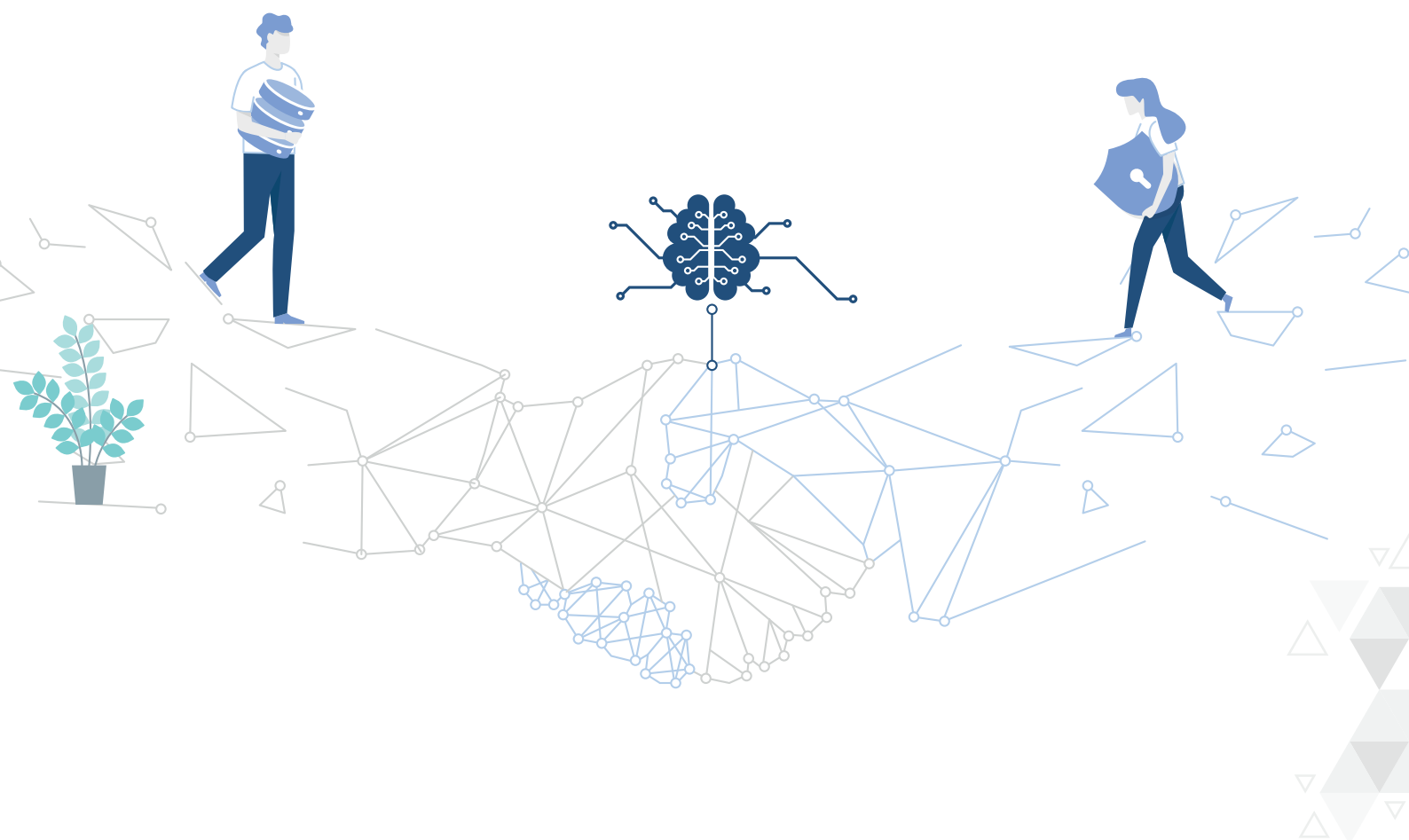




XayNet: Masked Cross-Device
Federated Learning Framework

Whitepaper

Lea Dänschel
Michael Huth
Leif-Nissen Lundbæk



Content

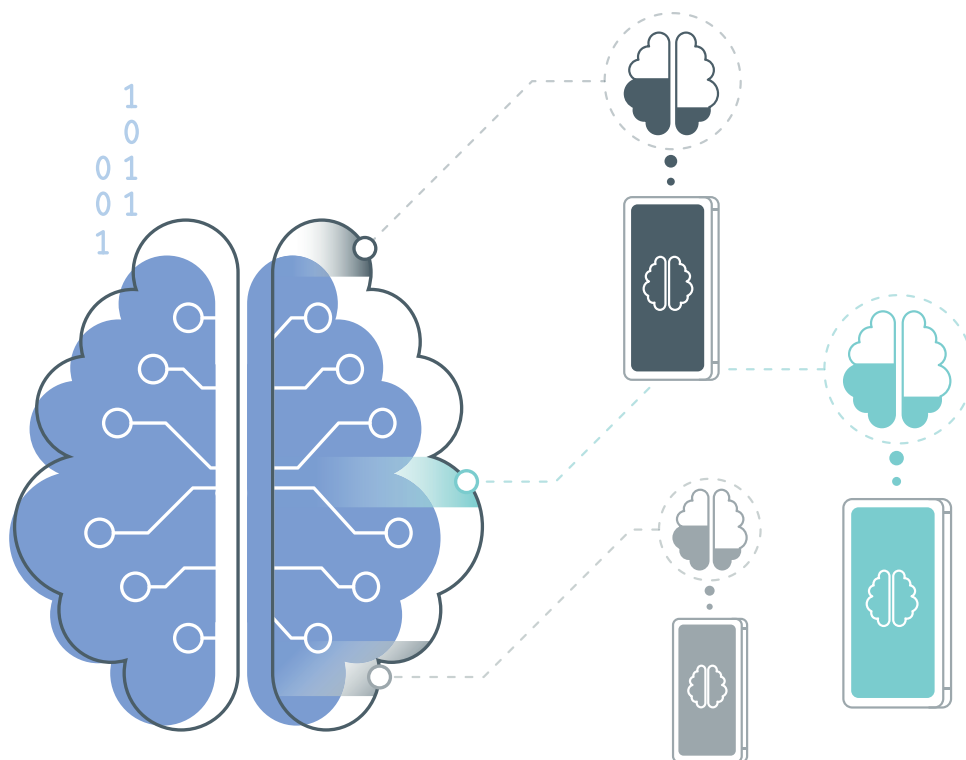
3	1. INTRODUCTION
5	2. BACKGROUND
	2.1 Centralized, distributed, and decentralized machine learning
	2.2 Adoption hurdles for machine learning
7	3. PRIVACY-ENHANCING TECHNOLOGY (PET)
	3.1 Data anonymization
	3.2 Synthetic data generation
	3.3 Homomorphic encryption
	3.4 Differential Privacy
	3.5 Multi-party computation
11	4. FEDERATED LEARNING
	4.1 Federated averaging
	4.2 Types of learning
	4.3 Benefits of federated learning
15	5. MASKED FEDERATED LEARNING
	5.1 PET protocol specification
	5.2 Example run of our PET protocol
	5.3 Privacy of our PET protocol
	5.4 Security of our PET protocol
	5.5 Attacks on federated learning
	5.6 Privacy in depth
	5.7 Post-quantum cryptography
27	6. OPEN-SOURCE AND FULLY MANAGED SERVICE APPROACHES
28	7. REPRESENTATIVE USE CASES
	7.1 Cross-silo use cases
	7.2 Cross-device use cases
30	8. CONCLUSIONS
31	References

Abstract

This Whitepaper describes XayNet, a platform for federated learning that offers privacy by design through an innovative collaboration between experts in AI, cryptography, platform architecture, and law. Federated learning holds great promise in removing adoption barriers of AI in production since it avoids the transfer of personal or sensitive raw data and allows training of such data in the environment in which it is generated and controlled. This overcomes technical, regulatory or other barriers for training AI models on a collection of such data sources, by aggregating the locally trained models into a model that contains all insights that are captured in such local models. To ensure the wider applicability of this approach, federated learning needs to be delivered in a way that preserves the privacy of training data and trained models, and protects data that is deemed to be commercially sensitive. XayNet is a federated-learning solu-

tion that offers such privacy and protection by design.

