introduction to regression analysis*. WIT press, 2004.

[54] Gareth James et al. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[55] Zhou Wang and Alan C Bovik. "Mean squared error: Love it or leave it? A new look at signal fidelity measures". In: *IEEE signal processing magazine* 26.1 (2009), pp. 98–117.

[56] Peter J Huber. "Robust estimation of a location parameter". In: *Breakthroughs in statistics: Methodology and distribution* (1992), pp. 492–518.

[57] Davide Proserpio, Sharon Goldberg, and Frank McSherry. "Calibrating data to sensitivity in private data analysis". In: *Proceedings of the VLDB Endowment* 7.8 (Apr. 2014), pp. 637–648. ISSN: 21508097. DOI: 10.14778/2732296.2732300. URL: http://dl.acm.org/citation.cfm?doid=2732296.2732300.

[58] Irit Dinur and Kobbi Nissim. "Revealing information while preserving privacy". In: Association for Computing Machinery (ACM), Nov. 2003, pp. 202–210. DOI: 10.1145/773153.773173.

[59] Nicolas Papernot. "A Marauder's Map of Security and Privacy in Machine Learning". In: October 2018 (2018), pp. 1–20. DOI: 10.1145/3270101.3270102. arXiv: 1811.01134. URL: http://arxiv.org/abs/1811.01134.

[60] Simson L Garfinkel, John M Abowd, and Sarah Powazek. "Issues encountered deploying differential privacy". In: *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*. ACM. 2018, pp. 133–137.

[61] C.E. Shannon. "A Mathematical Theory of Communication". In: *Bell System Technical Journal* 27.3 (July 1948), pp. 379–423, 623–656.

[62] Fernando Pérez-Cruz. "Kullback-leibler divergence estimation of continuous distributions". In: *IEEE International Symposium on Information Theory - Proceedings* (2008), pp. 1666–1670. ISSN: 21578101. DOI: 10.1109/ISIT.2008.4595271.

[63] Christopher Waites. "PyVacy: Towards Practical Differential Privacy for Deep Learning". In: (2019). URL: https://smartech.gatech.edu/bitstream/handle/1853/61412/WAITES-UNDERGRADUATERESEARCHOPTIONTHESIS-2019.pdf?sequence=1%7B%5C&%7DisAllowed=y.

[64] Martin Abadi et al. "Deep Learning with Differential Privacy". In: Ccs (2016), pp. 308–318. DOI: 10.1145/2976749.2978318. arXiv: arXiv:1607.00133v2.

[65] Ilya Mironov. "Rényi Differential Privacy". In: *Proceedings - IEEE Computer Security Foundations Symposium* (2017), pp. 263–275. ISSN: 19401434. DOI: 10.1109/CSF.2017.11. arXiv: arXiv:1702.07476v3.

[66] Yu-Xiang Wang, Borja Balle, and Shiva Kasiviswanathan. "Subsampled R\'enyi Differential Privacy and Analytical Moments Accountant". In: 1 (2018), pp. 1–29. arXiv: 1808.00087. URL: http://arxiv.org/abs/1808.00087.

[67] Jaewoo Lee and Daniel Kifer. "Concentrated differentially private gradient descent with adaptive per-iteration privacy budget". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM. 2018, pp. 1656–1665.

[68] Xinyang Zhang, Shouling Ji, and Ting Wang. "Differentially Private Releasing via Deep Generative Model (Technical Report)". In: *arXiv preprint arXiv:1801.01594* (2018).

[69] Ian J. Goodfellow et al. "Generative Adversarial Networks". In: (2014), pp. 1–9. arXiv: 1406.2661. URL: http://arxiv.org/abs/1406.2661.

[70] Michael Ben-Or, Shafi Goldwasser, 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. Chicago, Illinois, USA: ACM, 1988, pp. 1–10. ISBN: 0-89791-264-0. DOI: 10.1145/62212.62213. URL: http://doi.acm.org/10.1145/62212.62213.

[71] Andrew Chi-Chih Yao. "How to Generate and Exchange Secrets (Extended Abstract)". In: *FOCS*. 1986.

[72] Theo Ryffel et al. "A generic framework for privacy preserving deep learning". In: (2018), pp. 1–5. DOI: arXiv:1811.04017v2. arXiv: 1811.04017. URL: http://arxiv.org/abs/1811.04017.

