

PRECLUDE: PRIVACY-PRESERVING COLLABORATIVE LEARNING  
USING A DECENTRALIZED ENSEMBLE APPROACH

LARS R.B. VAN DE KAMP

to obtain a Master of Science in Computer Science  
Data Science & Technology  
with a Cyber Security specialization  
to be defended publicly on August 23, 2018

[Delft University of Technology](#)

Faculty of Electrical Engineering, Mathematics & Computer Science  
Intelligent Systems, Cyber Security Group



Lars R.B. van de Kamp: *PRECLUDE: PRivacy-prEserving Collaborative Learning Using a Decentralized Ensemble approach*, Master of Science, © August 2018

STUDENT NUMBER:

4501829

COMMITTEE MEMBERS:

Prof.dr.ir. Inald Lagendijk

Dr. Ir. Cynthia Liem

Dr. David Tax

Dr. Zekeriya Erkin

Chibuike Ugwuoke, MSc

SUPERVISORS:

Dr. Zekeriya Erkin

Chibuike Ugwuoke, MSc

## ABSTRACT

---

Machine learning techniques receive significant responsibilities, despite growing privacy concerns. Early-stage autonomous vehicles are increasingly appearing on the streets, carrying the burden of transporting human-lives to their destination. Meanwhile, doctors are involving Artificial Intelligence (AI) in their medical diagnoses, basing treatment of patients on the analyses AI provides. For these services to reach their full potential, a vast amount of training data is required, often gathered from a variety of sources. In many cases, the required data is considered to be privacy-sensitive (e.g., medical data). Due to the sensitivity of the underlying information, many individuals and organizations are not willing to entrust its protection to another party.

A field that attempts to limit the need to transfer training data openly is called collaborative learning, where multiple data generators cooperate to jointly train a classifier. In the proposed techniques the participants aim to limit the privacy loss of their collected training data to other collaborators. We contribute a clear overview of the current state-of-the-art and identify its limitations. Based on these limitations, we present two innovative protocol designs that pave the way towards private collaborative learning.

The ECONoMy protocol is developed to suit the needs of a high participant use case (i.e., Internet of Things (IoT)), under an assumed semi-honest adversarial model. The experimental results show that ECONoMy offers the desired privacy properties while remaining competitive to the non-privacy preserving alternative with which it is compared. However, in certain environments the incentives can grow exceedingly large rendering the 'semi-honest' adversary assumption impractical.

We, therefore, created the PRECLUDE protocol which uses traceable ring signatures to protect against adversaries in the covert adversarial model. The tracing capability allows to detect malpractice and leak the identity of the deviant while preserving the anonymity of honest participants. These additional privacy-preservation properties came at a high cost to the overall efficiency, which is what we aimed to reduce by designing our extended protocol called PRECLUDE<sup>+</sup>.

PRECLUDE<sup>+</sup> manages to drastically improve efficiency by reducing the number of participants included in a single signature. Further, we created a batch-verification phase that allows us to omit several exponentiations in each execution. We provide a detailed statistical analysis showing how to balance the efficiency improvements, with the required privacy parameters. The protocols presented in this thesis significantly improve upon the privacy guarantees offered by current alternatives, and provide a clear direction in which future work can continue to build.



*Everything should be made as simple as possible  
but not any simpler*

— **Albert Einstein** [63]

## ACKNOWLEDGMENTS

---

During my Bachelor's degree in international business, I never anticipated to perform such extensive research in computer science, let alone combining the fields of cryptography and machine learning. My time at the Delft University of Technology has been an exciting journey starting at the basis of computer science during my bridging year, followed by an extensive deep-dive in data-science and cyber security in the Master's. This journey, filled with ups and downs, would not have been as fulfilling without my close friends who were with me in the trenches. Together we have managed to overcome the obstacles in front of us and help each other in times of need.

I would like to thank my thesis supervisors Zeki and Chibuike for their help in guiding me through the last nine months. They have helped me to navigate the tremendously massive search-space of possibilities within the research problem, by providing relevant literature and critical feedback. Furthermore, I would like to thank the entire research group including Gamze and Oguzhan for making the experience more enjoyable.

Additionally, I would like to express my appreciation to the members of my thesis committee. Thank you Inald, David, and Cynthia, for the time and effort required to review my work and attend my presentation.

The thesis process has fueled a period of self-awareness and accompanying growth. I want to thank both Zeki and Cynthia for taking time to help me not only in the academic process but also at a personal level. The focus you have on mental well-being and personal development during the academic process is extremely valuable and has added an additional steep learning curve to the overall process.

Most importantly, I would like to thank all my friends and family members who have supported me during the Master's, I am honored to have you in my life. I would like to extend my gratitude to Hugo, Hari, Victor, Kevin, Jochem, and Tim for their efforts during the creation of this work. A special thanks to Nada, Sven, and my parents for providing the underlying support system I frequently needed. Without the help of all these people, especially in the moments where the end seemed endlessly far away, I would not have been able to finish the work that lays in front of you here today.



# CONTENTS

---

ABSTRACT	v
ACKNOWLEDGEMENTS	vii
TABLE OF CONTENTS	ix
LIST OF FIGURES	xii
LIST OF TABLES	xiv
LIST OF ALGORITHMS	xv
ACRONYMS	xvi
<b>I INTRODUCING THE PROBLEM</b>	<b>1</b>
1 INTRODUCTION	3
1.1 Machine Learning . . . . .	3
1.2 Privacy Awareness . . . . .	4
1.3 Collaborative Learning . . . . .	6
1.4 Adversary Objectives . . . . .	8
1.5 Research Statement . . . . .	8
1.6 Our Contributions . . . . .	9
1.7 Outline . . . . .	10
<b>II SETTING THE SCENE</b>	<b>11</b>
2 PRIMITIVES	13
2.1 Cryptographic primitives . . . . .	13
2.1.1 Cryptographic Encryption Schemes . . . . .	13
2.1.2 Zero-Knowledge Proofs . . . . .	13
2.1.3 Digital Signatures . . . . .	14
2.1.4 Cryptographic hashing . . . . .	15
2.1.5 Differential privacy . . . . .	16
2.1.6 Adversarial models . . . . .	17
2.2 Machine learning . . . . .	18
2.2.1 Ensemble Learning . . . . .	18
2.2.2 Collaborative Learning . . . . .	21
2.2.3 Adversarial Machine Learning . . . . .	21
2.3 Immutable Distributed Ledgers . . . . .	24
3 COLLABORATIVE MACHINE LEARNING	27
3.1 Contemporary approaches for privacy preservation in collaborative learning . . . . .	27
3.1.1 Centralized (Iterative) Parameter Aggregation . . . . .	28
3.1.2 Centralized Ensemble Model Aggregation . . . . .	32

3.1.3	Decentralized Iterative Parameter Aggregation . . . . .	34
3.1.4	Decentralized Ensemble Model Aggregation . . . . .	35
3.1.5	Adapting Existing Learning Algorithms . . . . .	36
3.2	Creating an Overview . . . . .	38
3.3	Discussion . . . . .	42
3.4	Key Takeaways . . . . .	43
4	RESEARCH CHALLENGES AND METHODOLOGY . . . . .	45
4.1	Application setting . . . . .	45
4.1.1	Assumptions . . . . .	45
4.1.2	How does our setting relate to previous work? . . . . .	46
4.2	Methodology . . . . .	47
4.2.1	Sub-questions . . . . .	48
4.2.2	Design science . . . . .	49
4.3	Challenges . . . . .	50
III	DESIGNING THE SOLUTIONS . . . . .	53
5	ECONOMY: ENSEMBLE COLLABORATIVE LEARNING USING MASKING . . . . .	55
5.1	Notation . . . . .	56
5.2	Detailed Primitives . . . . .	57
5.3	Protocol Overview . . . . .	59
5.4	Phase-by-phase Design Description . . . . .	61
5.4.1	Initial setup . . . . .	61
5.4.2	Random number generation . . . . .	62
5.4.3	Masking . . . . .	62
5.4.4	Aggregation . . . . .	62
5.4.5	Noise addition and final model generation . . . . .	64
5.5	Security Analysis . . . . .	64
5.6	Complexity Analyses . . . . .	69
5.6.1	Computational analysis . . . . .	69
5.6.2	Communication analysis . . . . .	69
5.7	Discussion . . . . .	70
6	PRELUDE: PRIVACY-PRESERVING COLLABORATIVE LEARNING USING A DECENTRALIZED ENSEMBLE APPROACH . . . . .	73
6.1	Notation . . . . .	74
6.2	Cryptographic Preliminaries . . . . .	74
6.3	Protocol Overview . . . . .	78
6.4	Phase-by-phase Design Description . . . . .	79
6.4.1	Initial setup . . . . .	79
6.4.2	Collaborative voting phase . . . . .	80
6.4.3	Verification and Tracing phase . . . . .	81
6.4.4	Aggregation phase . . . . .	81
6.4.5	Noise addition and final model generation . . . . .	82
6.5	Security Analysis . . . . .	82
6.6	Complexity analyses . . . . .	84
6.6.1	Computational analysis . . . . .	84



6.6.2	Communication analysis . . . . .	84
6.7	Discussion . . . . .	85
7	EXTENSIONS . . . . .	87
7.1	Ring Size Reduction . . . . .	87
7.1.1	Research challenges . . . . .	87
7.1.2	Protocol changes . . . . .	88
7.1.3	Factors dependent on $t$ . . . . .	90
7.1.4	Determining the size of $t$ . . . . .	92
7.2	Batch Verification . . . . .	95
7.2.1	Proposed alterations . . . . .	96
7.2.2	Performance improvement . . . . .	96
IV	EVALUATING THE PROTOCOLS AND DEFINING FUTURE WORK . . . . .	99
8	EVALUATION . . . . .	101
8.1	The Experiment Setting . . . . .	101
8.1.1	The code . . . . .	101
8.1.2	The used hardware . . . . .	102
8.2	The Alternative Approaches . . . . .	102
8.2.1	CrowdML . . . . .	102
8.2.2	AnonML . . . . .	103
8.3	Run-time Analyses . . . . .	105
8.3.1	ECONoMy . . . . .	105
8.3.2	PRECLUDE . . . . .	107
9	DISCUSSION AND FUTURE WORK . . . . .	113
9.1	Discussion . . . . .	113
9.2	Future Work . . . . .	115
9.3	Conclusion . . . . .	118
	BIBLIOGRAPHY . . . . .	121

## LIST OF FIGURES

---

Figure 1	An overview of the available limitations on data export across borders as given by Nigel Cory [24]. Note that this is before the inauguration of the GDPR.	5
Figure 2	An example where the samples encountered by two participants originate from different subsections of the population distribution. . . . .	7
Figure 3	A visualization of the cooperative nature of collaborative learning . . . . .	21
Figure 4	An example given by Goodfellow et al. [48], where an Imagenet picture of a panda combined with noise generated by the Fast Gradient Sign method to misclassify the Panda as a Gibbon with a high confidence level. . . . .	23
Figure 5	A list of all symbols used within the figures provided for each category resembling the components used. . .	28
Figure 6	An overview of a typical system structure of the systems belonging to the <a href="#">CIPA</a> category. . . . .	29
Figure 7	An overview of a typical system structure of the systems belonging to the <a href="#">CEMA</a> category. . . . .	33
Figure 8	An overview of a typical system structure of the systems belonging to the <a href="#">DIPA</a> category. . . . .	34
Figure 9	An overview of a typical system structure of the systems belonging to the <a href="#">DEMA</a> category. . . . .	35
Figure 10	The generic structure proposed by Papernot et al. [94] to improve privacy-preservation of a publicly released model generated by a single party. . . . .	46
Figure 11	The generic structure proposed by Hamm et al. [53] to improve privacy-preservation of a publicly released model generated by multiple parties in a collaborative setting, part of the <a href="#">CEMA</a> category. . . . .	47
Figure 12	The generic structure proposed in this thesis to improve privacy-preservation of a publicly released model generated by multiple parties in a collaborative setting, will be part of the <a href="#">DEMA</a> category. . . . .	48
Figure 13	Example showing the amount of iterations for which a participant was included in less than $k$ rings, for $t=[5,25]$ .	95
Figure 14	An overview showing how the distribution of signatures a participant is included in changes for $k=3$ , $n=50$ , $m=100$ , $\check{p} = 1 \cdot 10^{-50}$ , $\check{\epsilon} = 1 \cdot 10^{-6}$ . $t$ is set to 18, 19, or 20. . . . .	98

Figure 15	Comparing the per participant computation time of ECONoMy with CrowdML. . . . .	106
Figure 16	Highlighting the dependency on $m$ inherent to CrowdML, which is not as present in ECONoMy. . . . .	106
Figure 17	The efficiency of a protocol execution of PRECLUDE and PRECLUDE <sup>+</sup> for varying $m$ and $n$ . . . . .	108
Figure 18	The size of a single signature in the experiments for PRECLUDE and PRECLUDE <sup>+</sup> for varying $m$ and $n$ . . . . .	109
Figure 19	Trend comparison of PRECLUDE <sup>+</sup> and AnonML based on a single verification per item. . . . .	110

## LIST OF TABLES

---

Table 1	An overview of combination methods for ensemble learning as discussed by Lior Rokach [108] . . . . .	20
Table 2	The design decisions encountered in the considered papers, and their respective description . . . . .	38
Table 3	Overview of the discussed papers and their approach according to the provided criteria. . . . .	39
Table 4	An overview of the reviewed work which provides high-level insights into the techniques used in each approach. . . . .	40
Table 5	Application of the seven design guidelines of design science as presented by Hevner et al. [56] . . . . .	50
Table 6	Explanation of the symbols used in the ECONoMy design chapter. . . . .	56
Table 7	Review of all operations occurring in a single execution of the ECONoMy protocol to conclude the computation complexity. . . . .	70
Table 8	An overview of all communications needed for ECONoMy, and the size of the transferred information. . . . .	70
Table 9	Explanation of the additional symbols used in PRECLUDE. . . . .	74
Table 10	Intuition behind the need for $v = n - 1$ . . . . .	84
Table 11	An overview of the different types of operations encountered in the overall protocol described above. Amount shown both in its totality, per party, and per execution of the step. . . . .	85
Table 12	An overview of the communication in bits between participants. . . . .	85
Table 13	The improvements to the theoretical computational analysis provided by reducing the ring size. . . . .	92
Table 14	Example probabilities of retrieving complete sets of votes for low values of $k$ and $m$ . . . . .	93
Table 15	Variable values for determining $t$ in an example scenario. . . . .	94
Table 16	The experimental results obtained by running both CrowdML and ECONoMy. . . . .	107
Table 17	Comparison of the time needed to cast and verify $nm$ votes for PRECLUDE <sup>+</sup> and AnonML. . . . .	111
Table 18	Run-time experiment results comparing PRECLUDE, PRECLUDE <sup>+</sup> , and AnonML in their execution timings for different $n$ and $m$ . . . . .	111

## LIST OF ALGORITHMS

---

Algorithm 1	Random number generation protocol. . . . .	63
Algorithm 2	Masking procedure, hiding the prediction values. . . .	63
Algorithm 3	Communication of a participant $\pi_i$ with the protocol $\Phi$	65
Algorithm 4	Communication $S_i$ with the ideal functionality $f$ . . . .	66
Algorithm 5	The procedure highlighting the required steps to cast a signed vote. . . . .	80
Algorithm 6	The procedure used to verify and trace signatures. . . .	81
Algorithm 7	Communication between a party $\pi_i$ and the rest of the protocol $\Phi$ . . . . .	83
Algorithm 8	The generation of a random set of selected participants.	88
Algorithm 9	Vote signing and casting procedure based on a subset of $t$ parties. . . . .	89
Algorithm 10	Optimizing $t$ according to the set parameters $k$ and $\epsilon$ . .	93
Algorithm 11	Batch verification and tracing algorithm. . . . .	96

## ACRONYMS

---

AELA	Adapting Existing Learning Algorithms
AES	Advanced Encryption Standard
AML	Adversarial Machine Learning
AWS	Amazon Web Services
AI	Artificial Intelligence
CEMA	Centralized Ensemble Model Aggregation
CIPA	Centralized (Iterative) Parameter Aggregation
DEMA	Decentralized Ensemble Model Aggregation
DIPA	Decentralized (Iterative) Parameter Aggregation
DP	Differential Privacy
EU	European Union
GDPR	General Data Protection Regulation
GAN	Generative Adversarial Network
HE	Homomorphic Encryption
ITIF	Information Technology & Innovation Foundation
IoT	Internet of Things
MPC	Multiparty Computation
PATE	Private Aggregation of Teacher Ensembles
TRS	Traceable Ring Signatures
ZKP	Zero-Knowledge proof

## Part I

### INTRODUCING THE PROBLEM

The work presented in this thesis is divided into four different parts. First, we will introduce the research problem to acquaint the reader with the identified problem. In the second part, we will set the scene providing relevant background knowledge as well as the applied methodology. Afterward, we introduce the proposed solutions to the presented problem statement. In the final part of this thesis, we evaluate the presented work and identify open problems for future work.





## INTRODUCTION

---

The impact that AI will have on the future of society is the subject of intense debate [18, 130]. Organizations and their users want to benefit from improved service, offering higher utility and increased productivity, while recent data leaks incite a widespread awareness of privacy, as they highlight the risks accompanying the collection of large amounts of data needed to provide the desired benefits.

Despite growing concerns, the underlying techniques receive significant responsibilities. Early-stage autonomous vehicles are increasingly appearing on the streets, carrying the burden of transporting human-lives to their destination. Meanwhile, doctors are involving AI in their medical diagnoses basing treatment of patients on the analyses AI provides [12, 23, 69].

For these services to reach their full potential, a vast amount of training data is required, often gathered from a variety of sources. In many cases, the required data is considered to be privacy-sensitive (e.g., medical data). Due to the sensitivity of the underlying information, many individuals or organizations are not willing to entrust its protection to another party.

A field that attempts to limit the need to openly transfer training data is called collaborative learning, where multiple data generators cooperate to jointly generate a model. In the proposed techniques the participants aim to limit the privacy loss of their collected training data to other collaborators.

Although these efforts have improved the situation, new attacks can extract sensitive information from the used derivatives of the training data (e.g., the classifier). To continue to use data to provide us with the benefits AI has to offer, novel, privacy-preserving training techniques are essential. Within this thesis, we present two innovative protocol designs that pave the way towards private collaborative learning.

### 1.1 MACHINE LEARNING

Within the general concept of AI, machine learning has emerged as the method of choice for developing techniques such as computer vision, speech recognition, natural language processing, and robot control [66]. Machine learning can be defined as a field of study focusing on algorithms that can learn from and make predictions based on data [46]. The training process takes in example inputs to generate a model which aims to improve a predefined performance measure. For example, by training a particular machine learning algorithm on examples of both routine and fraudulent transactions, the resulting model can prevent future malicious transactions by flagging these transactions as such (i.e., [81], [82]).

Both conglomerates and private investors are increasingly moving into the machine learning industry to strengthen their competitive advantage for

years to come. McKinsey & Company estimates that in 2016, \$20-30 billion worth of AI investments originated from Google and Baidu, while the amount invested by private equity and venture capital totaled up to \$5-8 billion [23]. Nevertheless, privacy is often not considered a main priority, as shareholder interest trumps societal impact due to the fiduciary duty of those who are running the companies. On the other hand, there are initiatives such as the non-profit OpenAI (which received \$1 billion in funding), which focus on benefiting humanity as a whole, without any financial obligations [51], and accompanying expectations.

Due to the sharp increase in available resources originating from this increase in investments, many early-stage techniques are encountered in everyday life already. Innovations like the concept of a virtual assistant, early stage autonomous vehicles, and improved recommender systems are becoming ingrained into our daily lives. A good example is Netflix, a major entertainment company with over 117 million subscribers as reported in the fourth quarter of 2017 [91]. Netflix states that they prevent over \$1 billion of canceled subscriptions due to improved recommender systems every single year [47], benefiting the different stakeholders of the firm (e.g., shareholders, end-customers).

## 1.2 PRIVACY AWARENESS

Society is becoming more aware of the privacy risks associated with their digital presence [62]. Two primary drivers constitute this trend. First of all, numerous data breaches are highlighting the sensitivity of centralized solutions. Secondly, there is an increasing amount of legislation focusing on protecting one's privacy, forcing organizations and individuals to be aware of their rights and duties.

Recent scandals such as that of Cambridge Analytica, where private information was used to influence political preference [52], fuel awareness on both the value and inherent risk of the data generated and shared with third parties. Despite growing concerns, private data (e.g. bank records, photos, and videos) is still collected by different organizations that retain this sensitive information indefinitely or for the legal maximum allowed term.

Despite legislation lagging behind technological advances [113], privacy has a high priority for governments around the world. In 2017, the Information Technology & Innovation Foundation (ITIF) published research on the state of privacy regulation on data transfer across borders [24]. Nigel Cory, the writer of the published report, hypothesizes that the costs of the increase in legislation could exceed \$100 billion dollars for the European Union alone, under the given assumptions. Figure 1 shows which countries block what type of data at the time of publishing (2017). Note that this does not yet include the new legislation devised by the European Union European Union (EU), the general data protection regulation General Data Protection Regulation (GDPR), which came into effect in May 2018 [121].

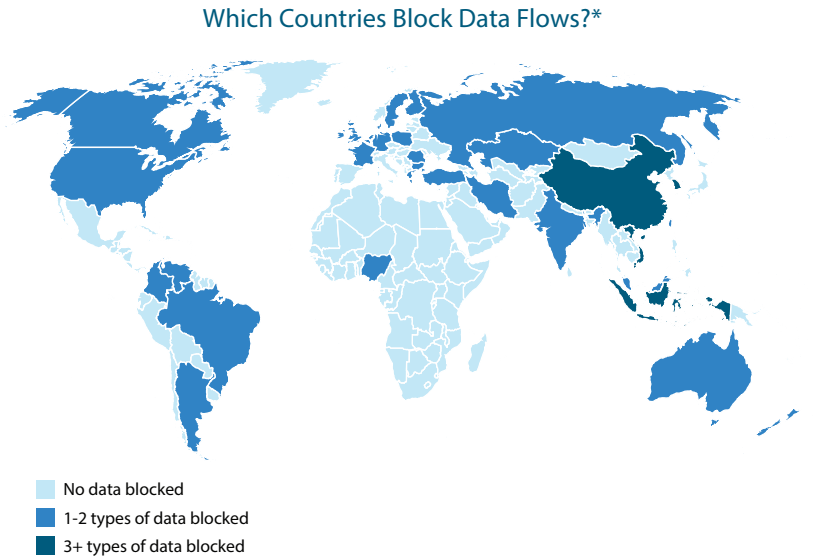


Figure 1: An overview of the available limitations on data export across borders as given by Nigel Cory [24]. Note that this is before the inauguration of the GDPR.

There are exorbitant costs associated with limitations to data usage as well as indirect costs due to involvement in a data breach. Therefore, there is a drive to develop novel techniques that allow organizations to continue their value-adding practices, while continuously adhering to current and future legislation. The creation of such techniques requires a multi-disciplinary approach to machine learning research, introducing relevant cryptographic primitives into protocols while keeping the performance of the final model a priority.

By having privacy-sensitive information collected at a central entity, a third party, the system becomes susceptible to several risks (i.e., the central entity becomes a single point of failure), which are not always apparent to those who will be affected when it is compromised. Lengthy, obfuscated user agreements that indicate what the collected data may be used for are often not read. A report published by the cybersecurity alliance states that only around 5% of Americans do read privacy policies [62]. Even if one reads the fine print, it often does not reflect the level of security used to protect these data sources, which is a critical piece of information required to evaluate a service [62]. Moreover, in certain parts of the world, the rule of law allows access to centrally stored information for surveillance during government investigation [83]. If an individual party retains his or her information, as opposed to transferring training samples to a central entity, these risks are removed or significantly reduced.

As mentioned, in collaborative learning collaborators share derivatives of training data, to achieve this. Nevertheless, due to newly introduced attacks, a potential adversary can still obtain private information from such derivatives. We aim to create a way to generate high-performing models that utilize multiple distributed data sources that belong to a variety of

participants, without compromising the privacy of the parties providing the data. For example, such a protocol could be relevant for institutes operating within the healthcare and finance industries which handle sensitive information, located at a variety of locations, alleviating their privacy concerns.

### 1.3 COLLABORATIVE LEARNING

Collaborative learning offers a solution to train a model based on such distributed data sources while adhering to relevant laws that aim to protect privacy. Whereas in a traditional machine learning setting different data sources transfer their collected training samples to a central location, collaborative learning eliminates the need for a trusted entity to protect the privacy of the individual data owners by safeguarding the actual samples. Instead, different techniques based on iterative or ensemble-based model aggregations are used to combine the underlying knowledge that is present within the training data. This distributed nature is useful for entities that prefer to retain control of their data or when regulation limits how data can be transferred among organizations and across country boundaries. Nevertheless, information derived from the training samples is in some cases still shared with a central party to coordinate the aggregation of the provided information. Removing the need to trust a central entity all-together can be accomplished by decentralizing a protocol, thereby distributing trust over multiple parties, as is currently being done by some researchers within the field of collaborative learning [3, 9].

**WHY LEARN COLLABORATIVELY?** Collaborative learning can offer different benefits as opposed to a more traditional machine learning setting. Firstly, the added value obtained by collaborating can originate from the number of available examples per participant, or from the subset of the entire population covered by a particular group of participants. Figure 2 shows how two example participants can complement each other to generate a better depiction of the entire population distribution by collaborating. The increased diversity in training data can help to produce more robust classifiers.

Secondly, by sharing abstractions from the original data, one can adhere to current privacy legislation. For example, the previously mentioned [GDPR](#) directive aims to protect the data that belongs to any European citizen, as it covers any data that directly, or indirectly identifies a natural person [17]. It applies to companies (both those that are responsible for the underlying data and those processing the data) located in the EU, even if data processing occurs outside of the EU. It is assumed that by only transferring data derivatives as an organization and allowing clients to contribute derivatives directly, an organization can limit the legal exposure as the information is no longer assumed to be personally identifiable.

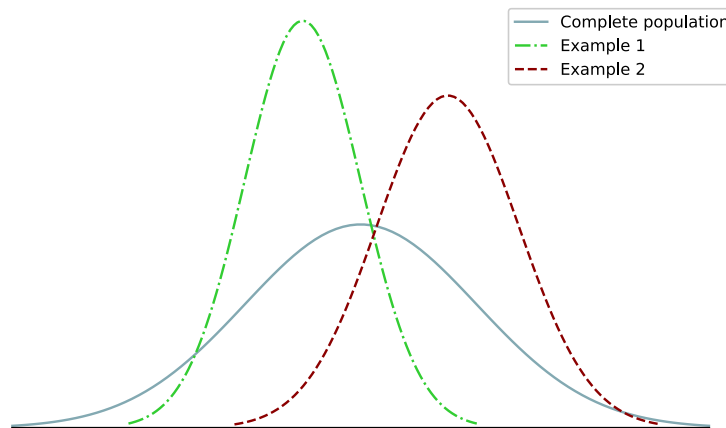


Figure 2: An example where the samples encountered by two participants originate from different subsections of the population distribution.

**USE CASES** Collaborative learning can deliver an anonymized solution to train a classifier on personal information while exporting only derivatives extracted based on the structure of the used training samples. By using such a protocol, rather than sharing training samples to a central location for training, one can obtain a robust model that represents the sample population across multiple nations without having to share identifiable information. Such an infrastructure can help multinational companies to generate robust, international models, within their organizational structure in a privacy-preserving manner.

Another possible use case would be a consortium of data providers who want to cooperate in generating a machine learning model. Such a group could have a peer-to-peer nature, where individuals would prefer to cooperatively train a particular model rather than having a central party, or a cooperative nature, where multiple organizations in the same industry cooperate to jointly provide an improved service without assigning full responsibility to a single party. For example, competing financial institutions could collaborate to jointly improve their fraud detection capabilities.

Finally, the rise of IoT devices being used to collect data offers a chance to see individual devices as participants in a collaborative setting. In such a way, the device would not need to transfer actual training samples, preserving the privacy of the environment in which it is located.

Due to an increase in attacks that can retrieve privacy-sensitive information from the shared derivatives, the protocol as a whole should have privacy as its main priority. The currently available approaches, especially in the decentralized categories, are lacking in privacy preservation, raising significant privacy issues. We will discuss these concerns in the next section.

#### 1.4 ADVERSARY OBJECTIVES

Even though the field of collaborative learning seems promising to help build a more privacy-aware future for machine learning, the currently available approaches that are reviewed for this thesis are lacking. Many of the proposed solutions either trust a central entity, or approach the decentralization in such a fashion that they do not consider the classifiers, the model predictions, nor the parameter updates as private information.

Recent advances in the field of adversarial machine learning show that an adversary can not only willingly cause a machine learning model to misinterpret its input [49, 70], he or she can also extract information that can be considered privacy-sensitive [57, 110] using only the abstractions of the training samples. An adversary who can access this type of information could commence in attacks that allow training data extraction [41] or membership inference [110]. Notably, this can be done in the 'semi-honest' threat model, as it does not pose any deviation from the prescribed protocols, and can be done purely by observing the transferred information. If an adversary successfully conducts these attacks, he or she can obtain insights into the privacy-sensitive training data used by a participant.

Another possible objective of an adversary could have in a decentralized setting would relate to their influence on the resulting global machine learning model. An adversary might be motivated to reduce the prediction capabilities of its peers rather than helping them improve. Such influence can be exerted by contributing intentionally faulty votes. Unfortunately, it is not evident whether the model did, in fact, consider that label to be most likely or the originating sender maliciously manipulated the message. What can be done, however, is limiting the ability of an adversary to contribute multiple times. Do note that when assuming that adversaries do in fact completely follow the protocol this does not directly apply.

Thus, to facilitate decentralized executions of collaborative learning, a privacy-preserving approach is needed that hides the required information and limits the number of allowed contributions, making the underlying attacks more difficult, if not infeasible to execute successfully. By doing so, the privacy of the training data used by the different participants is fortified, while still being able to generate globally shared machine learning models. Creating a method to do so, and validating its correctness together with a complexity analysis, are the primary research goals of this thesis.

#### 1.5 RESEARCH STATEMENT

As introduced in the previous paragraph, there are two primary adversary objectives which we aim to defend against.

We desire to provide execution verifiability, allowing participants to verify that the protocol steps have been executed correctly, whereas consensus is an agreement on the state of the shared transactions among participants. In addition, the individual participants do not want others to gain insights into the contributions they provide, and therefore it should be privacy-preserving.

Meanwhile, it should not be possible for a participant to wilfully degrade the global model's performance in excess of their own allocated share ( $1/n$ ). Every participant, who is a valid contributor to the protocol, should be entitled to a pre-defined number of votes. An adversary can degrade the performance of the final model by actively providing wrong votes, thereby transferring false information to the global model. While it is currently not possible to verify whether or not a particular vote originates from a hidden model, we can limit the impact of a potential adversary by preventing excessive contributions.

The presented research attempts to provide a novel collaborative learning system that provides these requirements while limiting the effect on prediction performance of the model. The research statement is as follows:

*How can we facilitate the joint generation of a shared machine learning model in a privacy-preserving manner, and refrain participants from degrading the final models' performance more than their own, allowed contribution, in a decentralized setting?*

Based on this research statement, the following sub-questions can be derived:

1. How can we create a transparent, decentralized joint machine learning model generation system?
2. How can we limit the number of contributions of a participant to a pre-specified amount?
3. How can we remove the linkability of their contributions to the originating user, while assuring a valid sender?
4. How can we leverage the setting of the protocol to improve efficiency?
5. How can the above sub-questions be achieved without significantly degrading the prediction accuracy as compared to centralized variants?

## 1.6 OUR CONTRIBUTIONS

In this thesis, we propose two protocols to preserve individual participants privacy in a collaborative learning setting. Both protocols utilize different techniques to provide privacy under their security assumptions. The protocols presented in this thesis are, to the best of our knowledge, the first to provide a privacy-preserving approach to decentralized, ensemble-based collaborative learning. The efficiency of the proposed protocols is determined, and the prediction performance compared to its alternatives, employing a naive implementation. The main contributions are as follows:

- A detailed overview of the work done in the field of collaborative learning, provided with its advantages and disadvantages.

- A masking based protocol that preserves participant privacy within a decentralized ensemble based collaborative learning setting, called ECONoMy, under a semi-honest threat model.
- A protocol that uses traceable ring signatures to preserve participant privacy within a decentralized ensemble based collaborative learning setting, called PRECLUDE, under a covert threat model.
- Problem specific efficiency improvements to the traceable ring signatures protocol, combined with a detailed analysis on how to determine the optimal values for the privacy-efficiency trade-off.

### 1.7 OUTLINE

The accrued information leading to these contributions, and ultimately the answer to the research question, will be introduced according to the following structure. Chapter 2 gives an overview of the important preliminaries which the following chapters will build upon. In Chapter 3, the current work in collaborative learning is discussed with both its strengths and weaknesses. Chapter 4 sets the foundation of the methodology that will be used to approach the research problem, where Chapters 5 and 6 introduce the protocol designs that were built on top of this. Chapter 7 proposes extensions that improve upon the base PRECLUDE protocol, whereas Chapter 8 evaluates and compares the performance of the models using naive implementations. Finally, Chapter 9 exposes the limitations and highlights the identified open problems and relevant future work.



## Part II

### SETTING THE SCENE

Now that we have introduced the problem and have explained our research goal, we can continue to exhibit all relevant context elements for this thesis. Here we guide the reader through necessary primitives to ensure an equal footing before presenting any technical terminology. Afterward, we will go through all reviewed related works and identify their limitations. Using these, we identify which area presents the most versatile opportunities for growth, which we will focus on when creating our designs. Finally, we will go through what methodology will be used to answer our research questions. Here we emphasize the use of design science and the challenges that are inherent to our problem scenario.



## PRIMITIVES

---

The work done to overcome privacy issues within the field of decentralized collaborative learning, as well as our work presented in this thesis, is based on a variety of concepts. In this chapter, we introduce the used primitives, which form a foundation on top of which the following chapters will continue to build. First, we discuss the relevant cryptographic primitives, after which we elaborate upon the applicable machine learning concepts. Finally, we explain blockchain technology and the desired properties for the field of collaborative learning.

### 2.1 CRYPTOGRAPHIC PRIMITIVES

The cryptographic concepts form the foundation of the security and privacy guarantees that can be made by a collaborative learning protocol. In this chapter, we introduce primitives relevant to the related work done in collaborative learning. Those primitives that are specific to our contributions will be presented in more detail in their respective design chapters.

#### 2.1.1 *Cryptographic Encryption Schemes*

The two types of cryptosystems will be addressed as primitives to this thesis: symmetric and asymmetric cryptosystems.

**SYMMETRIC CRYPTOSYSTEMS** A symmetric cryptosystem is dependent on a shared secret between multiple participating parties, the symmetric key. Encryption and decryption can be done using the same key  $k$ , where  $c = \mathcal{E}(k, m)$  and  $m = \mathcal{D}(k, c)$ . The mapping performed to retrieve the cipher text using the key and message should thus be invertible. Notable examples of symmetric cryptosystems include the one-time pad [10, 32], and the Advanced Encryption Standard (AES) [27] as is used in the first design proposed in this thesis.

**ASYMMETRIC CRYPTOSYSTEMS** Diffie and Hellman [32] introduced asymmetric cryptography. Within this strand of cryptography, each party holds both a public key  $pk_i$ , and secret key  $sk_i$ , where the first is used for encryption,  $\mathcal{E}_{pk_i}(m)$ , and the latter for decryption of the ciphertext  $c$ ,  $\mathcal{D}_{sk_i}(c)$ . This thesis utilizes this form of encryption in both its proposed protocols, where the public keys of all participants are known, and each party retains his or her secret to decrypt or sign information.

#### 2.1.2 *Zero-Knowledge Proofs*

The concept of a zero-knowledge proof, especially the more detailed descriptions that are given, is required to understand a crucial part of our

second protocol design, which employs these zero-knowledge proofs in its traceable ring signatures.

A Zero-Knowledge proof (ZKP), as presented by Fiege et al. [37], is the ability to construct a proof of knowledge of specific information, without disclosing this information. It consists of a series of message exchanges, typically dependent on random values to probabilistically prove a particular statement [85]. Rackoff and Simon [103] continued upon this work by removing the need for interaction between the prover and the verifier. A ZKP must adhere to three requirements: 1. completeness, if the statement is true, an honest verifier can be convinced, 2. soundness, it is not possible to convince the honest verifier of the correctness of a false statement, and 3. zero-knowledge, the verifier learns nothing about the statement despite its validity [85].

**WITNESS-INDISTINGUISHABLE INTERACTIVE PROOFS** Cramer et al. [25] have proposed a technique to generate interactive proofs of languages, where the prover reveals no information about the subset of knowledge he or she has (i.e., witness indistinguishable).

**FIAT-SHAMIR HEURISTIC** The heuristic presented by Fiat and Shamir [38] is used to create a signature based on a proof of knowledge. The combination of the Fiat-Shamir heuristic, and the interactive proof provided by Cramer et al. [25], is used to construct digital ring signatures which will be discussed next.

### 2.1.3 Digital Signatures

Another essential cryptographic primitive that is employed in our second design is the concept of a digital signature. In this subsection, we will go over the different types of digital signatures to provide the foundation upon which the used variant of traceable ring signatures builds.