To make this Whitepaper accessible to a wider readership, we first review the basics of machine learning and core concepts of privacy-preserving technology, and then give an introduction into federated learning. Our privacy-by-design solution for federated learning is then presented in more detail. This solution combines anonymization for federated learning of global models with pseudonymization of local models prior to their aggregation. Then we describe the current state and development plans of our open-source federated-learning project. Representative use cases, classified in two types, as well as experimental results illustrate the benefits of federated learning and of the use of our platform. These use cases also indicate the strategic focus on XayNet's future development.



1. Introduction

Artificial intelligence (AI) refers to the ability of machines to display intelligent behavior. Since the 1950s, AI is also a major research area across several disciplines, including Computer Science, Mathematics, and Statistics.

In the past 60 years, AI has seen waves of optimism and advances, interleaved with periods of disillusionment – the so-called “AI Winters”. In recent years, another wave of optimism began and is driven in no small part by the availability of cheaper and more powerful hardware and the open access of research findings, for example through open-source software packages for free experimentation and off-the-shelf hardware designed for solving AI problems.

These successes, particularly in the area of deep learning, have been widely publicized – for example, the Alpha Go AI that beat a world-class Go player. These advances have also moved AI up the strategic agenda of both politicians and decision-makers in the industry. AI is seen as a key technology in making territories and companies alike competitive in the future digital marketplaces.

Machine Learning It turns out that almost all of these recent advances stem from an area of AI called *machine learning*. This is a field that applies methods from statistics to solve two related problems:

- ▶ **Model Training:** training data is prepared and subjected to an algorithm that learns a mathematical model. The model represents patterns identified in the training data.
- ▶ **Model Inference:** the learned model is subjected to new data input in order to make inferences based on such learned patterns for decision support.

In mathematical terms, machine learning wants to learn an approximation of a given, but unknown, probability distribution and the training data is assumed to have been sampled according to that distribution. The inference then uses the learned, approximate distribution.