[73] Octavian Catrina and Sebastiaan De Hoogh. "Improved Primitives for Secure Multiparty Integer Computation". In: *Proceedings of the 7th International Conference on Security and Cryptography for Networks*. SCN'10. Amalfi, Italy: Springer-Verlag, 2010, pp. 182–199. ISBN: 3-642-15316-X, 978-3-642-15316-7. URL: http://dl.acm.org/citation.cfm?id=1885535.1885555.

[74] Praneeth Vepakomma et al. "Split learning for health: Distributed deep learning without sharing raw patient data". In: Nips (2018). DOI: arXiv:1812.00564v1. arXiv: 1812.00564. URL: http://arxiv.org/abs/1812.00564.

[75] Sameer Wagh, Divya Gupta, and Nishanth Chandran. "SecureNN : Efficient and Private Neural Network Training". In: *IACR Cryptology ePrint Archive* (2018), pp. 1–44.

[76] Craig Gentry et al. "Fully homomorphic encryption using ideal lattices." In: *Stoc*. Vol. 9. 2009. 2009, pp. 169–178.

[77] Ronald L Rivest, Adi Shamir, and Leonard Adleman. "A method for obtaining digital signatures and public-key cryptosystems". In: *Communications of the ACM* 21.2 (1978), pp. 120–126.

[78] Pascal Paillier. "Public-key cryptosystems based on composite degree residuosity classes". In: *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 1999, pp. 223–238.

[79] David Wu and Jacob Haven. *Using homomorphic encryption for large scale statistical analysis*. 2012.

[80] Yoshinori Aono et al. "Privacy-preserving deep learning via additively homomorphic encryption". In: *IEEE Transactions on Information Forensics and Security* 13.5 (2017), pp. 1333–1345.

[81] Wenjie Lu, Shohei Kawasaki, and Jun Sakuma. "Using Fully Homomorphic Encryption for Statistical Analysis of Categorical, Ordinal and Numerical Data." In: *IACR Cryptology ePrint Archive* 2016 (2016), p. 1163.

[82] Hao Chen, Kim Laine, and Rachel Player. "Simple encrypted arithmetic library-SEAL v2. 1". In: *International Conference on Financial Cryptography and Data Security*. Springer. 2017, pp. 3–18.

[83] Michael J Freedman, Kobbi Nissim, and Benny Pinkas. "Efficient private matching and set intersection". In: *International conference on the theory and applications of cryptographic techniques*. Springer. 2004, pp. 1–19.

[84] Yan Huang, David Evans, and Jonathan Katz. "Private set intersection: Are garbled circuits better than custom protocols?" In: *NDSS*. 2012.

[85] 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.

[86] Dana Dachman-Soled et al. "Efficient robust private set intersection". In: *International Conference on Applied Cryptography and Network Security*. Springer. 2009, pp. 125–142.

[87] Prasad Buddhavarapu et al. "Private Matching for Compute." In: *IACR Cryptol. ePrint Arch.* 2020 (2020), p. 599.

[88] Mihaela Ion et al. "On Deploying Secure Computing: Private Intersection-Sum-with-Cardinality". In: *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. 2020, pp. 370–389.

[89] Melissa Chase and Peihan Miao. "Private Set Intersection in the Internet Setting from Lightweight Oblivious PRF". In: *Annual International Cryptology Conference*. Springer. 2020, pp. 34–63.

[90] Benny Pinkas, Thomas Schneider, and Michael Zohner. "Scalable private set intersection based on OT extension". In: *ACM Transactions on Privacy and Security (TOPS)* 21.2 (2018), pp. 1–35.

[91] Daniel Demmler et al. "PIR-PSI: Scaling Private Contact Discovery". In: *IACR Cryptol. ePrint Arch.* (2018).

[92] Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. "Scalable multi-party private set-intersection". In: *IACR International Workshop on Public Key Cryptography*. Springer. 2017, pp. 175–203.

[93] Ergys Ristani et al. "Performance Measures and a Data Set for Multi-target, Multi-camera Tracking". In: *ECCV Workshops*. 2016.

[94] Kenny Peng. *Facial recognition datasets are being widely used despite being taken down due to ethical concerns. Here's how.* 2020. URL: https://freedom-to-tinker.com/2020/10/21/facial-recognition-datasets-are-being-widely-used-despite-being-taken-down-due-to-ethical-concerns-heres-how/ (visited on 10/21/2020).

[95] Jules. Harvey Adam. LaPlace. *Exposing.ai*. 2021. URL: https://exposing.ai (visited on 01/01/2021).

[96] Andrew Trask and Kritika Prakash. *Towards General-purpose Infrastructure for Protecting Scientific Data Under Study*. 2020.

[97] Alex Ingerman and Krzys Ostrowski. *Introducing tensorflow federated.* 2019. URL: blog.tensorflow.org.