Digital signatures aim to prove that a message originates from a particular sender, and thus provides authenticity to a message. As described by Rivest et al. in 1978 [105], such operations are essential in a digitalized world:

*"If electronic mail systems are to replace the existing paper mail system for business transactions, 'signing' an electronic message must be possible."*

Using public key cryptography, as discussed in Section 2.1.1, a digital signature provides integrity, non-repudiation, and authenticity of a signed message. A sender signs a message by using its secret key, only known to this particular sender, to 'encrypt' the message into a signature  $\sigma$ . A recipient can decrypt this message using the public key of the expected sender to validate that this results in the original message. Mathematically this would be denoted as:

$$\sigma = \mathcal{E}_{sk_i}(m), \quad (2.1)$$

$$m = \mathcal{D}_{pk_j}(\sigma), \text{ if } i = j. \quad (2.2)$$

**RING SIGNATURES** In 2001, Rivest et al. [106] proposed digital ring signatures. By signing a message with a ring signature rather than a direct digital signature as described before, the signer can preserve his or her anonymity behind a group of peers called a ‘ring.’ A verifier can determine that the signature is valid, while not knowing which of the ring members produced the message, nor are sequential signatures made by the same signer linkable. Ring signatures are often compared with group signatures [8, 19]. Group signatures require a group manager, designated setup, and have a limited ability to generate ad-hoc groups, whereas these dependencies are not present in ring signatures. Extensions of the original ring signatures protocol are abundant, adding new properties desirable in different circumstances [79, 90, 92]. The second protocol proposed in this work uses such an extension called Traceable Ring Signatures (TRS) [43], which state that unrestricted anonymity is not always desirable. Fujisaki and Sukuzi [43], add double-spending traceability while assuring one-more unforgeability. Double-spending traceability means that signing multiple items for the same ‘issue’, will be detected and the identity of the signer will be leaked. An issue can be seen as a unique object that corresponds with an event for which the signers are allowed to contribute a predefined number of times. By allowing such tracing, the protocol essentially limits the anonymity of the signer to execute a set number of signatures per ‘issue’. The term one-more unforgeability originates from the work proposed by Pointcheval and Stern [100, 101], who state that a receiver should not be able to receive more valid signatures than the number of interactions with a valid signer. A more detailed description of the protocol itself will be given in Chapter 6.

#### 2.1.4 Cryptographic hashing

A cryptographic hash function transforms an arbitrarily long input message into an output, the hash, consisting out of a predetermined number of bits. The mathematical representation is as follows [30]:

$$h : \{0, 1\}^* \rightarrow \{0, 1\}^n, m \rightarrow h(m). \quad (2.3)$$

Here  $n$  represents the predetermined bit length of the resulting hash, and  $m$  is the input message. The function  $h$  depends on three properties to be a valid one-way hash function. Determining the input based on a given hash should be computationally infeasible, which is called pre-image resistance. Furthermore, given a message  $m$ , it should not be possible for a computationally bounded individual to compute a message  $m'$ , where  $m' \neq m$ , that results in the same hash value,  $h(m) = h(m')$ . If this condition did not hold, the individual providing the hash could claim either of the two messages as their own as they both correspond with the hash. Finally, the function  $h$  should be collision resistant. In this case, the attacker can freely choose any combination of  $m$  and  $m'$ , where  $m' \neq m$ , and it should be computationally infeasible to determine a message pair that results in  $h(m) = h(m')$ .

The work presented in this thesis assumes a random oracle model, where a hash function is assumed to be a function that provides a truly random string [7]. However, if the input given to the function has already occurred before, the same output will be given. In our naive implementation, the NIST standard hash function SHA-3 has been used to approximate this [35].

### 2.1.5 Differential privacy

Differential Privacy (DP) is a commonly used approach to protect privacy. DP gives bounds on the probability that two neighboring datasets are differentiable. Neighboring datasets can differ in at most one element, and one of which is a proper subset of the other. These bounds are defined as  $\epsilon$  and  $\delta$  in  $(\epsilon, \delta)$ -differential privacy. The  $\delta$  represents the probabilistic component in providing privacy, where a lower probability symbolizes greater confidence.  $\epsilon$  represents the privacy protection bound, for which a lower value represents higher privacy guarantees. When there is no probabilistic factor, a stronger form of privacy can be proven:  $(\epsilon, 0)$ -differential privacy which is equal to  $\epsilon$ -differential privacy. The definition given by Papernot et al. [94] is as follows:

"A randomized algorithm  $M$  satisfies  $(\epsilon, \delta)$  differential privacy if for all pairs of neighbouring datasets  $(d, d')$ , for all subsets  $S$  of outputs:"

$$\Pr[M(d) \in S] \leq e^\epsilon \Pr[M(d') \in S] + \delta. \quad (2.4)$$

When publicly releasing a global model, there needs to be some form of DP present to hide information loss given when multiple classes are close too each other before labeling. By adding calibrated noise, the impact that a single person can make is reduced. It obfuscates the cases where the local models disagree and do not strongly agree [96].

Balcan et al. [5] describe three forms of differential privacy within the field of collaborative learning:

1. Differential privacy for individual records - each participant is responsible for each training sample (i.e. each image of one of their pets).
2. Differential privacy for the databases - a dataset has equal probability to belong to each of the participants. It can not be retrieved which set of pet images belongs to which participant.
3. Distributional privacy for the databases - limiting the information shared about your own samples to what is inherent to the population distribution itself. This would mean that a participant actively filters the provided images to remove outliers that provide too much information.

Approaches within collaborative learning often use DP. These approaches show that there is a trade-off between providing local differential privacy (within the private-local dataset) and global differential privacy (among all considered items over all participants). Thereby differentiating between the

level of privacy provided other than the privacy protection bound discussed earlier. A comprehensive survey on DP has been written by Cynthia Dwork [34], that can provide additional background knowledge on the topic. In addition, Ji et al. [64] present a survey on the application of the DP concept to machine learning.

Due to the extensive research performed in this field, we assume differential privacy as a primitive to our protocols. In our designs, we use the approach proposed by Papernot et al. [94], who use a moments accountant to track privacy costs. The accountant yields low privacy costs when there is a substantial agreement present among the individual models. Papernot et al. can attain a state-of-the-art privacy-utility trade-off using these techniques.

#### 2.1.6 *Adversarial models*

When interacting with other individuals, it is important to make certain assumptions on what capabilities a potential adversary has if one of the other participants would behave maliciously. This can be done by assuming an adversarial model which defines the capabilities of a potential adversary, and what needs to be protected against to be considered secure within this model. Hazay and Lindell [55], define three different threat models: semi-honest, covert, and malicious.

**SEMI-HONEST ADVERSARIAL MODEL** When an adversary is assumed to be semi-honest, also referred to as ‘passive’, the corrupted protocol participant will adhere to the pre-defined steps. Nevertheless, the adversary attempts to learn everything possible from observing the transmissions by keeping a record of all intermediate results. The semi-honest adversarial model is also referred to as ‘honest-but-curious’.

**MALICIOUS ADVERSARIAL MODEL** While the passive adversary follows the protocol description to the letter, a corrupted entity within the malicious model can arbitrarily deviate from the specifications according to the adversary’s instructions. The increased maneuverability of the corrupted entity earned this adversarial the name ‘active’.

**COVERT ADVERSARIAL MODEL** Security under this adversarial model is dependent on the concept of deterrence. Honest parties will detect malicious activity by a cover adversary with a given probability, allowing the cheating party to be penalized accordingly. Hazay and Lindell [55] refer to this adversarial model to be in between the semi-honest and malicious variants. They use an analogy of protecting a house, where malicious assumes a guard that is present 24/7, whereas covert would count on regular visits by a police patrol. The proposed protocols assume the semi-honest and covert adversarial model respectively.

## 2.2 MACHINE LEARNING

In this work, we focus on semi-supervised learning using ensemble machine learning. We do not make any constraints on the type of models combined by the ensemble, but focus on the majority vote, and the distribution summation weighing methods to determine what classification prediction to give. The local models originate from a collaborative setting, and the process of training the classifier is protected from attacks originating out of the field of AML. This section focuses on explaining every part of machine learning that we depend upon in this thesis.

Witten and Frank [123] define machine learning as finding and describing structural patterns within data, which offers additional knowledge on the data, and can help make predictions on new samples. The different data points, or samples, used to retrieve such structural insights can either be labeled, which refers to the concept of supervised learning, or unlabeled, unsupervised learning.

Theodoridis and Koutroumbas [116] define supervised learning as, designing a classifier by exploiting a priori known information, namely the labels corresponding to the available training data. Unsupervised learning, on the other hand, can be used without the presence of class labels. A given set of feature vectors attributes extracted from the training data, is used to determine underlying similarities or clusters, thereby grouping 'similar' items together [116]. Semi-supervised learning exists in between these two types, having access to a set of patterns without their corresponding class, and to a subset for which the class is known. The unlabeled data can then be used to obtain additional information about the general structure belonging to the data at hand [116].

### 2.2.1 Ensemble Learning

Ensemble learning is a form of supervised learning that uses an 'ensemble' of different classifiers rather than a single model. An ensemble is a collection of a (finite) number of predictors that are trained independently for the same task, after which their predictions are combined [112]. Lior Rokach [108] refers to an ensemble as a weighted combination of the individual classifier opinions to obtain a classifier that outperforms every single one. The combination of classifiers can be done by allowing the individual models to be dependent on, or independent of one another. In a dependent framework, a classifier is dependent on the output of the previous classifier, while independent frameworks allow each classifier to be built independently.

A common dependent ensemble framework is boosting. In boosting, a weak *base classifier* is selected and iteratively designed based on a different subset of the training data. Within boosting, we select the subset based on the computed distribution, which emphasizes those samples that are most difficult to classify. Finally, all individual classifiers are combined by taking the weighted average. An example algorithm is called AdaBoost, as proposed by Freund and Schapire [42].



Two common independent ensemble frameworks are *bagging*, and *random forest*. Bagging uses the output of multiple classifiers and outputs a label for a new unknown instance using based on which has been predicted the most. Random forests are very similar to bagging, despite selecting a random subset of features for the classifiers to use.

The employed classification technique of the ensemble is used to determine the class label corresponding to an unlabeled sample. Table 1 gives an overview of several available techniques to perform the label determination based on the input of individual classifiers.

Table 1: An overview of combination methods for ensemble learning as discussed by Lior Rokach [108]

Type	Name	Main formula	Explanation
Weighing methods	Majority vote	$\text{Class}(x) = \arg \max_{c_i \in \text{dom}(y)} \left( \sum_k g(y_k(x), c_i) \right)$	Also known as the plurality vote (PV) or basic ensemble method (BEM) where classification results in the class with the most votes.
	Performance weighting	$\alpha_i = \frac{(1-E_i)}{\sum_{j=1}^I (1-E_j)}$	An additional validation data set is used to assess the performance of the classifiers and weigh them accordingly.
	Distribution summation	$\text{Class}(x) = \arg \max_{c_i \in \text{dom}(y)} \sum_k \hat{P}_{M_k}(y = c_i x)$	The conditional probability vectors are aggregated, and the class with the maximum value within the total vector is selected.
	Bayesian combination	$\text{Class}(x) = \arg \max_{c_i \in \text{dom}(y)} \sum_k P(M_k S) \cdot \hat{P}_{M_k}(y = c_i x)$	The predictions made by the different classifiers are weighed based on their posterior probabilities given their respective training sets.
	Dembster-Shafer	$\begin{aligned} \text{bpa}(c_i, x) &= 1 - \prod_k (1 - \hat{P}_{M_k}(y = c_i x)) \\ \text{Bel}(c_i, x) &= \frac{1}{\lambda} \cdot \frac{\text{bpa}(c_i, x)}{1 - \text{bpa}(c_i, x)} \end{aligned}$	The basic probability assignment (BPA) is a density function. The outcome acts as the base to compute the belief function, where the class with the maximum belief will be selected.
	Entropy	$\text{Class}(x) = \arg \max_{c_j \in \text{dom}(y)} \sum_{k:c_i=} \arg \max_{c_j \in \text{dom}(y)} \hat{P}_{M_k}(y = c_j x) \quad E(M_k, x)$	The ensemble predictions are weighed that is inversely correlated to the amount of uncertainty, entropy, within the classification vector.
Density based	$\text{Class}(x) = \arg \max_{c_j \in \text{dom}(y)} \sum_{k:c_i=} \arg \max_{c_j \in \text{dom}(y)} \hat{P}_{M_k}(y = c_j x) \quad \hat{P}_{M_k}(x)$	The probability that the classifier would sample the proposed unknown item weighs the classifier, it assumes that classifiers trained on data originating from different parts of population distribution.	
Meta learning	Stacking	N/A <sup>1</sup>	A <i>meta</i> classifier is introduced that uses the predictions as input attributes instead of the original feature vectors.
	Arbiter trees	N/A <sup>1</sup>	Each data subset is used to design a classifier. For every pair of classifiers their classification is used to determine which elements of the joint dataset should be included in the subset used to train an arbiter on. This process continues until the tree converges to a single classifier.
	Grading	N/A <sup>1</sup>	The items most commonly wrongly predicted by a classifier are identified by a <i>meta</i> classifier, the predictions are combined for those classifiers for which their <i>meta</i> marks them as being correct.

<sup>1</sup> Due to the generic nature of these approaches, there is no mathematical representation that justifies the underlying approaches.

Xiaohua Hu [59] states that diversity within an ensemble is needed to dismiss the correlation between classifiers, and thereby increase overall classification performance. Such diversity can, for example, originate from the use of different training data by each participant, utilizing different classifier types, or having participants focus on different parts of the search space [108].

### 2.2.2 Collaborative Learning

A form of machine learning that can either be supervised, unsupervised or even a combination of both (i.e., semi-supervised) is collaborative learning. The primary focus of collaborative learning is on the cooperation of multiple data sources working together in generating a model and will be introduced next. Figure 3 illustrates the fact that five participants jointly provide inputs to generate a globally known model  $\Omega_G$ . The type of information that is iteratively shared, and the employed method using which the sharing occurs differ per approach.

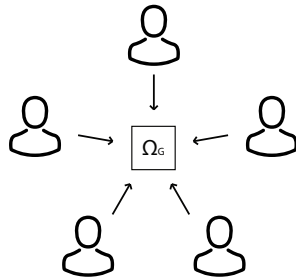


Figure 3: A visualization of the cooperative nature of collaborative learning

In a collaborative setting, the original training samples remain at its local host, a globally known model is constructed using distributed techniques that aggregate derivatives of the locally acquired knowledge. Such an approach can work in several ways: by either sharing a subset of updated model parameters, as described in the case of a neural network by [109] and [57], by sharing complete models, or by sharing predictions on specific items contained in a public dataset as done in [53]. The current approaches within the field will be discussed in detail in Chapter 3.

### 2.2.3 Adversarial Machine Learning

Adversarial Machine Learning (AML) is an upcoming research area that focuses on training and use of machine learning classifiers in adversarial environments [73]. This field assumes that the environment contains a potential adversary who can be defined as one's opponent in a contest, conflict or dispute [31], meaning the person or organization from whom an entity would want to protect their system. Adversaries conduct different types of attacks on the training or inference phases of machine learning to reach a particular objective. Huang et al. [60] have proposed a taxonomy for the research field in which they define these different types of attacks. The taxonomy classifies attacks on three different criteria: 1) the type of

influence exerted, 2) the resulting security violation, and 3) the specificity of the attack. The influence that an attack can be either causative or exploratory, where the first can actively adapt the training data whereas the latter attempts to obtain information about the training data or underlying model. The second requirement evaluates the impact a successful attack can have. An attack could lower the integrity of the system by allowing malicious samples to be labeled as benign, degrade the availability by increasing the number of wrong classifications in general, or by extracting privacy-sensitive information belonging to the learner. Finally, an attack can focus on a particular misclassification (or target), or any misclassification (all possible targets) determining the specificity of the aim of the attack. A more generic classification of attacks distinguishes between poisoning and evasion attacks. Poisoning attacks focus on the training phase, whereas evasion attacks focus on the inference phase of machine learning. The work presented in this thesis mainly focuses on the loss of privacy of the learners participating in a collaborative learning protocol, exercised to train a globally shared classifier. Using the previously described terminology, our focus can be described as exploratory attacks, extracting privacy-sensitive information focused on either all participants or a specific victim.

**ADVERSARIAL EXAMPLES** An adversarial example is a modified data sample that is misclassified by a classifier [70], allowing an adversary to trick a model to for example classify malicious traffic as benign. Figure 4 shows an example, visualizing a perturbation added to an image to generate an adversarial example that is misclassified. Usually, a perturbation is applied to a valid sample to achieve this and have the classifier decision disagree with what a human would see it as [86]. Such a perturbation can be specifically designed for a single model, a particular target class, or universal applications. The latter, proposed by Moosavi-Dezfooli et al. [88], computes the perturbation values in such a way that they can convert a diverse range of images into adversarial examples. The creation of an adversarial example can on the other hand already be achieved by the alteration of a single pixel as shown by Su et al. [114]. Kurakin et al. [70] have shown that the concept of adversarial examples also transfers to the physical world by physically printing the generated examples.

**ACCESS TYPE AND KNOWLEDGE** The amount of available knowledge needed to conduct such attacks depends on the assumed threat model. Within AML, one can differentiate between white-box, gray-box, and black-box access to a machine learning model as mentioned by Meng et al. [86]. In a black-box setting, an adversary only sees the output corresponding to an input sample, without any knowledge of the underlying classifier type. In a white box setting, an adversary is assumed to have all information available to conduct an attack successfully. Finally, a

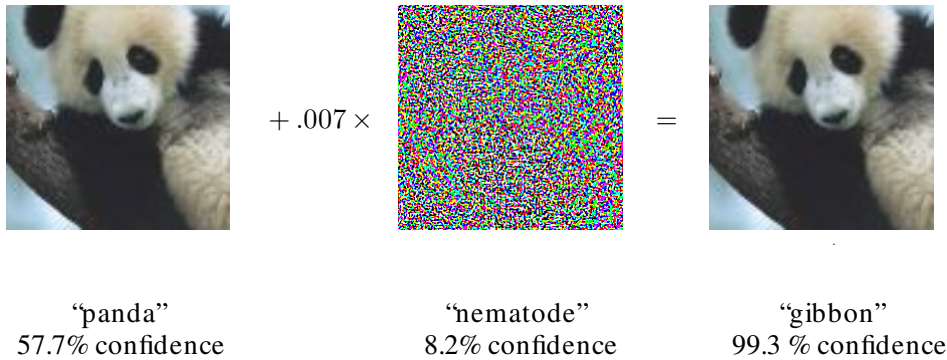


Figure 4: An example given by Goodfellow et al. [48], where an Imagenet picture of a panda combined with noise generated by the Fast Gradient Sign method to misclassify the Panda as a Gibbon with a high confidence level.

grey-box setting resembles a white-box threat model, without revealing the actual classifier parameter values.

**PRIVACY** The attacker can compromise privacy by obtaining the underlying model or the information derived from the training data. Thus, attacks and defenses for privacy focus on preventing an adversary from obtaining either. The most prevalent attacks that can harm privacy are: membership inference attacks [110], model inversion attacks [41], model extraction attacks [119], and Generative Adversarial Network (GAN) attacks [57].

Shokri et al. [110] have proposed a membership inference attack that given 1) a data record  $\delta_1$  and 2) black-box access to a model  $m_1$ , one can determine whether  $\delta_1$  was in the training dataset on which model  $m_1$  has been trained. Federikson et al. [41] introduced a model inversion approach abusing confidence levels that accompany label predictions to extract information of the used training data. Tramer et al. [119] propose a method to learn a close approximation of the original model function of a black-box model.

Finally, Hitaj et al. [57] devised an attack that uses GANs to extract prototypical examples of a class. A GAN as proposed by Goodfellow et al. [49] is a combination of two neural networks, a generative and a discriminative model. Both models commence in an iterative minimax game, where the generative model attempts to generate samples that fool the discriminator while the discriminator aims to detect any false samples. Using this construct, a malicious participant of an iterative, collaborative learning protocol generates samples that are drawn from the same distribution as the private training data of the victim.

**WHEN CAN PRIVACY LOSS OCCUR** Privacy-sensitive information can leak during the training-phase of machine learning or during inference. In early forms of machine learning that use distributed data sources, all training data is sent to a central authority to train a global model. Transferring data can be seen as a breach of privacy for all parties involved. Besides the loss of

actual data, attributes such as the size and dynamics of local datasets can be considered confidential as is assumed by Shokri et al. [109]. During inference, model information and training data attributes can be extracted using the earlier mentioned attacks.

Within this thesis, we attempt to limit the risk of privacy loss introduced due to the open, distributed trust setting. Rather than having a central entity from whom we are protecting our insights, we are now attempting to shield our derivatives from our fellow participants.

### 2.3 IMMUTABLE DISTRIBUTED LEDGERS

There exists a growing public concern about user privacy which has spurred a trend in decentralization. Decentralization concerns the removal of the dependency on a central entity for data protection and availability. It spurs the creation of new protocols that remove the need for a central entity, and the trust that is required in using such a system. The inception of blockchain technology, originating from the introduction of bitcoin by Satoshi Nakamoto [89], has been a significant driver for research in decentralized solutions in a wide variety of industries (i.e., energy [87] and supply chain industry [117]).

A blockchain is a distributed ledger where all transactions are stored at every participant's node. There are three key properties to a blockchain:

1. Decentralization - whereas traditional centralized systems have a single point of failure, the decentralization ensures the continuation of the system as a whole if a single node deactivates.
2. Immutability - it is infeasible to alter previously created transactions.
3. Distributed trust (trustless) - by reaching consensus on the correct state of the system, no individual party is assigned the responsibility to do so and thereby is rewarded the trust of all others.

**VARIANTS OF BLOCKCHAIN** Peters and Panayi [99] mention two main possible blockchain types. These types are permissioned, and permissionless. A permissionless blockchain allows any individual to join the group of peers and validate transactions. Most of these are public allowing anyone to both read and write transactions within the ledger. A permissioned blockchain, on the other hand, requires the permission of a subset of current members to perform any validation. The majority of such blockchains are private, restricting access to those individuals accepted into the system.

**CONSENSUS** In order to obtain a definitive version of the truth, there needs to be a consensus among the participants of the blockchain network. Current consensus algorithms such as proof of work [89] and Practical Byzantine fault tolerance (PBFT) [21] indicate how the truth should be determined.

Within the PBFT consensus model, the participation of a certain number of malicious nodes can be tolerated up to a certain degree. This consensus

model defines this according to the Byzantine Generals Problem (PGB) [72] on which it is based. The proof of work algorithm depends on a particular group of participants called miners to execute expensive mathematical operations. The miner is required to find a hash with a particular structure by adding a nonce, a number only used once, as input along with the underlying block of transactions. The computation time invested increases the difficulty of altering a transaction that has been validated since to do so all blocks that were created afterward have to be recomputed.

**RELEVANCE** In this work, there is no direct need for the chaining capability provided by blockchain. The decentralization, consensus, and the immutability are however desirable properties. Decentralization removes the need to trust a single party with the provided data and correct usage. Consensus ensures that all involved parties have obtained the same set of votes and that the votes can be validated. Finally, by having a state stored at every participant, the system obtains auditability, as any individual when needed can prove that certain votes have originated from them, while others corroborate the transaction did occur. It is important to note that we use this immutable distributed ledger technology to provide us with the capability to decentralize our system. We provide our method of 'transaction validation,' while still requiring the offered benefit of consensus allowing all participants to agree on the series of events performed in the protocol. In theory, for our design, the blockchain could be replaced by a trusted centralized server with access rights to all participants.





The principal objective of this chapter is to introduce the current state of the art in the field of collaborative learning. We first introduce five categories into which we divide the works that have been reviewed for this thesis. After which, these categories are evaluated one by one introducing the present works and identifying links between them. Based on the retrieved insights, we provide a comprehensive overview allowing easy comparison between the works. Finally, we discuss the advantages and disadvantages of each research category along with the key takeaways relevant to this thesis.

The goal of this chapter is to identify the type of approach most appropriate for our research goal. We evaluate the currently available approaches to attempt to find open problems in the field of collaborative learning. Additionally, the internal structure and the versatility of the aggregation technique is weighed to determine the protocol structure that will be our primary focus in this thesis. In our analysis we evaluate a variety of aspects (see Table 2). We analyze what privacy preserving technique is used, if the sender can be validated, whether the contribution of an individual participant can be retrieved, and whether the authors clearly assume a threat model. Table 3, shows whether or not these aspects were included in the reviewed works. Moreover, Table 4 gives an overview of the related work and their presented approaches.

### 3.1 CONTEMPORARY APPROACHES FOR PRIVACY PRESERVATION IN COLLABORATIVE LEARNING

As noted in the introduction, privacy in machine learning is a growing concern. The following section attempts to show the currently available alternatives to obtain this desired trait during the training phase of machine learning models. We have divided the research reviewed for this survey into five main categories. These categories have been created by looking at both similarities and differences in the approaches used to attain privacy when using distributed data sources. The categories are defined as follows:

1. Centralized (Iterative) Parameter Aggregation (CIPA)
2. Centralized Ensemble Model Aggregation (CEMA)
3. Decentralized (Iterative) Parameter Aggregation (DIPA)
4. Decentralized Ensemble Model Aggregation (DEMA)
5. Adapting Existing Learning Algorithms (AELA)

The first four categories exhaust all possible combinations focused on whether there is a central server required within the protocol and the

utilized combination technique. Decentralization indicates that no central entity is used and all participants train by communicating directly with one another. The centralized categories, on the other hand, do have a central server that handles communications and aggregates data or model-findings. The second differentiating factor (i.e., iterative/ensemble), defines how knowledge originating from the individual participants is combined into a single model. We have identified two possible approaches: aggregating model parameters to determine those of the global model or doing so using an ensemble structure that combines the local models through their predictions. The final category resembles a research direction that focuses on the adaptation of current machine learning algorithms to allow for encrypted processing of data. This category has been added to indicate other approaches within private data mining with possibilities for the field of collaborative learning. In the following sections, each of these categories will be discussed, and the available literature examined. We accompany each category with an image of the typical layout of a system belonging to its respective category. Figure 5 provides a list of the symbols used throughout these depictions.


$\theta$	A predetermined fraction of parameter updates that are downloaded from the central server.
	(Re)Train model using the newly available information
$\Delta$	The differences between the downloaded parameters and those retrieved after local retraining.
$\rightarrow$	Transition between participating parties, as computation is done iteratively.
$\Omega$	A model trained on $\Omega_L = \text{local data}$ , $\Omega_G = \text{public data}$ , and $\Omega = \{\Omega_L, \Omega_G, \dots, \Omega_L\}$
$\rho$	Public unlabeled dataset

Figure 5: A list of all symbols used within the figures provided for each category resembling the components used.

### 3.1.1 Centralized (Iterative) Parameter Aggregation

This category resembles protocols that utilize a central server and iteratively aggregate model parameters that converge to a global model. The majority in this category employs differential privacy mechanisms, whereas there are also approaches using multiparty computation and homomorphic encryption. Figure 6 shows a typical infrastructure belonging to this category. A central entity initiates the beforehand agreed upon model structure (i.e., a neural network with pre-specified layers) with random weights and defines both an upload and download percentage which are

communicated with the participants. After initialization, the first participant downloads a fraction  $\theta$  of these parameters. The local model with the same structure is updated with these weights and trained (on the locally stored data) for several epochs. After this (re-)training, the weights have shifted, and the participant uploads the difference between the weights on the server and the local values. Afterward, the other participants repeat the same process, until an ending condition is met. Each of the proposed schemes differ in their approach, and their method of choice to protect privacy.

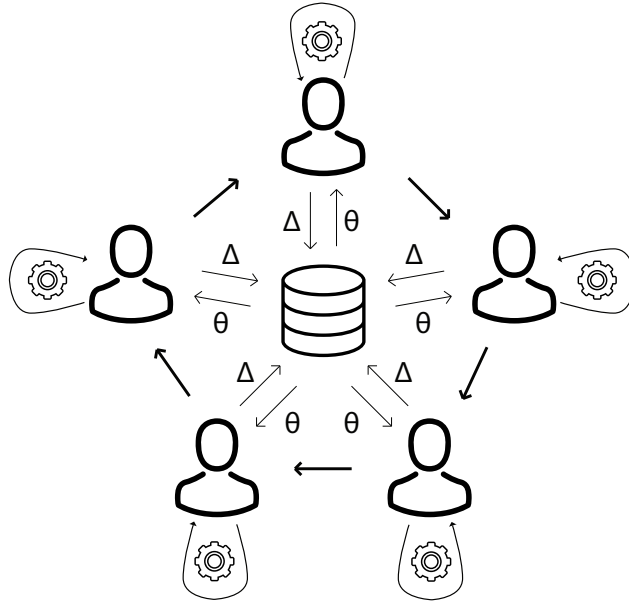


Figure 6: An overview of a typical system structure of the systems belonging to the [CIPA](#) category.

**DIFFERENTIAL PRIVACY APPROACHES** The first approach is by Pathak et al. [97]. Here, a central entity averages all the model parameters retrieved from the individual participants, without the iterative nature described earlier. The presence of a central entity maintains the relevance of the structure presented in Figure 6, with the slight adaptation that the interactions between the server and the participants should be interpreted as uni-directional. Using a private multiparty protocol, an untrusted curator named *Charlie* shares additive shares of the final classifier with all participants. The curator here does not gain insights into the parameters due to the use of both homomorphic encryption and data permutations. The individual data points are combined with a stochastic element during the averaging process to attain  $\epsilon$ -differential privacy. The authors consider the use of a logistic regression classifier but do claim that the techniques are generally applicable to any classification algorithm. Wu et al. [124] also focus on linear regression with the introduction of the Grid Binary LOGistic REgression (GLORE). However, this approach requires all participants to be

synchronized during the computation of the gradients. Also, since no secure multiparty computation is used, it does not provide any provable security.

Rajkumar et al. [104], questions why the authors of Pathak et al. [97] perform averaging over local regularized empirical risk minimization (ERM) classifiers learned by the individual parties stating that this might not be the right way to approximate the overall regularized ERM classifier. The approach devised by Rajkumar et al. [104], does, however, build on top of the work done by Pathak et al. [97], focusing on the use of stochastic gradient descent. Their approach provides the weaker variant,  $(\epsilon, \delta)$ -differentially privacy by perturbing the shared gradients by two different noise vectors. A new noise vector is retrieved every iteration, in addition to a constant noise vector that prevents the noise from canceling out over time. Nevertheless, it is more robust to the number of parties and the variability of the amount of data located at the different locations. The authors assume the loss function to be convex and differentiable, ensuring that gradients exist for which the minimizer of the objective is unique.

Shokri et al. [109] also state that the averaging performed by Pathak et al. [97] does not always lead to improved accuracy, stating that it is highly dependent on the underlying problem. Their approach performs a distributed form of gradient descent to train a deep learning model using a central parameter server through which the parameter gradients are communicated. This configuration has been used as a prototype for the example structure shown in Figure 6. It provides an upload and download parameter that defines the number of variables that a participant transfers. Limiting this is a form of selective stochastic gradient descent and reduces the chance of getting stuck in local optima. The sparse vector technique is used to prevent information leakage on the gradient selection technique or the actual values. Parameter updates above a threshold are selected and shared after being perturbed using Laplacian-noise. The protocol guarantees  $\epsilon$ -differential privacy, however, the privacy loss is computed per parameter and thus not for the entire model. Later, Abadi et al. [1] improved the privacy guarantees of the model proposed by Shokri et al. [109] by randomly perturbing parameter values during the stochastic gradient descent algorithm execution. Shokri et al. [109] state that an oblivious parameter server would be able to remove linkability between the updates and the participants, but this has been identified as future work.

Hamm et al. [54] took a different approach, mainly focusing on using smart-devices as different data sources. The protocol allows multiple devices (with a maximum of six) to update the central parameters at the same time, which improves the communication overhead when a large number of participants are involved. Thus, Figure 6 would be adapted to each participant representing a multiple performing their update protocol simultaneously. In addition, the proposed protocol uses stochastic gradient descent and provides  $\epsilon$ -differential privacy by locally adding Laplacian noise.

McMahan et al. [84] propose a similar federated approach to collaborative learning as that of Shokri et al. [109]. However, according to the McMahan et al. [84], Shokri et al. [109] do not focus on unbalanced, and independent and identically distributed (IID) data, whereas their algorithm is robust to both. Additionally, there are fewer (by orders of magnitude) communication rounds needed due to the introduction of new processes such as techniques reducing the complexity of gradient computation.

Rodriguez et al. [107] shift the focus towards personalized model creation extension to the category. The first step is to create a global model using either data collected from a small set of users, or using a public dataset. In the case where individual users data is used, the authors refer to the earlier mentioned approaches for differentially private training [53, 84, 109]. Users receive the shared model, where they can either retrain it to create a personal model or use the global model for the inference phase. They found that using a global model to create a personal model requires fewer samples than directly training a local model with the same amount of samples. Interesting to note is that this is the only research paper that explicitly considers possible attacks and how they impact their protocol. They investigate the accuracy of the final personal model under different proportions of corrupted training samples (poisoning attack), for both random perturbation, and targeted label perturbation.

Finally, it is important to highlight that Hitaj et al. [57] have shown that the iterative types of differentially private collaborative deep learning models are vulnerable against an attack that uses Generative Adversarial Networks (GAN). By training such a GAN actively during the iterative process, the malicious entity can generate prototypical class examples of a class belonging to another participant. This research focuses on the work by Shokri et al. [109], but other papers using a similar approach can also be vulnerable to such attacks.

**MULTI-PARTY COMPUTATION (MPC)** Tian et al. [118] describe an implementation to securely aggregate multiple locally trained machine learning models using MPC. This aggregation is done by having two central entities which are trusted not to collude with each other who complete a secure multiparty computation to aggregate encrypted shares of local models to obtain the global model. To the best of our knowledge, this is the first MPC-approach in this category.

The initial research proposed by Konecny et al. [68] lowered communication costs of distributed learning. However, one of the limitations of this approach is that the cloud-provider in their scenario learns all updates and aggregates them in the clear. Therefore, part of the research team continued to improve this work in [13]. Here, a communication-efficient, failure-robust protocol for secure aggregation of high-dimensional data is introduced that limits the knowledge of the cloud-provider to only the aggregated model. The secure multiparty computation used guarantees security in the honest-but-curious and active

adversary settings. Sender validation is actively considered, while input validation is defined as future work. Moreover, identifying and recovering from abuse of the protocol (i.e., a denial of service attack) is identified as an open problem. Although not mentioned explicitly, this also seems to be the case for the other works reviewed.

**HOMOMORPHIC ENCRYPTION** Yonetani et al. [127] propose another federated approach with a large focus on privacy. The individual parameters are aggregated using Double Permuted Homomorphic Encryption (DPHE), allowing all gradients to be encrypted during aggregation. The efficiency is claimed to be kept practical by using certain sparsity constraints limiting the number of needed encryptions. A permutation matrix that is only known to the user is used to hide which in indices are non-zero. However, a malicious user that intercepts the message is still able to find the weights. Therefore, an additional permutation matrix is shared between the user and the aggregator, preventing this from being possible. The name double permuted originates from the fact that there are two permutations used.

Overall, the current state of this category of private machine learning suffers from several drawbacks. First of all, communication often represents a significant bottleneck due to the centralized nature. Further, it has become apparent that sharing gradient updates can also reveal private information causing further improvements to become necessary in this space [111].

### 3.1.2 Centralized Ensemble Model Aggregation

This category focuses on Ensemble approaches in distributed learning that aggregates the individual predictions at a central point. Figure 7 shows an example of three participants transferring their models to a server, which uses their predictions to label a public dataset upon which a final global model is trained. Hamm et al. [53] highlight the importance of this category of distributed learning by stating that parameter averaging is not applicable when using different types of local classifiers or non-numerical ones like the decision tree. Here the approaches that do consider privacy, focus on the use of differential privacy.

One of the first occupants of this category is the research proposed by Folino et al. [39]. These researchers generated a novel approach to distributed intrusion detection systems, which aggregates locally generated decision trees, at a central location to train a *Genetic Programming Ensemble*. Each peer can communicate with its neighboring peers (its neighborhood) and the collector island (the central entity). The collector combines the hypotheses generated by each classifier to produce the ensemble. The paper does not indicate any privacy concerns, and it mentions no security or privacy precautions.

Xie et al. [125] take a similar approach where locally-trained classifiers are combined at the *aggregator*. The individual models trained on the sensitive data are assumed to be linear-binary classifiers trained using the

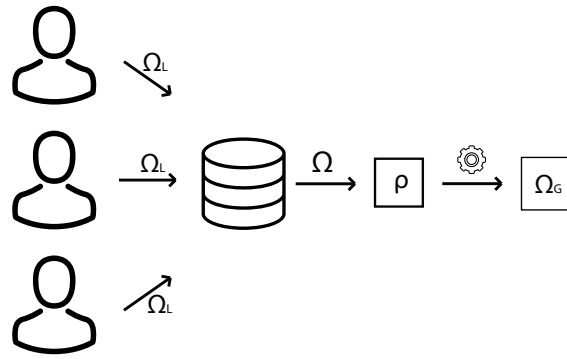


Figure 7: An overview of a typical system structure of the systems belonging to the [CEMA](#) category.

differentially private variant of empirical risk minimization (ERM) to achieve  $\epsilon$ -differential privacy. The aggregator uses the individual classifiers as features, projecting the public data onto them. By doing so, the aggregator can train a low-dimensional classifier which resembles a weighted-linear combination of all local classifiers.

Soon after, Hamm et al. [53] published their approach to distributed ensemble learning. In the first step, the participants share all classifiers with a trusted central entity. Then, the central entity labels an auxiliary, public dataset similar to the one used in Xie et al. [125]. This labeling is done by aggregating all individual predictions and assigning the full gradient to the public entry. The subsequent step uses this labeled dataset to train the final classifier, adding differential privacy using output perturbation to all samples involved, not just of those belonging to a single party. The used output perturbation does require a limited sample space or the truncation of feature values to assess the size of the to be added noise [65]. After noticing a lack of robustness in the majority vote procedure used to aggregate label predictions, soft labels were introduced as weights in the minimized loss function. Overall, the approach proposed by Hamm et al. is not constricted to a classifier type and provides global differential privacy, whereas Xie et al. [125] only allows for linear-binary classifiers and provides local differential privacy. This approach is extended upon by Papernot et al. [94] in an entirely centralized setting, as the authors state that there is no condition on the provided privacy as well as no restrictions on the chosen classifiers.

Cyphers et al. [26] use an entirely different approach. Here, the participants send partial feature vectors of the sensitive data to the aggregator. By combining these partial vectors, the aggregator can reconstruct the complete feature vector corresponding to a particular label. The distribution of a subset of peers, called a partition, is used to generate synthetic training data, upon which a model can be built. The authors also consider a validation step that checks whether the data originates from a valid participant, and focuses on unlinkability, removing any link to the provided data and the originating peer to the aggregator. This sender validation accompanied with double-spending prevention is done using Anonize tokens [58], which will be introduced more elaborately in Chapter

8 in more detail. Local differential privacy can also be guaranteed according to the author.