For example, a deep learning algorithm may be used on medical images to learn a model that can then predict from 3-D brain scans whether a patient has a particular brain disease. It is important to keep in mind that such predictions are merely statistical assertions, and not hard and qualitative facts. This means that machine learning needs to support decision making in a manner that is human-centric and meets expectations of ethics and legal requirements. And this is especially true when decisions are being made in an automated fashion, for example in an autonomously driving car.

FROM RESEARCH TO IMPACT

Machine learning has great potential to make decision-support processes smarter, cheaper, more automated, and self-improving. This makes the adoption of machine learning an important topic in the strategic planning of government agencies, industrial companies, and third-sector organizations.

It is also recognized that widespread commercial adoption of machine learning will require approaches that offer sufficient guarantees about the security and privacy of the process of learning models, especially when companies wish to learn collaboratively from datasets that they locally maintain and control. For example, a report [Res19] asked German enterprises to give their biggest reason for not considering the cloud for machine learning applications. Over a third of these companies cited data-protection issues as their biggest concern. In that same study, 20 percent of surveyed CEOs and Directors stated reservations about deploying AI solutions for fear that they could cause compliance and data-protection issues.

Federated learning has been proposed as a viable solution for this. In simple terms, federated learning allows parties to learn an updated model on their local datasets, communicate such an updated model to a third party who then aggregates these locally updated models into a global model. That global model is then the basis for subsequent local learning. In particular, all training data stays on-premise, only updated models get commu-

nicated in the network, and there is no need to perform costly data anonymization on local datasets. Therefore, this approach can accommodate data-protection directives such as the EU GDPR with greater ease, scalability, and cost-effectiveness.

Federated learning can also deal with the issue of an ever-increasing amount of data being generated, including IoT data stemming from edge devices. Due to the astronomical volume of such global data streams, it is not realistic – and certainly not sustainable – to persistently store all such information for further processing. But federated learning offers a vision of “compressing” such data streams into AI models that support decision making, where the training of AI models happens on or near the edge devices themselves.

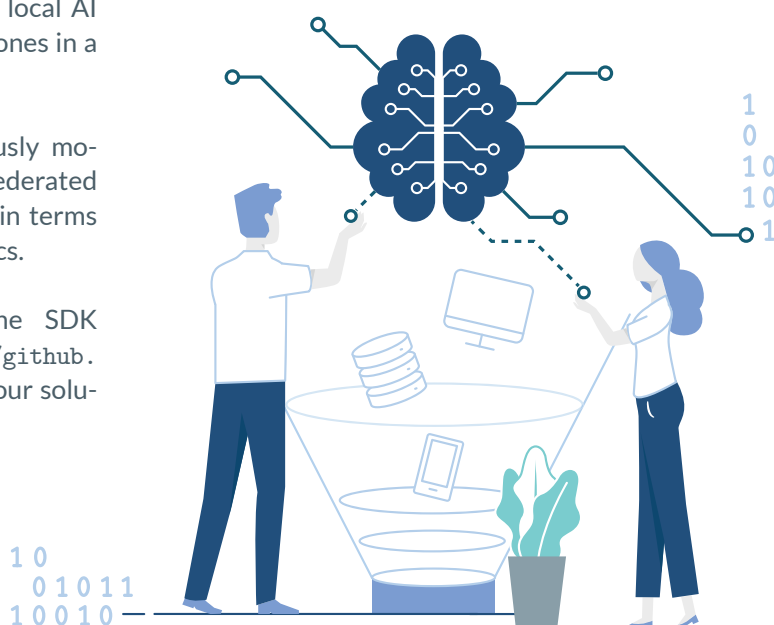
XayNet This context motivated the design and implementation of XayNet, our end-to-end platform for deploying and executing federated learning projects. XayNet is now entering its beta stage and offers the following highlights:

- ▶ A lean software development kit (SDK) that allows customers to make their machine-learning applications “federated” by the insertion of a few lines of code.
- ▶ A platform service for executing federated learning projects, aggregating local AI models into more accurate global ones in a privacy-compliant manner.
- ▶ Dashboard support for continuously monitoring the performance of a federated learning project on that platform in terms of model accuracy and other metrics.
- ▶ An open-source project for the SDK and the platform service (<https://github.com/xaynetwork/xaynet>), making our solution transparent.

- ▶ Open-source benchmarks and detailed documentation to ease the integration of XayNet into machine-learning applications and to assist the familiar machine-learning workflow.
- ▶ Ability to protect the IP of global models from the platform while still aggregating local AI models in a privacy-preserving manner.
- ▶ Designed so that personal raw data will not leave its existing data controller environment.

OUTLINE OF PAPER

In Section 2, we review machine-learning architectures and identify adoption hurdles for machine learning. A review of current privacy-enhancing technology is provided in Section 3. Federated learning is introduced and discussed in Section 4. Our solution for privacy-preserving federated learning, including a discussion of its data-privacy compliance, is developed in Section 5. The current state and road-map of XayNet are the subject of Section 6, uses cases are presented in Section 7, and the paper concludes in Section 8.



2. Background

We first provide some background to make this paper accessible to a wider readership.