[98] Wenbin Wei. *FATE Documentation*. 2018. URL: https://github.com/FederatedAI/FATE.

[99] Sebastian Caldas et al. "LEAF: A Benchmark for Federated Settings". In: *CoRR* abs/1812.01097 (2018). arXiv: 1812.01097. URL: http://arxiv.org/abs/1812.01097.

[100] Max Wong, H Moster, and Lin, Bryce. *Eggroll*. 2018. URL: https://github.com/WeBankFinTech/eggroll.

[101] Qinghe Jing, Dong Daxiang, and contributors. *PaddleFL*. 2019. URL: https://github.com/PaddlePaddle/PaddleFL.

[102] H. Wang et al. "Optimizing Federated Learning on Non-IID Data with Reinforcement Learning". In: *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*. 2020, pp. 1698–1707. DOI: 10.1109/INFOCOM41043.2020.9155494.

[103] Will Abramson et al. "A Distributed Trust Framework for Privacy-Preserving Machine Learning". In: *Lecture Notes in Computer Science* (2020), pp. 205–220. ISSN: 1611-3349. DOI: 10.1007/978-3-030-58986-8_14. URL: http://dx.doi.org/10.1007/978-3-030-58986-8_14.

[104] Nvidia. *Clara*. 2019. URL: https://blogs.nvidia.com/blog/2019/12/01/clara-federated-learning/.

[105] Sherpa. *We Research and Build Artificial Intelligence Technology and Services*. 2019. URL: https://github.com/sherpaai/Sherpa.ai-Federated-Learning-Framework.

[106] Heiko Ludwig et al. *IBM Federated Learning: an Enterprise Framework White Paper V0.1*. 2020. arXiv: 2007.10987 [cs.LG].

[107] Georgios Damaskinos et al. "FLeet". In: *Proceedings of the 21st International Middleware Conference* (Dec. 2020). DOI: 10.1145/3423211.3425685. URL: http://dx.doi.org/10.1145/3423211.3425685.

[108] Cao Hui et al. "IFed: A novel federated learning framework for local differential privacy in Power Internet of Things". In: *International Journal of Distributed Sensor Networks* 16 (May 2020), p. 155014772091969. DOI: 10.1177/1550147720919698.

[109] Chaoyang He et al. *FedML: A Research Library and Benchmark for Federated Machine Learning*. 2020. arXiv: 2007.13518 [cs.LG].

[110] Daniel J. Beutel et al. *Flower: A Friendly Federated Learning Research Framework*. 2020. arXiv: 2007.14390 [cs.LG].

[111] In Lee. "Cybersecurity: Risk management framework and investment cost analysis". In: *Business Horizons* (2021).

[112] "Regulation (EU) 2016/679 of the European Parliament and of the Council". In: (May 25, 2018). URL: https://www.legislation.gov.uk/eur/2016/679/article/5 (visited on 06/17/2019).

[113] Keith Bonawitz et al. "Towards Federated Learning at Scale: System Design". In: (2019). arXiv: 1902.01046. URL: http://arxiv.org/abs/1902.01046.

[114] Jeffrey Dean et al. "Large scale distributed deep networks". In: *Advances in neural information processing systems*. 2012, pp. 1223–1231.

[115] 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.

[116] Dipankar Das et al. "Distributed deep learning using synchronous stochastic gradient descent". In: *arXiv preprint arXiv:1602.06709* (2016).

[117] Eugene Bagdasaryan et al. “How To Backdoor Federated Learning”. In: 1 (2018). ISSN: 0007-1250. DOI: 10.1561/2200000016. arXiv: 1807.00459. URL: http://arxiv.org/abs/1807.00459.

[118] Eugene Bagdasaryan et al. “How to backdoor federated learning”. In: *arXiv preprint arXiv:1807.00459* (2018).

[119] 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.

[120] Qiang Yang et al. “Federated machine learning: Concept and applications”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 10.2 (2019), pp. 1–19.

[121] Siwei Feng and Han Yu. “Multi-Participant Multi-Class Vertical Federated Learning”. In: *arXiv preprint arXiv:2001.11154* (2020).

[122] Yang Liu, Xiong Zhang, and Libin Wang. “Asymmetrically Vertical Federated Learning”. In: *arXiv preprint arXiv:2004.07427* (2020).

[123] Wenliang Du and Mikhail J Atallah. “Privacy-preserving cooperative statistical analysis”. In: *Seventeenth Annual Computer Security Applications Conference*. IEEE. 2001, pp. 102–110.