To conclude, there are several drawbacks present in this category. Openly sharing models with a trusted entity violates one of the privacy goals as mentioned in section 2. Besides the technique proposed by Cyphers et al., no data or sender validation is done nor is the impact of malicious activity taken into account. Nevertheless, the approach proposed by Cyphers et al. does maintain a link between individual contributions and the obfuscated identities generated.

### 3.1.3 Decentralized Iterative Parameter Aggregation

This category decentralized the earlier work shown in section 3.1.1. Both of the reviewed approaches utilize differential privacy to preserve the privacy of the training data. Figure 8 shows a typical overview of a system that fits in this category. Each participant instantiates the shared model structure and shares its updates with a pre-determined amount of neighbors iteratively.

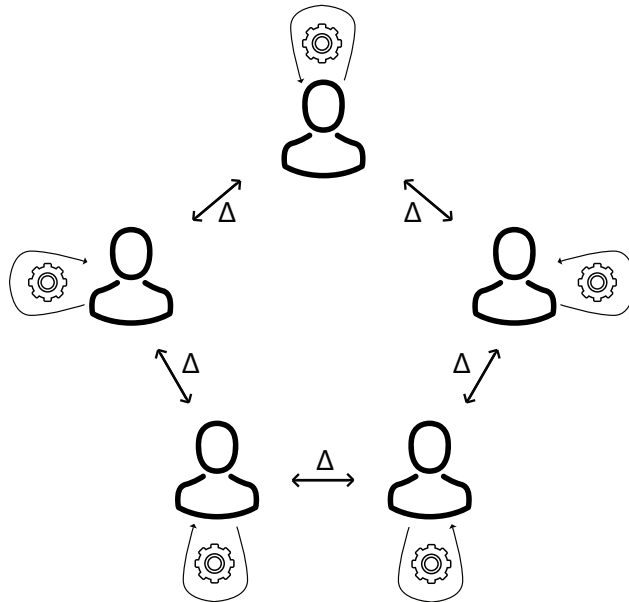


Figure 8: An overview of a typical system structure of the systems belonging to the [DIPA](#) category.

According to Huang et al. [61], this is the first solution in this category that also maintains privacy. The difference with the research done in the centralized iterative category is that in the system designed by Huang et al. the parameter updates belonging to a participant are shared with its neighbors. Initially, the gradient updates are perturbed using Laplacian noise that converges to the Dirac distribution achieving  $\epsilon$ -differential privacy.

Bellet et al. [9], advocate the creation of personalized models where the underlying process is similar to the one used by Huang et al. [61]. They propose a completely decentralized, asynchronous, iterative method of



aggregating the knowledge of local data and adapting the resulting model to the personalized problem statement. Again, the individual model parameter updates are communicated with the neighborhood of the participant in question, which consists of a fixed amount of other neighboring participants. Additionally, a confidence level is provided when sending weights to illustrate the amount of certainty of the sender in the delivered weight updates. Laplacian noise is added to the gradient of the local loss function to achieve  $(\epsilon, \delta)$ -differential privacy. The authors of Bellet et al. [9] state that they believe that the decentralized nature of the approach significantly reduces the effectiveness of the attack mentioned in Section 3.1.1, devised by Hitaj et al. [57]. Nevertheless, this has not been proven and left for future work.

Only two approaches have been identified to belong to this category. Nevertheless, these approaches do seem promising and warrant the need for further development.

#### 3.1.4 Decentralized Ensemble Model Aggregation

The result that falls into this category resembles attempts to decentralize the centralized ensemble approach discussed in section 3.1.2. Unfortunately, the research identified to belong to this category has a limited focus on privacy, presumably due to the limited number of approaches currently available. Figure 9 attempts to give the reader an example overview of a protocol belonging to this category. When compared to its centralized variant in Figure 7, one sees that the operations executed centrally are done at each participant separately. Also, the model belonging to a participant is transferred to every other participant.

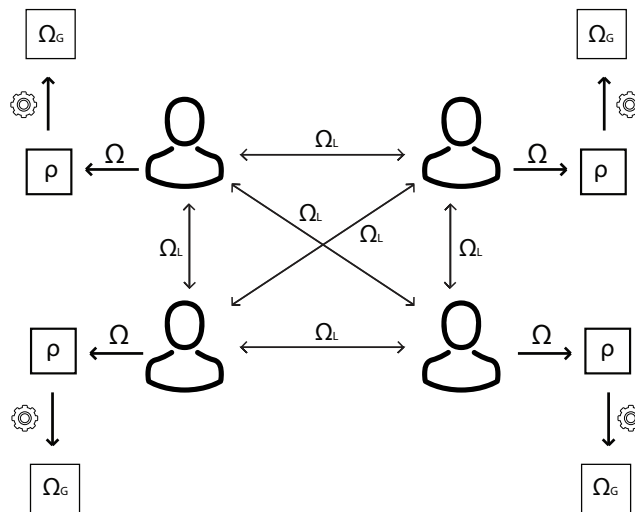


Figure 9: An overview of a typical system structure of the systems belonging to the [DEMA](#) category.

Early approaches such as proposed by Luo et al. [80] and Bhaduri et al. [11] introduce distributed classification using ensemble learning by

aggregating votes produced by local models. Therefore, the transferred information shown in Figure 9 should represent votes for these two papers. Both focus on the use of peer-to-peer (P2P) networks which are categorized as an at the time emerging technology for content sharing. Unfortunately, none of these incorporate any privacy precautions, leaving data-owner privacy lacking. The lack of privacy-preservation means that all communication is unprotected, allowing all neighbors with whom an individual communicates to see its votes in the clear, which also creates the opportunity for manipulation.

Li et al. [75] proposes adaptive distributed privacy-preserving algorithms. It provides every participant all individual models to aggregate any combination of sub-models as they please. However, the authors assume that a local model can be shared without any privacy impact as they state that sharing the models enables the creation of an integrated model, without sharing the individual datasets and thus preserving private of this training data. However, according to Amir-Khalili et al. [3], if the local models are known to all peers, there is a possibility for a malicious peer A to forward a crafted model to peer B and use B's response to inferring data at B. In addition, the system does not involve any failure precautions and allows an adversary to have detailed insights into the model, a white box scenario. Furthermore, the local sample size is used to assign weights to different classifiers in the global ensemble. Sharing this can be considered a privacy breach as it can leak sensitive information about the underlying entity. Finally, the desired combination models are selected by assessing performance on the local training data. Since the local data does not accurately describe the entire population distribution, this performance can be considered biased. Insulearn, proposed by Amir-Khalili et al. [3] themselves, incorporates a byzantine fault tolerance algorithm and clearly defines a threat model.

The decentralized ensemble approaches show a lack of privacy-preserving approaches, leaving room for significant improvement in future work.

### 3.1.5 *Adapting Existing Learning Algorithms*

This section aims to highlight a different approach to attain privacy in machine learning. The authors whose work falls into this category mainly focuses on the conversion of specific algorithms to allow for the use of encryption. Here, existing classification techniques are adapted to allow for training or inference on encrypted data. Encrypting the data protects the confidentiality of the individual inputs. First, we will discuss early work in the field and review more recent approaches afterward.

**THE FOUNDATION** Secure multiparty computation based approaches became prevalent in the early 2000's. In these methods, secure protocols replace a few critical steps within a specific classification algorithm and aggregate the results at a centralized node. Lindell and Pinkas [77] proposed a protocol to compute an essential step in the ID<sub>3</sub> algorithm needed to build a decision tree without revealing input values from either

party. Vaidya et al. [122] introduce secure dot protocols to perform privacy-preserving K-means clustering over vertically partitioned data. Kantarcioglu and Clifton [67] provided another approach, who focus on association rule mining. Finally, Yu et al. [128] focus on support vector machines by proposing secure dot product and sum protocols to retrieve the kernel matrix. These works show a variety of approaches to turn specific machine learning algorithm operations into their privacy preserving counterpart.

**NEURAL NETWORKS** More recently, Yuan et al. [129] proposed an adaptation of the back-propagation algorithm used in neural networks. This adaptation allows for encrypted data to be uploaded to the cloud, where one can then train a neural network without seeing any plaintext values. The data is homomorphically encrypted, and the decryption is done using an adaptation of the method proposed by Boneh et al. [15], allowing multiple entities to jointly decrypt values. The paper focuses on cloud computing where data from different sources is combined while encrypted. Nevertheless, there is a trusted authority involved that has access to the underlying information during the intermediate steps.

Gilad-Bachrach et al. [45] proposed Cryptonets, focusing on the inference stage of machine learning. Instead of training a network on encrypted data, an existing neural network is adapted such that the model can classify encrypted samples. Unfortunately, according to [22, 96], the overhead of the resulting model makes the method impractical, especially for deeper networks. In addition, there is a restricted set of arithmetic operations that are supported by homomorphic encryption.

**OTHER CLASSIFIERS** Aslett et al.[4] attempted a similar approach to Yuan et al. [129], focusing on the training phase of machine learning, generating models using encrypted data. The presented algorithms include random forest, and Naive Bayes, putting a coding scheme in place that facilitates the comparison of the values and a needed threshold. This protocol imposes strict requirements over the used data, more than are needed for the proposed work by Gilad-Bachrach et al. [45].

Graepel et al. [50] propose a method of training several binary-linear machine learning techniques using a somewhat homomorphic encryption scheme. This type of homomorphic encryption only allows a fixed number of multiplications; thus the approach uses a fixed number of iterations of gradient descent. Additionally, private inference is achieved in a weaker security model where the client is allowed to learn more about the model than only the label. Finally, Bost et al. [16] have developed a two-party computation framework using fully homomorphic encryption. Using this framework, three different classifier types are converted to allow classification of encrypted data namely: hyperplane decision, naive bayes, and decision trees. There is a large communication overhead due to the required interactions between the two *honest but curious* participants. Aslett

et al.[4] also mentioned this as a limitation of the work proposed by Bost et al. [16].

Even though this category does have its limitations, whether that is a trusted third party or impractically large overhead, the used techniques highlight a different research direction within the field.

### 3.2 CREATING AN OVERVIEW

After carefully analyzing the literature, we have obtained insights into the methods used in the field of collaborative learning. Most notably, we have identified what choices have been made for the vital design-decisions shown in Table 2. The first three variables indicate whether a specific privacy-preserving technique has been employed. The latter three indicate what type of protection is provided, and thus what type of limitations might still be present. Based on these aspects, we create an overview in Table 3 that shows a check-mark whether the subject is present in the conducted research.

A threat model indicates which type of adversary the authors consider, and by assuming one defines the scope of the research in terms of security. Here we define unlinkability as not having a link between private information and its owner. This property is desirable since if every parameter update or classification vote retains its link to its source, a clear overview of individual behavior will allow for more targeted attacks. Finally, verifiability is considered to represent whether or not both the sender, and the received data can be verified. In a collaborative setting, this is important to prevent malicious non-participants from participating in the protocol, and that each participant can only contribute its allowed portion of information to the aggregate model.

Table 2: The design decisions encountered in the considered papers, and their respective description

Variable	Description
Differential Privacy (DP)	Is differential privacy used in the presented work?
Multiparty Computation (MPC)	Is multiparty computation used in the presented work?
Homomorphic Encryption (HE)	Is homomorphic encryption used in the reviewed work?
Threat model (TM)	Does the work assume a clear threat model?
Unlinkability (U)	Throughout the protocol, is there a link between private information and the owner?
Verifiability (V)	Is there a check on the validity of sender and the conveyed information (vote/gradient)?

Table 3 presents the presence of the design decisions posed in Table 2 in the reviewed literature. This table provides a clear overview of the attributes present in the different approaches. The provided design decision overview shows that verifiability of the originator and the provided data is most often left disregarded. Moreover, the linkability between updates and their originators, which allows more intricate model inversion, is considered in a limited amount of cases. Furthermore, it seems that the older, more

Table 3: Overview of the discussed papers and their approach according to the provided criteria.

	Paper	DP	MPC	HE	TM	U	V
CIPA	Pathak et al. [97]	✓	✓	✓	✓	-	-
	Rajkumar et al. [104]	✓	-	-	-	-	-
	Shokri et al. [109]	✓	-	-	✓	-	-
	Hamm et al. [54]	✓	-	-	✓	-	-
	McMahan et al. [84]	✓	-	-	-	✓	-
	Rodriguez et al. [107]	✓	-	-	-	-	-
	Tian et al. [118]	-	✓	-	✓	✓	-
	Bonawitz et al. [13]	-	✓	-	✓	-	✓
Yonetani et al. [127]	-	-	✓	✓	✓	-	
CEMA	Folino et al. [39]	-	-	-	-	-	-
	Xie et al. [125]	✓	-	-	-	-	-
	Hamm et al. [53]	✓	-	-	-	-	-
	Cyphers et al. [26]	✓	-	-	✓	✓	✓
DIPA	Huang et al. [61]	✓	-	-	✓	-	-
	Bellet et al. [9]	✓	-	-	✓	-	-
DEMA	Luo et al. [80]	-	-	-	-	-	-
	Bhaduri et al. [11]	-	-	-	-	-	-
	Li et al. [75]	-	-	-	-	-	-
AELA	Yuan et al. [129]	-	-	✓	✓	-	✓
	Gilad-Bachrach et al. [45]	-	-	✓	-	-	-
	Aslett et al. [4]	-	-	✓	-	-	-
	Graepel et al. [50]	-	-	✓	✓	-	-
	Bost et al. [16]	-	-	✓	✓	-	-

researched categories cover a wider variety of approaches whereas both distributed categories (DIPA and DEMA) are insufficient in their offering.

Similarly, Table 4 shows an overview of paper attributes such as the evaluation techniques used, analyzed machine learning algorithms, and the used datasets. It becomes apparent that a variety of different models are used on a large variety of datasets, showing a lacking standardization in the field. Therefore, it remains complicated to compare the performance of different studies. Nevertheless, as can be seen in the evaluation column, certain papers such as the one written by Rajkumar et al. [104] do perform a comparison with their direct alternatives.

Furthermore, an overview focusing on practical details is provided in Table 4. Here features such as the use case, model type, and datasets used are mentioned. Even though not all papers mention everything, it does provide the reader with an indication of what is common practice within this field. For example, the majority of the reviewed research focuses on supervised learning, and there is a large variance in the datasets used to review the performance of the generated models.

Table 4: An overview of the reviewed work which provides high-level insights into the techniques used in each approach.

	Authors	Evaluation Use Case	Learning Paradigm	Model Type	Datasets
CIPA	Pathak et al. [97]	Empirical evaluation of privacy protection provided by predicting annual income based on census data.	Supervised	Logistic regression but extendable	UCI Adult data
	Rajkumar et al. [104]	Performance comparison of the proposed algorithm with the local aggregation algorithm proposed by Pathak et al. [97].	Supervised	Private stochastic gradient descent	Synthetic data, Wisconsin breast cancer data set
	Shokri et al. [109]	Determining the effect of $\theta$ and mini-batch size to select the version of SGD, while validating classification performance and the effect of collaboration of the proposed algorithm.	Supervised	MultiLayer perceptron, Convolutional Neural Network	MNIST, SVHN
	Hamm et al. [54]	Validating performance by making a proof of concept for a crowd sensing system on mobile devices, using Google's activity recognition as ground truth. In addition, additional validation is provided using digit recognition.	Unsupervised and supervised	Multiclass logistic regression	Actual phone movement data, MNIST
	McMahan et al. [84]	Evaluation of privacy costs within the proposed framework. Analyzing the impact of the allocated privacy budget, complexity, training efficiency, and model quality.	Supervised	Multilayer perceptron	MNIST, CIFAR-10
	Rodriguez et al. [107]	Determining the benefit of personalizing global models, and comparing this to pure local training and the unpersonalized global model.	Unsupervised and supervised	Multilayer Perceptron	WISDM, Wikipedia and NIPS datasets
	Tian et al. [118]	Verifying accuracy loss due to the aggregation using MPC focusing on high dimensional regression.	Supervised	Linear Discriminant Analysis	Synthetic data, Heartdisease dataset
	Bonawitz et al. [13]	Both a theoretical and a prototype evaluation are given to validate the proposed protocol, focusing on the efficiency and not on machine learning performance.	-	Neural Networks	-
Yonetani et al. [127]	Comparing classification performance of novel protocols to those proposed of Pathak et al. [97] and Rajkumar et al. [104].	Supervised	Linear Support Vector Machine (using SGD)	Caltech101, Caltech256	
CEMA	Folino et al. [39]	This work only validates classification accuracy (no privacy) on the dataset compared to other methods, focusing on distributed intrusion detection.	Supervised	Adaboost	KDD 1999 Cup
	Xie et al. [125]	Evaluating performance of this data-weighted ensemble approach for varying privacy guarantees, and the size of each local site.	Supervised	ERM with linear loss	MNIST, Coverttype

	Hamm et al. [53]	The proposed protocols are compared to a non-private setting, a variant where only local data is used, as well as the work proposed by Pathak et al. [97] on activity recognition, network intrusion detection, and malicious URL prediction.	Semi-supervised	Linear classifiers	UCI Human Activity Recognition, KDD-99, Malicious URL
	Cyphers et al. [26]	The aim is to build a discriminatory model with locally-private data. The performance of the resulting system is evaluated by comparing it to traditional adaboost and random forest algorithms.	Supervised	Ensemble classifier	MOOCDB EDX dropout data, Adult Data Set
DIPA	Huang et al. [61]	The proposed protocol is evaluated by analyzing the impact on the accuracy based on the set parameters, tweaking the noise distribution added throughout different iterations.	-	-	-
	Bellet et al. [9]	Personalizing globally generated model and comparing this with a baseline that only uses local data and validating increased performance by collaborating.	Supervised	Coordinate Descent	Synthetic data, MovieLens-100K
DEMA	Luo et al. [80]	This work evaluates the proposed algorithms in different P2P topologies in terms of classification performance and communication overhead and converge time.	Supervised	Ensemble	Covtype from UCI repository
	Bhaduri et al. [11]	Using synthetic data, the proposed solutions are evaluated against a completely centralized alternative, and the impact of noise in data is analyzed.	Supervised	Decision tree	Synthetic data
	Li et al. [75]	The proposed framework is evaluated by comparing the aggregated model to one that is trained solely on its own EHR data.	Supervised	Adaboost	Diabetis EHR data
AELA	Yuan et al. [129]	In addition to a complexity analysis, three different experiments are performed comparing the proposed scheme to two alternatives as well as a not privacy preserving version.	Supervised	Neural Network	Iris, Diabetes and kr-vs- kp
	Gilad-Bachrach et al. [45]	The provided network conversion is reviewed in terms of classification performance and the efficiency of the different steps is assessed in terms of transmission size and training time.	Supervised	Convolutional Neural Network	MNIST
	Aslett et al. [4]	The proposed solution is implemented and the efficiency timings are shown for different scalar operations.	Supervised	Naive Bayes, Random Forest	Synthetic data
	Graepel et al. [50]	The focus of the evaluation is on efficiency due to the inherent complexity of homomorphic encryption, for both algorithms.	Supervised	Linear Means, Fisher's Linear Discriminant	Synthetic data
	Bost et al. [16]	Performance is evaluated as opposed to their unencrypted counterparts, while execution time, and transmission size of the various stages are also measured.	Supervised	Linear, Naive Bayes, Decision Tree	Breast cancer, Credit, Audiology, Nursery, ECG

### 3.3 DISCUSSION

In an ideal scenario, a protocol is able to protect the used training data, the private models made by the individual participants, and remove any linkability between released information and the data owner. Furthermore, malicious attempts to alter the outcomes or provide falsified data, from both a valid and non-valid participants should be detected and prevented. The current state of the art does not provide the combination of these features, but this survey has identified a trend towards more privacy-aware machine learning approaches.

First of all, when reflecting upon Table 4, it becomes apparent that in order to compare the performance of future protocol designs, a more standardized dataset selection should be made. By selecting a standardized dataset for a specific model type, it will be easier to compare papers and their contributions and thus improve the interpretability of the contribution presented in future papers. Currently, there is a significant variance between the used datasets making it difficult to compare the performance of the resulting models directly. By being able to compare performance directly, it also becomes easier to make the trade-off between the attained privacy and performance.

Furthermore, Table 3 shows us what decisions have been made by the authors of the reviewed papers. Most of the protocols did not provide any form of verifiability or linkability between the data, and its sender remains present. To achieve these aspects, different requirements need to be met. These consist of the implementation of both anonymity and unlinkability of the participants, verifiability of the sender and the accompanying data, consensus on the correct execution of the used techniques, and a certain degree of protection against the different attacks mentioned in Section 2. Especially the implementation of the inherently opposing concepts of unlinkability and verifiability can be a challenge.

Moreover, an explicit threat model should be adopted providing insights into what information is available to the attacker and what his or her goal is. Providing this assumption allows future researchers as well as potential users to evaluate for which scenarios the proposed protocol is or is not destined. Even though a notable portion of the papers did include this, not all did, and security should become a primary concern when designing novel protocols. It warrants a cross-over between both research areas to jointly conduct research into on the one hand maintaining and improving performance while on the other hand providing increased privacy.

When we zoom in on the categories as a whole, we can see higher level patterns. It can be seen that the centralized approaches (CIPA and CEMA) are comprehensive and vary in their offerings. Nevertheless, many do not consider and evaluate every aspect of privacy, resulting in open problems, often denoted as future work. For example, an oblivious parameter server



can add unlinkability to the work presented by [109], while Bonawitz et al. [13] leave the identification of well-formed inputs to future work.

The distributed collaborative learning categories offer significant opportunities for future work as it does show promising aspects for privacy-preservation. In these settings, one does not need to trust a central entity resulting in there being no single point of failure. Nevertheless, the iterative approaches within DIPA might still be vulnerable to the GAN attack [57], and the ensemble category (DEMA) is significantly lacking in terms of privacy protection as the authors do not take it into account at all. Distributed methods should also focus on establishing a consensus methodology, on when the process has indeed been properly executed. The value of the final result produced would increase if every participant can themselves verify that all steps belonging to the protocol have in fact been executed properly.

The final category (AELA) gives insights into works that attempt to convert specific model types to handle the use of encrypted data. The techniques proposed can be used to improve performance and privacy protection. Nevertheless, the use of homomorphic encryption can lead to significant computational overhead, and even though performance might be improved, efficiency should also be compared to allow readers to make a careful trade-off between the two. The main downside of such methods in distributed learning include both the use of a trusted central entity ([129]), as well as impractical computational overhead ([45]). Moreover, there is still only a limited amount of available models that allow for encrypted training or inference. However, unlike using an ensemble approach, there is no loss in classification performance since there is no information lost in transition as everything is encrypted.

Overall, the discussed approaches show that collaborative learning has compelling use cases and warrants future work focused on making it usable in an era where privacy awareness and legislation are playing an increasingly more prominent role.

### 3.4 KEY TAKEAWAYS

We have discussed current approaches to collaborative learning, which have been divided into five categories each with their respective advantages and disadvantages. Based on our findings, an overview has been presented both on the properties attained by each paper as well as practical facts exposing a lack of standardization. It has become apparent that the currently available options are lacking a focus on privacy. The DEMA architecture offers several benefits such as the fact that the local models can be of any desired type allowing a group of participants to use already available classifiers. The decentralized nature removes the need to transfer these models to a single entity. The generic model creation can be abstracted from the global process and initiated locally, where every participant can decide on the preferred architecture if so desired. Moreover, the training process can be very intensive, especially for those model types that used in the CIPA/DIPA

categories. The ensemble-based approaches benefit from the fact that the communication is depended on the number to be labeled items rather than training epochs. In general, an ensemble-based approach is a more flexible approach that can be used for and freely tuned to a variety of different problem statements. Additionally, the ensemble-based approaches do not allow for iterative attacks such as the [GAN](#) attack. Therefore, we have decided for our primary focus to be on the [DEMA](#) category in finding a solution for our research problem. This and the rest of our application setting and how we aim to answer our research question will be discussed next.

Privacy needs to be the foundation of the design process to enable the decentralization of collaborative machine learning model generation. The current alternatives have been evaluated, and we argue that these do not meet the current privacy requirements. The related work shows a limited focus on privacy, leaving the state of participant privacy to be inadequate. In this chapter we are going to introduce how we aim to overcome these limitations and provide an improved solution. Firstly, we will go over the generic application setting for which we will design our solution. Secondly, we will discuss how we attempt to answer the questions derived from our principal research goal (Chapter 1) in our research. Finally, we will identify the challenges that are present in overcoming the currently present limitations.

#### 4.1 APPLICATION SETTING

Our primary goal is to create a protocol design that facilitates privacy-preserving collaborative learning. In this section, we will identify what assumptions we make of the environment to justify our problem statement. We continue to discuss how our setting compares to currently available alternatives solving similar issues.

##### 4.1.1 *Assumptions*

In attempting to answer our research question, we have made several assumptions about the underlying application setting. In this setting, we are addressing the relevant assumptions on the global level, whereas the protocol designs will include a clear overview of solution specific assumptions. In our problem definition and overall solution structure we make the following assumptions:

- There exists a need to combine insights retrieved from training data located at three or more different locations, in a privacy-preserving manner.
- These parties are willing to cooperate to obtain better-performing classifiers, as long as the privacy of their training data can be guaranteed.
- Participants have a means of identifying themselves during the protocol setup, to validate the people with whom the protocol will commence.
- We assume that participants are willing to commit extra resources for the additional training time to attain training data privacy, making the protocol practically feasible.

- We take the approach presented by Papernot et al. [94], and blockchain technology, as primitives. We apply the state-of-the-art differential privacy techniques presented by Papernot et al. Further, we use blockchain technology to distribute our communication channels with consensus. We assume the presence of these techniques as this is not the core of our contributions.

#### 4.1.2 How does our setting relate to previous work?

The inspiration for the generic application structure originates from Papernot et al. [94], which achieves a similar goal in an individualistic, non-collaborative setting. Their approach is comparable with the protocol proposed by Hamm et al. [53], who assumes different originating data sources as mentioned in Chapter 3. We attempt to convert their completely centralized and central collaborative approach into a decentralized, privacy-preserving variant. In order to do so, we first go over how each of these solutions construct their final labels of the public data set.

Figure 10 indicates the approach proposed in the Private Aggregation of Teacher Ensembles (PATE) framework. The proposed framework assumes a white box adversary who can target the global model  $\Omega_G$ . The user splits the original data set into different partitions upon which models are trained, who jointly label the public dataset as the ensemble model  $\Omega_E$ . By transferring the knowledge of the ensemble to the public dataset  $\rho$ , the authors set a bound to the amount of privacy loss to the original training data, dependent on the number of queries, i.e., the cardinality of  $\rho$ . The main difference with our setting is that within their approach there is no possible adversary within the ensemble creation phase of the protocol. It is their semi-supervised structure that we adopt as a primitive in our protocol designs, in addition to their differential privacy approach to protect the final global model.

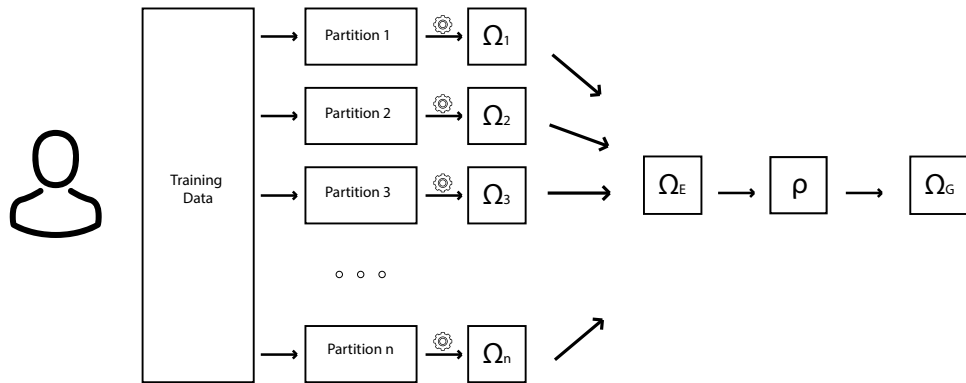


Figure 10: The generic structure proposed by Papernot et al. [94] to improve privacy-preservation of a publicly released model generated by a single party.

A more detailed visualization of the protocol devised by Hamm et al. [53] as compared to Figure 7 is given in Figure 11. All participants transfer their models to the central entity  $c$ , who generates an ensemble out of the different models and labels of a public dataset  $\rho$ . In addition to the privacy risks

concerning the public release of a final model, the internal communication becomes a potential risk. Access to a machine learning model represents a threat to privacy, as introduced in Chapter 2, which we do not want to have in our system.

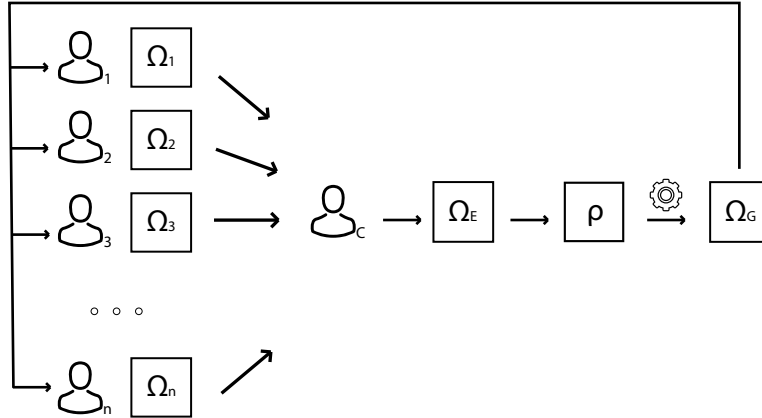


Figure 11: The generic structure proposed by Hamm et al. [53] to improve privacy-preservation of a publicly released model generated by multiple parties in a collaborative setting, part of the CEMA category.

We, on the other hand, assume that the different participants do not (want to) trust a central entity, but still want to collaborate with others. Such an assumption could hold for large multinational companies, forced by law to act as independent entities, as well as a consortium of participants. By doing so, we enable the inclusion of data sources whose knowledge might have previously not been accessible for training purposes. Therefore, we want to decentralize these types of protocols, resulting in a depiction as given in Figure 12.

The decentralization of these protocols, as depicted by the use of a distributed ledger, generates challenges for the execution of the protocol. However, before we continue to address the challenges that will be present, we will address what methodology we will employ to create such a system.

## 4.2 METHODOLOGY

The inadequacy of the currently available alternatives is why our main research goal is to provide a new protocol that overcomes currently present limitations and delivers a way to train models in a privacy-preserving fashion. This goal has been formulated as:

*How can we facilitate the joint generation of a shared machine learning model in a privacy-preserving manner, and refrain participants from degrading the final models' performance more than their own, allowed contribution, in a decentralized setting?*

In other words, we want to propose a protocol that allows multiple participants to perform a classifier training operation jointly. In doing so, we

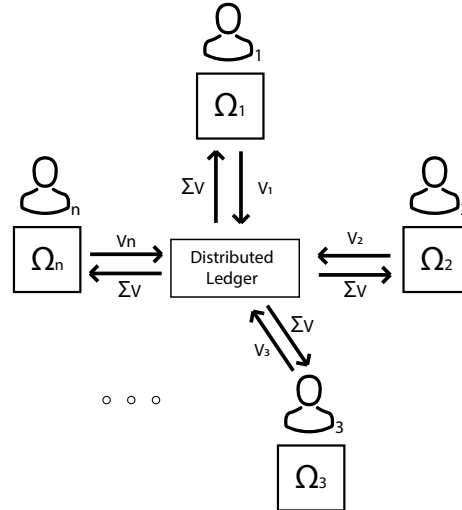


Figure 12: The generic structure proposed in this thesis to improve privacy-preservation of a publicly released model generated by multiple parties in a collaborative setting, will be part of the DEMA category.

want to ensure that the amount of information that any participant can learn with regards to the training data present locally at any other participant to a determinable threshold. Meanwhile, we want to prohibit participants from contributing more than their  $1/n$  equal right to the final label of a public sample. Finally, we want to be able to do so by removing the need for a central entity. From these insights, we derive the following five sub-questions which we will discuss individually.

#### 4.2.1 Sub-questions

To answer the presented research question and achieve its corresponding goal, we will design and implement two privacy-preserving protocols. Different underlying techniques are used, each taking a different route to obtain the required properties. In order to evaluate the research questions, we execute proof-of-concept implementations, offering runtime analysis to complement the theoretical computation and communication complexity analyses. We will now go over each sub-question and discuss which aspects are required to successfully answer them.

*How can we create a transparent, decentralized joint machine learning model generation system?* Transparency is crucial in order to allow participants to value the resulting global model. If any participant is able to recompute intermediate steps and see that these have been validly constructed, one can ensure herself from the correct execution of the protocol. In order to answer this question, we should test the final protocol designs, and validate whether any participant can reconstruct intermediate steps and ensure proper execution. Meanwhile, this should not endanger privacy by allowing insights into individual behavior.

*How can we limit the number of contributions of a participant to a pre-specified amount?* A technique must be employed that prevents a participant from contributing more than a pre-specified amount. By doing so, we prevent excess contributions as mentioned earlier. We can evaluate this question by performing a proper analysis whether a participant, under the given assumptions, is able to contribute more than allowed to the final protocol outcome.

*How can we remove the linkability of their contributions to the originating user, while assuring a valid sender.* In order to carefully evaluate this question, we need to provide a detailed analysis on whether a contribution can be linked to its sender. This analysis includes validating that there is no link between a 'pseudonym' or random derivation of the sender identity and all sender contributions. Meanwhile, we require a valid sender, meaning that the sender is included in the group of participants who are validated when the protocol commences.

*How can we leverage the setting of the protocol to improve efficiency?* Since our application setting assumes the involvement of multiple parties, how can we use this as an advantage to improve efficiency? It might be the case that in more substantial amounts of participants, there is no need for a message to be equally likely to have originated from *all* participant. To evaluate this question, we need to develop both a simple protocol design and compare its efficiency with an approach that leverages the number of participants.

*How can the above sub-questions be achieved without significantly degrading the prediction accuracy as compared to centralized variants?* The usage of privacy-preservation techniques within the field of machine learning forms a delicate trade-off, where the costs must not severely impact the attained benefit. The impact to the equilibrium of this trade-off can be evaluated by analyzing the applied techniques, and assessing whether or not they impact the knowledge transfer that occurs. If the knowledge transfer is adversely impacted, we need to evaluate the performance with and without these changes. Otherwise, the analysis of the fact that the output of local classifiers is not altered would assert that this is trivial.

#### 4.2.2 Design science

In this thesis, we employ the design science methodology. Research conducted according to this methodology aims to develop artifacts with the explicit intention of improving the performance of the artifact [56]. The design science framework [56] is used to ensure a proper process, and add to the scientific nature of this thesis. Hevner et al. [56] have presented seven guidelines to which valid design science-based research must cohere. Table 5 shows these seven guidelines and how this research attempts to abide by them.

Table 5: Application of the seven design guidelines of design science as presented by Hevner et al. [56]

DS guideline	Our approach
Design as an artifact	Chapters 5 and 6 show the design of two IT artifacts in an elaborate manner, making implementation, and validation easily possible.
Problem relevance	The combination of Chapter 1 and 3 sketches the underlying problem statement and identifies it as an open issue when compared to the available literature. The significant costs involved due to misclassification or the inability to export data make the business need increasingly present.
Design evaluation	Chapter 8 focuses on the evaluation of the proposed artifacts. The design will be implemented experimentally validated. We will compare the run-time results to peer-reviewed alternative approaches.
Research contribution	This thesis presents the contributions made as summarized in Chapter 1. The extensive literature review in Chapter 3 indicates open issues which the design is made to solve.
Research rigor	Peer reviewed works from the shared body of knowledge are used as building blocks in the presented design. Furthermore, these works are also used in the evaluation of the presented design.
Design as a search process	Two variants solving the underlying problem are presented. These are designed after actively evaluating different alternative cryptographic techniques to achieve the required properties.
Communication of research	The report is structured in such a way that both business and technological identify the need for the proposed solutions. The technical details might not necessarily be applicable to both technical and managerial audiences. However, we aim to make the rest of the report insightful for any interested reader.

### 4.3 CHALLENGES

To eliminate the need for a central entity for both the ensemble generation process as well as the execution of the labeling phase, we observe several challenges that need to be overcome. Do note that these challenges are only relevant within the scope of the assumed threat model and the abilities of the adversary that come with it.

By allowing multiple participants to be part of the ensemble generation process, we allow for adversaries to gain the increased power to disrupt the machine learning process. These adversaries can now either be internal or external to the protocol environment. An internal adversary gains limited insights into what other participants contribute, and should not be able to skew the outcome of the ensemble excessively. The challenges inherent to decentralization are as follows:



**LIMITED INSIGHTS INTO PARTICIPANT BEHAVIOR** A participant should not be able to relate the votes to the participant who has cast them. If an internal adversary can track the behavior of each participant, he or she can obtain knowledge from the used classifier. Similarly to the protocol as a whole, the adversary can use this knowledge to make a new classifier based on the knowledge provided by the participant. To prevent this from happening, either the identity linked to a vote should be obfuscated, or the content of the vote itself should be hidden.

**NO DOUBLE SPENDING** The contribution given by a participant should be limited to a predefined amount of votes per item. If the given votes exceed the allowed values or multiple votes are cast, the outcome of the ensemble can be skewed towards a particular dimension, possibly moving the decision boundary of the ensemble and thereby misclassifying malicious traffic as benign or vice versa.

**VERIFIABILITY AND CONSENSUS** The entire process should be verifiable. Every participant should be able to verify that each vote has been cast correctly within the given threat model. Also, all participants should agree upon the state of the collected votes, meaning that no participant can claim to have voted differently by changing their contribution later on.

**AUDITABILITY** For a participant to allow verification of their contribution, a participant should be able to prove that a specific vote value originated from him or her. For example, imagine that due to the actions of an adversary, the decision boundary moves slightly causing significant monetary damage. In such a situation, the participants should be able to prove to both judicial and regulatory entities that they behaved as they should have.