2.1 CENTRALIZED, DISTRIBUTED, AND DECENTRALIZED MACHINE LEARNING

Machine learning, as explained above, is executing an algorithm over a set of training data on a dedicated computing machine. We may refer to this as *centralized* machine learning. For medium to large scale machine learning problems, it is pretty common to apply *distributed* machine learning instead.

A popular approach for that is to partition the set of training data \mathcal{D} into different buckets \mathcal{D}_i , each hosted on a respective machine M_i , to run the same learning algorithm on each machine and bucket, and to then have a means of combining the learned local models into a global one. For that, let us assume that there is no adversary that may manipulate local data, local learning or message content and delivery. Then the same model can be learned as if centralized machine learning were to be used. Therefore, such distribution does not “break things”, but why would we choose it?

The advantage of distributed machine learning is that it can complete a large learning task at greater speed and scale by distributing the learning task to multiple machines. In order to accomplish this, trade-offs between speed gains and model accuracy have to be considered (see for example [GZW17]). Moreover, the tuning of hyperparameters such as the epoch size [GZW17] may be required, communication protocols may need to be optimized to reduce communication complexity and payload (see for example [KJA18]), and the machine-learning algorithms should be robust against communication failure and downtime of machines (see for example [YTR⁺19]).

Distributed machine learning is also useful when the training set \mathcal{D}_i is already physically split up, for example when each \mathcal{D}_i represents data from an autonomous vehicle i across an entire fleet of vehicles. In such use cases, economic and other considerations may prevent a combination of all such data into a sole data-

set on which centralized machine learning could then be performed.

The combination of results from local learning described above is often done by a third party called a “Coordinator”. Alternatively, the combination of local models may use communication protocols that do not rely on a third party and we then speak of *decentralized* machine learning as a special form of distributed machine learning [yA18].

The area of Distributed Systems has a lot to offer in answering open research questions in distributed machine learning. Let us illustrate this with two such questions: To what degree can model combination happen asynchronously in order to improve parallel processing, without compromising the mathematical behavior of the combined learning? How can we make mechanisms for model combination resilient to adversarial manipulation? Answering such questions needs to draw from non-trivial insights of Distributed Systems and we will revisit the latter question further below.

2.2 ADOPTION HURDLES FOR MACHINE LEARNING

Although machine learning and AI are high on the strategic agenda, there are significant hurdles faced in the adoption of machine-learning technology in enterprises, and similarly in non-commercial institutions. Let us list and then discuss important hurdles for such adoption:

- AH1** Machine learning technology is the output of research and its transfer to commercial usage is non-trivial and requires high levels of expertise [ea18] [Res19].
- AH2** Companies such as SMEs or use cases such as usage data from smart-phones deal with “small data” on which machine learning may be less or not at all effective [ea18] [Res19].
- AH3** Training data of a company often resides in *data silos* separated across physical boundaries (for example dif-

ferent IT systems) or political/legal boundaries (for example across departments or subsidiaries) which creates obstacles to combining such data [oCCA19].

AH4 Training of a model involves a complex workflow, including the pre-processing of training data and tuning of so-called “hyperparameters” such as the topological structure of a deep learning network. These are typically manual and high-skill tasks [oCCA19].

Hurdle AH1 is a reflection of the fact that machine learning is research-driven. Fortunately, knowledge and technology transfer from research-based machine learning tools to their usability and integration in commercial decision-support workflows is improving.

The hurdle AH2 suggests that there are incentives in designing and using *collaborative* forms of machine learning. Such collaboration may take on many forms but will certainly have to accommodate constraints that stem from hurdle AH3. In fact, Google developed *federated learning* as a form of collaborative, distributed machine learning for predicting which word a user wants to type on her keyboard as a function of already typed in keys [MMRyA16]. Trust perceptions and perhaps also regulatory constraints around privacy prevented the combination of all user keyboard data into a global set of training data. Rather, the training was done on local training data for each user/device and only model information was then combined.¹

Therefore, the development and deployment of technology that mitigates against the factors of hurdle AH3 will make a pivotal contribution to the wider and successful adoption of machine learning in practice. Moreover, such technology will positively impact the issues in hurdle AH2 as such mitigating effects will also apply to datasets of smaller size.

Hurdle AH4 also is quite significant. The workflows of machine learning are fairly complex,

and small mistakes at steps of such workflows can then corrupt the entire learning process and make the learned models unusable. These issues are amplified by a substantial shortage in available AI engineers/developers and AI knowledge in the global workforce, which severely limits the successful pursuit of AI projects in companies.

Tools and technologies that can realize the execution of machine-learning workflows more easily, without compromising the quality of the obtained outputs, will thus have strategic value.

We think that *Automated Machine Learning* (AutoML) has great potential to that end, which is reflected in our product planning.



3. Privacy-enhancing technology (PET)

Given the central role of data privacy in the removal of adoption barriers for AI in applications, we review technology that can enhance privacy in machine learning. In doing so, we also point out potential shortcomings of such approaches or how their integration into federated learning may have synergistic effects. We begin with a discussion of traditional data anonymization

3.1 DATA ANONYMIZATION

This is the de facto standard for enhancing privacy in machine learning and data sharing in current practice. Datasets can be stripped of personal information, which requires more than the removal of names and other directly identifying information. It also typically requires the removal or abstraction of structural information that could be used to make inferences about personal information.