[124] Wenliang Du, Yunghsiang S Han, and Shigang Chen. “Privacy-preserving multivariate statistical analysis: Linear regression and classification”. In: *Proceedings of the 2004 SIAM international conference on data mining*. SIAM. 2004, pp. 222–233.

[125] Jaideep Vaidya and Chris Clifton. “Privacy preserving association rule mining in vertically partitioned data”. In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2002, pp. 639–644.

[126] Alan F Karr et al. “Privacy-preserving analysis of vertically partitioned data using secure matrix products”. In: *Journal of Official Statistics* 25.1 (2009), p. 125.

[127] Ashish P Sanil et al. “Privacy preserving regression modelling via distributed computation”. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2004, pp. 677–682.

[128] Li Wan et al. “Privacy-preservation for gradient descent methods”. In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2007, pp. 775–783.

[129] 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.

[130] Chandra Thapa, M. A. P. Chamikara, and Seyit Camtepe. *SplitFed: When Federated Learning Meets Split Learning*. 2020. arXiv: 2004.12088 [cs.LG].

[131] Stephen Hardy et al. “Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption”. In: *arXiv preprint arXiv:1711.10677* (2017).

[132] Richard Nock et al. “Entity resolution and federated learning get a federated resolution”. In: *arXiv preprint arXiv:1803.04035* (2018).

[133] Chaoyang He et al. “Fedml: A research library and benchmark for federated machine learning”. In: *arXiv preprint arXiv:2007.13518* (2020).

[134] Otkrist Gupta and Ramesh Raskar. “Distributed learning of deep neural network over multiple agents”. In: *Journal of Network and Computer Applications* 116 (2018), pp. 1–8.

[135] Praneeth Vepakomma et al. “Split learning for health: Distributed deep learning without sharing raw patient data”. In: *arXiv preprint arXiv:1812.00564* (2018).

[136] P. Vepakomma et al. “Reducing leakage in distributed deep learning for sensitive health data”. In: 2017 (2019), pp. 1–6.

[137] 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.

[138] Si Chen, Ruoxi Jia, and Guo-Jun Qi. *Improved Techniques for Model Inversion Attacks*. 2020. arXiv: 2010.04092 [cs.LG].

[139] Yuheng Zhang et al. "The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[140] Maoqiang Wu et al. *Evaluation of Inference Attack Models for Deep Learning on Medical Data*. 2020. arXiv: 2011.00177 [cs.LG].

[141] Zecheng He, Tianwei Zhang, and Ruby B Lee. "Model inversion attacks against collaborative inference". In: *Proceedings of the 35th Annual Computer Security Applications Conference*. 2019, pp. 148–162.

[142] Sharif Abuadbba et al. "Can We Use Split Learning on 1D CNN Models for Privacy Preserving Training?" In: *arXiv preprint arXiv:2003.12365* (2020).

[143] Cynthia Dwork. "Differential Privacy: A Survey of Results". In: *Theory and Applications of Models of Computation*. Ed. by Manindra Agrawal et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1–19. ISBN: 978-3-540-79228-4.

[144] Fatemehsadat Mireshghallah et al. "Shredder: Learning noise distributions to protect inference privacy". In: *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. 2020, pp. 3–18.

[145] Sean D Holcomb et al. "Overview on deepmind and its alphago zero ai". In: *Proceedings of the 2018 international conference on big data and education*. ACM. 2018, pp. 67–71.

[146] Jenna Wiens and Erica S Shenoy. "Machine learning for healthcare: on the verge of a major shift in healthcare epidemiology". In: *Clinical Infectious Diseases* 66.1 (2017), pp. 149–153.

[147] Ibrahim Abaker Targio Hashem et al. "The role of big data in smart city". In: *International Journal of Information Management* 36.5 (2016), pp. 748–758.

[148] Pamela Spence. *How we can place a value on health care data*. 2019. URL: https://www.ey.com/en%5C_gl/life-sciences/how-we-can-place-a-value-on-health-care-data.

[149] Paul Voigt and Axel Von dem Bussche. "The eu general data protection regulation (gdpr)". In: *A Practical Guide, 1st Ed., Cham: Springer International Publishing* (2017).

[150] Kaliya Young and Steve Greenberg. *A Field Guide to Internet Trust*. 2014. URL: https://identitywoman.net/wp-content/uploads/TrustModelFieldGuideFinal-1.pdf.

[151] Aaron M Hoffman. "A conceptualization of trust in international relations". In: *European Journal of International Relations* 8.3 (2002), pp. 375–401.