**REASONABLE EFFICIENCY** When executing the protocol, the cooperation of multiple parties should be executed with reasonable efficiency. Efficiency, in this case, is considered as being executable using a regular IT setup without taking significantly longer than the currently available alternatives. The protocol should be able to evaluate a large public dataset items, both when a few participants, as well as large number of participants are included. We assume that the actions do not necessarily need to be real-time, since an updated version would be trained periodically in terms of multiple days, weeks, or even months.



## Part III

### DESIGNING THE SOLUTIONS

Part two has provided us with the necessary context on what approaches are currently available, and how we aim to answer our research goal. Part three we present our designs and provide theoretical analyses on their performance. For both designs, we first provide an overview of the entire protocol, after which we examine every phase separately in more detail. Afterward, we provide an extension to our second design, in which we attempt to improve the efficiency of the protocol without losing our desired privacy properties.



## ECONOMY: ENSEMBLE COLLABORATIVE LEARNING USING MASKING

---

The first protocol of two protocols that we have designed in our attempt to attain our research goal is called ECONoMy, private collaborative learning using an ensemble approach. The principal objective of the presented design is to provide an efficient training scheme, applicable to a use case where the protocol contains a large number of participants, such as in an IoT setting.

Since we desire the use of a decentralized architecture, we need to convert the solutions presented in a centralized setting, and need to overcome the research challenges discussed in Chapter 4. One of these challenges raises the need to remove the link between the participant identity and the prediction values. We can accomplish this by either anonymizing the identity behind the prediction values or by making the data confidential to all other participants of the protocol. ECONoMy achieves the required property using the data confidentiality alternative. The prediction votes are made confidential using masking, for which we generate values in such a way that when the individually masked votes are aggregated, the randomness is canceled out. The aggregate is subject to a label selection technique to label the underlying samples, after the application of noise to achieve differential privacy. The right combination technique is dependent on the analyzed problem. Thus, we focus on the basic ensemble method as described in Table 1, based on soft labels (note that this is the same as distribution summation).

To the best of our knowledge, ECONoMy is the first collaborative learning system that allows for privacy preservation in a decentralized ensemble setting, assuming an 'honest-but-curious' threat model. No individual can extract information on the individual models as it requires the combination of all masked values to retrieve the total.

We have structured this chapter as follows. First, we introduce the required notation in Section 5.1, and discuss the utilized preliminaries in greater, mathematical detail, continuing on the explanations given in Chapter 2. After, we provide an all-encompassing overview of the ECONoMy protocol in Section 5.3, which we dissect in a phase-by-phase analysis in Section 5.4. Section 5.5 analyses the security of the proposed scheme. Furthermore, the evaluation in Section 5.6 is done in terms of complexity (5.6.1) and communication (5.6.2). Finally, we provide a small discussion of the presented protocol.

## 5.1 NOTATION

We will frequently need to refer to certain variables within this design chapter. In order to make this as efficient as possible, Table 6 contains a full overview of all the used notations.

Table 6: Explanation of the symbols used in the ECONoMy design chapter.

SYMBOL	EXPLANATION
$n$	The number of participants in the protocol.
$m$	The number of items for which there is to be voted.
$u$	The number of possible classes the items can be classified as.
$\eta$	Number of zeros appended to the plaintext encoding.
$\beta$	The number of bits required to represent one class in the vote encoding.
$\xi$	The bit size of the created masking value.
$\psi$	The size of an encrypted random number share.
$\lambda$	The size of a label.
$\mathcal{E}$	Denotes the use of the AES encryption function.
$\mathcal{E}'$	Denotes the use of the Paillier encryption function.
$\mathcal{D}$	Denotes the use of the AES decryption function.
$\mathcal{D}'$	Denotes the use of the Paillier decryption function.
$\mathcal{J}$	Set of identifiers where $\mathcal{J} = \{1, 2, \dots, n\}$ , where $ \mathcal{J}  = n$ .
$\overset{r}{\leftarrow}_x$	Uniformly selecting $x$ random values out of a specific space, if omitted $x = 1$ .
$\Pi$	Set of participants $\Pi = \{\pi_1, \pi_2, \dots, \pi_i\}$ , where $ \Pi  = n$ and $i \in \mathcal{J}$
$\mathcal{S}$	Unlabeled public samples, $\mathcal{S} = \{s_1, s_2, \dots, s_m\}$ , where $ \mathcal{S}  = m$ .
$\mathcal{C}$	The possible classes, $\mathcal{C} = \{c_1, c_2, \dots, c_u\}$ , where $ \mathcal{C}  = u$ .
$\mathcal{L}$	Set of final labels, $\{L_1, L_2, \dots, L_m\}$
$\Gamma_{\pi_i}$	Set of predictions for $m$ items from participant $i$ , $\{q_{\pi_i, s_1}, q_{\pi_i, s_2}, \dots, q_{\pi_i, s_m}\}$ .
$R_{\pi_i}$	Denotes the original set of random numbers generated by participant $\pi_i$ such that $R_{\pi_i} = \{r_{i1}, r_{i2}, \dots, r_{ij}\}$ for $\pi_j \in \Pi$ and $i \neq j$ .
$R_{\pi_i, \pi_j}$	Denotes the random shares generated by participant $\pi_i$ , meant for participant $\pi_j$ where $\pi_i$ and $\pi_j \in \Pi$ and $i \neq j$ .
$R_{\pi_j, \pi_i}$	Denotes the set of random numbers that participant $\pi_i$ received from participant $\pi_j$ , generated using the random number generation. $R_{\pi_j, \pi_i} = \{r_{\pi_j, \pi_i}(1), r_{\pi_j, \pi_i}(2), \dots, r_{\pi_j, \pi_i}(m)\}$ , where $ R_{\pi_j, \pi_i}  = m$ .
$\check{R}_{\pi_i, s_k}$	Denotes the set of random numbers of $\pi_i$ , after aggregating all received part per item $1 \leq k \leq m$ .
$\mathcal{M}_{\pi_i}$	The set of masked votes where $\mathcal{M}_{\pi_i} = \{\check{m}_{\pi_i, 1}, \check{m}_{\pi_i, 2}, \dots, \check{m}_{\pi_i, m}\}$ where $\check{m}_{\pi_i, k} = \check{R}_{\pi_i, s_k} + q_{\pi_i, k}$ .
$\mathcal{M}_{s_k}$	The set of masked votes where $\mathcal{M}_{s_k} = \{\check{m}_{1, s_k}, \check{m}_{2, s_k}, \dots, \check{m}_{n, s_k}\}$ where $\check{m}_{k, s_k} = \check{R}_{\pi_k, s_k} + q_{\pi_k, k}$ .
$\Delta$	The set of all aggregated masked vote distributions.

## 5.2 DETAILED PRIMITIVES

In this section, we will introduce the four different cryptographic preliminaries that are explicitly used in the ECONoMy protocol. The protocol uses additive secret sharing, Diffie-Hellman key exchange, as well as both a symmetric and asymmetric cryptosystem, to attain privacy preservation. The underlying principles and a brief introduction of these techniques have been given in Chapter 2. In this section, we will give a more detailed, mathematical representation to provide a clear foundation for the design chapters.

Additive secret sharing is used to split random values into different shares. These shares form the foundation to generate random numbers, one present with each participant, that sum up to zero and which we can use for masking. By performing this masking, the intermediately shared votes cannot be retrieved by an adversary, limiting the exposure to the total of all individual votes. Furthermore, even after addition, no information of the used random numbers is available using this technique.

To distribute the generated shares, the use of a symmetric cryptosystem assures that only the destined party has access to their shares. By sharing the messages in a single batch, we attain lower communication costs as opposed to sending all parts in a separate secure channel. The other primitives are used to assert a symmetric key between the different parties of the protocol.

**ADDITIVE SECRET SHARING** Our first protocol uses additive secret sharing to compute the masking values that will obfuscate the transmitted votes. We use a variant of the additive secret sharing procedures employed by Kursawe et al. [71], and Gracia et al. [44]. Their approach uses the assignment of leaders and aggregators, whereas we will not use these constructs.

Instead, every individual starts out by generating  $m$  random numbers  $R_{p_i}$  for  $i \in \mathcal{J}$  (defined in Table 6), once for each item that is to be labeled. These random numbers are then divided into  $n$  random shares, one for each of the participants, such that the shares sum up to the original random value. The following example assumes there is only one item,  $m = 1$ :

$$\text{Alice (1): } R_1 = r_{11} + r_{12} + r_{13} \pmod{p} \quad (5.1)$$

$$\text{Bob (2): } R_2 = r_{21} + r_{22} + r_{23} \pmod{p} \quad (5.2)$$

$$\text{Charlie (3): } R_3 = r_{31} + r_{32} + r_{33} \pmod{p} \quad (5.3)$$

where  $p$  is a large integer. Alice will retain  $r_{11}$ , and receive both  $r_{21}$  and  $r_{31}$  as the second letter in the denominator indicates the receiving party. Alice then computes her masking value using the following function:

$$M_1 = \sum_{i=1}^n r_{i1} - R_1 \quad (5.4)$$

By deducting the original random value, we can be convinced that the final parts that are used in the masking will cancel out to zero and thus

leave us with an accurate aggregate of the, to be hidden, messages. All shares meant for every different participant will be published in a single operation to reduce communication costs. However, we publish these in the clear without any masking; participants will gain insights into other shares than their own. This limitation would allow them to compute the masking value used by other participants, and break the confidentiality gained by masking in the first place.

Imagine the same scenario as before, where  $m = 1$ , and Alice, Bob, and Charlie have computed their shares. If all the to be transferred shares are published Charlie would see the following shares:

$$\begin{aligned} & r_{12}, r_{13} \\ & r_{21}, r_{23} \\ & r_{31}, r_{32}, r_{33}. \end{aligned} \tag{5.5}$$

If Charlie wants to compute  $M_2$  he can now do so, as shown below. By substituting the underlying values, it becomes apparent that  $M_2$  can be computed using only values present in the list above.

$$M_2 = r_{12} + r_{22} + r_{32} - R_2 \tag{5.6}$$

$$M_2 = r_{12} + r_{22} + r_{32} - r_{21} + r_{22} + r_{23} \tag{5.7}$$

$$M_2 = r_{12} + r_{32} - r_{21} + r_{23} \tag{5.8}$$

In order to preserve the benefit provided by the masking operation, we need to hide the shares from participants for whom they are not meant. Therefore, we employ a symmetric block-cipher to encrypt the randomly generated parts.

**ADVANCED ENCRYPTION STANDARD** The [AES](#), presented by Daemen and Rijmen [27], is a symmetric block cipher commonly used in practice. The scheme utilized a varying number of rounds, depending on the key size (128, 192, 256) in which there are different executions of substitution and permutation steps to generate a ciphertext. The symmetric nature results from the fact that decryption can be done using the same key and by reversing the order of the executed rounds.

Block ciphers have different modes of operation [36], which differ in how the different blocks interact. In this thesis, we use counter mode due to its parallelizable encryption and decryption, and a respectable security reputation. Counter mode prepends a selected counter value which acts as an initialization vector (IV). This mode can be replaced by any other mode which is proven to be semantically secure.

The random numbers that will be encrypted are uniformly selected with a bit size of  $\xi$ . The probability of a plaintext collision offers a negligible advantage. The selected algorithm is [AES-128](#) in counter-mode, which is assumed to be semantically secure.

**DIFFIE-HELLMAN** Diffie and Hellman [32] presented a key exchange protocol that allows the generation of a symmetric key based on the public



key information. In a two-party setting, each party selects a secret key  $sk_1, sk_2$ . Using two commonly known prime numbers  $p$  and  $g$ , each participant computes and publishes their public key:

$$pk_i = g^{sk_i} \pmod p. \quad (5.9)$$

Each party can now obtain their shared-secret by computing the following function:

$$\text{shared\_secret}_1 = pk_2^{sk_1} = (g^{sk_2})^{sk_1} \quad (5.10)$$

$$\text{shared\_secret}_2 = pk_1^{sk_2} = (g^{sk_1})^{sk_2} \quad (5.11)$$

where  $\text{shared\_secret}_1 = \text{shared\_secret}_2$ . In our application of this primitive, we hash the shared secret after adding a counter value. By doing so, we enable the creation of a new symmetric key over time without additional communication.

**PAILLIER CRYPTOSYSTEM** The Paillier cryptosystem [93] can be used to encrypt a message  $m$ , and decrypt a ciphertext  $c$  using the following functions:

$$\mathcal{E}_{pk}(m) = g^m \cdot r^n \pmod{n^2} \quad (5.12)$$

$$\mathcal{D}_{sk}(c) = L(c^\lambda \pmod{n^2}) \cdot \mu \pmod n \quad (5.13)$$

These operations require the use of a keypair  $(pk, sk) = ((g, n), (\lambda, \mu))$ . Here,  $g$  represents a value within  $\mathbb{Z}_{n^2}^*$ ,  $n$  represents the product of primes  $p$  and  $q$ , and  $r$  represents a random value for which the greatest common divider of both  $r$  and  $n$  equals one,  $\text{gcd}(r, n) = 1$ .  $\lambda$  represents the least common multiplier of  $\text{lcm}(p-1, q-1)$ , while  $\mu$  represents  $(L(g^\lambda \pmod{n^2}))^{-1} \pmod n$ , where  $L(x) = (x-1)/n$ .

The scheme is probabilistic due to the use of a random value  $r$  and therefore will give different outputs for the same message, it is semantically secure.

### 5.3 PROTOCOL OVERVIEW

Before going into the details concerning every individual phase of the protocol, we will first discuss the complete overview. The ECONoMy protocol consists of five phases:

1. The initial startup - A phase where the required infrastructure gets initialized. All participants generate their keys, publish their public keys, and compute their shared secrets.
2. The random number generation phase - This phase enables the generation of masking numbers in such a way that when all masked individual votes  $\mathcal{M}_{s_j}$  are aggregated for a specific item  $s_j$ , the aggregate of the original votes is retrieved. It results in a set  $\check{R}_{\pi_i, s_j}$  for each participant  $\pi_i \in \Pi$ , containing a random masking value for each item in  $S$ .

3. Masking phase - All predictions  $q_{\pi_i, s_j}$  are obfuscated by adding the corresponding computed masking value,  $\check{r}_{\pi_i, s_j}$ , to obtain  $\check{m}_{\pi_i, s_j}$  for all  $\pi_i \in \Pi$  and all  $s_j \in S$ . Each participant contributes their masked votes in  $\mathcal{M}_{\pi_i}$ .
4. Aggregation phase - For each item,  $s_j \in S$ , the obtained masked values in set  $\mathcal{M}_{s_j}$  are aggregated to obtain the aggregate of all predictions.
5. Noise addition and final model generation - The noise required to obtain differential privacy on the predictions given by the released global model. Similarly, the final labels are devised using the selected classifier combination technique, and the training of a final model can commence.

The design of ECONoMy originates from several required properties: 1) prevent double voting, 2) the result should represent a unique vote per party per item, and 3) the vote needs to be confidential from other participants. We have opted for masking as our technique to hide vote contents from other participants. Masking does not require exponentiations of large prime numbers and results in relatively small masked values (as compared to cipher text in public cryptosystems). The randomly generated numbers will cancel out in aggregation, allowing us to obtain an unmasked aggregate.

By having a random number generation phase, where shares are destined for specific participants, every participant receives one valid masking value per item. This constraint directly limits the number of votes per item per participant to one and requires the addition of all numbers to obtain a valid total, assuring the aggregation of precisely  $n$  votes when demasking has been successful. Thus, by constraining our adversary by designing the protocol description, he or she is not able to deviate from it and will not be able to double vote.

The encryption of the secret shares before transmission is done using [AES](#) since it requires no exponentiations using large values. A participant will post his or her messages in a single transaction, making it essential that there is no correlation between ciphertexts originating from the same input plaintext. Thus, we require a semantically secure block-cipher mode. In our described protocol we use the counter mode, which also has the added benefit of allowing parallel encryption and decryption to allow system optimization. Any other encryption scheme that provides us with the ability to hide the transferred contents similarly could be used if preferred over [AES](#) in counter mode.

In establishing this protocol, we have made the following assumptions that should be taken into account when evaluating the design.

- We assume a 'semi-honest' threat model, where adversaries follow the protocol description but are curious enough to use all obtained information to the fullest extent.

- In a semi-honest structure, an adversary can store all known information. We, therefore, assume that less than half of all parties are sharing their information.
- We assume that the participants can jointly validate certain execution steps, meaning that there is a certain degree of trust. If  $x$  participants have approved a contribution or the initial infrastructure, the others are willing to trust those  $x$  people to contain at least one honest verifier. There is thus a threshold of  $x$  approvals in order for an operation to succeed.
- We assume that the use of AES in counter mode provides semantically secure encryptions.
- We assume that the  $G$  and  $g$  variables required for the Diffie-Hellman key exchange are properly chosen.
- We assume that the discrete log problem holds for properly selected bit sizes, upon which the Diffie-Hellman protocol depends as well as the Paillier Cryptosystem.

#### 5.4 PHASE-BY-PHASE DESIGN DESCRIPTION

In this section we will present the design of the ECONoMy protocol per phase. Each of the phases will show the required procedures and their respective analyses.

##### 5.4.1 Initial setup

The first phase of the protocol initializes the required infrastructure and enables the participants to identify themselves by allowing them to participate in the protocol. Each individual  $\pi_i \in \Pi$ , in turn generates a key-pair  $Pk_i = g^{x_i}, Sk_i = x_i$ , corresponding to the Diffie-Hellman key-pairs as discussed in Section 5.2. Additionally, every participant creates a Paillier key pair is created, and both its public key and that corresponding to the Diffie-Hellman key-pair are published. Each participant continues to generate shared AES keys in the following way. First, every participant  $\pi_i \in \mathcal{P}$ , publishes  $n - 1$  random values  $\check{r}_{ij}$  encrypted using the public Paillier key of each receiving participant  $\pi_j \in \Pi$  where  $i \neq j$ . Afterward, the shared AES-Key can be generated by each participant by computing the following function:

$$\text{shared\_secret} = (pk_j)^{sk_i \cdot \check{r}_{ij} \cdot \check{r}_{ij}}$$

resulting in a shared secret. This shared secret is then hashed to provide a shared symmetric key of 128 bits suitable for AES-128, that is unique for that combination of participants. Finally, the participants agree on the vote encoding and a time frame within which all votes should be received.

At the end of this phase, the protocol is ready to commence, and every participant knows who is participating. All participant have access to an AES

key for its communication with each participant in the protocol. Now we are ready to commence to the next phase, the generation of the masking values that will be used to hide each participant  $\pi_i$ 's to be shared prediction values  $\Gamma_{\pi_i}$ .

#### 5.4.2 *Random number generation*

We create random number numbers using additive secret sharing, such that they sum up to zero, to mask each vote for all items in  $\mathcal{J}$ . By doing so, when all votes corresponding to an item  $s_i \in \mathcal{S}$  are aggregated, the random values that masked the individual votes will be negated leaving us with the total of the vote distributions. To properly hide the underlying vote of bit size  $(u \cdot \beta + \alpha)$ , a random value of a specific bit-length would need to be chosen, corresponding to the currently required bit security,  $\kappa$  bits, according to NIST standards [6], where  $\kappa$  is our security parameter. The required bit size will be further discussed in the security analysis in Section 5.5.

Every participant will need to generate  $m$  random numbers that each mask a prediction in  $\Gamma$ , as is done in the `GenerateRandomNumberShares` procedure in Algorithm 1. Therefore, the procedure first generates  $m$  random numbers that are  $\xi$  bits in length. Afterward,  $n$  parts are generated which modulo a prime  $p$  sum up to the original random value. While generating these parts, the computed shares are immediately ordered according to their destination participant towards whom these particular shares are intended.

In order to transmit all shares in a single publishment, the shares are encrypted using the public key of the intended recipient using the `EncryptShares` procedure. A participant can, upon receiving all his or her parts, decrypt using the `DecryptShares` procedure, which aggregates the parts corresponding to the same item and subtracts her original random value. This operation leaves the participant with a final random value which can be used in the next phase, masking.

#### 5.4.3 *Masking*

We mask each vote to hide the contents of the vote provided by a participant. This masking can be done by simply adding the previously computed random value, to the corresponding vote value creating a masked value now containing the contents of the participant's vote (note that this can also be negative). Now, the resulting masked value can be shared with the other participants as can also be seen in Algorithm 2. Every participant  $\pi_i \in \Pi$ , where  $i$  is the participant identifier, executes this protocol for each item  $i \in \mathcal{J}$ . All masked votes can be shared in a single transaction, where a vote is contributed in concatenation with the hash of the original vote value. By doing so, it prevents the originating party to alter the original contributed value if he or she were to reveal the hidden information.

#### 5.4.4 *Aggregation*

Now that the votes have been cast during the previous phase of the protocol, the totals can be computed. In doing so, this phase only requires the addition of the votes belonging to a to be labeled item due to the extensive work

---

**Algorithm 1** Random number generation protocol.

---

```

1: procedure GENERATERANDOMNUMBERSHARES( $n, m$ )
2:    $R_{\pi_i} \xleftarrow{r} \{0, 1\}^\xi$ 
3:   sharesPerRecipient  $\leftarrow \emptyset$ 
4:   totals  $\leftarrow []$ 
5:   for  $j$  in range( $n$ ),  $j \neq i$  do
6:      $R_{\pi_i, \pi_j} \leftarrow \emptyset$ 
7:     for item in range( $m$ ) do
8:       newShare  $\xleftarrow{r} \{0, 1\}^\xi$ 
9:       totals[item] += newShare
10:       $R_{\pi_i, \pi_j}.$ include(newShare)
11:     sharesPerRecipient.include( $R_{\pi_i, \pi_j}$ )
12:    $R_{\pi_i, \pi_i} \leftarrow \emptyset$ 
13:   for item in range( $m$ ) do
14:      $R_{\pi_i, \pi_i}.$ include( $(R_{\pi_i}[\text{item}]) - \text{totals}[\text{item}] \bmod p$ )
15:   sharesPerRecipient.include( $R_{\pi_i, \pi_i}$ )
16:   return sharesPerRecipient

17: procedure ENCRYPTSHARES( $R, PK_N$ )
18:   encryptions  $\leftarrow \emptyset$ 
19:   ctr = newCounter()
20:   for  $x$  in range( $n$ ) do
21:     encryptions.include( $\mathcal{E}(R_{\pi_i, \pi_x} | \text{ctr})$ )
22:   return encryptions

23: procedure DECRYPTSHARES( $\mathcal{E}(R), SK_i, R_{\pi_i}$ )
24:    $R_{f, \pi_i} \leftarrow \emptyset$ 
25:   for  $e$  in range( $m$ ) do
26:      $r_{f, \pi_i}(e) \leftarrow 0$ 
27:     for  $d$  in range( $n$ ) do
28:        $r_{f, \pi_i}(e) += \mathcal{D}(\mathcal{E}(R | \text{ctr})[d])$ 
29:      $r_{f, \pi_i}(e) - = R_a[p]$ 
30:      $R_{f, \pi_i}.$ include( $r_{f, p_i}(e)$ )
31:   return  $R_{f, p_i}$ 

```

---



---

**Algorithm 2** Masking procedure, hiding the prediction values.

---

```

1: procedure MASKVOTES( $R_{\pi_i, s_j}, \Gamma_{\pi_i}$ )
2:    $\mathcal{M}_{\pi_i} \leftarrow \emptyset$ 
3:   for  $j$  in range( $m$ ) do
4:     maskedValue  $\leftarrow R_{\pi_i, s_j} + \Gamma[i]$ 
5:      $\mathcal{M}_{\pi_i}.$ insert(maskedValue)
6:   return maskedVotes

```

---

done in the random number generation phase. When adding all seemingly random-looking numbers, there will be a result that corresponds to the same value that would have been attained when all individual votes would have been aggregated without the use of the random numbers, as these values sum up to zero.

#### 5.4.5 Noise addition and final model generation

Once the entire dataset is labeled, noise needs to be added according to the process described by Papernot et al. [95]. A participant proposes a noise distribution to extract perturbations from, which needs to be approved by the majority of other participants. Finally, each individual can choose to train their personal model, with desired structure and parameter settings, or propose a global model which participants can validate and use.

### 5.5 SECURITY ANALYSIS

It is crucial for this protocol to provide a secure environment within the assumed semi-honest or "honest-but-curious" threat model. Masking is the selected technique for this protocol to hide the sensitive information, the vote, and thereby providing confidentiality of the data. All participants generate the random value used to perform the masking. Thus, a proper unmasking of the value provides a guarantee that all participants have only submitted one vote for this item, and that all allowed participants have contributed at least one vote.

We assume that the input  $q_{\pi_i, s_j}$  for each  $\pi_i \in \Pi$  and  $s_j \in S$  needs to be protected. Additionally, the intermediately computed random parts that a participant  $\pi_i$  sends to a participant  $\pi_j$  and vice versa are confidential, only available to the receiving and originating participants. This confidentiality must be provided to such a degree that the final masking values  $\tilde{R}_{\pi_i, s_k}$  are indistinguishable from truly random numbers. The output of the protocol is visible to all participants and does not need to be private.

We assume a semi-honest adversary in a protocol setting where we have an honest majority. The nodes controlled by the adversary follow the protocol specification precisely, while collecting all information from the transcript of messages and the nodes internal states. The privacy of the contributions provided by the participants is argued using a simulation-based security deduction [76].

**Definition 5.1.** A function  $\alpha(\cdot)$  represents a negligible function if for every polynomial  $p(\cdot)$  and a sufficiently large security parameter  $\kappa \in \mathbb{N}$ ,  $\alpha(\kappa) < 1/p(\kappa)$ .

**Definition 5.2.** Computational Indistinguishability Given security parameter  $\kappa$ , and the input provided, let  $X = X(\text{input}, \kappa)$  and  $Y = Y(\text{input}, \kappa)$ . The functions  $X$  and  $Y$  are indistinguishable, denoted as  $\stackrel{c}{\equiv}$  if for every non-uniform probabilistic-time algorithm  $D$ , the distinguisher, a negligible function  $\alpha(\cdot)$  exists such that:

$$|\Pr[D(X(\text{input}, \kappa)) = 1] - \Pr[D(Y(\text{input}, \kappa)) = 1]| \leq \alpha(\kappa) \quad (5.14)$$

**Definition 5.3.** Security

- Let  $f = (f_1, f_2, \dots, f_n)$  be a functionality. We say that  $\Phi$  securely computes  $f$  in the presence of static semi-honest adversaries, if an adversary cannot distinguish the protocol  $\Phi$  from an ideal functionality  $f$ . Meaning that the adversary cannot obtain any additional information.
- The view of the  $i$ -th party ( $i \in \mathcal{J}$ ) during an execution of  $\Phi$  on  $(\Gamma_{\pi_i}, R_{\pi_i})$  and security parameter  $\kappa$  is denoted by  $\text{view}^\Phi((\Gamma_{\pi_1}, R_{\pi_1}), (\Gamma_{\pi_2}, R_{\pi_2}), \dots, (\Gamma_{\pi_n}, R_{\pi_n}), \kappa)$  and equals  $(w, r^i, m_1^i, \dots, m_k^i)$ , where  $w \in (\Gamma_{\pi_i}, R_{\pi_i})$  (its input depending on the value of  $i$ ),  $r_i$  equals the contents of the  $i$ -th party's internal random tape  $r$ , and  $m_k^i$  represents the  $k$ -th message that it received.
- The output of the protocol  $\Phi$  on input  $(\Gamma_{\pi_1}, R_{\pi_1}, \Gamma_{\pi_2}, R_{\pi_2}, \dots, \Gamma_{\pi_n}, R_{\pi_n})$  and security parameter  $\kappa$  is denoted by  $\text{output}^\Phi(\mathcal{M}'_{\Pi})$  and can be computed from all views of the execution.

We say that Protocol  $\Phi$  securely computes  $f$  in the presence of  $\rho$  semi-honest adversaries - simulators -  $S_1, S_2, \dots, S_\rho$  such that:

$$S_i(1^\kappa, \Gamma'_{S_i}, R'_{S_i}), f_i(\Gamma_{\pi_1}, R_{\pi_1}, \Gamma_{\pi_2}, R_{\pi_2}, \dots, \Gamma_{\pi_n}, R_{\pi_n}) \stackrel{c}{=} \text{view}_i^\Phi((\Gamma_{\pi_1}, R_{\pi_1}, \Gamma_{\pi_2}, R_{\pi_2}, \dots, \Gamma_{\pi_n}, R_{\pi_n}), \kappa) \quad (5.15)$$

The communications of a single participant  $\pi_i$  with the other participants is visualized in Algorithm 3. All outgoing arrows represents information send to its peers by participant  $\pi_i$ , whereas the incoming arrows are all messages that are received from the other participants. The transmission

**Algorithm 3** Communication of a participant  $\pi_i$  with the protocol  $\Phi$ 

$\pi_i$	$\Phi$
$\text{input}_{\pi_i}^\Phi(\Gamma_{\pi_i}, R_{\pi_i})$	$\text{output}^\Phi(\mathcal{M}_{\pi_i})$
$R = \{R_{\pi_i, \pi_j}   j \neq i\}$	$R_J = \{R_{\pi_j, \pi_i}   j \neq i\}$
$\check{R}_{\pi_i, s_k} = \sum_{j=0}^n R_{\pi_j, \pi_i}(k) - r_{ik}$	
$\check{R}_{\pi_i} = \{\check{R}_{\pi_i, s}   s \in S\}$	
$\mathcal{M}_{\pi_i} = \Gamma_{\pi_i} + \check{R}_{\pi_i}$	$\mathcal{M} = \{\mathcal{M}_s   s \in S\}$
	$\Delta_s = \sum_{i=0}^n \check{m}_{i, s}$
	$\Delta = \{\Delta_s   s \in S\}$

tape and internal state of a single participant  $\pi_i$  consists of the following attributes:

1. His or her own to be hidden contributions  $\Gamma_{\pi_i}$ .
2. The received random parts  $R_{\pi_j, \pi_i}$  from all participants  $\pi_j$  where  $j \neq i$ .
3. All encrypted random shares destined for other participants, denoted as  $\mathcal{E}(R_{\pi_v, \pi_w})$ , where  $w \neq i$  and  $v \neq w$ .
4. The final output of the aggregate of all inputs  $\Delta$ .

We assume an ideal functionality  $f$  that simulates a trusted third party that exchanges the relevant information, which has access to perfect encryption and secure communication channels. Additionally, the random numbers originate from true randomness. The transferred information between a participant and the functionality  $f$  is visualized in Algorithm 7.

---

**Algorithm 4** Communication  $S_i$  with the ideal functionality  $f$

---

$S_i$		$f$
$\text{input}_{S_i}^f(\Gamma'_{S_i}, R'_{S_i})$		$\text{output}^f(\mathcal{M}_{S_i})$
$R' = \{R'_{S_i, \pi_j}   j \neq i\}$	$\xrightarrow{R'}$	
	$\xleftarrow{R'_j}$	$R'_j = \{R'_{\pi_j, S_i}   j \neq i\}$
$\check{R}'_{S_i, s_k} = \sum_{j=0}^n R'_{\pi_j, S_i}(k) - r'_{ik}$		
$\check{R}'_{S_i} = \{\check{R}'_{S_i, s}   s \in S\}$		
$\mathcal{M}'_{S_i} = \Gamma'_{S_i} + \check{R}'_{S_i}$	$\xrightarrow{\mathcal{M}'_{S_i}}$	
		$\mathcal{M}' = \{\mathcal{M}'_s   s \in S\}$
		$\Delta'_s = \sum_{i=0}^n \check{m}'_{i, s}$
	$\xleftarrow{\mathcal{M}', \Delta'}$	$\Delta' = \{\Delta'_s   s \in S\}$

---

The ideal functionality  $f$  makes it impossible for participants to access random shares not meant for them, as only those can be decrypted. Meanwhile, the masks are generated in such a way that an adversary cannot distinguish between a random value and the masked votes. Only allowing the adversary to gain insights into the total aggregate of votes.

**Theorem 1.** The protocol  $\Phi$  securely and privately computed the aggregate functionality  $f$  to obtain  $\text{output}^\Phi(\mathcal{M}_\Pi)$ .

Simulator  $S_i$  executes the following steps to simulate the view of party  $i$ :

1. The Simulator generates true random parts  $R'_{S_i}$ , different from those generated by a generic participant  $\pi_i$  who generates random parts  $R$ .
2. The Simulator generates contributions  $\Gamma'_{S_i}$  out of the same distribution as an actual  $\Gamma_{\pi_i}$ .
3. The Simulator computes masked values to attain  $\mathcal{M}'_{S_i}$ , in a similar way a participant  $\pi_i$  would compute  $\mathcal{M}_{\pi_i}$ .



The simulated view can be represented as:

$$\{S_i(1^\kappa, \Gamma'_{S_i}, R'_{S_i}), f_i(\Gamma_{\pi_1}, R_{\pi_1}, \Gamma_{\pi_2}, R_{\pi_2}, \dots, \Gamma_{\pi_n}, R_{\pi_n})\} := \{(1^\kappa, \Gamma'_{S_i}, R'_{S_i}), \mathcal{M}'_{S_i}, \Delta'\} \quad (5.16)$$

$$\text{view}_i^\Phi((\Gamma_{\pi_1}, R_{\pi_1}, \Gamma_{\pi_2}, R_{\pi_2}, \dots, \Gamma_{\pi_n}, R_{\pi_n}), \kappa) := \{(1^\kappa, \Gamma_{\pi_i}, R_{\pi_i}), \mathcal{M}_{\pi_i}, \Delta\} \quad (5.17)$$

For any distinguisher  $D$  and negligability function  $\alpha(\cdot)$ :

$$|\Pr[D(1^\kappa, \Gamma'_{S_i}, R'_{S_i}), \mathcal{M}'_{S_i}, \Delta'] = 1] - \Pr[D(1^\kappa, \Gamma_{\pi_i}, R_{\pi_i}), \mathcal{M}_{\pi_i}, \Delta] = 1]| \leq \alpha(\kappa) \quad (5.18)$$

An adversary controlling  $\rho$  nodes, where  $\rho$  represents a minority, thus has access to a subset of the shared information. The random shares <sup>1</sup> in  $R_j$  are encrypted using the assumed semantically secure [AES](#). The provided confidentiality causes an adversary only to observe the shares aimed for or originating from the  $\rho$  participants. Every masking value is dependent on one random share from each participant, and the original random value generated by the victim. This dependency can be shown as follows:

$$\check{R}_{\pi_v, s_k} = \sum_{j=0}^{\rho} R_{\pi_j, \pi_v}(k) + \sum_{l=\rho+1}^n R_{\pi_l, \pi_v}(k) - r_{vk} \quad (5.19)$$

for victim  $\pi_v$ . The adversaries are unable to reconstruct  $r_{vk}$ , as this value is split into  $n$  parts from which the adversary is only able to access  $\rho$ . Furthermore, the second term of equation 5.19 consists of unknown random shares that form the majority of the final masking value. Therefore, due to the inability of reconstructing both the original random value and the total share aggregation term, an adversary controlling  $\rho$  nodes is unable to obtain masking values used by honest participants.

If an independent random value generated from a uniform distribution is added to a value, the result remains random. Thus, an adversary cannot retrieve any information from the masked votes if a sufficiently large masking value is used.

Therefore, by analyzing all the information seen by the adversaries, even when controlling multiple parties, he or she is not able to retrieve information on the individuals' vote. Therefore, we can state that the proposed protocol securely evaluates the function  $f$  in the presence of semi-honest adversaries.

The bit size of the used variables play a vital role in providing suitable security in the encryption, and for the masked values to provide suitable security. In the following paragraphs, we address both the encryption and masking steps, and the minimum required security bit sizes be proposed.

The mask is required to be of a particular bit size to obfuscate the original message. Over time, the minimum required bit size changes and should thus

<sup>1</sup> Generated using the Yarrow algorithm [126] standard on MacOS assumed to generate truly random numbers based on real-world entropy sources.

be adjusted according to the appropriate value at the time of implementation [6]. Currently, the bit size that provides proper security corresponds with the bit size of the message plus 112 bits according to the current NIST standards for key lengths [6]. Here the bit size of the message with the added 112 bits can be described as follows:

$$\xi = u \cdot \beta + \eta + 112 \quad (5.20)$$

During the initialization a prime  $p$  of bit size  $\xi$  is determined,  $p \in \{0, 1\}^\xi$ . To generate random mask values of that particular bit size, the first randoms  $R$  that are generated by the participants per item should originate from the set  $R \in \{0, 1\}^{\lfloor \log_2 p \rfloor}$ , where the values are of bit size  $\xi$ , and less than  $p$ . Afterward,  $n - 1$  random values are generated with the same bit size for each of the other participants. As shown in Protocol 1, the final random parts are computed to make sure that the sum of all parts mod  $p$  is equal to the original random value. Finally, these parts received per item are aggregated, and the original random numbers are deducted, leaving one with a random value of  $\xi$  bits.

The encryption done to send the masked parts requires a specific key size. Today's security standards warrant the use of a 128-bit key, mapping every encryption to a ciphertext consisting of a multiple of this bit size rounded up. An additional 128 bits are required to provide the counter used for encryption.

In order to assume AES to be secure, we need to ensure that the number of ciphertexts and encrypted blocks generated with the same key preserves security. Therefore, we need to be able to alternate the used symmetric key after a certain amount of encryptions which can be determined as follows. Let us assume that the ratio of the exposed key information to the key size should be less than  $2^{-32}$ . The following equation will need to hold for the counter-mode of operation:

$$\frac{Q^2 l^2}{2^{128}} < 2^{-32} \quad (5.21)$$

$$Q^2 l^2 < 2^{96} \quad (5.22)$$

$$Ql < 2^{48} \quad (5.23)$$

where  $Q$  is the number of encryptions and thus always a positive integer and  $l$  is the number of blocks. Every symmetric key corresponds to a pair of participants and thus will be used by two individuals for transmitting  $m$  random parts of  $\lceil \frac{\xi}{128} \rceil + 1$  blocks. This makes the total number encryptions  $= m \cdot \lceil \frac{\xi}{128} \rceil + 1$ , which even for high levels of  $m$  is less than  $2^{48}$ . Therefore, the AES scheme is secure if the key is renewed with every protocol execution. Every pair of participants has a shared secret counter, allowing them to alter their shared secret without additional communication. This counter can either be increased or decreased by one every new protocol execution allowing both participants to compute a new AES key without any additional interaction between the parties.