For example, the US Health Insurance Portability and Accountability Act (HIPAA) effectively makes use of a k -anonymization technique, saying that a dataset is compliant if 17 specific identifiers have been removed. This is problematic as there can be datasets that meet this requirement but definitely allow inferences about personal data.

There are attempts to generalize k -anonymity for its use in federated learning [CGDS⁺20] so that datasets are abstracted in terms of relational and transactional attributes and where this abstraction trades off the loss of information between these two types of data. While this seems to offer benefits in meeting the regulatory demands of HIPAA, it is less clear whether it can meet those of GDPR, given the concerns raised in [Par14] about such anonymization techniques. Moreover, in the approach of [CGDS⁺20] the aggregation service learns the syntactic mappings (i.e. abstraction functions) of each participant, which an attacker may exploit to make inferences about personal data. This approach may also have practical limitations since input for model inference needs to be mapped to an abstraction that approximates the input in some optimal manner.

There are few benefits of relying on such data anonymization techniques, apart from regulation such as the GDPR mentioning anonymization as a means for getting outside the scope of the GDPR. The risks of relying on such data anonymizations are well documented in a well-cited legal opinion by the Article 29 Working Party [Par14]. There is compelling evidence that these risks are increasing over time, as exemplified by the recent advances in re-identification techniques (see e.g. [RHdM19]).

Plans by government agencies and public/private partnerships to offer data through such anonymization on clouds, such as potentially through the Gaia-X project in Germany, the EU's vision for a cloud to drive innovation, and the Polish Cloud project, should be assessed with caution. At least they seem to require means for managing risks that stem from the use of such traditional anonymization techniques.

Moreover, the creation of anonymous datasets is not fully automatable and so is costly and does not scale well. It also seems to be not well suited to the incremental capture and processing of data, which is becoming increasingly important in edge computing and IoT.

3.2 SYNTHETIC DATA GENERATION

An emerging privacy-enhancing technique for data sharing and AI is that of synthetic data generation. This comprises methods that exploit the statistical nature of data to generate synthetic data from it that has very similar statistical properties.

The machine learning tools used for this are Generative Adversarial Networks [GPM⁺14] (GAN). A GAN defines an interactive process for generating synthetic data from raw data. It does this through an adversarial "game" between a discriminator and a generator. The discriminator wants to distinguish raw data from its synthetic version. The generator learns how to produce better and better synthetic data – based on feedback from the discriminator – until the discriminator can

no longer detect any differences between raw and synthetic data. The learning engines used in GANs can be convolutional neural networks, recurrent neural networks or other machine learning techniques.

GANs can generate quite powerful synthetic data, such as completely realistic portrait images of people that have and will never exist – so-called “deep fakes”. Given the statistical similarity of the generated synthetic data and the raw data from which it was generated, it seems that GANs offer a viable alternative to the aforementioned anonymization techniques. Another really nice aspect of GANs is that the synthetic data they generate are not subject to re-identification attacks based on ancillary data. This is so since one cannot map the ancillary data to the synthesized distribution.

An important question, though, is how this approach is interpreted legally, where [BDR19] gives a good overview. Generally speaking, there does not seem to be a data-privacy solution that retains the full utility of machine learning and completely eliminates all risks of data privacy violations. So how close do synthetic datasets come to resolving this dilemma? We believe that this approach has great potential, but that – as a more universally applicable technique – it is somewhat limited:

- S1** Synthetic data may allow inference about natural persons, e.g. speech data – especially for small and personalized raw datasets,
- S2** this approach is suited for static data and seems more complex to operationalize for dynamic learning,
- S3** it appears to be not very effective for rare event learning, e.g. anomaly detection,
- S4** there is legal uncertainty around its usage, with conflicting interpretations of its compliance across different regulations and territories.

The issue S1 can be mitigated against by using Differential Privacy techniques within the GAN used for generating the synthetic data (see e.g. [XLW⁺18]). But this can and typically will lead to a loss of utility due to the noise inserted by Differential Privacy in this process.

More research seems to be needed to address the issues S2 and S3 above. As for issue S4, the problem is in part that different regulations can have inconsistent approaches in judging the compliance of a technique.

Let us recall that the US Health Insurance Portability and Accountability Act (HIPAA) uses a k -anonymization technique, saying that a dataset is compliant if 17 specific identifiers have been removed. This is also problematic for synthetic data as there can be synthetic datasets that meet this requirement but do allow inferences about personal data.

On the other hand, a regulation that bans the use of synthetic data even if the risk of re-identification is very small (smaller than the chance of winning a national lottery, for example) appears to be overly restrictive. There is, therefore, scope for the legal space to catch up with advances in anonymization techniques, and lawmakers may want to encourage the adoption of those techniques whose effectiveness can be shown with mathematical rigor.

To summarize, we see that in a narrower use scope synthetic data will be a very valuable PET tool with the potential for a genuine market share in PET AI in the future. We emphasize that federated learning can be used to harness the full utility of synthetic data while using an aggregation that protects against re-identification attacks.