[152] Esther Keymolen. "Trust on the line: a philosophycal exploration of trust in the networked era". In: (2016).

[153] Julia Powles and Hal Hodson. "Google DeepMind and healthcare in an age of algorithms". In: *Health and technology* 7.4 (2017), pp. 351–367.

[154] Elizabeth Denham. *Royal Free - Google DeepMind trial failed to comply with data protection law*. Tech. rep. Information Commisioner Office, 2017. URL: https://ico.org.uk/about-the-ico/news-and-events/news-and-blogs/2017/07/royal-free-google-deepmind-trial-failed-to-comply-with-data-protection-law/.

[155] Owen Hughes. *Royal Free: 'No changes to data-sharing' as Google absorbs Streams*. Nov. 2018. URL: https://www.digitalhealth.net/2018/11/royal-free-data-sharing-google-deepmind-streams/.

[156] Jeffrey De Fauw et al. "Automated analysis of retinal imaging using machine learning techniques for computer vision". In: *F1000Research* 5 (2016).

[157] Carlton Chu et al. "Applying machine learning to automated segmentation of head and neck tumour volumes and organs at risk on radiotherapy planning CT and MRI scans". In: *F1000Research* 5 (2016).

[158] Drummond Reed et al. *Decentralized Identifiers (DIDs) v1.0*. Jan. 2020. URL: https://w3c.github.io/did-core/.

[159] W3C Credential Community Group. *DID Method Registry*. Tech. rep. 2019. URL: https://w3c-ccg.github.io/did-method-registry/.

[160] Daniel Hardman. *Peer DID Method Specification*. Tech. rep. 2019. URL: https://openssi.github.io/peer-did-method-spec/index.html.

[161] Daniel Hardman. *DID Communication*. Github Requests for Comments. RFC. Jan. 2019. URL: https://github.com/hyperledger/aries-rfcs/tree/master/concepts/0005-didcomm.

[162] Hyperledger. *Hyperledger Aries*. URL: https://www.hyperledger.org/projects/aries.

[163] Oliver Terbu. *DIF starts DIDComm Working Group*. Decentralized Identity Foundation, 2020. URL: https://medium.com/decentralized-identity/dif-starts-didcomm-working-group-9c114d9308dc.

[164] Taher ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms". In: *IEEE transactions on information theory* 31.4 (1985), pp. 469–472.

[165] JEREMY Wohlwend. *Elliptic curve cryptography: Pre and post quantum*. Tech. rep. MIT, Tech. Rep, 2016.

[166] Manu Sporny, Dave Longely, and David Chadwick. *Verifiable Credentials Data Model 1.0*. Tech. rep. W3C, Nov. 2019. URL: https://w3c.github.io/vc-data-model/.

[167] M Jones, J Bradley, and N Sakimura. *JSON Web Signatures*. RFC. May 2015. URL: https://tools.ietf.org/html/rfc7515.

[168] Dave Longley, Manu Sporny, and Christopher Allen. *Linked Data Signatures 1.0*. Tech. rep. 2019. URL: https://w3c-dvcg.github.io/ld-signatures/.

[169] Jan Camenisch and Anna Lysyanskaya. "A Signature Scheme with Efficient Protocols". en. In: *Security in Communication Networks*. Ed. by Gerhard Goos et al. Vol. 2576. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 268–289. ISBN: 978-3-540-00420-2 978-3-540-36413-9. DOI: 10.1007/3-540-36413-7_20. URL: http://link.springer.com/10.1007/3-540-36413-7%5C_20.

[170] M Davie et al. *The Trust Over IP Stack*. RFC 289. Hyperledger, Oct. 2019. URL: https://github.com/hyperledger/aries-rfcs/tree/master/concepts/0289-toip-stack.

[171] Man Ho Au et al. "Dynamic Universal Accumulators for DDH Groups and Their Application to Attribute-Based Anonymous Credential Systems". In: *Topics in Cryptology – CT-RSA 2009*. Ed. by Marc Fischlin. Vol. 5473. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 295–308. URL: http://link.springer.com/10.1007/978-3-642-00862-7_20 (visited on 08/22/2019).

[172] Keith Bonawitz et al. "Practical Secure Aggregation for Federated Learning on User-Held Data". In: *CoRR* abs/1611.04482 (2016). arXiv: 1611.04482. URL: http://arxiv.org/abs/1611.04482.

[173] Nikolaos Pitropakis, Christos Lyvas, and Costas Lambrinoudakis. "The Greater The Power , The More Dangerous The Abuse : Facing Malicious Insiders in The Cloud". In: c (2017), pp. 156–161.