## 5.6 COMPLEXITY ANALYSES

We conduct both a communication and computation analysis in Sections 5.6.1, and 5.6.2 respectively, through which we attempt to assess the proposed protocol.

### 5.6.1 Computational analysis

The computational complexity of ECONoMy is analyzed by reviewing the required number of operations per phase per party. The required operations are dependent on the number of participants  $n$ , as well as the number of items in the public data set  $m$ . The chosen size of the public dataset will depend on multiple factors, such as availability (i.e., what data can be acquired), as well as privacy requirements. The privacy obtained by transferring knowledge from the ensemble to the public data provides a bound to the number of queries done to the ensemble and limits privacy. The number of items in this public dataset resembles this bound and represents a trade-off between the amount of knowledge transferred that is available to train the global model on, and the level of privacy offered to the local training data. The number of participants is assumed to be smaller than the number of items to be labeled, even though this is not a constraint of the protocol. At least three parties should participate. When  $n = 2$ , the masking will be ineffective as a participant can retrieve the vote of its collaborator by detracting her own vote from the total.

Table 7 shows the number of mathematical operations that occur in each phase. ECONoMy provides a lightweight solution, where the random number generation can be done efficiently due to the use of the symmetric cryptosystem AES opposed to an asymmetric cryptosystem. Moreover, the masking only consists of adding the random values to the votes, and the aggregation only consists of a limited amount of linear operations. Furthermore, it is important to reiterate that the operations are performed locally, in a distributed fashion. This parallelization means that the encryptions in the per party column are most relevant as each party will be executing the operations in parallel.

### 5.6.2 Communication analysis

Information is transferred between participants in different phases of the ECONoMy protocol. The sizes of these messages are shown in Table 8, and use the symbols defined in Table 6. The **Total** column shows the addition of all messages combined in each of the phases, whereas the **Per Party** column indicates the message sizes send or received per participant. It becomes apparent that the total amount of bits transferred in one execution of the protocol is equal to  $mn^2\psi - mn\psi + nm\xi + m\lambda$ .

The majority of the transferred bits originate from the random number generation, which depends on the number of participants, the number of items, as well as the size of the ciphertext transferred. Inference voting requires a smaller message size, since it does not have a quadratic dependency on the number of included participants. An example size of a

Table 7: Review of all operations occurring in a single execution of the ECONoMy protocol to conclude the computation complexity.

Phase	Operation	Total	per party
Initial Setup	Key Generations	$n$	1
Random Number Generation	Random number generation	$n \cdot m$	$m$
	Additive secret sharing	$n \cdot m$	$m$
	Encryptions/Decryptions	$2n^2m$	$2nm$
	Additions	$m \cdot n$	$m$
Masking	Additions	$n \cdot m$	$m$
Aggregating	Additions	$n \cdot m$	$n \cdot m$
	Maximum	$m$	$m$

masking value can be constructed as follows. If we assume the presence of 5 different classes ( $u = 5$ ),  $\beta = 10$ , and we append 10 zeros to identify a correct demasking, the correct masking size would be 172 bits. This would require two separate AES-128 blocks, in the encryption, accompanied with an IV! of 128 bits.

Table 8: An overview of all communications needed for ECONoMy, and the size of the transferred information.

Phase	Operation	Total	Per party
Random Number Generation	Send	$mn^2\psi - mn\psi$	$mn\psi - m\psi$
	Retrieve	$mn^2\psi - mn\psi$	$mn\psi - m\psi$
Inference voting	Sharing votes	$nm\xi$	$m\xi$
Labeling	Publishing final labels	$m\lambda$	$m\lambda$

## 5.7 DISCUSSION

To the best of our knowledge, ECONoMy represents the first privacy-preserving collaborative model generation using the distributed ensemble learning approach. Assuming a passive, semi-honest adversary the privacy risks inherent to the distributed nature are overcome using symmetric encryption, masking, and differential privacy. After distributing random numbers among the participant, a privacy-preserving voting procedure allows for the accrued knowledge within the local models to be transferred to the labels given to the public data set.

First of all, by using encryption, the required number of communications is reduced, and the revelation of random shares is prevented as discussed in Section 5.2. Furthermore, during the inference voting phase, the lightweight masking procedure hides all individual contributions in such a way that no intermediate value can be determined. This obfuscation protects individual models' training data against targeted attacks by removing the link between the individual vote and the participant. Furthermore, the intuitive privacy benefit inherent to this approach semi-supervised learning, suggested by

Papernot et al. [95], allows for a fixed privacy loss of the original training data set (combined over all local models) as only  $m$  queries are made to the ensemble as a whole. If this ensemble were to be released directly, the privacy loss does not have such an upper bound. Finally, by adding calibrated noise, the outcomes generated by white box attacks are hidden under differential privacy.

The most expensive computations are needed during the random number generation phase, originating from the encryption of the random shares. These operations bring the overall computational complexity of the protocol to  $O(n^2m)$ , and  $O(nm)$  when considering the parallel nature of the protocol. The inference voting phase itself requires every participant posting each of their votes, which is the under-bound required by the operation at hand. Similarly, the most significant communication costs also occur within the random number generation phase. Many encrypted parts need to be transferred and retrieved separately.

Chapter 4 indicates several research challenges that needed to be taken into account, given the assumed threat model.

- *Limited insights into participant behavior* - The masking hides the sensitive information when a sufficient security parameter is chosen. Other participants can see that a specific participant has contributed, but is unable to inspect the obfuscated content.
- *No double spending* - As all participants can verify which participants have contributed to a specific item, it is infeasible for a vote to be included if the originating participant has already voted before.
- *Verifiability and consensus* - The use of a permissioned blockchain allows all participants to agree on the state of the protocol. Because the blockchain is open to all participants, everyone can verify the executed steps and recompute the labels themselves if so desired.
- *Auditability* - Every participant retains a link to their vote, and if insights to a regulatory entity have to be given, the participant can provide their mask as well as original vote. This auditability becomes feasible as a participant provides a hash of the original vote with a contribution, acting as a commitment to the underlying vote value.
- *Reasonable efficiency* - The use of masking rather than more expensive techniques such as homomorphic encryption makes that the inference phase is very efficient. The random number generation has a quadratic complexity, where AES encryption is the most computationally expensive operation. The practical computational and communication complexity will be analyzed in Chapter 8.

ECONoMy provides a very lightweight solution to privacy-preserving decentralized collaborative learning. The protocol is ideal for a use case where a large number of participants are considered to be cooperating in a 'semi-honest' environment. A frequent use case present within the field of

collaborative learning, as also used by Hamm et al. [54], is for the internet of things (IoT) devices. Where Hamm et al. [54] require transferring the entire model with the privacy issues accompanying such an approach, ECONoMy allows for a similar ensemble approach with added privacy to the environment in which the device gathers its data. The lightweight nature allows for fast computation even on resource-constrained devices often present in the IoT space, especially when a dedicated AES hardware solution were to be present in the device. Furthermore, a manufacturer of IoT devices would not need to update the present Diffie-Hellman key once deployed due to the ability to generate symmetric keys using the random number communicated between participants. Nevertheless, we can argue that the 'semi-honest' threat model does not accurately reflect reality. This is why we have constructed our next design that assumes a stronger, more maneuverable adversary.

## PRECLUDE: PRIVACY-PRESERVING COLLABORATIVE LEARNING USING A DECENTRALIZED ENSEMBLE APPROACH

---

ECONoMy offers a variety of benefits for participants who are willing to assume a 'semi-honest' threat model. In practice the potential benefits a deviant could attain by misbehaving can be extremely large. A corrupt organization profiting from the lack of fraude detection might not want the detection models to be improved across different participants, as it could potentially cause them to lose their income. Incentives such as these have driven us to create another solution that is robust against stronger security assumptions. The second protocol that we present is PRECLUDE, a privacy-preserving approach to collaborative learning that uses traceable ring signatures([TRS](#)) to provide sender verifiability while retaining anonymity.

In PRECLUDE, rather than hiding the contents of the individual votes, we remove the identity of the sender. By using a [TRS](#), each participant can prove that he or she is allowed to vote for a specific item, without leaking any information about from whom the vote originates. Only when a malicious participant attempts to vote more than once for a particular item, the identity of the person in question will be made public when these signatures are traced. All verified and traced votes are included in the final aggregate. This aggregate is subject to a label selection technique, i.e., majority vote, to label the underlying samples, after the application of noise to achieve differential privacy. The fact that there remains complete transparency of the executed protocol steps during the entire process adds to the overall confidence the individual parties have in the final result.

The robustness provided by these [TRSS](#) allows the influence of an adversarial participant to be limited to its own allowed input, meaning  $1/n$ . Furthermore, there is no direct need to hide the vote values as a vote can not directly be associated with a particular sender. So in short, it can be verified that the sender is part of a valid set, double voting leaks the senders' identity, and we can transmit votes in the clear due to the computational infeasibility of reconstructing a sufficiently large set of corresponding votes.

To the best of our knowledge, the PRECLUDE protocol is the first distributed collaborative learning protocol that protects against adversaries under the covert threat model. It enables organizations who attempt to adhere to legislation while building the best possible models across borders.

This chapter consists of both the description and evaluation of PRECLUDE. First, we will provide an overview of the required mathematical notations used throughout the chapter in Section [6.1](#). The included cryptographic primitives will be introduced in more detail in Section [6.2](#). First section [6.3](#) provides an intricate description of the design

of the protocol, after which section 6.4 provides a phase-by-phase deep-dive. Sections 6.5, and 6.6 analyze both the computational costs of the protocol and the corresponding security. Finally, a discussion on the protocol is included in Section 6.7.

## 6.1 NOTATION

We will frequently need to refer to certain variables within this design chapter. Table 6 contains a full overview of all the used notations, to which the reader can refer for any unknown definitions.

Table 9: Explanation of the additional symbols used in PRECLUDE.

SYMBOL	EXPLANATION
$\omega$	The size of a vote.
$\gamma$	The bitsize of the used safe-primes $p, q$ .
$\zeta$	The size of a single signature.
$\chi$	The size of the item number.
$\tau$	The number of traces per to be labeled item.
$\nu$	The number of verifications per provided signature.
$\upsilon$	The number of equal $\sigma$ 's in the tracing procedure.
$V_{\pi_i}$	Set of verify outcomes provided by participant $\pi_i$ .
$T_{\pi_i}$	Set of trace outcomes provided by participant $\pi_i$ .
$\mathcal{H}$	Denotes the use of a hash function.
$G$	A multiplicative group.
$g$	Generator of the multiplicative group $G$ .
$q$	The order of the multiplicative group $G$ .
$PK_J$	Ordered list of all public keys used in ring signature.
$\Sigma$	Denotes the signature returned by performing a traceable ring signature.
$L$	Denotes the used tag (issue, $PK_n$ ).
$\Lambda$	Set of signature-vote pairs $\{\lambda_{\pi_i}   \pi_i \in \Pi\}$ .
$\lambda_{\pi_i, s_k}$	A signature-vote pair from participant $\pi_i$ for an item $s_k$ , $\lambda_{\pi_i, s_k} = (\Sigma_{\pi_i, q, \pi_i, s_k})$ .

## 6.2 CRYPTOGRAPHIC PRELIMINARIES

In this section, we will expand upon the introduction given for a cryptographic preliminary that is applied within the PRECLUDE protocol. The PRECLUDE protocol uses an adaptation of the traceable ring signature scheme proposed by Fujisaki and Suzuki [43]. Traceable ring signatures allow a signer to prove that he or she originates from a group of participants, without identifying the actual signer. Furthermore, the ability to trace signatures limits this anonymity to a predefined number of allowed signatures per tag. If such anonymity were not to be contained, an



adversary would be able to arbitrarily sign messages, obstructing the protocol, without any repercussions.

**TRACEABLE RING SIGNATURES** Traceable ring signatures are used to sign a particular message  $m'$  concerning a tag  $L$ . The tag consists of an *issue* and the list of all public keys  $PK_j$ . The issue represents an election for which a participant can vote a predefined amount of times. In our use of the traceable ring signatures protocol, the participants are allowed to sign exactly once. The identity of the signer is released when this frequency is exceeded for a specific tag, thereby absolving the signer of his or her anonymity. The protocol assumes the presence of three hash functions  $\mathcal{H}, \mathcal{H}'$ , and  $\mathcal{H}''$  that act as random oracles, which have as introduced in Chapter 2. The functions  $\mathcal{H}, \mathcal{H}'$  map to a value within the group  $G$ , and  $\mathcal{H}''$  maps to a value within  $\mathbb{Z}_q$ , where  $q$  represents the prime-order of group  $G$ .

Fujisaki and Suzuki [43] provide four security requirements, with corresponding security proofs, for their traceable ring signature protocol.

1. **Public traceability** - Any participant that publishes multiple signatures for different messages corresponding to the same tag will successfully be traced and identified.
2. **Tag-linkability** - All signatures based on the same tag, generated by a specific participant that will be linked. This constraint means that there can be at most  $n$  valid signatures, corresponding to a single tag, when there are  $n$  ring-members if none of them are linked.
3. **Anonymity** - The identity of the signer remains hidden if the signer does not sign different messages with respect to the same tag. If a participant generates two signatures for distinct tags, it is infeasible for anyone to determine that the same signer computed them.
4. **Exculpability** - An adversary cannot force identity leakage of another participant by replaying a signature, even if  $n - 1$  participants are compromised, and a polynomial amount of signatures of the targeted participant are available.

The original traceable ring signature scheme consists of four different algorithms. We will give a detailed description of each of these algorithms.

**GENERATE** The **GEN** algorithm focuses on the generation of the key-pairs for each of the participants. A security parameter  $\kappa$  is provided as input to generate a keypair  $(pk_i, sk_i)$  for  $i \in \mathcal{I}$ , where  $\mathcal{I}$  is the set of all identifiers. The secret key  $sk_i$  is a random value  $x_i \in \mathbb{Z}_q$ , where  $q$  is the prime order of the Abelian group  $G$ . The public key is derived using this private  $x_i$ , and the generator  $g$  of the group  $G$ , to compute  $pk_i = y_i = g^{x_i}$ .

- Input: A security parameter  $\kappa$ , an Abelian group  $G$ , its prime order  $q$ , and the generator  $g$ .
- Output: Key pair  $(pk_i, sk_i) = (g^{x_i}, x_i)$ .

**SIGNING** The first step in the signing process is the computation of the signer's own sub-signature  $\sigma_i$  using the **SIG** algorithm, where  $i \in \mathcal{J}$  is the identifier belonging to the signer. First a variable  $h$  is computed by hashing the tag  $L$ ,  $h = \mathcal{H}(L)$ . We sign this hash using the secret key by computing  $\sigma_i = h^{s_{k_i}} = h^{x_i}$ .

Next, the signer generates values representing the  $\sigma$ 's of the other participants. To generate these  $\sigma$ 's, the participant generates base values  $A_0$  and  $A_1$  using the following functions:

$$A_0 = \mathcal{H}'(L, m) \text{ and } A_1 = \left( \frac{\sigma_i}{A_0} \right)^{1/i} \quad (6.1)$$

where  $m$  is the message being signed. For each participant  $j \in \mathcal{J}$  for  $j \neq i$  we compute  $\sigma_j = A_0 A_1^j$ . All the computed values serve as an input to generate the actual signature values, which is done using a non-interactive zero-knowledge proof on the following language:

$$\mathcal{L} \triangleq \{(L, h, \sigma_N) \mid \exists i' \in \mathcal{J} \text{ such that } \log_g(y_{i'}) = \log_h(\sigma_{i'})\} \quad (6.2)$$

This statement states a secret key is included in the generation of one of the  $\sigma$ 's, which is proven by executing the following four steps.

1. Choose a random  $r_i \in \mathbb{Z}_q$  to compute  $a_i = g^{r_i}$  and  $b_i = h^{r_i}$ .
2. For every  $j$ , where  $j \neq i$ , choose two random values  $z_j, c_j \in \mathbb{Z}_q$  and compute  $a_j = g^{z_j} y_j^{c_j}$  and  $b_j = h^{z_j} \sigma_j^{c_j}$ .
3. Compute  $c = \mathcal{H}''(L, A_0, A_1, a_N, b_N) \pmod q$  where  $a_j, b_j$  are lists of all computed  $a$  and  $b$  values.
4. Determine  $c_i = c - \sum_{j \neq i} c_j$  and  $z_i = r_i - c_i \cdot x_i \pmod q$  where the values of  $c$  and  $z$  are returned as the proof of language  $\mathcal{L}$ .

Finally, the signing algorithm returns a traceable ring signature which is defined as follows  $\sigma = (A_1, c_j, z_j)$ . Where  $c_j$  and  $z_j$  are a collection of  $c$  and  $z$  values corresponding to all participants included in the signature.

- **Input:** The set of public keys  $Pk_j$ , the signer's private key  $x_i$ , the message to be signed, and the corresponding tag.
- **Output:** A signature that contains a proof that there a private key that corresponds with a public key included in the ring, has been used to generate it  $\Sigma = (A_1, c_j, z_j)$ .

**VERIFICATION** In order to start the verification, the provided values need to be validated. Thus, a verifier checks whether the provided  $A_1$ , and all  $y_i$  are within the group  $G$ , and all  $c_i$  and  $z_i$  values are within  $\mathbb{Z}_q$ . Afterward, similar to the signing process, variables  $h$  and  $A_0$  are computed. The verification of a signature slightly differs from the signing process, as it needs to compute all values without knowing who the signer  $i$  is. This

means that all  $\sigma$ 's,  $a$ 's and  $b$ 's are generated using the same function for all  $i \in \mathcal{J}$ , namely:

$$\sigma_i = A_0 A_1^i \quad (6.3)$$

$$a_i = g^{z_i} y_i^{c_i} \quad (6.4)$$

$$b_i = h^{z_i} \sigma_i^{c_i}. \quad (6.5)$$

In order for the signature to be valid, computing the value  $c$  should be equal to the sum of all the provided values within  $c_j$ .

$$\mathcal{H}''(L, A_0, A_1, a_j, b_j) \bmod q \stackrel{?}{=} \sum_{i \in \mathcal{J}} c_i \bmod q \quad (6.6)$$

Any of the participants could be the signer, since the  $c$  and  $z$  values for that individual are dependent on all others. This attribute also indicates how the ring is closed. The  $\sigma_i$ ,  $a_i$ , and  $b_i$  of the signer need to be result in the same value while being computed using a different function in the verification phase. We will show that the verify function follows from the function used in signing, in equations 6.7.

$$\begin{aligned} \sigma_i &= A_0 A_1^i \\ &= A_0 \left( \left( \frac{\sigma_i}{A_0} \right)^{1/i} \right)^i \\ &= A_0 \left( \left( \frac{h^{x_i}}{A_0} \right)^{1/i} \right)^i \\ &= h^{x_i} \end{aligned} \quad (6.7)$$

Similarly, we will show that the computation of the  $a$  and  $b$  variables converts as well. We reason from the formula used in the verification towards the formula used in the signature. The conversion for both  $a$  and  $b$  depend on the final computation performed in the signing process:  $z_i = r_i - c_i \cdot x_i \bmod q$ . as shown in Equations 6.8 and 6.9.

$$a_i = g^{z_i} y_i^{c_i} = g^{z_i} (g^{x_i})^{c_i} = g^{z_i} g^{x_i c_i} = g^{z_i + x_i c_i} = g^{r_i} \quad (6.8)$$

$$b_i = h^{z_i} \sigma_i^{c_i} = h^{z_i} (h^{x_i})^{c_i} = h^{z_i} h^{x_i c_i} = h^{z_i + x_i c_i} = h^{r_i} \quad (6.9)$$

- Input: The to be verified signature  $\Sigma$ , the corresponding message  $m$  and tag  $L$ .
- Output: Accept or Reject the signature based on the final equality statement.

**TRACING** The **Trace** algorithm can be used to trace two signatures and verify whether the same participant has signed them with respect to the same tag. When tracing two signatures, there are three possible outcomes for signer  $i$  and  $i' \in \mathcal{J}$  as is shown in Equation .

$$\text{Trace}(L, m, \sigma, m', \sigma') = \begin{cases} \text{Independent} & \text{if } i \neq i' \\ \text{Linked} & \text{if } m = m' \\ \text{pk}_i & \text{if } i = i' \text{ and } m \neq m' \end{cases}$$

The first case covers distinct signers corresponding to the same tag. The second case is aimed to prevent adversaries to replay a signature to be able to leak the identity of the original sender. Finally, if both the signer id is the same and the messages differ, we release the identity  $\text{pk}_i$ .

Similarly to the verification, we compute all sigmas for both signatures after initializing the variables  $h$  and  $A_0$ . In order to determine which of the above-described scenario is applicable, we compare the computed sigma vectors and count which signatures are exact copies denoted as  $v$ . If  $v = 0$ , or  $1 < v < n$ , the signatures are independent, if  $v = n$ , the signatures are linked, and finally if  $v = 1$  the signatures fall into the last category, signalling a successful tracing.

- Input: Two signatures to be traced,  $\Sigma = (A_1, c_j, z_j)$  and  $\Sigma' = (A'_1, c'_{jN}, z'_j)$ , with their messages and the corresponding tag.
- Output: The result of the trace, either "Independent", "Linked", or the identity of the person attempting to sign multiple messages.

### 6.3 PROTOCOL OVERVIEW

We will first discuss the complete of the PRECLUDE protocol, after which the different steps will be highlighted separately. The PRECLUDE protocol consists of five phases:

1. The initial startup - A phase where the required infrastructure gets initialized and agree on the procedure. All participants generate their keys, publish their public keys, and compute their shared secrets.
2. Collaborative voting phase - Each participant contributes their votes with an accompanying TRS. Every identifier  $j$  corresponding to an item  $s_j \in S$  is used as the issue for TRS, and the message correspond to the prediction for the same item,  $q_{\pi_i, s_j} \in \Gamma_{\pi_i}$ .
3. Verification and Tracing phase - All provided votes are verified and traced according to the number of traces  $\tau$ , and verifications  $v$ , required for a vote to be considered valid. If this threshold is not met, the votes are discarded.
4. Aggregation phase - For each item,  $s_j \in S$ , those votes that have been successfully verified and traced, are aggregated to obtain the aggregate of all predictions.
5. Noise addition and final model generation - The noise required to obtain differential privacy on the predictions given by the released global model. Similarly, the final labels are devised using the selected classifier combination technique, and the training of a final model can commence.

Similar to the ECONoMy protocol introduced in Section 5, the aim is to prevent double voting, and remove the link between the vote and the originating participant. The PRECLUDE protocol attempts to do this in the covert threat model while reducing the possible impact an adversary can exert within the protocol by increasing the robustness.

The application of the TRS protocol allows a clear distinction between valid votes and those that need to be discarded. We are able to prevent double voting, remove replayed votes, and still retain a valid aggregate of the approved votes.

In establishing this protocol, we have made the following assumptions that should be taken into account when evaluating the design.

- We assume a ‘covert’ threat model, where adversaries can actively perform malicious activities and have a high probability of being caught when doing so.
- We assume that the participants can jointly validate certain execution steps, meaning that there is a certain degree of trust. If  $x$  participants have approved a contribution or the initial infrastructure, the others are willing to trust those  $x$  people to contain at least one honest verifier. There is thus a threshold of  $x$  approvals in order for an operation to succeed.
- We assume that the participants have access to anonymous routing, not allowing their originating IP-address to identify their contributions.
- We assume that the use of TRS provides secure signatures. This assumption inherently assumes the validity of the random oracle model and the decisional Diffie–Hellman assumption.
- We assume that all shared parameters are properly chosen as a participant’s identity is linked to their suggestion.

## 6.4 PHASE-BY-PHASE DESIGN DESCRIPTION

We will present the design of the PRECLUDE protocol per phase. Each of the phases will show the required procedures and their respective analyses.

### 6.4.1 *Initial setup*

The setup phase initializes the needed infrastructure for the protocol to commence. All participants identify themselves by linking their identity to their public key generated using the GEN procedure, denoted by  $y_{p_i}$  for  $1 \leq i \leq n$ . The necessary hash functions are communicated, and the corresponding keys for the initial two hash functions are determined. The global variables proposed, including the unlabeled public data set. These variables include the primes  $p$  and  $q$ , the generator  $g$ , and the list of all public keys  $PK_N$ . Any participant proposes the primes and generator and validated by the others before moving on to the next round. The validation requires a threshold value of the minimally required number of participants

to approve a certain operation. Finally, a time frame needs to be agreed upon within which all votes should be received.

#### 6.4.2 Collaborative voting phase

The collaborative voting phase allows every participant to share their predictions  $v$ , for each item  $w$  in the public dataset, while remaining anonymous. The sender can remain anonymous by providing a traceable ring signature as discussed in 6.2, that simultaneously validates that the sender is from the group  $\Pi$ , while defending against double spending. Algorithm 5 identifies three procedures: SignVote is the generation of the signature for a particular vote and PublishVote shares the combination of vote, signature, and item identifier with the other participants. Finally, the CastVotes procedure is invoked by all participants that aim to cast their predictions, linking the other procedures together.

---

**Algorithm 5** The procedure highlighting the required steps to cast a signed vote.

---

```

1: procedure CASTVOTES(votes, signerID, itemID)
2:   for  $v$  in votes do
3:      $\Sigma \leftarrow \text{SignVote}(\text{signerID}, \text{itemID}, v)$ 
4:     Invoke: PublishVote(vote,  $\Sigma_v$ , itemID)

5: procedure SIGNVOTE(signerID, itemID, vote)
6:    $i, w \leftarrow \text{signerID}, \text{itemID}$ 
7:    $h \leftarrow \mathcal{H}(L)$  for  $L = (w, \text{pk}_j)$ 
8:    $\sigma_i \leftarrow h^{x_i}$  for  $x_i \in \mathbb{Z}_q$ 
9:    $A_0 \leftarrow \mathcal{H}'(L, \text{vote})$ 
10:   $A_1 \leftarrow \begin{pmatrix} \sigma_i \\ A_0 \end{pmatrix}$ 
11:  for All  $j$  in range(1, n) do
12:    if  $j \neq i$  then
13:       $\sigma_j \leftarrow A_0 A_1^j \in G$ 
14:       $a_j \leftarrow g^{z_j} y_i^{c_j}, b_j \leftarrow h^{z_j} \sigma_j^{c_j}$  for  $z_j, c_j \xleftarrow{r} \mathbb{Z}_q$ 
15:       $a_i \leftarrow g^{r_i}, b_i \leftarrow h^{r_i}$  for  $r_i \xleftarrow{r} \mathbb{Z}_q$ 
16:       $c \leftarrow H''(L, A_0, A_1, a_N, b_N)$ 
17:       $c_i \leftarrow c - \sum_{j \neq i} c_j \pmod{q},$ 
18:       $z_i \leftarrow w_i - c_i x_i \pmod{q}$ 
19:      return  $\Sigma = (A_1, c_N, z_N)$ 

20: procedure PUBLISHVOTE(vote, signature, itemID)
21:   return vote || signature || itemID

```

---

Every participant in  $\Pi$  will provide their prediction for each item in  $S$ , as shown in the CastVotes procedure, which executes the three other procedures  $m$  times. Every participant will sign, encrypt, and publish each vote in order to move on to the next stage. As mentioned before, a deadline

condition will be set to prevent any malicious participant from preventing the protocol to finish.

### 6.4.3 Verification and Tracing phase

This phase contains two procedures: the `VerifySignature` procedure, which verifies whether someone from the allowed participants has created a valid signature, and the `TraceSignature` procedure that checks whether two signatures originate from the same person, using the same tag. In order to verify the incoming votes, the procedure `VerifySignature` will need to be invoked  $\beta$  times in order for the vote to be counted. Moreover, the execution of the `TraceSignatures` procedure will identify whether a participant has double voted. If this is the case, the tracing participant can publish the public key of that participant, and the vote will be disregarded during the aggregation phase. Furthermore, if the `TraceSignatures` procedure returns "Linked," it means that there has been a replay of the exact same signature, meaning that it can be disregarded in the aggregation phase as well.

---

**Algorithm 6** The procedure used to verify and trace signatures.

---

```

1: procedure VERIFYSIGNATURE( $\Sigma$ )
2:    $h \leftarrow \mathcal{H}(L)$  for  $L = (w, pk_N)$ 
3:    $A_0 \leftarrow \mathcal{H}'(L, vote)$ 
4:    $\sigma_i \leftarrow A_0 A_1^i \in G$  for all  $i$ 
5:    $a_j \leftarrow g^{z_j} y_i^{c_j}$ ,  $b_j \leftarrow h^{z_j} \sigma_j^{c_j}$  for all  $i$  for  $z_j, c_j \in \mathbb{Z}_N, \mathbb{C}_N$ 
6:    $c \leftarrow H''(L, A_0, A_1, a_N, b_N)$ 
7:   return  $c \pmod{q} == c_N \pmod{q}$ 

8: procedure TRACESIGNATURES( $\Sigma_1, \Sigma_2$ )
9:    $A_{01} \leftarrow \mathcal{H}'(L_1, vote_1)$ 
10:   $A_{02} \leftarrow \mathcal{H}'(L_2, vote_2)$ 
11:   $\sigma_i \leftarrow A_{01} A_1^i \in G$  for all  $i$ 
12:   $\sigma'_i \leftarrow A_{02} A_1^i \in G$  for all  $i$ 
13:  similarityList  $\leftarrow \emptyset$ 
14:  for all sigmas do
15:    if  $\sigma_i == \sigma'_i$  then
16:      similarityList.insert( $PK_N[i]$ )
17:  if |similarityList| == 1 then
18:    return similarityList
19:  else if |similarityList| == n then
20:    return "Linked"

```

---

### 6.4.4 Aggregation phase

Once all votes are registered, the aggregation protocol can commence. For each item, the total will be computed by summing the votes which are accompanied by a valid signature. The totals will be broadcasted after completion giving each participant both the desired labels, and the opportunity to verify the correct computation of the encrypted totals.

#### 6.4.5 Noise addition and final model generation

Adding calibrated noise to the final aggregated and training a global model will be done as presented in Section 5.4. Both the noise addition and model generation consists of a proposal and verify process where a participant proposes a noise distribution or final model. After which the other participants verify the proposal.

### 6.5 SECURITY ANALYSIS

The privacy-preserving nature of the PRECLUDE protocol is inherent to the security of the traceable ring signature. Only a verified participant of the protocol must be able to produce a valid signature, any second vote must be traced and discarded, the identity of the sender must remain hidden, and a sender cannot be framed as being dishonest when this is not the case. If these requirements are met, we can be sure that every validated, and traced vote, originates from a valid participant and that participant has only voted once. Additionally, if a malicious participant attempts to increase their influence on the outcome, he or she is not able to double vote and must, therefore, obtain the secret key from another participant to reconstruct signatures in his or her name.

The secret key used by each of the participants is a random value  $x_{p_i} \in \mathbb{Z}_q$ . An adversary could attempt to retrieve this value by observing the participant's public key  $y_{p_i} = g^{x_i}$ , and attempting to compute  $\text{dlog}_g(y_{p_i})$ . In order to do this, an adversary would need to solve the discrete log problem which is infeasible for a computationally bounded adversary. The bit security offered by the scheme must be less than or equal to 112 bits, to ensure that such an operation is indeed computationally binding, as discussed in the Section 5.5. Such a security standard requires the use of primes with the appropriate bit size,  $\gamma = 2048$ .

We assume that the identity of the participant  $\pi_i \in \Pi$  should not be linked to the contributions he or she provides for all items  $s_j \in S$ . The signatures are required to validate the participant as being a part of the protocol and limit the number of contributions. Meanwhile, these same signatures should not leak any information of the protocol, providing anonymity to the participant. The output of the protocol is visible to all participants and does not need to be protected.

The transmission tape and internal state of a single participant  $\pi_i$  consists of the following attributes:

1. His or her own contributions  $\Gamma_{\pi_i}$ , with accompanying signatures.
2. The received signature-vote pairs.
3. Verify and trace output performed by  $\pi_i$ .
4. All verify and trace outputs performed by other participants.
5. The final output of the aggregate of all inputs  $\Delta$ .



**Algorithm 7** Communication between a party  $\pi_i$  and the rest of the protocol  $\Phi$

$\pi_i$		$\Phi$
$\text{input}_{\pi_i}^{\Phi}(\Sigma_{\pi_i}, \Gamma_{\pi_i})$		$\text{output}^{\Phi}(\Delta)$
$\Sigma_{\pi_i} = \{\Sigma_{s_k}(A_1, c_j, z_j)   s_k \in S\}$	$\xrightarrow{\Sigma_{\pi_i}, \Gamma_{\pi_i}}$	$\lambda = \{\Sigma_{\pi_i}(s_k), q_{\pi_i, s_k}   \pi_i \in \Pi, s_k \in S\}$
	$\xleftarrow{\Lambda}$	$\Lambda = \{\lambda_{\pi_i}   \pi_i \in \Pi\}$
$V_{\pi_i} = \mathbf{VER}(\Lambda)$		
$T_{\pi_i} = \mathbf{TRACE}(\Lambda)$	$\xrightarrow{V_{\pi_i}, T_{\pi_i}}$	$V = \{V_{\pi_i}   \pi_i \in \Pi\}$
		$T = \{T_{\pi_i}   \pi_i \in \Pi\}$
		$\Delta_{s_k} = \sum_{i=0}^n q_{\pi_i, s_k}$
	$\xleftarrow{\Delta, V, T}$	$\Delta = \{\Delta_{s_k}   s_k \in S\}$

In the assumes covert adversarial model, an adversary can act in three different ways: 1) he or she can abide by the prescribed steps of the protocol, 2) the malicious nodes can abort the protocol (abort), or 3) the participants can cheat (cheat).

**ABIDING PROTOCOL STEPS** If the controlled nodes abide by the protocol steps, they will not learn any additional information. This property is inherent to the security proof provided for our primitive, traceable ring signatures.

**ABORT** The protocol can commence when nodes abort. There is a threshold of the required number of participants to proceed, which is similar to ECONoMy set to a majority share. Therefore, the minority of compromised nodes cannot move the protocol along before others have voted. Moreover, there also exists an allocated period within which would need to be voted, when no vote is received within this period the protocol continues.

**CHEAT** We define cheating as attempting to have excessive influence over the final label. An adversary can cheat by providing malformed votes, or by providing multiple well-formatted votes. The initial case is made infeasible by allowing votes to remain open, thus allowing verification on whether the probabilities do in fact sum up to one. An adversary cannot execute the latter case because multiple votes it is not possible to include multiple votes in the final aggregate. If an exact replay is detected, the votes are dismissed. If multiple votes originating from the same participant are detected, the identity is leaked and the identity excluded from future executions of the protocol without allowing a double vote to be entered in the aggregate.

Table 10: Intuition behind the need for  $v = n - 1$ .

	a	b	c	d	e
a			x	x	x
b				x	x
c	x	x			x
d	x	x	x		
e		x	x	x	

## 6.6 COMPLEXITY ANALYSES

This section will evaluate the theoretical performance of the PRECLUDE protocol. Section 6.6.1 evaluates the complexity of the executed operations, whereas Section 6.6.2 discusses the transferred information.

### 6.6.1 Computational analysis

Each phase of the PRECLUDE protocol is evaluated, and the number of operations required in total and per party is computed. The required operations are dependent on the number of participants  $n$ , the number of items  $m$  in the public data set  $\mathcal{S}$ . Similarly to the previously described protocol, the number of public items considered for the knowledge transfer is determined by determining the acceptable privacy guarantees, with the precision of the global model. In general, the number of participants is assumed to be smaller than the number of items,  $n < m$ , with  $n \geq 3$ . Within a two-party scenario, the anonymity provided by the traceable ring signatures is ineffective as each participant could directly link a vote to the other participant.

Table 11 shows the number of operations included in a single execution of the protocol, per phase. The variable  $v$  resembles the number of verifications needed per signature in order to be accepted as valid.  $\tau$  represents the required number of tracings each participant needs to do per item to be labeled. Since tracing has a larger space of possible combinations, we initially assumed that  $\tau > v$ . However, if not every participant traces all signature combinations, there will be signature pairs for which only a single participant has verified them. Table 10 shows the intuition behind this. In this scenario, there are five parties whom each trace the combinations of three signatures. As can be seen, there are several combinations of signatures which are only traced by one additional, non-signer, participant. An example would be the signatures originating from  $c$  and  $e$ , which combination is only traced by one participant, thereby entrusting that participant with the validity of this signature.

### 6.6.2 Communication analysis

Table 12 shows the amount of transferred information during the different phases. The largest phase regarding the amount of transferring information involves the inference voting. The cause of this is the size required in sharing a signature accompanying a vote and corresponding item number. The size

Table 11: An overview of the different types of operations encountered in the overall protocol described above. Amount shown both in its totality, per party, and per execution of the step.

Phase	Step	Operation type	Total complexity	Per participant	Per execution
Inference voting	Signing	Hashing	$3nm$	$3m$	3
		Exponentiation	$5n^2m - 2nm$	$5nm - 2m$	$5n - 2$
		RNG	$2n^2m - nm$	$2nm - m$	$2n - 1$
		$\times / \div$	$3n^2m - 2nm + n$	$3nm - m$	$3n - 1$
		$+/-$	$n^2m$	$nm$	$n$
Aggregation	Verification	Hashing	$3nmv$	$3mv$	3
		Exponentiation	$5n^2mv$	$5nmv$	$5n$
		$\times / \div$	$3n^2mv$	$3nmv$	$3n$
		$+/-$	$n^2mv - nmv$	$nmv - mv$	$n - 1$
	Tracing	Hashing	$3nm\tau$	$3m\tau$	3
		Exponentiation	$2n^2m\tau$	$2nm\tau$	$2n$
		$\times / \div$	$2n^2m\tau$	$2nm\tau$	$2n$
	Labeling	$+/-$	$n^2m - nm$	$nm - m$	$n - 1$

of the shared item identifier depends on  $m$ , but will remain fairly small, even in extreme cases of  $m$  being multiple hundreds of thousands, would still require less than 25 bits to represent. The bit-size of the vote,  $\chi$ , depends on the number of possible classes, which would not require a significant bit size. A signature on tag  $L$ , and message  $m$  consists of the variables  $\Sigma = (A1, c_j, z_j)$ . Each of the transferred values have a worst-case size of 2048 bits, leading the size of a single signature ( $\zeta$ ) to be  $(2n + 1) \cdot 2048$ . This means that  $\zeta$  is significantly larger than  $\omega$  and  $\chi$ . For example, for  $n = 50$ , a single signature will have an approximate size of 206.848 bits, or 0,025856 MB.