3.3 HOMOMORPHIC ENCRYPTION

Fully homomorphic encryption (FHE) represents decades of progress in cryptography, with first work originating in a paper by Rivest et al. in 1978 [RAD78].

The intuitive idea is that a third party can compute over data without actually getting

to know that data. This conundrum is solved with key-based encryption where the encryption process preserves algebraic operations. For the addition operator, for example, we would have $e(k, a) + e(k, b) = e(k, a+b)$ for an encryption scheme $e(\cdot, \cdot)$, encryption key k , and plaintexts a and b .² A third party could thus compute the ciphertext of the value of the addition $a+b$ from the ciphertexts of a and b , and return this to the owner who could decrypt this to get the computation result on the plaintext.

However, getting encryption schemes that are homomorphic for both addition and multiplication without any restrictions on the algebraic circuits it can compute – so called fully homomorphic encryption – has proved to be very difficult. Essentially, fully homomorphic schemes realize such full homomorphisms for simple polynomials and then lift such capabilities for more complex polynomials by allowing for errors in the computations and by “re-encrypting” ciphertext to reduce its error before feeding that ciphertext into further homomorphic computation. In most schemes, this error increases with each multiplication gate in the algebraic circuit. In many of these schemes, one of the operations (say multiplication) doubles the error whereas the other operation does not increase the error by much.

This growth of errors and the need to contain these errors introduces complexity that also impedes the performance of such schemes. On the brighter side, more efficiency can be had when the algebraic circuits that are being computed have specific properties, e.g. bounded depth or bounded depth of multiplications in the circuit. This is of interest for federated learning, as we have quite specific algebraic expressions here that do not change or expand their shape much. For example, federated averaging is a tail-recursive weighted sum whose algebraic circuit does not have nested multiplications, thus containing the complexity of error management.

Since FHE is a key-based encryption, its use does not provide perfect (also known as

information-theoretic) security. This can be legally problematic for its use in federated learning as XayNet would be able, in principle, to make some inferences from such ciphertexts. For example, since FHE schemes are malleable, XayNet could produce related ciphertext that could facilitate a cryptanalysis that would want to learn parts or all of the plaintext – which in our concrete case would be a local model.

This is potentially problematic. Furthermore, it means that all inputs for an algebraic computation need to be encrypted with the same key. If the local models are for example held on end-devices of different participants, this seems to require a key-agreement protocol by which many participants agree on such a key. And this key would then be stored on each participant’s device, increasing the attack surface for stealing such a key.

It is also not clear whether the state of the art of fully homomorphic encryption could encrypt and decrypt large vectors that represent models of about 500MB or even 1GB.

3.4 DIFFERENTIAL PRIVACY

Differential privacy (DP) [Dwo06, DR14] was invented by Dwork, McSherry, Nissim, and Smith for transforming the outputs of queries to databases so that the answers to such queries do not compromise the privacy inherent in the data that this database contains. It realizes a trade-off between the accuracy of the answer and the level of privacy that the answer can provide. In particular, this does not give us perfectly correct answers to such queries. But it can guarantee that the answers preserve privacy, with a small probability of violating such privacy. Since this probability can be engineered to be very small, this is likely to help with compliance for regulation such as the GDPR.

One shortcoming of DP is that its guarantees, when made for repeated answers to queries, are made within an initial privacy budget. Techniques, such as the Momentum Accountant [ACG⁺16] can be used to keep a balance of this budget. But this makes DP somewhat

complex to operationalize in environments that have dynamic uncertainty. For one, this needs a monitor that checks whether the budget has not yet been consumed and would block answers to queries when the budget has been used up. For another, uncertainty in the dynamic operation (e.g. when federated learning terminates once sufficient accuracy of a model has been achieved) can make it difficult to allocate a budget that is sufficient for the computational task at hand.

While we think that it is useful to keep up with progress in R&D for differential privacy, DP does not seem to be a good candidate for XayNet's chief PET mechanism in the aggregation of local models in the short to medium term.

3.5 MULTI-PARTY COMPUTATION

Secure multi-party computation (MPC) [ZZZ⁺19] is an approach by which a set of parties can protect the privacy of their inputs in computing the output of some function that is known to all parties. As an example, if three parties P_x , P_y , and P_z have private input x , y , and z – respectively – and the function f outputs $o = f(x, y, z)$, then party P_x cannot learn anything about the values of y or z that it would not be able to learn from knowing f , o , and x . The ability to learn from public information is known as acceptable leakage. One potential concern is whether the leakage is indeed acceptable or whether parties can

manipulate their inputs to maximize what they can learn from public information provided by this protocol (see e.g. [AH19]).

Protocols that realize this PET functionality can provide either computational or unconditional guarantees of such privacy protection:

- ▶ Computational guarantees say that an adversary with specific bounded computing resources cannot compromise the privacy of the protocol.
- ▶ Unconditional guarantees say that an adversary with “infinite” computing resources (a mathematical idealization) cannot compromise the privacy.

There are also different attack models, mostly pertaining to whether participants would obey the rules of the protocol or not. MPC protocols that are resilient to active attacks require more computational effort, for example through the use of commitments and challenges embedded in protocol steps.