[174] Sans Intitute. "Interested in learning SANS Institute InfoSec Reading Room Defending Against the Wrong Enemy : 2017 SANS Defending Against the Wrong Enemy : 2017 SANS Insider Threat Survey". In: (2017).

[175] Institute Ponemon. "2017 Cost of Data Breach Study, Global Overview". In: *IBM Security* March (2017), pp. 1–34. URL: https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=SEL03130WWEN%7B%5C&%7D.

[176] Computer Associates. "Insider Threat 2018 Report". In: (2018), p. 41. URL: https://%20www.cybersecurity-insiders.com/wp-content/uploads/2016/09/Insider-Threat-Report-2018.pdf.

[177] Iffat A. Gheyas and Ali E. Abdallah. "Detection and prediction of insider threats to cyber security: a systematic literature review and meta-analysis". In: *Big Data Analytics* 1.1 (2016), p. 6. ISSN: 2058-6345. DOI: 10.1186/s41044-016-0006-0. URL: http://bdataanalytics.biomedcentral.com/articles/10.1186/s41044-016-0006-0.

[178] William Eberle and Lawrence Holder. "Insider Threat Detection Using Graph-Based Approaches". In: *2009 Cybersecurity Applications & Technology Conference for Homeland Security* (2009), pp. 237–241. ISSN: 1936-1610. DOI: 10.1109/CATCH.2009.7. URL: http://ieeexplore.ieee.org/document/4804450/.

[179] Yu Yingbing and James H. Graham. "Anomaly instruction detection of masqueraders and threat evaluation using fuzzy logic". In: *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics* 3 (2007), pp. 2309–2314. ISSN: 1062922X. DOI: 10.1109/ICSMC.2006.385207.

[180] F. Meng et al. "Deep Learning Based Attribute Classification Insider Threat Detection for Data Security". In: *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*. June 2018, pp. 576–581. DOI: 10.1109/DSC.2018.00092.

[181] Aaron Tuor et al. “Deep Learning for Unsupervised Insider Threat Detection in Structured Cybersecurity Data Streams”. In: 2012 (2017). arXiv: 1710.00811. URL: http://arxiv.org/abs/1710.00811.

[182] Yessir Hashem et al. “Inside the Mind of the Insider: Towards Insider Threat Detection Using Psychophysiological Signals *”. In: *Journal of Internet Services and Information Security* 6.1 (2016), pp. 20–36. DOI: 10.1145/2808783.2808792. URL: http://isyou.info/jisis/vol6/no1/jisis-2016-vol6-no1-02.pdf.

[183] Gaurang Gavai et al. “Supervised and unsupervised methods to detect insider threat from enterprise social and online activity data”. In: *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)* 6.4 (2015), pp. 47–63. ISSN: 20935382. DOI: 10.1145/2808783.2808784. URL: http://isyou.info/jowua/papers/jowua-v6n4-2.pdf.

[184] *Weka 3 - Data Mining with Open Source Machine Learning Software in Java*. https://www.cs.waikato.ac.nz/ml/weka/. Accessed: 2018-10-05.

[185] Ian H Witten, Eibe Frank, and Mark a Hall. *Data Mining: Practical Machine Learning Tools and Techniques (Google eBook)*. 2013, p. 664. ISBN: 0080890369. DOI: 0120884070 , 9780120884070. arXiv: arXiv:1011.1669v3. URL: http://books.google.com/books?id=bDtLM8CODsQC%7B%5C&%7Dpgis=1.

[186] J. Paulin; A. Calinescu; M. Wooldridge. “Agent-Based Modeling for Complex Financial Systems have concluded that the dynamics of networked market”. In: *IEEE Intelligent Systems* 33.2 (2018), pp. 74–82. ISSN: 1541-1672. DOI: 10.1109/MIS.2018.022441352.

[187] H Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *arXiv preprint arXiv:1602.05629* (2016).

[188] Jakub Konečnỳ et al. “Federated learning: Strategies for improving communication efficiency”. In: *arXiv preprint arXiv:1610.05492* (2016).

[189] Brendan McMahan and Daniel Ramage. "Federated learning: Collaborative machine learning without centralized training data". In: *Google Research Blog* 3 (2017). URL: https://ai.googleblog.com/2017/04/%20federated-learning-collaborative.html.

[190] Keith Bonawitz et al. "Towards federated learning at scale: System design". In: *arXiv preprint arXiv:1902.01046* (2019).