Table 12: An overview of the communication in bits between participants.

Phase	Operation	Total	Per party
Inference voting	Sharing votes	$n \cdot m \cdot (\omega + \zeta + \chi)$	$m \cdot (\omega + \zeta + \chi)$
Labeling	Publishing final labels	$m \cdot \lambda$	$m \cdot \lambda$
	Retrieving final labels	$n \cdot m \cdot \lambda$	$m \cdot \lambda$

## 6.7 DISCUSSION

PRECLUDE provides a robust, modular, approach to collaborative model generation using the distributed ensemble learning approach. To the best of our knowledge, PRECLUDE represents the first privacy-preserving approach to do so, allowing more maneuverable adversaries than ECONoMy. PRECLUDE assumes a covert adversary model, where adversaries who attempt to increase their influence on the outcome of the protocol have a chance of being caught.

The introduction of traceable ring signatures increases the number of exponentiations required to  $4mn^2 - 4mn + 3m$ . In addition, the signatures

themselves are quite large as they contain  $2n + 1$  large values, depending on the security parameter.

As also mentioned in Chapter 5, the intuitive privacy benefit inherent to this semi-supervised learning approach allows for a fixed privacy loss of the original training data set (combined over all local models) as only  $m$  queries are made to the ensemble. If this ensemble were to be released directly, the privacy loss does not have such an upper bound. Finally, by adding calibrated noise, the outcomes generated by white box attacks are hidden under differential privacy.

Chapter 4 indicates several research challenges that needed to be taken into account, given the assumed threat model. We will discuss these, one by one.

- *Limited insights into participant behavior* - The individual votes are visible for any participant, thereby validating the structure of the provided votes. Nevertheless, it is infeasible to detect who cast a particular vote and thus finding the correct vote combinations belonging to a specific person has a chance of  $n^{-m}$ .
- *No double spending* - The tracing capability provided by the traceable ring signatures prevents any double vote to be included in the aggregation. If an exact replay is detected, it will be discarded, and if a second signature on the same tag belonging to a different message is found, the identity of the signer is released.
- *Verifiability and consensus* - Similar to ECONoMy, the use of a permissioned blockchain allows all participants to agree on the state of the protocol. Because the blockchain is accessible to all participants, everyone can verify the executed steps and recompute the labels themselves if so desired. We obtain consensus by employing the practical Byzantine fault tolerance algorithm. Another consensus algorithm can replace this algorithm if it adheres to the same requirements. As noted multiple times, this is not the core of this research and is assumed as a primitive.
- *Auditability* - Only the creator of a signature can prove that he or she signed this item. Therefore, if an external party would like to find out who has votes what, each participant is able to prove it was or was not them.
- *Reasonable efficiency* - The increased robustness obtained by the addition of traceable ring signatures comes at a cost. Both the theoretical communication and computation complexities have increased. The high complexity could especially become a problem for the tracing operations when there are a lot of participants. The proof of concept computational analyses evaluation will be given in Chapter 8. But as can be seen from the theoretical analysis, the complexity is in need of improvement which we will aim to do in Chapter 7.

## EXTENSIONS

After completing the PRECLUDE protocol described in Chapter 6, we constructed further improvements to the efficiency of the underlying protocol. Two adaptations to the traceable ring signature scheme allow for significant performance improvements for our use case. In this chapter, we will introduce these improvements to the protocol called PRECLUDE<sup>+</sup>.

## 7.1 RING SIZE REDUCTION

Instead of signing a message originating from a group of  $n$  individuals, selecting  $t$  participants could already suffice in certain circumstances. Reducing the number of individuals included in a ring exposes a direct trade-off between the provided anonymity and the efficiency of both the signing and verification of the scheme. First, we introduce the challenges that originate from adapting the original protocol. Next, we will elaborate on how we propose to change the protocol to achieve increased efficiency while preserving privacy. Afterward, we perform an analysis of the required number of participants, which maximizes the efficiency gain, while adhering to the given privacy constraints.

## 7.1.1 Research challenges

In order for the proposed adaptation to be valid, the four requirements of the protocol given by the authors of [43] need to be maintained. Below, we address each of these requirements and the changes that follow the proposed alterations to the protocol.

1. When a participant  $\pi_e$  signs a vote, he or she hides within a group of  $n - 1$  others. Furthermore, these  $n - 1$  participants can also include the identity of  $\pi_e$ , allowing our initial participant to be associated with multiple signatures. A direct link between the signer and a ring signature will remain if a signer's identity does not occur in other participants' signatures. By reducing the number of participants included in a signature, there are fewer possible signers from which the signature could have arisen from. This is because the chance that the identity of the participant  $\pi_e$  is incorporated in the signatures produced by others becomes smaller, as rather than including all individuals, a randomly selected subselection is used. When a participant is included in fewer signatures, there is a higher chance that an adversary links a valid participant-signature pair, thus reducing the original *anonymity* requirement described in Section 6.2.
2. The tracing capabilities providing *tag-linkability* to the protocol need to be maintained. Any two signatures with the same tag, message, and overlapping sub-group need to be linked.

3. The ability to trace prevents double spending and thus performs a critical task in the correct execution of the protocol. Signatures based on different messages or different sub-groups out of the population concerning the same tag need to be traceable. The original definition of *public traceability* as given in Section 6.2 is extended to allow tracing of signatures based on the same message, but using a different sub-group of size  $t$ , as this cannot represent a direct replay of the signature/vote combination.
4. The concept of *exculpability* introduced in Section 6.2 needs to be maintained, prohibiting adversaries from entrapping an honest participant, and leaking his or her identity with the corresponding votes. In general, allowing individuals to determine a sub-group  $t - n$  extends the possibilities an adversary has within the protocol. It needs to be determined what an adversary who controls a variable number of participants can do with these new capabilities.

#### 7.1.2 Protocol changes

The changes needed to adapt the protocol mainly instantiate within the signing phase, while both the verification and tracing need to take the selected participants into account when validating a signature. Note that the initial setup, and the **Gen** portion of traceable ring signatures, do not change.

**SIGNING** Before signing a vote, a participant computes a random array of selected participants. Each element in this list will contain either a 1 or a 0, representing that the participant with that particular index within the  $PK_N$  array is either selected or not, respectively. Do note that the identity belonging to the signer will need to be set to 1 to generate a verifiable signature. Algorithm 8 shows how this list can be generated in the `generateSelectedParticipants` procedure.

---

**Algorithm 8** The generation of a random set of selected participants.

---

```

1: procedure GENERATESELECTEDPARTICIPANTS(signerIndex, n, t)
2:   selectedParticipants  $\leftarrow$  [0]  $\cdot$  n
3:   selectedParticipants[signerIndex] = 1
4:   included  $\leftarrow$  signerIndex
5:   i  $\leftarrow$  1
6:   while len(included) < t do
7:     random  $\leftarrow$  randomInt(0, n - 1)
8:     if random  $\neq$  signerIndex then
9:       selectedParticipants[random] = 1
10:      included.append(random)
11:   return selectedParticipants

```

---

To implement these alterations, we have adapted the original signing protocol as shown in Algorithm 9, by using this generated list of selected

participants of size  $t$ . Primarily, we reduce the number of computations to only include the values corresponding to the selected participants, thereby excluding those who have a 0 in their respective place within the selection list. As a result, the computed  $c_i$  and  $z_i$  are only dependent on the included participants. By doing so, we reduce the size of the signature by  $(n-t)/n\%$  since we do not need to add the previously required  $c$  and  $z$  values. We do also have to transfer the selected participants to be made visible to all other participants.

---

**Algorithm 9** Vote signing and casting procedure based on a subset of  $t$  parties.

---

```

1: procedure CASTVOTES(votes)
2:   for vote in Votes do
3:      $sp \leftarrow \text{generateSelectedParticipants}(\text{signerIndex}, n, t)$ 
4:      $\sigma \leftarrow \text{SignVote}(\text{signerID}, \text{itemID}, \text{vote}, sp)$ 
5:     Invoke: PublishVote(vote,  $\sigma(\text{vote})$ , itemID, sp)

6: procedure SIGNVOTE(signerID, itemID, vote, selectedParticipants)
7:    $i, w \leftarrow \text{signerID}, \text{itemID}$ 
8:    $h \leftarrow \mathcal{H}(L)$  for  $L = (w, pk_N)$ 
9:    $\sigma_i \leftarrow h^{x_i}$  for  $x_i \in \mathbb{Z}_q$ 
10:   $A_0 \leftarrow \mathcal{H}'(L, \text{vote})$ 
11:   $A_1 \leftarrow \begin{pmatrix} \sigma_i \\ A_0 \end{pmatrix}$ 
12:   $a_i \leftarrow g^{r_i}, b_i \leftarrow h^{r_i}$  where  $r_i \xleftarrow{r} \mathbb{Z}_q$ 
13:  for  $j$  in range( $t, n$ ) do
14:    if  $j \neq i$  and  $\text{selectedParticipants}[j] = 1$  then
15:       $\sigma_j \leftarrow A_0 A_1^j \in G$ 
16:       $a_j \leftarrow g^{z_j} y_i^{c_j}, b_j \leftarrow h^{z_j} \sigma_j^{c_j}$  for  $z_j, c_j \xleftarrow{r} \mathbb{Z}_q$ 
17:    else if  $\text{selectedParticipants}[j] = 0$  then
18:       $a_j \leftarrow 0, b_j \leftarrow 0$ 
19:   $c \leftarrow H''(L, A_0, A_1, a_N, b_N)$ 
20:   $c_i \leftarrow c - \sum_{j \neq i} c_j \pmod{q},$ 
21:   $z_i \leftarrow w_i - c_i x_i \pmod{q}$ 
22:  return  $\sigma \leftarrow (A_1, c_N, z_N)$ 

23: procedure PUBLISHVOTE(vote, signature, itemID, selectedParticipants)
24:  return vote || signature || itemID || selectedParticipants

```

---

**VERIFICATION AND TRACING** The adaptations for both the verification and tracing steps are very similar. The verifier (tracer) only computes those variables corresponding to the selected sub-group. However, due to the adaptations, the original tracing requirements have to be adapted.

The sigmas belonging to excluded participants are assigned the value  $-1$  in the tracing stage of the protocol. As a consequence, it should not be possible for an intermediately computed sigma to be negative. Otherwise, a potential

false positive trace could be obtained. The original computation of a sigma value is done using the following function:

$$\sigma_j = A_0 A_1^j \quad (7.1)$$

where both  $A_0$  and  $A_1$  are elements from the multiplicative group. Thus, this function will always result in a value within the group  $G$ , due to its multiplicative nature, and thus be non-negative under the assumed data representation class.

The resulting  $\sigma$  and  $\sigma'$  arrays are of length  $n$ , where  $n - t$  entries contain the value  $-1$ . We adopt the following alterations to possible outcomes the tracing operation can result in:

$$\text{Trace}(L, m, \sigma, m', \sigma') = \begin{cases} \text{Independent} \rightarrow i \neq i' \\ \text{Linked} \rightarrow i = i', m = m' \text{ and } |sp \cap sp'| > 1 \\ \text{pk}_i \rightarrow i = i' \text{ and } (m \neq m' \text{ or } |sp \cap sp'| = 1) \end{cases}$$

where  $sp$  stands for the selected participants. These rules are translated into a specific requirement of the intersection cardinality of the resulting sigma arrays computed by the tracer. In these operations, the replacement value  $-1$  should not be counted as an intersection and there is still a requirement of exactly one exactly one item to overlap to successfully trace two signatures. Overall, the correctness of the protocol does not change. Note that similarly to the original protocol, if multiple selected participants overlap and the message remains stable, we can only link the signatures without leaking the identity.

### 7.1.3 Factors dependent on $t$

Within this section we will analyze the different factors involved in selecting the appropriate size of  $t$ , the number of participants to be included in a single ring signature. First of all, we will evaluate the chance that an adversary can determine the combination of votes originating from a specific participant. After, we will generalize the chance that a participant is chosen in the rings generated by other participants. Finally, we will touch upon the benefits accrued by the change, analyzing the change in the complexity analyses.

**LINKABILITY** Within this paragraph, we analyze how well we can protect the anonymity of a specific participant when all votes and accompanying signatures are available to an adversary. Equation 7.2 summarizes the initial probability of finding a complete set of votes belonging to a single participant.

$$g(n) = \left(\frac{1}{n}\right)^m \quad (7.2)$$



The chance that the other participants included participant  $\pi_e$  in their signatures was 100%. For a given variable  $t$ , the probability that another participant includes  $\pi_e$  in its signature becomes:

$$\frac{t-1}{n-1} \quad (7.3)$$

as there need to be  $t$  individuals included in the signature out of which the original signer is one. By multiplying this value by the total number of participants, and combining this with the signature we generate ourselves, we obtain  $t$  as the average number of signatures that are linked to a participant.

$$\left(\frac{t-1}{n-1}\right) \cdot (n-1) + 1 = t \quad (7.4)$$

The total chance of selecting all votes belonging to your local model becomes:

$$g(t) = \left(\frac{1}{t}\right)^m. \quad (7.5)$$

However, this chance parameter reflects the average number of rings an individual is included in, later we will include the worst-case scenario in this computation which will cause the probability to increase due to a lowered denominator.

**ANONYMITY** The hiding property of the scheme determines how well a participant can hide among multiple signatures for a specific item/election. We can represent the probability that a participant is used in  $k$  different signatures according to Equation 7.6.

$$P(k|n, t) = \left(\frac{t-1}{n-1}\right)^k \cdot \left(1 - \frac{t-1}{n-1}\right)^{n-1-k} \cdot \left(\frac{n-1!}{k!(n-1-k)!}\right) \quad (7.6)$$

Within this function, the first term represents the probability of being included in a signature, the second term constitutes not being included in the others, and the final term denotes the number of possible combinations.

**COMPLEXITY IMPROVEMENTS** Other important factors that instantiate from this extension are the reduction in both the communication and computational analyses that originate from this alteration. Table 13 indicates the difference in computational complexity. Signature generation requires several exponentiations that is dependent on  $n$ . By lowering the number of individuals included in the ring to  $t$ , we improve the efficiency as long as  $t < n$ . Similarly, we reduce the bit size of a valid signature. Rather than providing  $n$ ,  $c$ , and  $z$  values, we provide  $t$ . By omitting these variables, we reduce the size of a signature by  $n - t \cdot \gamma$  as each  $c$  and  $z$  value has a worst-case bit size of  $\gamma$ .

Table 13: The improvements to the theoretical computational analysis provided by reducing the ring size.

Phase	Step	Total complexity	Per $\pi$	Per $s$
Inference voting	Signing	$n^2m - nm$	$nm - m$	$n$
	Adapted signing	$ntm - nm$	$tm - m$	$t$
Aggregation	Verification	$n^2m\nu$	$nm\nu$	$n$
	Adapted verification	$ntm\nu$	$tm\nu$	$t$
	Tracing	$n^2m\tau$	$nm\tau$	$n$
	Adapted tracing	$ntm\tau$	$tm\tau$	$t$
Total	Sign vote	$n^2m\tau$	$nm\tau$	$n$
	Adapt sign vote	$ntm\tau$	$tm\tau$	$t$

#### 7.1.4 Determining the size of $t$

The adjustments made to the protocol must retain their privacy-preservation. The difference between PRECLUDE and its plus variant only exists in the subgroup of participants among which one hides. In this section, we will show that a predefined level of privacy can be maintained while improving efficiency.

The level of anonymity provided by the adapted protocol is dependent on three parameters that are manually assigned by the executors of the protocol. These parameters are used as constraints to compute the optimum value of  $t$  corresponding to the security needs of the participants. These parameters are  $k$ , which we define as the minimum number of rings an individual participant should be included in, and the probability  $\check{e}$ , representing the required maximum likelihood that a participant is in less than  $k$  rings, and  $\check{p}$  which represents the chance of finding a linked set of votes. In order to determine  $k$ , one can use the linkability probability as a constrained by computing,  $g(k) = \check{p}$ .

In a worst case scenario, an individual is included in the minimum number of signatures  $k$ , for each of the votes. This would reduce the chance of finding the set of votes originating from this participant to:

$$g(k) = \left(\frac{1}{k+1}\right)^m \quad (7.7)$$

$$\check{p} > \left(\frac{1}{k+1}\right)^m \quad (7.8)$$

where one is added to  $k$  as  $k$  is the number of *additional* signatures that refer to a participant. Notice that the minimum probability is bounded by  $k = n - 1$ , reducing the ring size is infeasible for a problem with a likelihood requirements that results in a  $k \geq n$ .

Table 14 gives an example in how  $k$  can be determined using a given  $m$  and  $\check{p}$ . It visualizes the lowest values of  $k$  for low values of  $m$  to show that

Table 14: Example probabilities of retrieving complete sets of votes for low values of  $k$  and  $m$ .

$k / m$	1	5	10	50
1	0,5	0,031	0,98E-3	8,88E-16
2	0,33	0,41E-2	0,17E-4	1,39E-24
3	0,25	0,98E-3	0,95E-6	7,89E-31
4	0,2	0,32E-3	0,1E-6	1,126E-35
5	0.17	0,13E-3	0,17E-7	1,24E-39

the chance decreases significantly even for these low values. Once  $k$  has been determined according to this formula, thus guaranteeing a satisfactory level of vote unlinkability, one can determine  $t$ . The minimum required  $k = 2$  because if a participant is included in two rings, the other selected participant can directly see the signer's identity.

To compute the required value for  $k$ , we first look at how to compute the probability that an individual participant is included in at least  $k$  rings. This probability is computed using the following function:

$$P(a < k | n, t) = \sum_{j=0}^{k-1} P(j | n, t) \quad (7.9)$$

$$P(\text{any} < k | n, t) = \sum_{j=0}^{k-1} P(j | n, t) \cdot \left(1 - \sum_{j=0}^{k-1} P(j | n, t)\right)^{n-1} \cdot n \quad (7.10)$$

which we require to be lower than the preset maximum bound  $\epsilon$ . Here Equation 7.9 represents the probability that there is a participant which is included in less than  $k$  rings. Equation 7.10, computes the probability that **any** participant is included in less than  $k$  rings. Algorithm 10 shows how to determine the lowest level of  $t$  that still adheres to the bounds set by the parameters  $k$  and  $\epsilon$ .

---

**Algorithm 10** Optimizing  $t$  according to the set parameters  $k$  and  $\epsilon$ .

---

```

1: procedure OPTIMIZE_T( $k, n, \epsilon$ )
2:   for  $t$  in range(1, (n-1)) do
3:     chance  $\leftarrow$  0
4:     for  $i$  in range(0, (k-1)) do
5:       chance +=  $P(i)$ 
6:     if chance <  $\epsilon$  then
7:       return  $t$ 

```

---

In general, it appears that higher values of  $n$  require a lower value of  $t$  (proportional to  $n$ ), and can, therefore, gain more considerable efficiency improvements. The relationship of  $t$  and its constraint can be explained using Equation 7.11.

$$t^*(k, \check{\epsilon}) = \arg \max_t \left( \sum_{j=0}^{k-1} P(j|t, n) \right), \text{ for } \sum_{j=0}^{k-1} P(j|t, n) < \check{\epsilon} \quad (7.11)$$

An example method of establishing the required probability  $\check{\epsilon}$  could be to set it in relation to the total number of signatures  $nm$ . This chance could be set equal to one over the size of  $nm + 2$ . By using this formula, there will always be less than one percent chance of an undesired result. If it is desired to obtain a lower probability, the constant in the denominator (i.e., 2) can be incremented to reduce the maximum chance further.

We will now discuss an example to validate and illuminate the above claims, with the parameters as shown in Table 15. We ran extensive simulations to verify our claims using these example values. When we enter these parameters into the equations described above, the selected ring size becomes  $t = 20$ . Each signature now only includes 20 individuals rather than the initial 50, improving efficiency. A total of one million simulations are done rather than the 5000 signatures that would be performed in a standard protocol execution using these parameters. Figure 14 shows the effect of the selected  $t$  under such extreme conditions. Here, when a lower-than-computed  $t$  is utilized, the distribution fitted on all occurrences moves to the left-hand side. When  $t$  is selected to be either 18 or 19, there are 16 and 5 instances respectively for which a participant was included in less than  $k$  rings. When the computed value 20 is used, there is no participant present in less than  $k$ , thus adhering to the probability  $\check{\epsilon}$ , as the chance that this does happen becomes:  $3,0743 \cdot 10^{-7}$ . After a million executions there is an approximate 30% chance of **one** participant being included in less than  $k$  rings. The probabilistic nature can also be seen in the following analysis where we indeed encounter an iteration for which a participant is in less than  $k$  rings.

Figure 13 shows the number of iterations for which there was a person who was included in less than  $k$  rings. As can be seen, selecting  $t$  in the proposed way offers the set probabilities, while achieving the highest possible efficiency improvement.

Table 15: Variable values for determining  $t$  in an example scenario.

Variable	Used value
$n$	50
$m$	100
$\check{p}$	$1 \cdot 10^{-50}$
$k$	3
$\check{\epsilon}$	$1 \cdot 10^{-6}$
$t$	20

There exists a risk of exposing the participant from which a set of votes originates that was not present before. An adversary could analyze the

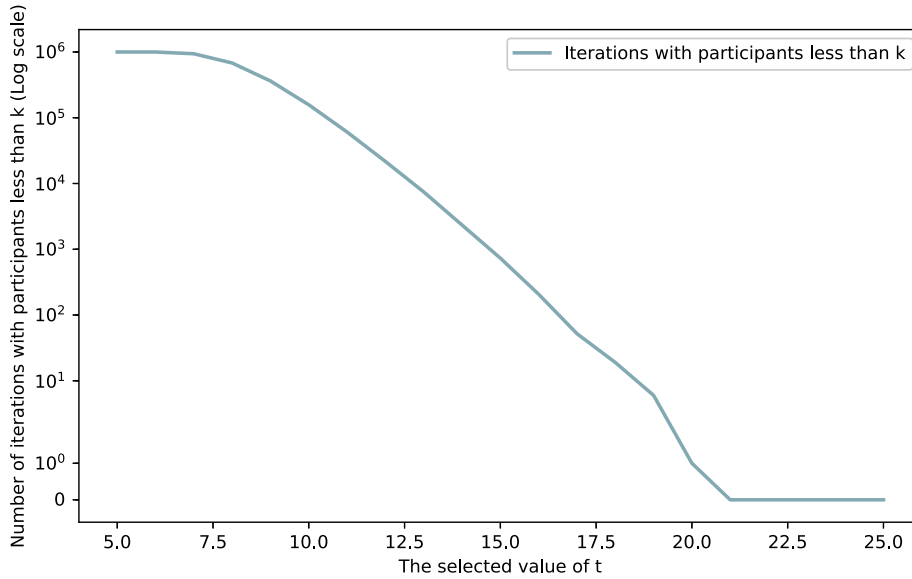


Figure 13: Example showing the amount of iterations for which a participant was included in less than  $k$  rings, for  $t=[5,25]$ .

items in the public dataset that have a strong correlation. If we assume that a participant  $\pi_i$  gives an outlier prediction for these samples, overlapping the individuals included in the signatures could expose  $\pi_i$ . Therefore, a correlation analysis on the public samples should be used for signers to select overlapping subsets of participants for correlating items rather than purely random. By doing so, we prevent an adversary from exploiting the structure of public data set to obtain additional information. We assume that there is a limited amount of sample pairs that have a strong correlation in our agreed upon public dataset, which would not significantly affect the analyses described above.

Additionally, we need to set the minimal value  $k$  in parallel with the number of parties within the protocol an adversary can control. If an adversary controls  $k$  or more nodes, no non-malicious participants may be selected within the ring signature, allowing the adversary to link the vote directly to the originating party.

Overall, the final determination of  $t$  thus depends on the required level of linkability, corresponding with the maximum probability of an individual participant being included in less than  $k$  rings. These dependencies guarantee that the unlinkability and anonymity properties of the proposed protocol are maintained according to the desired bounds while lowering the complexity exponentially for growing values of  $n$  and  $m$ .

## 7.2 BATCH VERIFICATION

We achieve another performance improvement by merging the verification and tracing phases into a batch verification phase. Due to the large number of signatures present in our protocol, the traceable ring signature structure is sub-optimal as the re-computation of the  $\sigma$  elements done for both the

verification and tracing adds a significant amount of exponentiations, and in turn computation time.

### 7.2.1 Proposed alterations

Every participant is currently required to verify  $v$  signatures and accompanying votes. After which, a participant traces  $\tau$  signature pairs, in order to detect potential fraud. By merging these two phases, all  $\sigma$  computations, containing one exponentiation each, are performed within the verification. All intermediate values are stored according to their item number, after which we can identify duplicates. When a duplicate is found, the duplicate location within the  $pk_n$  array will reveal which user signed those two particular signatures. Algorithm 11 highlights how this merger can be done in pseudocode based on an input item number and the corresponding set of signatures.

---

**Algorithm 11** Batch verification and tracing algorithm.

---

```

1: procedure BATCHVERIFY( $\sigma_I$ )
2:   sigmaHashSet, seen, tracing, linking, invalidSigns  $\leftarrow \emptyset$ 
3:    $h \leftarrow \mathcal{H}(L)$  for  $L = (w, pk_N)$ 
4:    $A_0 \leftarrow \mathcal{H}'(L, vote)$ 
5:   for all sigma in  $\sigma_I$  do
6:      $\sigma_i \leftarrow A_0 A_1^i \in G$  for all  $i$ 
7:     if  $\sigma_i$  in seen then
8:       if  $(i, \text{sigmaHashSet})(\sigma_i)$  not in tracing then
9:         tracing.append( $i, \text{sigmaHashSet}(\sigma_i)$ )
10:        linking[ $i, \text{sigmaHashSet}(\sigma_i)$ ]+ = 1
11:      else
12:        tracing.pop( $(i, \text{sigmaHashSet}(\sigma_i))$ )
13:      else if  $\sigma_i \geq 0$  then
14:        sigmaHashSet[ $\sigma_i$ ] =  $i$ 
15:        seen.append( $\sigma_i$ )
16:         $a_j \leftarrow g^{z_j} y_i^{c_j}$ ,  $b_j \leftarrow h^{z_j} \sigma_j^{c_j}$  for all  $i$  for  $z_j, c_j \in \mathbb{Z}_N, c_N$ 
17:         $c \leftarrow H''(L, A_0, A_1, a_N, b_N)$ 
18:        if  $c \pmod{q} \neq c_N \pmod{q}$  then
19:          invalidSigns.append(sigma)
20:      for  $x$  in linking do
21:        if linking[ $x$ ] <  $n$  then
22:          linking.pop( $x$ )
23:      return [invalidSigns, tracing, linking]

```

---

### 7.2.2 Performance improvement

The overall complexity of the tracing, as shown in Table 13, is reduced from  $n^2 m \tau$  to  $n^2 m$ .

Every participant  $n$  will need to find duplicates within  $m$  lists, where the use of hash sets would suffice to assign a time complexity of  $O(n)$  to the duplicate detection. When merging this with the complexity still required

for the verification, we get  $n^2 \cdot m \cdot \nu + n^2 \cdot m = n^2 \cdot m \cdot (1 + \nu)$ . Before this alteration, this complexity would have totaled to  $n^2 \cdot m \cdot \nu + n^2 \cdot m \cdot \tau$ . Also, the complexity is no longer dependent on the number of sigmas that are traced, as the tracing is done over one complete array per item, rather than a trace per signature pair.

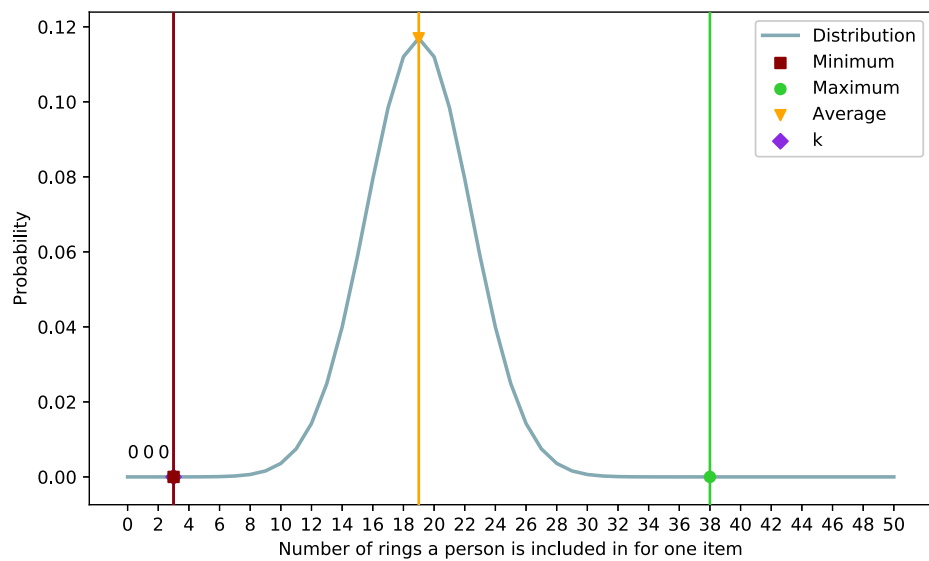
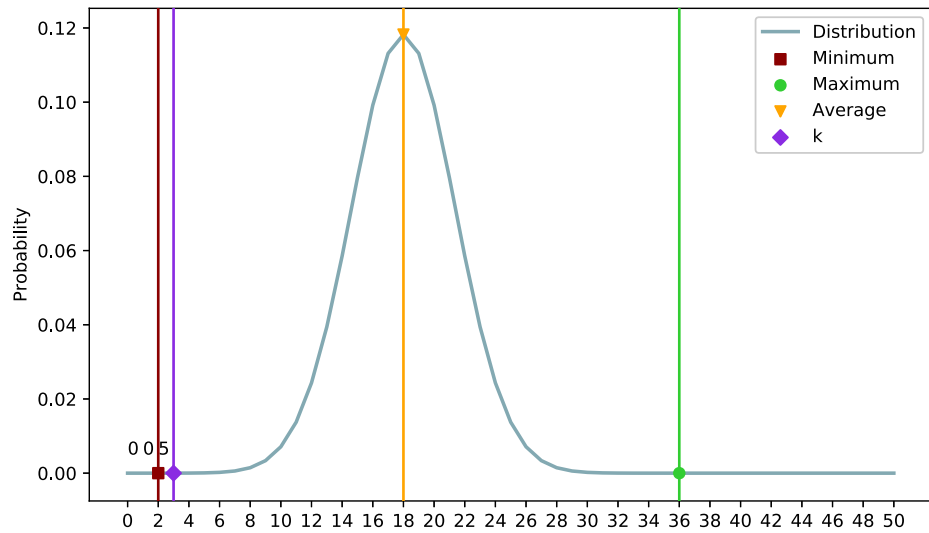
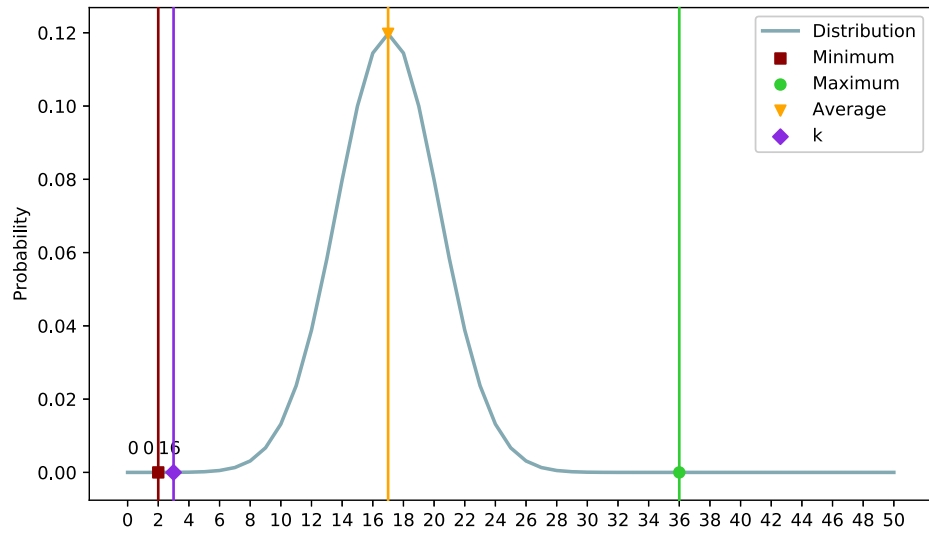


Figure 14: An overview showing how the distribution of signatures a participant is included in changes for  $k=3$ ,  $n=50$ ,  $m=100$ ,  $\hat{p} = 1 \cdot 10^{-50}$ ,  $\hat{\epsilon} = 1 \cdot 10^{-6}$ .  $t$  is set to 18, 19, or 20.



## Part IV

### EVALUATING THE PROTOCOLS AND DEFINING FUTURE WORK

Now that we have a complete overview of the proposed designs, we will now evaluate their performance. In order to do this, we have developed naive implementations of our protocol designs, as well as comparable alternatives. Our experiments are discussed and visualized, and we analyze the attained results. We continue to evaluate our research sub-questions to obtain an answer to our primary research question. Finally, we go over the limitations inherent to this thesis and describe open problems available for future work.



## EVALUATION

---

In order to establish the relevance of our contributions, we will compare the performance of the proposed work with alternative approaches. The currently available alternatives show a general lack in privacy focus. Nevertheless, the approaches used for the comparison do approximate a certain degree of privacy preservation. When comparing, we will focus on the efficiency and thereby attempt to assess the cost of implementing the improved privacy guarantees provided by our protocols. We will first introduce the setting in which we evaluate all protocols, after which we present the alternatives. Finally, we showcase the experimental results and analyze the outcome.

### 8.1 THE EXPERIMENT SETTING

To provide a clear context to the provided execution times, we will introduce aspects of the produced code as well as the hardware on which it ran. These details will aid any interested reader in reproducing the experiments or justify any deviations from the execution times based on differences in these primitives. In selecting these primitives, the primary objective has been to provide an even playing ground for each protocol, thus not skewing any of the retrieved results in any direction. Moreover, all experiments have been executed five times, and their results have been averaged to negate any potential outlier test results.

#### 8.1.1 *The code*

For each of the evaluated protocols, we have developed a naive implementation. These have not been optimized for performance, but do reflect the proportional efficiency by executing the required operations.

The naive implementations are written in Python 2.7 [40], because this scripting language is the most popular within data science and machine learning in particular [102]. Also, the accessibility of relevant packages such as *pycrypto* [78], providing the necessary cryptographic primitives, and *scikit-learn* [98], providing relevant machine learning models, makes it a suitable language for our problem. Even though Python is considered to be less efficient than its alternatives, it should not hinder our ability to compare alternatives as we are performing a relative comparison.

Unfortunately, to the best of our knowledge there is no available, validated implementation for the traceable ring signature protocol, and therefore this has been implemented and checked for correctness. We have made a custom implementation that provides an equal playing field when comparing with the other protocols implemented in python, inhibiting the language from becoming a decisive factor in the comparison. We ensure

that any inefficiencies introduced by the programming style are present in both PRECLUDE with PRECLUDE<sup>+</sup>, having changed the minimum required to attain the new mode of operation.

### 8.1.2 *The used hardware*

Selecting the hardware to execute our tests on, we evaluated three possibilities. A personal computer, with limited capacity, the Kova cluster provided by the Delft University of Technology [115], and the use of an external server such as Amazon Web Services (AWS) [2]. Since the test executions require the simulation of multiple participants, it is relatively easy to deplete the resources provided by a personal laptop for increasing values of  $n$ . In order to facilitate the more computationally intensive executions, and be able to execute these multiple times to average the execution time, we require a more powerful infrastructure. However, the use of the Kova cluster offered by the university has certain limitations. In previous experiments, it became apparent that the test results vary over time, due to others utilizing the infrastructure. We have therefore decided to use AWS to have the highest chance of a fair comparison.

More specifically, we have selected the `c5d.18xlarge` instance, to provide sufficient computing power to prevent resource constraints from playing an active role in the comparison. This machine is used for all tests of all sizes related to the PRECLUDE protocol. The selected instance offers 72 virtual CPU's and 144 GiB memory, where the number of vCPUs is the main reason for choosing the instance. We use the `c4.4xlarge` machine to execute the experiments related to ECONoMy, which offers 16 virtual vCPUs and 30 GiB memory [2].

## 8.2 THE ALTERNATIVE APPROACHES

It is important to compare our protocols with peer-reviewed alternatives and see how our approach performs in contrast and highlight the contribution of the presented work. As discussed in Chapter 3, there are a variety of alternatives present in the field of collaborative learning. The most relevant alternatives are within the CEMA and DEMA categories as they also aim to generate ensemble models. Unfortunately, none of the reviewed entries for distributed ensemble model aggregation focus on privacy preservation. In the centralized category, however, two approaches are good candidates for comparison, which we will briefly introduce in this section.

### 8.2.1 *CrowdML*

First of all, the CrowdML approach proposed by Hamm et al. [54] follows the same semi-supervised approach to machine learning thereby bounding the privacy loss of the final released model. Nevertheless, this approach does trust a central entity in such a way that the participants transfer their models from the participants without taking into account potential privacy loss this can cause. Nevertheless, our ECONoMy approach aims to be used in similar situations where a lot of small devices are used to generate an

ensemble of the underlying data. Therefore, a naive implementation of the Hamm approach will be compared to our ECONoMy model in terms of efficiency.