The aggregation of local models in federated learning can be seen as an instance of such a multi-party computation where participants submit local models and want to learn the value of the global model that these determine through aggregation, while preserving the privacy of their own local model. Our PET solution for federated learning can be interpreted as a form of secure multi-party computation.



4. Federated Machine Learning

Federated learning, as a form of collaborative and distributed machine learning, seems imminently suitable in overcoming regulatory constraints around the use of machine learning. Federated learning was proposed for learning from datasets generated by and stored on small devices – keyboards [MMRyA16], where different datasets will not show identical patterns, may vary considerably in size, and where there may be millions of such datasets. Such use cases are called *cross-device* in [KMA⁺19], as opposed to *cross-silo* use cases in which a much smaller number of datasets are hosted in more powerful environments such as servers.

For an in-depth discussion of federated learning and its various aspects and trade-offs in R&D we refer the reader to [KMA⁺19]. Next, we offer a gentle introduction into this topic.

4.1 FEDERATED AVERAGING

We will now present the original federated-learning algorithm first informally and then at a more technical level.

The FederatedAveraging algorithm of [MMR⁺17] is depicted in Figure 1 and involves a set of K participants, identified as 1, 2, and so forth. These participants want to train the same model structure on local data and aggregate these models across participants to boost model accuracy, which will then support better decision making at model inference time.

The intuition of the algorithm is that participants will learn local models based on their local datasets and that a Coordinator will aggregate these local models into a global one that will then be used again for local learning.

At each round, the Coordinator selects a set of participants of fixed size, asks each selected participant k to update their local model based on the current global model and the local dataset D_k of that participant and to send this updated model to the Coordinator. Once the Coordinator has received all these updated models, the weighted average of all of these is computed as the updated global model which is then sent to all selected participants of the next round.

```
// executed by a central, trusted „Coordinator“
initialize  $w_0$ ;
for (round  $t = 1, 2, \dots$ ) {
     $S_t = \text{random set of } \lceil C \cdot K \rceil \text{ participants};$ 
    for (each participant  $k \in S_t$  in parallel) {
         $w_{t+1}^k = \text{ParticipantUpdate}(k, w_t);$ 
    }
     $w_{t+1} = \sum_{k=1}^K \frac{n_k}{n} w_{t+1}^k;$ 
}

// executed by each participant locally
ParticipantUpdate( $k, w$ ) {
     $\mathcal{B} = \text{partition of } D_k \text{ into batches of size } B;$ 
    for ( $i = 1, \dots, E$ ) { // local training epochs
        for ( $b \in \mathcal{B}$ ) {
             $w = w - \eta \cdot \nabla l(w; b);$  // local Stochastic Gradient Descent steps
        }
    }
    return  $w$ ;
}
```

Figure 1: Pseudo-code for algorithm FederatedAveraging of [MMR⁺17]: C is the percentage of clients that randomly participate in model updates in each round, $\{1, \dots, K\}$ is the set of all clients, w_t is the global model learned after round t , whereas w_t^k is the local model learned after round t (both w_t and w_t^k are d -dimensional real-valued vectors), variable n_k denotes the size of local data set D_k where $n = \sum_{k=1}^K n_k$, parameter η is the learning rate, and $l(\cdot, \cdot)$ is the loss function used for local learning.

Function *ParticipantUpdate* updates local models; its details are not important here, except for noting that all parties apply the same learning algorithm (here a Gradient Descent method) and on the same model structure (abstracted here into a vector of size d).

In *FederatedAveraging*, participants are selected *randomly*, making this *Stochastic Gradient Descent*. Other selection strategies are possible. The weights are determined by the relative size of a participant's dataset, but other choices are possible. Further, we note that federated learning is fairly agnostic in the choice of learning algorithm used by participants, i.e. how function *ParticipantUpdate* is being implemented, as long as the models have meaningful representations as d -dimensional vectors and the resulting instance of federated learning offers good performance or theoretical convergence guarantees.

These degrees of freedom in choice illustrate that federated learning can be seen as a generic algorithm in which mechanisms such as participant selection, weight determination, aggregation function (here a weighted sum), and locally used learning algorithm are additional parameters whose instantiation results in a concrete case of federated learning. One may thus think of federated learning as a software framework. In particular, one may, therefore, optimize such choices for particular use cases and their requirements. And such optimizations may be achieved through standard methods, including those of machine learning itself.

Another aspect to consider is that federated learning is also agnostic to how participants integrate globally learned models into their own decision-support systems. For example, they may set their own criteria for when the global model would be the source of truth for their own decision support and thus replace a local model. This flexibility is nice but it has a flip side: other participants and the Coordinator do not have any guarantees that the updated model w_{t+1}^k that participant k returns to the Coordinator has really been computed by the intended implementation of function

ParticipantUpdate. We will revisit this issue below.