[191] Murat Kantarcioglu and Chris Clifton. "Privacy-preserving distributed mining of association rules on horizontally partitioned data". In: *IEEE transactions on knowledge and data engineering* 16.9 (2004), pp. 1026–1037.

[192] Sabine McConnell and David B Skillicorn. "Building predictors from vertically distributed data". In: *Proceedings of the 2004 conference of the Centre for Advanced Studies on Collaborative research*. 2004, pp. 150–162.

[193] S.R. Kadhe et al. "Privacy-Preserving Federated Learning over Vertically and Horizontally Partitioned Data for Financial Anomaly Detection". In: *arXiv preprint arXiv:2310.19304* (2023). URL: https://arxiv.org/abs/2310.19304.

[194] Cynthia Dwork et al. "Calibrating Noise to Sensitivity in Private Data Analysis". In: *Theory of Cryptography*. Ed. by Shai Halevi and Tal Rabin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 265–284. ISBN: 978-3-540-32732-5.

[195] Nick Angelou et al. *PSI Source Code*. 2020. URL: https://github.com/OpenMined/PSI.

[196] Pavlos Papadopoulos et al. "Privacy and Trust Redefined in Federated Machine Learning". In: *Machine Learning and Knowledge Extraction* 3.2 (2021), pp. 333–356. ISSN: 2504-4990. DOI: 10.3390/make3020017. URL: https://www.mdpi.com/2504-4990/3/2/17.

[197] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial Intelligence and Statistics*. PMLR. 2017, pp. 1273–1282.

[198] Theodora S Brisimi et al. “Federated learning of predictive models from federated electronic health records”. In: *International journal of medical informatics* 112 (2018), pp. 59–67.

[199] Georgios A Kaissis et al. “Secure, privacy-preserving and federated machine learning in medical imaging”. In: *Nature Machine Intelligence* (2020), pp. 1–7.

[200] Reza Shokri et al. “Membership inference attacks against machine learning models”. In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2017, pp. 3–18.

[201] Nicholas Carlini et al. “Extracting Training Data from Large Language Models”. In: *arXiv preprint arXiv:2012.07805* (2020).

[202] Binghui Wang and Neil Zhenqiang Gong. “Stealing hyperparameters in machine learning”. In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2018, pp. 36–52.

[203] Eugene Bagdasaryan et al. “How to backdoor federated learning”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 2938–2948.

[204] John R. Douceur. “The Sybil Attack”. In: *Revised Papers from the First International Workshop on Peer-to-Peer Systems*. IPTPS ’01. Berlin, Heidelberg: Springer-Verlag, 2002, pp. 251–260. ISBN: 3540441794.

[205] Peter Kairouz et al. “Advances and open problems in federated learning”. In: *arXiv preprint arXiv:1912.04977* (2019).

[206] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[207] Martin Abadi et al. “Deep learning with differential privacy”. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 2016, pp. 308–318.

[208] Nicolas Papernot et al. “Semi-supervised knowledge transfer for deep learning from private training data”. In: *arXiv preprint arXiv:1610.05755* (2016).

[209] Nicolas Papernot et al. “Scalable private learning with pate”. In: *arXiv preprint arXiv:1802.08908* (2018).

[210] Mathias Lecuyer et al. “Certified robustness to adversarial examples with differential privacy”. In: *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2019, pp. 656–672.

[211] Luis Muñoz-González et al. “Towards poisoning of deep learning algorithms with back-gradient optimization”. In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM. 2017, pp. 27–38.

[212] Deyan Chen and Hong Zhao. “Data security and privacy protection issues in cloud computing”. In: *2012 International Conference on Computer Science and Electronics Engineering*. Vol. 1. IEEE. 2012, pp. 647–651.

[213] Omer F Ahmad, Danail Stoyanov, and Laurence B Lovat. “Barriers and Pitfalls for Artificial Intelligence in Gastroenterology: Ethical and Regulatory issues”. In: *Techniques in Gastrointestinal Endoscopy* (2019), p. 150636.

[214] Hyperledger. *Hyperledger Aries Cloud Agent - Python*. 2019. URL: `https://github.com/hyperledger/aries-cloudagent-python`.

[215] Government of British Columbia. *British Columbia’s Verifiable Organizations*. 2018. URL: `https://orgbook.gov.bc.ca/en/home`.

[216] Adrian Nilsson et al. "A performance evaluation of federated learning algorithms". In: *Proceedings of the Second Workshop on Distributed Infrastructures for Deep Learning*. 2018, pp. 1–8.

[217] Matthew Fredrikson et al. "Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing". In: *23rd USENIX Security Symposium USENIX Security 14)*. 2014, pp. 17–32.