**FUNCTIONAL COMPARISON** The CrowdML setting requires all entities to trust the central entity with their models. As has been discussed in Chapter 2, this can allow a malicious central entity to retrieve information from the underlying training data. Nevertheless, the bulk of the complexity is removed as the central party can directly predict the public data set with the available models, and provide a wider variety of classifier combiner techniques. In the CrowdML setting, the central party can directly see which predictions originate from which model, and perhaps even provide an additional test set. This advantage allows the protocol to provide classifier weighing which is currently not possible within the ECONoMy setting. ECONoMy, on the other hand, focuses on being as maneuverable as possible, allowing all possible local classifiers without sharing these details with others, while only exposing the effect has on the final aggregate of votes. It is unclear to any other participant what type of local model any participant has, nor are any details of the training data exposed. This results in a black-box situation for any internal adversary, whereas CrowdML grants the central party white-box access. Meanwhile, the techniques that ECONoMy employs are light-weight and can be executed in parallel whereas the prediction for each of the models on the public data sets in CrowdML has to be executed sequentially. In an IoT setting, also described by Hamm et al. themselves, a significant number of participating entities could require additional computing resources at this central entity.

**THE IMPLEMENTATION** We will compare the CrowdML approach with our ECONoMy approach based on a large number of participants. Therefore, we simulate the prediction of a public dataset of different sizes, for a varying number of participants, on a single computer. We then continue to compare these results with the vote-based approach presented in ECONoMy. All model types are allowed when participating in a protocol such as ECONoMy and CrowdML. Since prediction time varies among classifier types, we have selected a variety of basic, well-known classifier types from which every participant randomly chooses. These classifiers include Naive Bayes, Decision Tree, Linear Regression, Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), and Random Forrest (RF) with default settings provided by sklearn in their documentation. The classifiers are trained on samples from the MNIST dataset [74] using the pixel values as features.

### 8.2.2 *AnonML*

The second approach, called AnonML, as devised by Cyphers et al. [26] will be used to compare to our PRECLUDE and PRECLUDE<sup>+</sup> protocols. AnonML was the only reviewed approach that focused on aspects such as sender validation, prevention of double-spending, and anonymity.

Therefore, the approach attempts to provide similar privacy guarantees to our protocols.

**A MORE DETAILED VIEW OF ANONML** As previously discussed, AnonML splits the various nodes into participant partitions of size  $k_p$ . Each data sample is divided into groups of  $K_s$  features per non-overlapping subset  $k_h$ . The central entity receives a separate data packet from each participant for each feature subset requested, accompanied with a verification token using the Anonize protocol [58]. In total there are  $k_p \cdot k_h \cdot n$  data transfers, after which the central entity trains a classifier on each of the received  $k_p \cdot k_h$  partitions. In our protocols we assume local model generation to be part of pre-processing. Therefore, we will do the same for the AnonML, considering any data partitioning as pre-processing. Thus we will only focus on the generation of the Anonize verification tokens. These tokens allow the aggregator to validate that the sender is authorized to contribute as part of a particular partition, for a survey as described by Hohenberger et al., without identifying the sender. Moreover, a second contribution for a specific request from the aggregator is discarded, and thus double-spending is prevented. The central entity detects a secondary input by using a pseudo-random function to compute a unique code for a participant to which is committed in the initialization. However, by doing so there remains a link between individual contributions from a particular entity, even though there is no link to the original identity. In order to compare the AnonML protocol with PRECLUDE and PRECLUDE<sup>+</sup>, we will generate and verify these verification tokens according to the different amount of participants and features.

**FUNCTIONAL COMPARISON** AnonML protects data from participants from the central party by using differential privacy. The added noise does perturb the feature values and could lead to slightly lowered prediction performance. This decrease is also seen in the evaluation performed by Cyphers et al. [26]. There is no link to the identity of the participant, despite there being a link between the different data samples provided to the aggregator by one of the participants. Thus, while the protocol prevents excess contributions, there is no ability to trace which participant is exhibiting this deviant behavior. Additionally, the central entity still needs to train models after receiving the partitioned data, to achieve the final ensemble. The PRECLUDE protocols do not concede in terms of local classifier performance as there is no need to add noise locally. The use of traceable ring signatures allows each participant to hide every vote among the included ring members. Not only are we able to prevent double contributions, but additional contributions can also be traced, and misbehaving participants can be excluded from the following protocol executions (while protecting Exculpability).

**THE IMPLEMENTATION** The Anonize verification tokens have been implemented according to the descriptions presented in [58]. The `bplib` [28] package has been used together with `petlib` [29], to execute asynchronous bilinear mappings. We have implemented both the Dodis-yampolskiy PRF [33] and the Boneh-Boyen signature scheme [14] on which the Anonize token construction is dependent, using these dependencies. Unfortunately, the AWS Linux host OS used in the other experiments did not facilitate the required Elliptic Curve cryptography operations used in `bplib`; therefore Ubuntu has been used instead.

### 8.3 RUN-TIME ANALYSES

In this section, we will show the results gathered by running experiments using the described implementations. First, we show the experimental results obtained from the experiments aimed to compare ECONoMy and CrowdML. Afterward, we will present the experimental results for the comparison between PRECLUDE, PRECLUDE<sup>+</sup>, and AnonML.

#### 8.3.1 *ECONoMy*

The comparison performed between CrowdML and ECONoMy is made for varying values of participants ( $n$ ) and to be predicted items ( $m$ ). Table 16 shows the run-time results obtained from these experiments. Upon review, it becomes clear that there is a difference between the performance of CrowdML and ECONoMy. As previously mentioned, CrowdML offers no privacy on the base machine learning models, which are shared openly. Hence, we will concern ourselves with the increased costs of the added privacy in ECONoMy.

As can be seen in Table 16, the amount of time taken per vote appears to increase linearly by the number of included participants. Due to the significantly low time required per vote, the protocol can accommodate high values of both  $n$  and  $m$ . Especially so, if the implementation were to be optimized and the inherent parallelism of the protocol exploited. Each participant executes their respective share of computations locally, meaning that the computational work-load is dispersed among all participants. In CrowdML on the other hand, the work-load is concentrated in a single entity, the central server. Figure 15, shows the run-time if we were to divide the single-threaded experimental execution time of ECONoMy by the number of participants ( $n$ ), displaying the work-load per party represented in time. CrowdML still outperforms ECONoMy for both  $m = 500$  and  $m = 1000$ . At  $n = 500$  ECONoMy takes approximately 5,4 times as long with an  $m$  of 500, while for  $m = 1000$  this is reduced to 3,1. These values show that there is an additional dependence on  $m$  for CrowdML that is not present in ECONoMy.

We continue to analyze this by looking at a per vote basis as depicted in Figure 16. When we compare the experimental results for  $m = 500$  and  $m = 1000$ , we see that the execution time linear in the number of to be predicted items, while ECONoMy has a linear dependency on the number

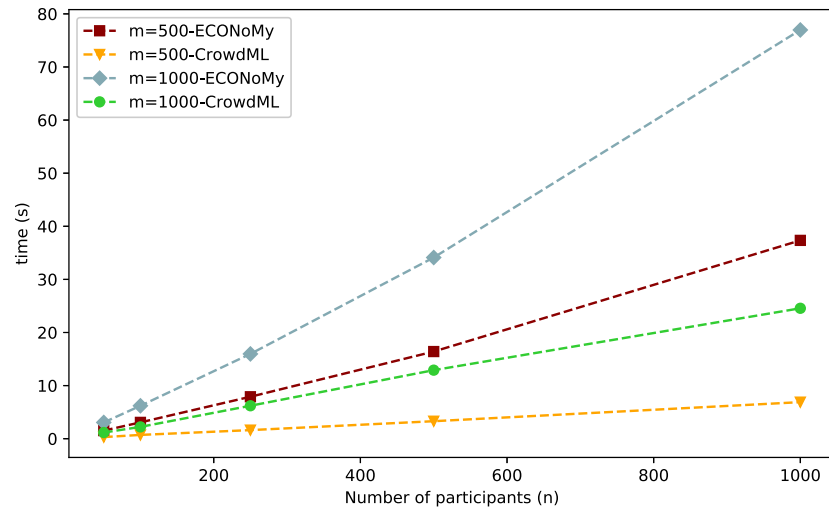


Figure 15: Comparing the per participant computation time of ECONoMy with CrowdML.

of participants while it is not affected by the number of to be labeled items. In the graph, we see both ECONoMy variants very close to one another, and an approximate doubling of execution time for CrowdML.

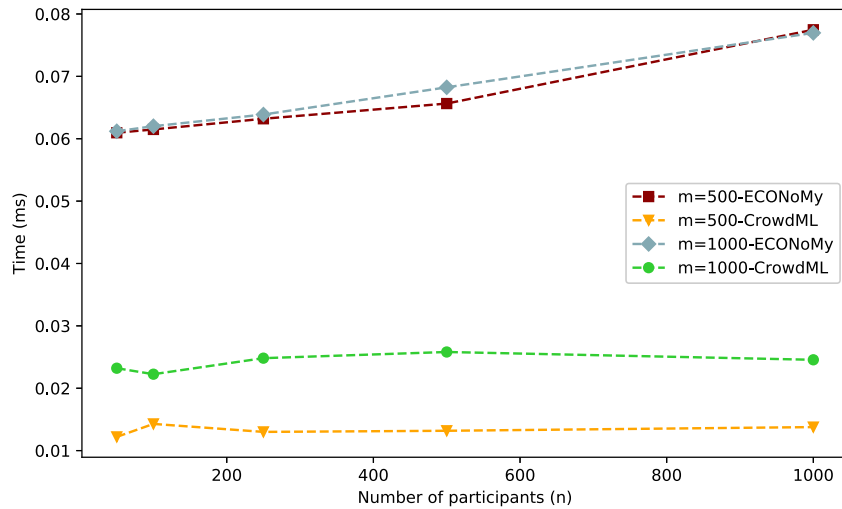


Figure 16: Highlighting the dependency on  $m$  inherent to CrowdML, which is not as present in ECONoMy.

For a participant to make a valid contribution to the ECONoMy protocol, he or she needs to participate in the random number generation protocol. The generation of a single mask is dependent on all other participants. Thus, despite computing the execution per party per vote, there remains a dependency on the number of participants.



The CrowdML protocol is directly dependent on the number of data partitions transferred to the central entity. When this is removed by dividing by the number of tokens, we see a constant performance. The curve is lifted due to an apparent dependency on the number of public items. The experimental results show that the average time of a single prediction increases with increasing values of  $m$ . To corroborate this result, additional tests have been executed on each of the utilized classifiers separately, which shows that all appear to have this dependency on  $m$ .

Table 16: The experimental results obtained by running both CrowdML and ECONoMy.

			ECONoMy			CrowdML	
n	m	nm	time (s)	time/n	(time/n)/vote	time (s)	time/p
50	500	25.000	76,193	1,524	6,096E-5	0,305	1,22E-5
	1000	50.000	153,008	3,060	6,120E-5	1,156	2,321E-5
100	500	50.000	307,432	3,074	6,149E-5	0,714	1,428E-5
	1000	100.000	620,036	6,200	6,201E-5	2,225	2.225E-5
250	500	125.000	1975,087	7,900	6,320E-5	1,619	1,30E-5
	1000	250.000	3991,550	15,966	6,388E-5	6,205	2,482E-5
500	500	250.000	8205,644	16,411	6,564E-5	3,296	1,318E-5
	1000	500.000	17.059,580	34,119	6,824E-5	12,907	2.581E-5
1000	500	500.000	37.345,950	37,346	7,747E-5	6,886	1,377E-5
	1000	1.000.000	76.970,465	76,970	7,697E-5	24,556	2.456E-5

### 8.3.2 PRECLUDE

We have executed experiments designed to evaluate the PRECLUDE protocols have for different values for  $n$  and  $m$ . When executing PRECLUDE<sup>+</sup>, a conservative  $t$  value has been used, computed using the methods described in Chapter 7. This conservative nature is indicated by the use of a low  $\check{p}$ ,  $\check{e}$ , and  $v = n$ . By putting the selected constraints on the determination of  $t$ , the PRECLUDE<sup>+</sup> model can be used from  $n = 12$ , as before the required properties cannot be guaranteed. AnonML, on the other hand, is given a optimistic variable setup by assigning  $k_s = 1$ . The factor with which this value were to increase can be directly multiplied with the result as the required number of tokens is dependent on this parameter.

PRECLUDE<sup>+</sup> Table 18 shows the results obtained from the experiments. The execution time of the original PRECLUDE model grows significantly due to the sharp increase in the number of exponentiations required for each signature with increasing  $n$ . The alterations made in PRECLUDE<sup>+</sup> combat this effectively as can be seen by the lower amount of time required per signature. Figure 17 shows the relative efficiency improvements of the plus variant over the original PRECLUDE.

Within this figure, we see the execution times in contrast to the number of items for two different values of  $n$ . The curves appear to be linear, this is

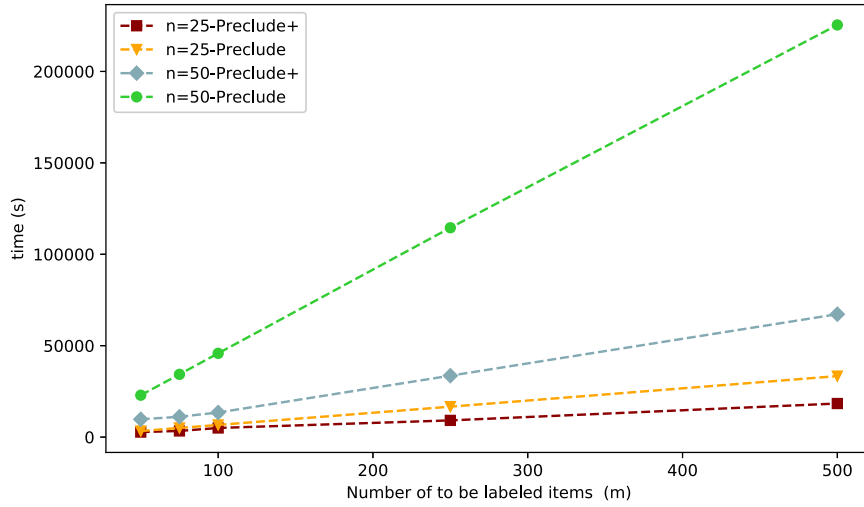


Figure 17: The efficiency of a protocol execution of PRECLUDE and PRECLUDE<sup>+</sup> for varying  $m$  and  $n$ .

because the implementation uses parallelization to simulate the presence of multiple parties, removing the quadratic nature of the protocol into the second column of the provided theoretical analyses in Section 6.6.1. The parallelization is only a naive approximation not resembling the actual use of  $n$  different machines.

For increasing values of  $n$  (50 rather than 25), the difference between PRECLUDE and PRECLUDE<sup>+</sup> grows significantly. When comparing both variants for  $n=25$ , we see a reduction in execution time when using the plus variant. This improvement can be directly attributed to the corresponding values of  $t$  as computed according to the guidelines described in Chapter 7. The required  $t$  decreases to 15 (or 60% of  $n$ ) with increasing values of  $m$ , requiring fewer exponentiations in the process. For the executions performed on  $n = 50$ , we see that  $t$  decreases to 20 (or 40%) for increasing  $m$ . Thereby gaining an efficiency improvement of roughly 60% as opposed to PRECLUDE.

The steepness of the curve increases drastically for the PRECLUDE protocol, which will continue to grow for increasing values of  $n$ . The PRECLUDE<sup>+</sup> protocol shows similar behavior, but it can significantly reduce this effect.

The same improvement also occurs in the transmission size of each signature as 60% fewer values are included when providing a TRS. Figure 18 visualized this effect for the obtained experimental values. The size of a single signature is directly dependent on the number of included parties. When  $t$  decreases as opposed to its corresponding  $n$ , we see a larger discrepancy between the signature size of PRECLUDE and PRECLUDE<sup>+</sup>.

**ANONML** At first glance, Table 18 shows AnonML outperforming both PRECLUDE and PRECLUDE<sup>+</sup> significantly. The majority of this difference

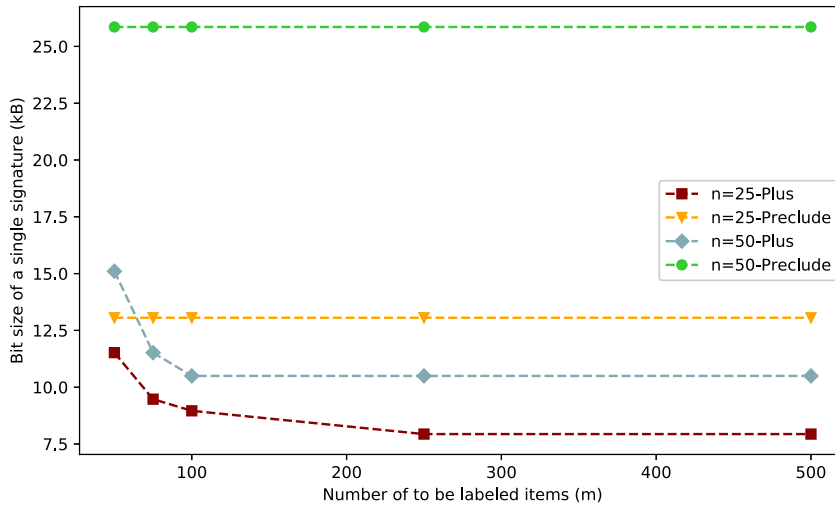


Figure 18: The size of a single signature in the experiments for PRECLUDE and PRECLUDE<sup>+</sup> for varying  $m$  and  $n$ .

for our proposed protocols originates from the Verification phases. Within AnonML, an average of 45% of the execution time can be attributed to the token generation, whereas the complementing 55% corresponds to the verification of the supplied information. For the PRECLUDE protocols, the signature creation contribution drops significantly for increasing  $n$  and  $m$ .

The massive growth in execution time of the verification phases arises from the number of verifications needed per signature. AnonML currently requires precisely one verification per produced token without performing any tracing. Our protocols, on the other hand, have been set to require each signature to be verified by each participant. The significant growth shown in these verification phases could be reduced by tuning this parameter to be closer to the majority who are assumed to be honest.

If we were to convert our batch verify into a per item verify, we would significantly reduce the difference between the two protocols. Table 17 shows the results of this when approximating the batch verification time from PRECLUDE<sup>+</sup> from the attained data. Notice that this includes tracing on the PRECLUDE<sup>+</sup> side, due to the extension called batch verification, shown in Section 7.2. The values from Table 17 are visualized in Figure 19. The figure depicts that in this scenario, PRECLUDE<sup>+</sup> outperforms AnonML for both  $n = 25$  and  $n = 50$ , where there does appear to be some overlap for lower values of  $m$ . This indicates that our selected technique in similar circumstances is more efficient, and provides the additional privacy properties which we want to attain. In this scenario, we show that when comparing PRECLUDE<sup>+</sup> in a CEMA fashion to AnonML also belonging to that category, our protocol outperforms AnonML.

Nevertheless, Table 18 does highlight the cost of decentralization when looking at the total experimental results. The need to verify additional signatures, by multiple parties, significantly increases the time required to

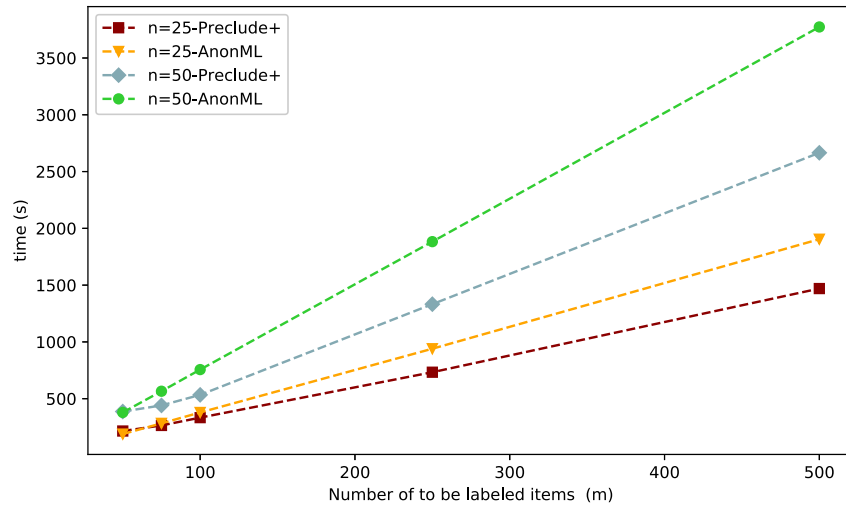


Figure 19: Trend comparison of PRECLUDE<sup>+</sup> and AnonML based on a single verification per item.

complete the verification stage. Of course, if we would loosen our stringent privacy requirements, fewer individuals will need to be included in signatures thereby further increasing efficiency. Furthermore, since we assume that the majority of the individuals are indeed benign, we can lower the  $\nu$ , which will reduce the time needed for the batch verify stage.

Table 17: Comparison of the time needed to cast and verify nm votes for PRECLUDE<sup>+</sup> and AnonML.

			PRECLUDE <sup>+</sup>	AnonML
n	m	nm	time (s)	time (s)
25	50	1250	215,243	188,329
	75	1875	264,435	282,917
	100	2500	333,234	378,586
	250	6250	733,123	939,989
	500	12500	1469,768	1904,356
50	50	2500	386,846	378,871
	75	2750	440,684	565,697
	100	5000	533,658	756,282
	250	12500	1332,234	1882,932
	500	25000	2665,355	3773,365

Table 18: Run-time experiment results comparing PRECLUDE, PRECLUDE<sup>+</sup>, and AnonML in their execution timings for different n and m.

			PRECLUDE			PRECLUDE <sup>+</sup>						AnonML	
n	m	nm	Sign (s)	Ver + Tra (s)	t/σ	k	č	t	Sign (s)	Verify (s)	t/σ	Token (s)	t/σ
5	50	250	24,880	99,551	0,50	9	1E <sup>-5</sup>	-	-	-	-	37,734	0,151
	75	375	37,258	149,293	0,50	4	1E <sup>-5</sup>	-	-	-	-	56,437	0,151
	100	500	49,689	199,203	0,50	3	1E <sup>-5</sup>	-	-	-	-	76,150	0,152
	250	1250	124,124	496,474	0,50	2	1E <sup>-6</sup>	-	-	-	-	188,344	0,151
	500	2500	249,993	1010,007	0,50	2	1E <sup>-6</sup>	-	-	-	-	374,837	0,150
10	50	500	49,768	452,360	1,00	9	1E <sup>-5</sup>	-	-	-	-	75,402	0,151
	75	750	75,507	676,110	1,00	4	1E <sup>-5</sup>	-	-	-	-	113,432	0,151
	100	1000	99,419	902,081	1,00	3	1E <sup>-6</sup>	-	-	-	-	150,854	0,151
	250	2500	230,521	2284,479	1,01	2	1E <sup>-6</sup>	-	-	-	-	375,820	0,150
	500	5000	501,336	4518,664	1,00	2	1E <sup>-6</sup>	-	-	-	-	757,553	0,152
25	50	1250	124,714	3204,857	2,66	9	1E <sup>-6</sup>	22	109,960	2632,069	2,19	188,329	0,151
	75	1875	187,418	4818,364	2,67	4	1E <sup>-6</sup>	18	134,915	3238,001	1,80	282,917	0,151
	100	2500	249,914	6408,944	2,67	3	1E <sup>-6</sup>	17	170,100	4078,342	1,70	378,586	0,151
	250	6250	624,347	16014,039	2,67	2	1E <sup>-6</sup>	15	374,367	8968,896	1,50	939,989	0,150
	500	12500	1248,274	32113,258	2,67	2	1E <sup>-7</sup>	15	750,200	17989,198	1,50	1904,356	0,152
50	50	2500	337,023	22564,565	9,16	9	1E <sup>-6</sup>	29	195,738	9555,381	3,90	378,871	0,152
	75	3750	505,262	33825,127	9,15	4	1E <sup>-6</sup>	22	222,763	10896,054	2,97	565,697	0,151
	100	5000	674,111	45135,718	9,16	3	1E <sup>-6</sup>	20	270,121	13176,872	2,69	756,282	0,151
	250	12500	1522,987	112977,013	9,16	2	1E <sup>-7</sup>	20	673,916	32915,921	2,69	1882,932	0,151
	500	25000	3326,417	222097,166	9,15	2	1E <sup>-7</sup>	20	1346,887	65873,398	2,69	3773,365	0,151



## DISCUSSION AND FUTURE WORK

---

The recent advances in machine learning allow everyone to experience the potential that is inherent to the underlying techniques on a daily basis. Privacy scandals are reducing the trust people have in central entities, and a more general increase in privacy awareness fuels demand for privacy-preserving techniques. The utility that is provided by the use of machine learning should, however, be maintained in this quest to preserve privacy. In this thesis, we have reviewed a wide variety of approaches that aim to allow participants to collaborate in learning a machine learning model. However, the majority of the reviewed prior art lacks commitment when it comes to privacy. The protocol designs proposed in this work aim to change the industry mentality to a privacy-first approach, handling user data with the care it deserves. This research shows that people can collaborate in training a machine learning model without the need for a central entity while preserving privacy. The main research question reviewed in this work was as follows:

*How can we facilitate the joint generation of a shared machine learning model in a privacy-preserving manner, and refrain participants from degrading the final models' performance in excess of their own, allowed contribution, in a decentralized setting?*

In this chapter, we revisit the original research question and discuss how the proposed approaches achieve our research goal. Also, we discuss the limitations of our proposed methods, and possible future improvements are identified.

### 9.1 DISCUSSION

We have presented two privacy-preserving protocols for decentralized collaborative learning, applicable to two different use cases. Both protocols are based on the semi-supervised ensemble learning approach suggested by Papernot et al. [95], providing differential privacy to the resulting global model. ECONoMy uses random number generation with symmetric key encryption to mask vote values in a 'semi-honest' setting. Every participant has the opportunity and freedom to select their desired model type for both the local and the global classifier. Nevertheless, in practice, the assumed 'semi-honest' threat model might not be sufficient. It may very well be the case that there are compelling incentives that would drive participants to deviate from the protocol description. This motivated us to create our second protocol, PRECLUDE, which uses traceable ring signatures to not only limit the contributions made by participants but allow for active tracing to identify malicious protocol participants. However, the

computational complexity presented by the second protocol depreciated quickly as opposed to ECONoMy. To improve the efficiency, we continued our work and extended the PRECLUDE protocol to increase its efficiency. The result of this, called PRECLUDE<sup>+</sup>, considerably reduces both the communication and computation load, while maintaining privacy-preservation. Our protocols limit the insights obtained from the training data by moving towards a vote based system while hiding the identity linked to a vote. We move the privacy leakage from an individual's classifier to the contributed part of the global total.

In Chapter 1 we identified five sub-questions to be answered by our research. The first sub-question describes the need for transparency in a decentralized approach. Both of the presented protocols offer the ability to verify the aggregation of votes and allow for a verification step when participants suggest global variables. Furthermore, the PRECLUDE protocols allow every participant to validate and trace each vote, ensuring themselves of the value he or she can attribute to the final labeling. It can thus be argued that transparency is, in fact, a large factor in the proposed solutions. Additionally, both protocols can prove that a participant made a particular contribution. In ECONoMy, the participants provide a hash of the original vote, acting as a commitment on the vote value. In PRECLUDE this can be proven by computing a second signature on the same tag in such a way that it 'leaks' the identity. These features allow a participant to demonstrate what he or she has voted to an auditing party.

The second question focuses on the contributions allowed per participant, and how to limit these. Both of the suggested protocols take a different approach to achieve this. ECONoMy requires the use of a random number that is jointly generated by all participants in order to mask a value. When a participant makes an additional contribution, either separate randomness is added to the total, or the vote will not be masked. The extra randomness will cause the 'unmasking' to fail whereas a vote in the clear is easily recognizable as such due to the particular encoding used. PRECLUDE on the other hand limits contributions due to the tracing capability of its signatures. An additional input is caught as such and allows the validating participant to leak the identity of the deviant, or discard a duplicate vote.

In the third sub-question, we identify the need to remove the link between the contribution and the originating user, while providing sender validation. Again the two protocols take a different approach in affirming this question. ECONoMy hides the content of the contribution by masking the value, thereby not linking the contents to the sender identity. Moreover, the random number generation includes open identities, and if a person can contribute a valid random number, he or she is thus part of the validated identities. PRECLUDE hides the identity corresponding to the message while allowing any participant to verify that a valid party has indeed signed the vote.

The fourth sub-question, the need for efficiency is mentioned, and the possibility of leveraging an increasing number of participating parties is



highlighted. PRECLUDE<sup>+</sup> clearly shows that for an increasing amount of parties, we can remain anonymous under the assumed constraints while reducing the number of parties we are hiding among. Doing so increases the efficiency of the entire protocol significantly. Nevertheless, both the communication and computational complexity weigh heavily on the ability to implement the PRECLUDE protocols in practice.

Finally, in the fifth sub-question, we inquire how the previously described aspects can be attained while limiting the degradation of prediction accuracy. By not perturbing the intermediate votes, we ensure that the labeling is not disturbed more than is done during the addition of global noise. As shown by Papernot et al. high prediction performance can still be achieved while adding this noise to the overall totals. Therefore, by not introducing additional limitations on the actual labeling process, while attaining the previously mentioned desired attributes, we do not alter the actual labeling and thus do not affect the prediction performance.

## 9.2 FUTURE WORK

Although the presented protocols show promising results, there are several limitations currently present that allow for improvements. In this section, we will introduce several limitations and thereby introduce opportunities for future work.

**SECURITY ASSUMPTIONS** The presented PRECLUDE protocol depends on the security provided by the TRS protocol. Fujisaki et al. [43] assume the random oracle model upon which they prove the security of the TRS protocol. This security assumption is often criticized, e.g. Canetti et al. [20] state that being secure in the random oracle model cannot be taken as evidence, nor indication, to the security of (possible) implementations of the scheme. There is a misalignment between the theoretical proof and its implementation. This discrepancy originates from the fact that the random oracle model is approximated by the use of currently available hash functions, which we assume to act accordingly. Future work will need to adapt the protocol to be proven secure under a different security assumption, where the security proof could extend to a practical implementation.

**MALICIOUS MODEL** We currently assume non-malicious adversarial models in our protocols. In practice, the adversary could have significant incentives to prevent the protocol from completing successfully. An industry leader might not want the other participating parties to attain more robust machine learning models, thereby threatening their position in the market with the accompanying financial consequences. These incentives could entice the industry leader to provide malformed input, prevent others from contributing, or attaining IoT devices that participate in a competitor's protocol to poison the input data. Future work can focus on upgrading the assumed adversarial model to offer protection in high-stakes environments.

Our first protocol ECONoMy is proven to be efficient, allowing it to be used with a larger number of participants, labeling a large number of items. However, the protocol currently assumes the ‘semi-honest’ adversarial model, i.e., the adversaries are expected to follow the protocol description, while attempting to extract sensitive information from the information they obtain. It would be interesting to see if this efficiency can be maintained by devising a joint random number generation protocol that can cope with participants misbehaving, and possibly offer a tracing capability similar to that in PRECLUDE. Similarly, in certain situations, the covert adversarial model might not be sufficient.

**PERFORMANCE** The ECONoMy protocol attains high computational performance, by providing a relatively fragile protocol. The successful completion of the protocol depends on whether all participants following precisely follow the protocol description (although privacy is preserved if a party would deviate from the intended plan). A possible direction to improve the performance of ECONoMy would be to, similar to PRECLUDE<sup>+</sup>, divide the trust among fewer parties. An extensive analysis would be required to see how this impacts the possible outcomes of a vote aggregation and the potential loss of anonymity. To provide stronger privacy guarantees, we have introduced PRECLUDE which offers a much more robust protocol that can come to its completion even if participants misbehave. Nevertheless, the computational performance drastically deteriorated due to the use of more computationally expensive operations. PRECLUDE<sup>+</sup> manages to thoroughly reduce the added complexity while remaining quite intensive due to the high number of verifications required on the proposed signatures. The same goes for the communication complexity of PRECLUDE. To apply the proposed protocols in practice, the TRS protocol will need to be adapted to provide smaller signatures to prevent the transmission from being a bottleneck in more heavy protocol executions. The extension provided in PRECLUDE<sup>+</sup> reduces the communication size of a signature compared to the original TRS protocol, but it can be argued that for large values of  $n$ ,  $t$ , and  $m$  this is reduction is not sufficient to become practical. Future work should focus on reducing the size of a single signature. Currently, a signature includes  $2t$  values of 2048 bits, resulting in significantly large signatures. Each of the provided values is required in the verification step of the protocol. The size of the signature could be reduced by allowing verification to be done a fixed number of values rather than two sets depending on the number of participants included in the ring.

**APPLICABILITY OF THE PROTOCOLS** The proposed protocols focus on the DEMA category of collaborative learning. We can adapt the underlying techniques that remove the link between the identity and the contribution for the CEMA, DIPA, and CIPA categories. In the evaluations chapter, we already indicated the performance of PRECLUDE<sup>+</sup> in a centralized setting.

Similarly, the contributed votes which now represent predictions on the public data set could represent gradient updates on a shared global model. Initially, we did not focus on these fields since we preferred the versatility offered by the [DEMA](#) category, allowing for a more generalizable solution. However, it would be interesting to see how these techniques can be applied to the iterative environment and analyze how the different gradient updates interrelate and could potentially leak identifiable information after several epochs.

**PUBLIC DATASET ALTERATION** In our protocols, we add differential privacy to the total number of votes, as is done in CrowdML or the Papernot approach we have used as a primitive. In the PRECLUDE approach, we openly share votes with the accompanying signature to provide input validation. Over multiple executions of the protocol, if the underlying model does not change, there is a chance that the contributions of specific individuals participating in a subset of executions overlap, possibly providing the link that we aim to remove. This risk can be reduced by altering the public dataset and assuming that the re-execution of the protocol is only relevant when local models are updated, and thus the contributed votes would most likely differ. An interesting problem would be to find a relation between when the participants desire to update a global model, and how much the public data needs to vary to maintain participant anonymity over time.

**OPEN PROTOCOL** The proposed protocols are assumed to be executed using a curated set of participants, no unknown participants can freely join without registering their identity in the public key generation phase. Future work can focus on developing a similar approach as described in this research, which allows anyone who wants to contribute their data to a model generation to participate. However, this would raise significant challenges in order to maintain participant anonymity. An open protocol would require stricter key management, and active participation would need to be required to gain insights into the executed protocol. Otherwise, external individuals could attain a machine learning model without contributing which can be considered unfair to the other participants. Also, the ability to freely participate could encourage malicious participants to register multiple times and thereby still be able to contribute more than their intended allowance.

**ADVERSARIAL TRAINING** The use of adversarial training shows promising results to make a model more robust to adversarial inputs. Techniques such as Ensemble Adversarial Training [120], show how to do such training efficiently. A collaborative model training scheme such as those proposed in this thesis needs to accommodate the use of such techniques to provide more robust global models. The current versions allow for such training in a 'semi-honest' setting, as every participant can

themselves generate adversarial examples and include them in their training data according to specified steps. However, when a more complicated environment is assumed, for which adversarial training is more relevant, it needs to be possible to assure that participants are doing this correctly.

**COMBINING TECHNIQUES** As discussed in Chapter 2, there is a wide variety of combining techniques within ensemble learning. The best performing one depends on the underlying problem, its data, and the extracted features. Our protocols currently offer a limited variety of combining possibilities, especially since we remove the link between the classifier and the vote. Pursuing the ability to weigh classifiers according to their prediction performance could add to the range of problems the protocols can be used for. In a semi-honest setting, we could assume that the participants provide an accurate performance representation to their peers. However, in a covert or malicious adversarial model, this would not be feasible and can be seen as an open problem.

**IMPLEMENTATION** There are several possible optimizations possible for our naive implementations that can drastically improve the presented numbers. The distributed nature would need to be evaluated in a more thoroughly constructed environment. The parallelization of function calls might not perfectly mimic the behavior of different instances in practice. Thus, a full-scale optimized implementation, utilizing multiple nodes, would be very interesting to evaluate the run-time more realistically. Further, the use of more efficient components in the implementation could help elevate the current implementation from showing correctness, to be practically feasible. Such elements might be along the lines of a speed-centric programming language, more efficient use of data structures, and the use of faster cryptographic constructs to implement our primitives.

### 9.3 CONCLUSION

The main goal of this thesis has been to maintain the ability to use the benefits offered by machine learning while preserving the privacy of participating individuals. The reviewed previous work is limited in its ability to provide this, especially in the wake of new adversarial machine learning attacks, while user demand for privacy is increasing. This thesis presents two protocols that move the collaborative learning space in the right direction. The protocols aim to alleviate privacy concerns for two different use cases, both offering stronger privacy properties than currently available alternatives.

The first protocol offers a light-weight alternative suitable for high participant application setting. It allows for a large number of users to interact with efficiency, under the 'semi-honest' threat model. However, in practice, strong incentives can be present for adversaries to ensure a sub-optimal protocol outcome. Therefore, we upgraded the threat model in our second protocol, allowing for the protocol to complete successfully,

even if a participant aims to contribute additional votes. However, the significant computational costs associated with the generation of the underlying traceable ring signatures hinders the practicality of the protocol. By adapting the underlying technique, we have been able to reduce this problem by significantly reducing run-time complexity by up to 60% as shown in Chapter 8. The presented work shows that the cost associated with privacy, regarding efficiency, can be used in a cost-benefit analysis allowing the balancing of the benefits of machine learning with the protection of training data.

The protocols described in this work provide an initial step towards solving a multi-disciplinary challenge of collaborative private model generation. The conflicting interests of machine learning researchers with those in the field of privacy need to be overcome to continue to build improved approaches to private machine learning. By aggregating insights obtained by various participants in a privacy-preserving manner, this research shows how such a multi-disciplinary mindset can allow us all to privately learn together.