[218] Arjun Nitin Bhagoji et al. "Analyzing federated learning through an adversarial lens". In: *arXiv preprint arXiv:1811.12470* (2018).

[219] Yingqi Liu et al. "Trojaning attack on neural networks". In: *Purdue University Libraries e-Pubs* (2017).

[220] OWASP. "TOP 10 2017". In: *The Ten Most Critical Web Application Security Risks. Release Candidate* 2 (2018).

[221] Patrick Hall. *Proposals for model vulnerability and security*. 2019. URL: `https://www.oreilly.com/ideas/proposals-for-model-vulnerability-and-security`.

[222] Cynthia Dwork. "Differential privacy". In: *Encyclopedia of Cryptography and Security* (2011), pp. 338–340.

[223] Chris Waites. *PyVacy: Privacy Algorithms for PyTorch*. 2019. URL: `https://pypi.org/project/pyvacy/`.

[224] MAP Chamikara et al. "Local differential privacy for deep learning". In: *arXiv preprint arXiv:1908.02997* (2019).

[225] Yehida Lindell. "Secure multiparty computation for privacy preserving data mining". In: *Encyclopedia of Data Warehousing and Mining*. IGI Global, 2005, pp. 1005–1009.

[226] D Hardman. *Message Trust Contexts*. RFC 29. Hyperledger, May 2019. URL: `https://github.com/hyperledger/aries-rfcs/tree/master/concepts/0029-message-trust-contexts`.

[227] Ruth Gavison. "Privacy and the Limits of Law". In: *The Yale law journal* 89.3 (1980), pp. 421–471.

[228] Alan F Westin. "Privacy and freedom". In: *Washington and Lee Law Review* 25.1 (1968), p. 166.

[229] Ronald Cramer, Ivan Bjerre Damgård, et al. *Secure multiparty computation*. Cambridge University Press, 2015.

[230] Jung Hee Cheon et al. "Homomorphic encryption for arithmetic of approximate numbers". In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2017, pp. 409–437.

[231] Whitfield Diffie and Martin Hellman. "New directions in cryptography". In: *IEEE transactions on Information Theory* 22.6 (1976), pp. 644–654.

[232] Jan Camenisch and Anna Lysyanskaya. "An efficient system for non-transferable anonymous credentials with optional anonymity revocation". In: *International conference on the theory and applications of cryptographic techniques*. Springer. 2001, pp. 93–118.

[233] Facebook Differential Privacy. *Opacus Differential Privacy*. URL: `https://github.com/pytorch/opacus`.

[234] *Microsoft SEAL (release 3.6)*. `https://github.com/Microsoft/SEAL`. Microsoft Research, Redmond, WA. Nov. 2020.

[235] *TenSEAL (release 0.3.0)*. `https://github.com/OpenMined/TenSEAL`. Feb. 2021.

[236] Donald Beaver. "Efficient Multiparty Protocols Using Circuit Randomization". In: vol. 576. Aug. 1991, pp. 420–432. ISBN: 978-3-540-55188-1. DOI: `10.1007/3-540-46766-1_34`.

[237] Brian Knott et al. *CRYPTEN: Secure Multi-Party Computation Meets Machine Learning*. 2021. URL: `https://crypten.ai/` (visited on 02/18/2021).

[238] Will Abramson, Adam Hall, Lohan Spies, Tom Titcombe. *PyDentity*. 2021. URL: `https://github.com/OpenMined/PyDentity`.

[239] Alan Aboudib. *Syfertext*. 2021. URL: https://github.com/OpenMined/Syfertext.

[240] Yann LeCun, Corinna Cortes, Christopher J.C. Burges. *THE MNIST DATABASE of handwritten digits*. 2019. URL: http://yann.lecun.com/exdb/mnist/.

[241] Jung Hee Cheon, Seungwan Hong, and Duhyeong Kim. "Remark on the Security of CKKS Scheme in Practice". In: *IACR Cryptol. ePrint Arch* 2020 (2020), p. 1581.

[242] Daniel Takabi et al. "Privacy preserving Neural Network Inference on Encrypted Data with GPUs". In: *CoRR* abs/1911.11377 (2019). arXiv: 1911.11377. URL: http://arxiv.org/abs/1911.11377.

[243] Praneeth Vepakomma et al. *No Peek: A Survey of private distributed deep learning*. 2018. arXiv: 1812.03288 [cs.LG].

[244] Ionesio Junior, Patrick Cason. *PyGrid*. 2021. URL: https://github.com/OpenMined/PyGrid.

1.5