## BIBLIOGRAPHY

---

- [1] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. “Deep Learning with Differential Privacy.” In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*. 2016, pp. 308–318. DOI: [10 . 1145 / 2976749 . 2978318](https://doi.org/10.1145/2976749.2978318). URL: <http://doi.acm.org/10.1145/2976749.2978318>.
- [2] Amazon. *Amazon EC2 C5 Instances*. 2018. URL: <https://aws.amazon.com/ec2/instance-types/c5/> (visited on 03/26/2018).
- [3] Alborz Amir-Khalili, Soheil Kianzad, Rafeef Abugharbieh, and Ivan Beschastnikh. “Scalable and Fault Tolerant Platform for Distributed Learning on Private Medical Data.” In: *Machine Learning in Medical Imaging - 8th International Workshop, MLMI 2017, Held in Conjunction with MICCAI 2017, Quebec City, QC, Canada, September 10, 2017, Proceedings*. 2017, pp. 176–184. DOI: [10 . 1007 / 978 - 3 - 319 - 67389 - 9 \\_ 21](https://doi.org/10.1007/978-3-319-67389-9_21). URL: [https://doi.org/10.1007/978-3-319-67389-9\\_21](https://doi.org/10.1007/978-3-319-67389-9_21).
- [4] Louis J. M. Aslett, Pedro M. Esperança, and Chris C. Holmes. “Encrypted statistical machine learning: new privacy preserving methods.” In: *CoRR abs/1508.06845* (2015). arXiv: [1508 . 06845](https://arxiv.org/abs/1508.06845). URL: <http://arxiv.org/abs/1508.06845>.
- [5] Maria-Florina Balcan, Avrim Blum, Shai Fine, and Yishay Mansour. “Distributed Learning, Communication Complexity and Privacy.” In: *COLT 2012 - The 25th Annual Conference on Learning Theory, June 25-27, 2012, Edinburgh, Scotland*. 2012, pp. 26.1–26.22. URL: <http://www.jmlr.org/proceedings/papers/v23/balcan12a/balcan12a.pdf>.
- [6] Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid. “Recommendation for key management part 1: General (revision 3).” In: *NIST special publication 800.57* (2012), pp. 1–147.
- [7] Mihir Bellare and Phillip Rogaway. “Random Oracles are Practical: A Paradigm for Designing Efficient Protocols.” In: *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993*. 1993, pp. 62–73. DOI: [10 . 1145 / 168588 . 168596](https://doi.org/10.1145/168588.168596). URL: <http://doi.acm.org/10.1145/168588.168596>.
- [8] Mihir Bellare, Haixia Shi, and Chong Zhang. “Foundations of Group Signatures: The Case of Dynamic Groups.” In: *Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005, Proceedings*. 2005, pp. 136–153. DOI: [10 . 1007 / 978 - 3 - 540 - 30574 - 3 \\_ 11](https://doi.org/10.1007/978-3-540-30574-3_11). URL: [https://doi.org/10.1007/978-3-540-30574-3\\_11](https://doi.org/10.1007/978-3-540-30574-3_11).

- [9] Aurélien Bellet, Rachid Guerraoui, Mahsa Taziki, and Marc Tommasi. "Personalized and Private Peer-to-Peer Machine Learning." In: *NIPS 2017 Workshop on Machine Learning on the Phone and other Consumer Devices*. 2017.
- [10] Steven M. Bellovin. "Frank Miller: Inventor of the One-Time Pad." In: *Cryptologia* 35.3 (2011), pp. 203–222. DOI: [10.1080/01611194.2011.583711](https://doi.org/10.1080/01611194.2011.583711). URL: <https://doi.org/10.1080/01611194.2011.583711>.
- [11] Kanishka Bhaduri, Ran Wolff, Chris Giannella, and Hillol Kargupta. "Distributed Decision-Tree Induction in Peer-to-Peer Systems." In: *Statistical Analysis and Data Mining* 1.2 (2008), pp. 85–103. DOI: [10.1002/sam.10006](https://doi.org/10.1002/sam.10006). URL: <https://doi.org/10.1002/sam.10006>.
- [12] Mariusz Bojarski et al. "End to End Learning for Self-Driving Cars." In: *CoRR abs/1604.07316* (2016). arXiv: [1604.07316](https://arxiv.org/abs/1604.07316). URL: <http://arxiv.org/abs/1604.07316>.
- [13] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. "Practical Secure Aggregation for Privacy-Preserving Machine Learning." In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. 2017, pp. 1175–1191. DOI: [10.1145/3133956.3133982](https://doi.acm.org/10.1145/3133956.3133982). URL: <http://doi.acm.org/10.1145/3133956.3133982>.
- [14] Dan Boneh and Xavier Boyen. "Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles." In: *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*. 2004, pp. 223–238. DOI: [10.1007/978-3-540-24676-3\\_14](https://doi.org/10.1007/978-3-540-24676-3_14). URL: [https://doi.org/10.1007/978-3-540-24676-3\\_14](https://doi.org/10.1007/978-3-540-24676-3_14).
- [15] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. "Evaluating 2-DNF Formulas on Ciphertexts." In: *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*. 2005, pp. 325–341. DOI: [10.1007/978-3-540-30576-7\\_18](https://doi.org/10.1007/978-3-540-30576-7_18). URL: [https://doi.org/10.1007/978-3-540-30576-7\\_18](https://doi.org/10.1007/978-3-540-30576-7_18).
- [16] Raphael Bost, Raluca Ada Popa, Stephen Tu, and Shafi Goldwasser. "Machine Learning Classification over Encrypted Data." In: *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015*. 2015. URL: <https://www.ndss-symposium.org/ndss2015/machine-learning-classification-over-encrypted-data>.
- [17] MW. MR. N.M. Brouwer. "De Algemene Verordening Gegevensbescherming." In: (2018). URL: [https://www.dirkzwager.nl/media/21631/de\\_algemene\\_verordening\\_gegevensbescherming.pdf](https://www.dirkzwager.nl/media/21631/de_algemene_verordening_gegevensbescherming.pdf).



- [18] Miles Brundage et al. "The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation." In: *CoRR* abs/1802.07228 (2018). arXiv: [1802.07228](https://arxiv.org/abs/1802.07228). URL: <http://arxiv.org/abs/1802.07228>.
- [19] Jan Camenisch and Markus Stadler. "Efficient Group Signature Schemes for Large Groups (Extended Abstract)." In: *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*. 1997, pp. 410–424. DOI: [10 . 1007 / BFb0052252](https://doi.org/10.1007/BFb0052252). URL: <https://doi.org/10.1007/BFb0052252>.
- [20] Ran Canetti, Oded Goldreich, and Shai Halevi. "The random oracle methodology, revisited." In: *J. ACM* 51.4 (2004), pp. 557–594. DOI: [10 . 1145/1008731.1008734](https://doi.org/10.1145/1008731.1008734). URL: <http://doi.acm.org/10.1145/1008731.1008734>.
- [21] Miguel Castro and Barbara Liskov. "Practical Byzantine Fault Tolerance." In: *Proceedings of the Third USENIX Symposium on Operating Systems Design and Implementation (OSDI), New Orleans, Louisiana, USA, February 22-25, 1999*. 1999, pp. 173–186. DOI: [10 . 1145 / 296806 . 296824](https://doi.org/10.1145/296806.296824). URL: <http://doi.acm.org/10.1145/296806.296824>.
- [22] Hervé Chabanne, Amaury de Wargny, Jonathan Milgram, Constance Morel, and Emmanuel Prouff. "Privacy-Preserving Classification on Deep Neural Network." In: *IACR Cryptology ePrint Archive 2017* (2017), p. 35. URL: <http://eprint.iacr.org/2017/035>.
- [23] Michael Chui. "Artificial intelligence the next digital frontier?" In: *McKinsey and Company Global Institute* (2017), p. 47.
- [24] Nigel Cory. "Cross-border data flows: Where are the barriers, and what do they cost." In: *Information Technology and Innovation Foundation, May 1* (2017).
- [25] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. "Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols." In: *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*. 1994, pp. 174–187. DOI: [10 . 1007 / 3 - 540 - 48658 - 5 \\_ 19](https://doi.org/10.1007/3-540-48658-5_19). URL: [https://doi.org/10.1007/3-540-48658-5\\_19](https://doi.org/10.1007/3-540-48658-5_19).
- [26] Bennett Cyphers and Kalyan Veeramachaneni. "AnonML: Locally Private Machine Learning over a Network of Peers." In: *2017 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2017, Tokyo, Japan, October 19-21, 2017*. 2017, pp. 549–560. DOI: [10 . 1109 / DSAA . 2017 . 80](https://doi.org/10.1109/DSAA.2017.80). URL: <https://doi.org/10.1109/DSAA.2017.80>.
- [27] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002. ISBN: 3-540-42580-2. DOI: [10 . 1007 / 978 - 3 - 662 - 04722 - 4](https://doi.org/10.1007/978-3-662-04722-4). URL: <https://doi.org/10.1007/978-3-662-04722-4>.

- [28] George Danezis. *A bilinear pairing library for petlib*. 2018. URL: <https://pypi.org/project/bplib/> (visited on 06/26/2018).
- [29] George Danezis. *A library implementing a number of Privacy Enhancing Technologies (PETs)*. 2018. URL: <https://pypi.org/project/petlib/> (visited on 06/26/2018).
- [30] Hans Delfs and Helmut Knebl. *Introduction to Cryptography - Principles and Applications, Third Edition*. Information Security and Cryptography. Springer, 2015. ISBN: 978-3-662-47973-5. DOI: 10.1007/978-3-662-47974-2. URL: <https://doi.org/10.1007/978-3-662-47974-2>.
- [31] Oxford Dictionary. *Definition of the word 'adversary'*. 2018. URL: <https://en.oxforddictionaries.com/definition/adversary> (visited on 06/30/2018).
- [32] Whitfield Diffie and Martin E. Hellman. "New directions in cryptography." In: *IEEE Trans. Information Theory* 22.6 (1976), pp. 644–654. DOI: 10.1109/TIT.1976.1055638. URL: <https://doi.org/10.1109/TIT.1976.1055638>.
- [33] Yevgeniy Dodis, Aleksandr Yampolskiy, and Moti Yung. "Threshold and Proactive Pseudo-Random Permutations." In: *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*. 2006, pp. 542–560. DOI: 10.1007/11681878\_28. URL: [https://doi.org/10.1007/11681878\\_28](https://doi.org/10.1007/11681878_28).
- [34] Cynthia Dwork. "Differential Privacy: A Survey of Results." In: *Theory and Applications of Models of Computation, 5th International Conference, TAMC 2008, Xi'an, China, April 25-29, 2008. Proceedings*. 2008, pp. 1–19. DOI: 10.1007/978-3-540-79228-4\_1. URL: [https://doi.org/10.1007/978-3-540-79228-4\\_1](https://doi.org/10.1007/978-3-540-79228-4_1).
- [35] Morris J Dworkin. *SHA-3 standard: Permutation-based hash and extendable-output functions*. Tech. rep. 2015.
- [36] Morris Dworkin. *Recommendation for block cipher modes of operation. methods and techniques*. Tech. rep. NATIONAL INST OF STANDARDS and TECHNOLOGY GAITHERSBURG MD COMPUTER SECURITY DIV, 2001.
- [37] Uriel Feige, Amos Fiat, and Adi Shamir. "Zero-Knowledge Proofs of Identity." In: *J. Cryptology* 1.2 (1988), pp. 77–94. DOI: 10.1007/BF02351717. URL: <https://doi.org/10.1007/BF02351717>.
- [38] Amos Fiat and Adi Shamir. "How to Prove Yourself: Practical Solutions to Identification and Signature Problems." In: *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*. 1986, pp. 186–194. DOI: 10.1007/3-540-47721-7\_12. URL: [https://doi.org/10.1007/3-540-47721-7\\_12](https://doi.org/10.1007/3-540-47721-7_12).

- [39] Gianluigi Folino, Clara Pizzuti, and Giandomenico Spezzano. “GP Ensemble for Distributed Intrusion Detection Systems.” In: *Pattern Recognition and Data Mining, Third International Conference on Advances in Pattern Recognition, ICAPR 2005, Bath, UK, August 22-25, 2005, Proceedings, Part I*. 2005, pp. 54–62. DOI: [10.1007/11551188\\_6](https://doi.org/10.1007/11551188_6). URL: [https://doi.org/10.1007/11551188\\_6](https://doi.org/10.1007/11551188_6).
- [40] Python Software Foundation. *Python 2.7*. 2010. URL: <https://www.python.org/download/releases/2.7/> (visited on 07/03/2010).
- [41] 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, Denver, CO, USA, October 12-6, 2015*. 2015, pp. 1322–1333. DOI: [10.1145/2810103.2813677](https://doi.org/10.1145/2810103.2813677). URL: <http://doi.acm.org/10.1145/2810103.2813677>.
- [42] Yoav Freund and Robert E. Schapire. “A decision-theoretic generalization of on-line learning and an application to boosting.” In: *Computational Learning Theory, Second European Conference, EuroCOLT '95, Barcelona, Spain, March 13-15, 1995, Proceedings*. 1995, pp. 23–37. DOI: [10.1007/3-540-59119-2\\_166](https://doi.org/10.1007/3-540-59119-2_166). URL: [https://doi.org/10.1007/3-540-59119-2\\_166](https://doi.org/10.1007/3-540-59119-2_166).
- [43] Eiichiro Fujisaki and Koutarou Suzuki. “Traceable ring signature.” In: *International Workshop on Public Key Cryptography*. Springer. 2007, pp. 181–200.
- [44] Flavio D. Garcia and Bart Jacobs. “Privacy-Friendly Energy-Metering via Homomorphic Encryption.” In: *Security and Trust Management - 6th International Workshop, STM 2010, Athens, Greece, September 23-24, 2010, Revised Selected Papers*. 2010, pp. 226–238. DOI: [10.1007/978-3-642-22444-7\\_15](https://doi.org/10.1007/978-3-642-22444-7_15). URL: [https://doi.org/10.1007/978-3-642-22444-7\\_15](https://doi.org/10.1007/978-3-642-22444-7_15).
- [45] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin E. Lauter, Michael Naehrig, and John Wernsing. “CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy.” In: *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. 2016, pp. 201–210. URL: <http://jmlr.org/proceedings/papers/v48/gilad-bachrach16.html>.
- [46] “Glossary of Terms.” In: *Mach. Learn.* 30.2-3 (Feb. 1998), pp. 271–274. ISSN: 0885-6125. URL: <http://dl.acm.org/citation.cfm?id=288808.288815>.
- [47] Carlos A. Gomez-Uribe and Neil Hunt. “The Netflix Recommender System: Algorithms, Business Value, and Innovation.” In: *ACM Trans. Management Inf. Syst.* 6.4 (2016), 13:1–13:19. DOI: [10.1145/2843948](https://doi.org/10.1145/2843948). URL: <http://doi.acm.org/10.1145/2843948>.

- [48] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples." In: *CoRR* abs/1412.6572 (2014). arXiv: 1412 . 6572. URL: <http://arxiv.org/abs/1412.6572>.
- [49] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. "Generative Adversarial Networks." In: *CoRR* abs/1406.2661 (2014). arXiv: 1406.2661. URL: <http://arxiv.org/abs/1406.2661>.
- [50] Thore Graepel, Kristin E. Lauter, and Michael Naehrig. "ML Confidential: Machine Learning on Encrypted Data." In: *Information Security and Cryptology - ICISC 2012 - 15th International Conference, Seoul, Korea, November 28-30, 2012, Revised Selected Papers*. 2012, pp. 1–21. DOI: 10 . 1007 / 978 - 3 - 642 - 37682 - 5 \_ 1. URL: [https://doi.org/10.1007/978-3-642-37682-5\\_1](https://doi.org/10.1007/978-3-642-37682-5_1).
- [51] Ilya Sutskever Greg Brockman and OpenAI. *Introducing OpenAI*. 2015. URL: <https://blog.openai.com/introducing-openai> (visited on 03/26/2018).
- [52] The Guardian. *Cambridge Analytica News Overviews*. 2018. URL: <https://www.theguardian.com/uk-news/cambridge-analytica> (visited on 03/26/2018).
- [53] Jihun Hamm, Yingjun Cao, and Mikhail Belkin. "Learning privately from multiparty data." In: *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. 2016, pp. 555–563. URL: <http://jmlr.org/proceedings/papers/v48/hamm16.html>.
- [54] Jihun Hamm, Adam C. Champion, Guoxing Chen, Mikhail Belkin, and Dong Xuan. "Crowd-ML: A Privacy-Preserving Learning Framework for a Crowd of Smart Devices." In: *35th IEEE International Conference on Distributed Computing Systems, ICDCS 2015, Columbus, OH, USA, June 29 - July 2, 2015*. 2015, pp. 11–20. DOI: 10 . 1109 / ICDCS . 2015 . 10. URL: <https://doi.org/10.1109/ICDCS.2015.10>.
- [55] Carmit Hazay and Yehuda Lindell. *Efficient Secure Two-Party Protocols - Techniques and Constructions*. Information Security and Cryptography. Springer, 2010. ISBN: 978-3-642-14302-1. DOI: 10 . 1007 / 978 - 3 - 642 - 14303 - 8. URL: <https://doi.org/10.1007/978-3-642-14303-8>.
- [56] Alan Hevner and Samir Chatterjee. "Design science research in information systems." In: *Design research in information systems*. Springer, 2010, pp. 9–22.
- [57] Briland Hitaj, Giuseppe Ateniese, and Fernando Pérez-Cruz. "Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning." In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. 2017, pp. 603–618. DOI: 10 . 1145 / 3133956 . 3134012. URL: <http://doi.acm.org/10.1145/3133956.3134012>.

- [58] Susan Hohenberger, Steven Myers, Rafael Pass, and Abhi Shelat. "ANONIZE: A Large-Scale Anonymous Survey System." In: *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*. 2014, pp. 375–389. DOI: [10.1109/SP.2014.31](https://doi.org/10.1109/SP.2014.31). URL: <https://doi.org/10.1109/SP.2014.31>.
- [59] Xiaohua Hu. "Using Rough Sets Theory and Database Operations to Construct a Good Ensemble of Classifiers for Data Mining Applications." In: *Proceedings of the 2001 IEEE International Conference on Data Mining, 29 November - 2 December 2001, San Jose, California, USA*. 2001, pp. 233–240. DOI: [10.1109/ICDM.2001.989524](https://doi.org/10.1109/ICDM.2001.989524). URL: <https://doi.org/10.1109/ICDM.2001.989524>.
- [60] Ling Huang, Anthony D. Joseph, Blaine Nelson, Benjamin I. P. Rubinstein, and J. D. Tygar. "Adversarial machine learning." In: *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence, AISEC 2011, Chicago, IL, USA, October 21, 2011*. 2011, pp. 43–58. DOI: [10.1145/2046684.2046692](https://doi.acm.org/10.1145/2046684.2046692). URL: <http://doi.acm.org/10.1145/2046684.2046692>.
- [61] Zhenqi Huang, Sayan Mitra, and Nitin Vaidya. "Differentially Private Distributed Optimization." In: *Proceedings of the 2015 International Conference on Distributed Computing and Networking, ICDCN 2015, Goa, India, January 4-7, 2015*. 2015, 4:1–4:10. DOI: [10.1145/2684464.2684480](https://doi.acm.org/10.1145/2684464.2684480). URL: <http://doi.acm.org/10.1145/2684464.2684480>.
- [62] 16 TRUSTe NCSA Consumer Privacy Infographic. *U.S. CONSUMER PRIVACY INDEX 2016*. 2016. URL: <https://download.trustarc.com/download.php?f=7UU0QTHS-597> (visited on 06/02/2018).
- [63] Quote Investigator. *Everything Should Be Made as Simple as Possible, But Not Simpler*. 2018. URL: <https://quoteinvestigator.com/2011/05/13/einstein-simple/> (visited on 07/26/2018).
- [64] Zhanglong Ji, Zachary Chase Lipton, and Charles Elkan. "Differential Privacy and Machine Learning: a Survey and Review." In: *CoRR abs/1412.7584* (2014). arXiv: [1412.7584](https://arxiv.org/abs/1412.7584). URL: <http://arxiv.org/abs/1412.7584>.
- [65] Zhanglong Ji, Zachary Chase Lipton, and Charles Elkan. "Differential Privacy and Machine Learning: a Survey and Review." In: *CoRR abs/1412.7584* (2014). arXiv: [1412.7584](https://arxiv.org/abs/1412.7584). URL: <http://arxiv.org/abs/1412.7584>.
- [66] Michael I Jordan and Tom M Mitchell. "Machine learning: Trends, perspectives, and prospects." In: *Science* 349.6245 (2015), pp. 255–260.
- [67] Murat Kantarcioglu and Chris Clifton. "Privacy-Preserving Distributed Mining of Association Rules on Horizontally Partitioned Data." In: *IEEE Trans. Knowl. Data Eng.* 16.9 (2004), pp. 1026–1037. DOI: [10.1109/TKDE.2004.45](https://doi.org/10.1109/TKDE.2004.45). URL: <https://doi.org/10.1109/TKDE.2004.45>.

- [68] Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. “Federated Optimization: Distributed Machine Learning for On-Device Intelligence.” In: *CoRR* abs/1610.02527 (2016). arXiv: 1610.02527. URL: <http://arxiv.org/abs/1610.02527>.
- [69] Igor Kononenko. “Machine learning for medical diagnosis: history, state of the art and perspective.” In: *Artificial Intelligence in Medicine* 23.1 (2001), pp. 89–109. DOI: 10.1016/S0933-3657(01)00077-X. URL: [https://doi.org/10.1016/S0933-3657\(01\)00077-X](https://doi.org/10.1016/S0933-3657(01)00077-X).
- [70] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. “Adversarial examples in the physical world.” In: *CoRR* abs/1607.02533 (2016). arXiv: 1607.02533. URL: <http://arxiv.org/abs/1607.02533>.
- [71] Klaus Kursawe, George Danezis, and Markulf Kohlweiss. “Privacy-Friendly Aggregation for the Smart-Grid.” In: *Privacy Enhancing Technologies - 11th International Symposium, PETS 2011, Waterloo, ON, Canada, July 27-29, 2011. Proceedings*. 2011, pp. 175–191. DOI: 10.1007/978-3-642-22263-4\_10. URL: [https://doi.org/10.1007/978-3-642-22263-4\\_10](https://doi.org/10.1007/978-3-642-22263-4_10).
- [72] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. “The Byzantine Generals Problem.” In: *ACM Trans. Program. Lang. Syst.* 4.3 (1982), pp. 382–401. DOI: 10.1145/357172.357176. URL: <http://doi.acm.org/10.1145/357172.357176>.
- [73] Pavel Laskov and Richard Lippmann. “Machine learning in adversarial environments.” In: *Machine Learning* 81.2 (2010), pp. 115–119. DOI: 10.1007/s10994-010-5207-6. URL: <https://doi.org/10.1007/s10994-010-5207-6>.
- [74] Yann LeCun and Corinna Cortes. “MNIST handwritten digit database.” In: (2010). URL: <http://yann.lecun.com/exdb/mnist/>.
- [75] Yan Li, Changxin Bai, and Chandan K. Reddy. “A distributed ensemble approach for mining healthcare data under privacy constraints.” In: *Inf. Sci.* 330 (2016), pp. 245–259. DOI: 10.1016/j.ins.2015.10.011. URL: <https://doi.org/10.1016/j.ins.2015.10.011>.
- [76] Yehuda Lindell. “How to Simulate It - A Tutorial on the Simulation Proof Technique.” In: *Tutorials on the Foundations of Cryptography*. 2017, pp. 277–346. DOI: 10.1007/978-3-319-57048-8\_6. URL: [https://doi.org/10.1007/978-3-319-57048-8\\_6](https://doi.org/10.1007/978-3-319-57048-8_6).
- [77] Yehuda Lindell and Benny Pinkas. “Privacy Preserving Data Mining.” In: *J. Cryptology* 15.3 (2002), pp. 177–206. DOI: 10.1007/s00145-001-0019-2. URL: <https://doi.org/10.1007/s00145-001-0019-2>.
- [78] Dwayne C. Litzenberger. *Python Cryptography Toolkit (pycrypto)*. 2018. URL: <https://pypi.org/project/pycrypto/> (visited on 06/26/2018).

- [79] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. "Linkable Spontaneous Anonymous Group Signature for Ad Hoc Groups (Extended Abstract)." In: *Information Security and Privacy: 9th Australasian Conference, ACISP 2004, Sydney, Australia, July 13-15, 2004. Proceedings*. 2004, pp. 325–335. DOI: [10.1007/978-3-540-27800-9\\_28](https://doi.org/10.1007/978-3-540-27800-9_28). URL: [https://doi.org/10.1007/978-3-540-27800-9\\_28](https://doi.org/10.1007/978-3-540-27800-9_28).
- [80] Ping Luo, Hui Xiong, Kevin Lü, and Zhongzhi Shi. "Distributed classification in peer-to-peer networks." In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California, USA, August 12-15, 2007*. 2007, pp. 968–976. DOI: [10.1145/1281192.1281296](https://doi.org/10.1145/1281192.1281296). URL: <http://doi.acm.org/10.1145/1281192.1281296>.
- [81] Sam Maes, Karl Tuyls, Bram Vanschoenwinkel, and Bernard Manderick. "Credit card fraud detection using Bayesian and neural networks." In: *Proceedings of the 1st international naiso congress on neuro fuzzy technologies*. 2002, pp. 261–270.
- [82] Nader Mahmoudi and Ekrem Duman. "Detecting credit card fraud by Modified Fisher Discriminant Analysis." In: *Expert Syst. Appl.* 42.5 (2015), pp. 2510–2516. DOI: [10.1016/j.eswa.2014.10.037](https://doi.org/10.1016/j.eswa.2014.10.037). URL: <https://doi.org/10.1016/j.eswa.2014.10.037>.
- [83] Winston Maxwell and Christopher Wolf. "A Global Reality: Governmental Access to Data in the Cloud." In: *white paper, Hogan Lovells* 18 (2012).
- [84] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. "Communication-Efficient Learning of Deep Networks from Decentralized Data." In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*. 2017, pp. 1273–1282. URL: <http://proceedings.mlr.press/v54/mcmahan17a.html>.
- [85] Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996. ISBN: 0-8493-8523-7.
- [86] Dongyu Meng and Hao Chen. "MagNet: A Two-Pronged Defense against Adversarial Examples." In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. 2017, pp. 135–147. DOI: [10.1145/3133956.3134057](https://doi.org/10.1145/3133956.3134057). URL: <http://doi.acm.org/10.1145/3133956.3134057>.
- [87] Esther Mengelkamp, Benedikt Notheisen, Carolin Beer, David Dauer, and Christof Weinhardt. "A blockchain-based smart grid: towards sustainable local energy markets." In: *Computer Science - R&D* 33.1-2 (2018), pp. 207–214. DOI: [10.1007/s00450-017-0360-9](https://doi.org/10.1007/s00450-017-0360-9). URL: <https://doi.org/10.1007/s00450-017-0360-9>.

- [88] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. "Universal Adversarial Perturbations." In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. 2017, pp. 86–94. DOI: [10.1109/CVPR.2017.17](https://doi.org/10.1109/CVPR.2017.17). URL: <https://doi.org/10.1109/CVPR.2017.17>.
- [89] Satoshi Nakamoto. "Bitcoin: A peer-to-peer electronic cash system." In: (2008).
- [90] Moni Naor. "Deniable Ring Authentication." In: *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*. 2002, pp. 481–498. DOI: [10.1007/3-540-45708-9\\_31](https://doi.org/10.1007/3-540-45708-9_31). URL: [https://doi.org/10.1007/3-540-45708-9\\_31](https://doi.org/10.1007/3-540-45708-9_31).
- [91] Netflix. *Results 4th quarter 2017*. 2018. URL: <https://ir.netflix.com/static-files/0c060a3f-d903-4eb9-bde6-bf3e58761712> (visited on 03/30/2018).
- [92] Shen Noether, Adam Mackenzie, et al. "Ring confidential transactions." In: *Ledger 1* (2016), pp. 1–18.
- [93] Pascal Paillier. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes." In: *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*. 1999, pp. 223–238. DOI: [10.1007/3-540-48910-X\\_16](https://doi.org/10.1007/3-540-48910-X_16). URL: [https://doi.org/10.1007/3-540-48910-X\\_16](https://doi.org/10.1007/3-540-48910-X_16).
- [94] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. "Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data." In: *CoRR abs/1610.05755* (2016). arXiv: [1610.05755](http://arxiv.org/abs/1610.05755). URL: <http://arxiv.org/abs/1610.05755>.
- [95] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. "The Limitations of Deep Learning in Adversarial Settings." In: *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*. 2016, pp. 372–387. DOI: [10.1109/EuroSP.2016.36](https://doi.org/10.1109/EuroSP.2016.36). URL: <https://doi.org/10.1109/EuroSP.2016.36>.
- [96] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. "Towards the science of security and privacy in machine learning." In: *arXiv preprint arXiv:1611.03814* (2016).
- [97] Manas A. Pathak, Shantanu Rane, and Bhiksha Raj. "Multiparty Differential Privacy via Aggregation of Locally Trained Classifiers." In: *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*. 2010, pp. 1876–1884. URL: <http://papers.nips.cc/paper/4034-multiparty-differential-privacy-via-aggregation-of-locally-trained-classifiers>.



- [98] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.
- [99] Gareth W Peters and Efstathios Panayi. "Understanding modern banking ledgers through blockchain technologies: Future of transaction processing and smart contracts on the internet of money." In: *Banking Beyond Banks and Money*. Springer, 2016, pp. 239–278.
- [100] David Pointcheval and Jacques Stern. "Provably Secure Blind Signature Schemes." In: *Advances in Cryptology - ASIACRYPT '96, International Conference on the Theory and Applications of Cryptology and Information Security, Kyongju, Korea, November 3-7, 1996, Proceedings*. 1996, pp. 252–265. DOI: [10 . 1007 / BFb0034852](https://doi.org/10.1007/BFb0034852). URL: <https://doi.org/10.1007/BFb0034852>.
- [101] David Pointcheval and Jacques Stern. "Security Arguments for Digital Signatures and Blind Signatures." In: *J. Cryptology* 13.3 (2000), pp. 361–396. DOI: [10 . 1007 / s001450010003](https://doi.org/10.1007/s001450010003). URL: <https://doi.org/10.1007/s001450010003>.
- [102] Jean-Francois Puget. *The Most Popular Language For Machine Learning Is ...* 2016. URL: <https://blog.openai.com/introducing-openai> (visited on 12/29/2016).
- [103] Charles Rackoff and Daniel R. Simon. "Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack." In: *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*. 1991, pp. 433–444. DOI: [10 . 1007 / 3 - 540 - 46766 - 1 \\_ 35](https://doi.org/10.1007/3-540-46766-1_35). URL: [https://doi.org/10.1007/3-540-46766-1\\_35](https://doi.org/10.1007/3-540-46766-1_35).
- [104] Arun Rajkumar and Shivani Agarwal. "A Differentially Private Stochastic Gradient Descent Algorithm for Multiparty Classification." In: *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2012, La Palma, Canary Islands, April 21-23, 2012*. 2012, pp. 933–941. URL: <http://jmlr.csail.mit.edu/proceedings/papers/v22/rajkumar12.html>.
- [105] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems." In: *Commun. ACM* 21.2 (1978), pp. 120–126. DOI: [10.1145/359340.359342](https://doi.org/10.1145/359340.359342). URL: <http://doi.acm.org/10.1145/359340.359342>.
- [106] Ronald L. Rivest, Adi Shamir, and Yael Tauman. "How to Leak a Secret." In: *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*. 2001, pp. 552–565. DOI: [10 . 1007 / 3 - 540 - 45682 - 1 \\_ 32](https://doi.org/10.1007/3-540-45682-1_32). URL: [https://doi.org/10.1007/3-540-45682-1\\_32](https://doi.org/10.1007/3-540-45682-1_32).

- [107] Sandra Servia Rodríguez, Liang Wang, Jianxin R. Zhao, Richard Mortier, and Hamed Haddadi. “Personal Model Training under Privacy Constraints.” In: *CoRR* abs/1703.00380 (2017). arXiv: 1703.00380. URL: <http://arxiv.org/abs/1703.00380>.
- [108] Lior Rokach. “Ensemble-based classifiers.” In: *Artif. Intell. Rev.* 33.1-2 (2010), pp. 1–39. DOI: 10.1007/s10462-009-9124-7. URL: <https://doi.org/10.1007/s10462-009-9124-7>.
- [109] Reza Shokri and Vitaly Shmatikov. “Privacy-Preserving Deep Learning.” In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*. 2015, pp. 1310–1321. DOI: 10.1145/2810103.2813687. URL: <http://doi.acm.org/10.1145/2810103.2813687>.
- [110] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. “Membership Inference Attacks Against Machine Learning Models.” In: *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*. 2017, pp. 3–18. DOI: 10.1109/SP.2017.41. URL: <https://doi.org/10.1109/SP.2017.41>.
- [111] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S. Talwalkar. “Federated Multi-Task Learning.” In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. 2017, pp. 4427–4437. URL: <http://papers.nips.cc/paper/7029-federated-multi-task-learning>.
- [112] Peter Sollich and Anders Krogh. “Learning with ensembles: How overfitting can be useful.” In: *Advances in Neural Information Processing Systems 8, NIPS, Denver, CO, USA, November 27-30, 1995*. 1995, pp. 190–196. URL: <http://papers.nips.cc/paper/1044-learning-with-ensembles-how-overfitting-can-be-useful>.
- [113] Daniel Solove. “A brief history of information privacy law.” In: (2006).
- [114] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. “One pixel attack for fooling deep neural networks.” In: *CoRR* abs/1710.08864 (2017). arXiv: 1710.08864. URL: <http://arxiv.org/abs/1710.08864>.
- [115] Delft University of Technology. *IBM Power Systems*. 2018. URL: <http://www.ce.ewi.tudelft.nl/ce-infra/servers/ibm-power-systems/> (visited on 03/26/2018).
- [116] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition, fourth edition*. Elsevier, 2009. ISBN: 9780123744869. URL: <https://www.elsevier.com/books/pattern-recognition/theodoridis/978-1-59749-272-0>.
- [117] Feng Tian. “An agri-food supply chain traceability system for China based on RFID & blockchain technology.” In: *Service Systems and Service Management (ICSSSM), 2016 13th International Conference on*. IEEE. 2016, pp. 1–6.

- [118] Lu Tian, Bargav Jayaraman, Quanquan Gu, and David Evans. "Aggregating Private Sparse Learning Models Using Multi-Party Computation." In: *NIPS Workshop on Private Multi-Party Machine Learning, Barcelona, Spain*. 2016.
- [119] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. "Stealing Machine Learning Models via Prediction APIs." In: *USENIX Security Symposium*. 2016, pp. 601–618.
- [120] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick D. McDaniel. "Ensemble Adversarial Training: Attacks and Defenses." In: *CoRR abs/1705.07204* (2017). arXiv: [1705.07204](https://arxiv.org/abs/1705.07204). URL: <http://arxiv.org/abs/1705.07204>.
- [121] European Union. *Homepage of EU GDPR*. 2018. URL: <https://www.eugdpr.org> (visited on 03/26/2018).
- [122] Jaideep Vaidya and Chris Clifton. "Privacy-preserving  $k$ -means clustering over vertically partitioned data." In: *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 24 - 27, 2003*. 2003, pp. 206–215. DOI: [10 . 1145 / 956750 . 956776](https://doi.org/10.1145/956750.956776). URL: <http://doi.acm.org/10.1145/956750.956776>.
- [123] Ian H. Witten, Frank Eibe, and Mark A. Hall. *Data mining: practical machine learning tools and techniques, 3rd Edition*. Morgan Kaufmann, Elsevier, 2011. ISBN: 9780123748560. URL: <http://www.worldcat.org/oclc/262433473>.
- [124] Yuan Wu, Xiaoqian Jiang, Jihoon Kim, and Lucila Ohno-Machado. "Grid Binary LOGistic REgression (GLORE): building shared models without sharing data." In: *JAMIA 19.5* (2012), pp. 758–764. DOI: [10 . 1136 / amiajnl - 2012 - 000862](https://doi.org/10.1136/amiajnl-2012-000862). URL: <https://doi.org/10.1136/amiajnl-2012-000862>.
- [125] Liyang Xie, Sergey M. Plis, and Anand D. Sarwate. "Data-weighted ensemble learning for privacy-preserving distributed learning." In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016*. 2016, pp. 2309–2313. DOI: [10 . 1109 / ICASSP . 2016 . 7472089](https://doi.org/10.1109/ICASSP.2016.7472089). URL: <https://doi.org/10.1109/ICASSP.2016.7472089>.
- [126] Maurice Yarrow, Rob VanderWijngaart, and Paul Kutler. "Communication improvement for the LU NAS parallel benchmark: A model for efficient parallel relaxation schemes." In: (1997).
- [127] Ryo Yonetani, Vishnu Naresh Boddeti, Kris M. Kitani, and Yoichi Sato. "Privacy-Preserving Visual Learning Using Doubly Permuted Homomorphic Encryption." In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. 2017, pp. 2059–2069. DOI: [10 . 1109 / ICCV . 2017 . 225](https://doi.org/10.1109/ICCV.2017.225). URL: <https://doi.org/10.1109/ICCV.2017.225>.

- [128] Hwanjo Yu, Xiaoqian Jiang, and Jaideep Vaidya. "Privacy-preserving SVM using nonlinear kernels on horizontally partitioned data." In: *Proceedings of the 2006 ACM Symposium on Applied Computing (SAC), Dijon, France, April 23-27, 2006*. Ed. by Hisham Haddad. ACM, 2006, pp. 603–610. DOI: [10.1145/1141277.1141415](https://doi.org/10.1145/1141277.1141415). URL: <http://doi.acm.org/10.1145/1141277.1141415>.
- [129] Jiawei Yuan and Shucheng Yu. "Privacy Preserving Back-Propagation Learning Made Practical with Cloud Computing." In: *Security and Privacy in Communication Networks - 8th International ICST Conference, SecureComm 2012, Padua, Italy, September 3-5, 2012. Revised Selected Papers*. 2012, pp. 292–309. DOI: [10.1007/978-3-642-36883-7\\_18](https://doi.org/10.1007/978-3-642-36883-7_18). URL: [https://doi.org/10.1007/978-3-642-36883-7\\_18](https://doi.org/10.1007/978-3-642-36883-7_18).
- [130] Eliezer Yudkowsky. "Artificial intelligence as a positive and negative factor in global risk." In: *Global catastrophic risks* 1.303 (2008), p. 184.