4.2 TYPES OF LEARNING

Federated learning can support different types of learning in order to meet the varying needs of use cases. In [YLCT19], three types were identified which we discuss here briefly:

- ▶ **Horizontally federated learning:** Datasets from participants have (almost) the same features and labels but identities (the rows in a data matrix) do not overlap significantly.
- ▶ **Vertically federated learning:** Datasets from participants have the same or almost the same identities but where features and labels do not overlap significantly.
- ▶ **Federated Transfer Learning (“Transfer federated learning”):** Datasets from participants have no significant overlap between their identities, features, and labels.

Horizontally federated learning is a natural starting point for technical development and commercial use cases for the following reasons:

- ▶ The matching of links between features and labels does not seem to raise any privacy or other regulatory concerns, so this is easier to operationalize.
- ▶ With no links between identities, we do not have to worry about how to implement such links without compromising regulatory demands such as those pertaining to privacy.
- ▶ Many use cases involve structurally similar or identical datasets for participants, where the semantic link structure is reflected very well by horizontally federated learning.

Federated transfer learning seems to have similar benefits; it does not appear to require the establishment of semantic links between identities, and so this seems to not raise any regulatory issues – where such issues may or may not be addressable with technological means.

Strategic focus of XayNet

Therefore, our federated learning platform development focuses on *horizontally federated learning* first and considers *federated transfer learning* use cases next. While our platform will also support *cross-silo* use cases, its strategic emphasis and most of its innovations will be concerned with *cross-device* use cases.

4.3 BENEFITS OF FEDERATED LEARNING

Let us now demonstrate the benefits of federated learning when compared to the *centralized* setting in which machine learning takes place on each dataset separately, without any sharing of locally updated models or computation of a global model.

This comparison uses a standard machine-learning benchmark and its experimental results are depicted in Figure 2.

Experimental setup: Our results in Figure 2 are based on the previously described partitioning schemes applied to the benchmark Fashion-MNIST. We train a convolutional neural network (CNN) consisting of

- ▶ a convolutional layer with 64 filters and kernel size 2,

- ▶ a second convolutional layer with 32 filters and kernel size 2,
- ▶ each followed by ReLU activations and max pooling,
- ▶ then a fully-connected layer with 256 units and ReLU activation, and finally
- ▶ the output layer which uses a soft max activation.

This model has a total number of 412,778 trainable parameters (more than four-hundred thousand). The three hidden layers use dropout probabilities of 0.3, 0.3, and 0.5 respectively. All layers use Glorot uniform initialization.

In this experiment, the machine learning problem is the classification of inputs. Intuitively, a model captures a set of classes and assigns each input to one of these classes. The experiment takes an entire dataset and then applies different partitioning schemes to that dataset, where each such scheme gives us a set of partitions. Each partition would then amount to a local dataset in the terminology of federated learning.

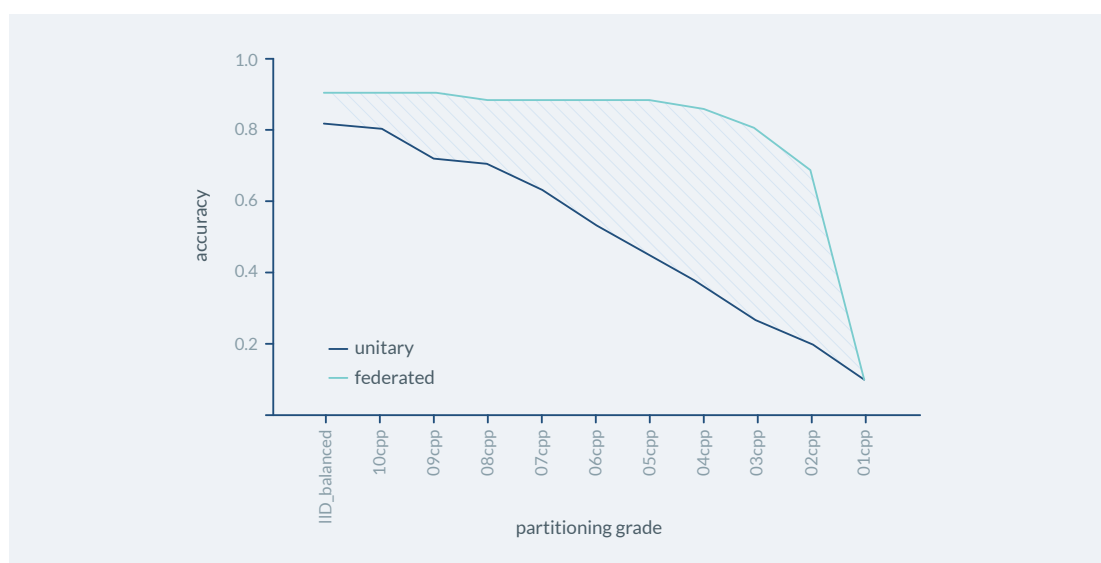


Figure 2: Benefits of federated learning over the centralized approach of machine learning: model accuracy (y-axis) plotted over the grade of a partitioning scheme (x-axis). Partitions are local datasets for federated learning, which makes much more accurate predictions when data is evenly or very unevenly balanced across partitions. The "unitary" approach trains one model for each partition and reports the most accurate of these models. The unitary and federated approaches are ineffective in the pathological case, where each partition contains only one class used for the classification of inputs.

Discussion of results

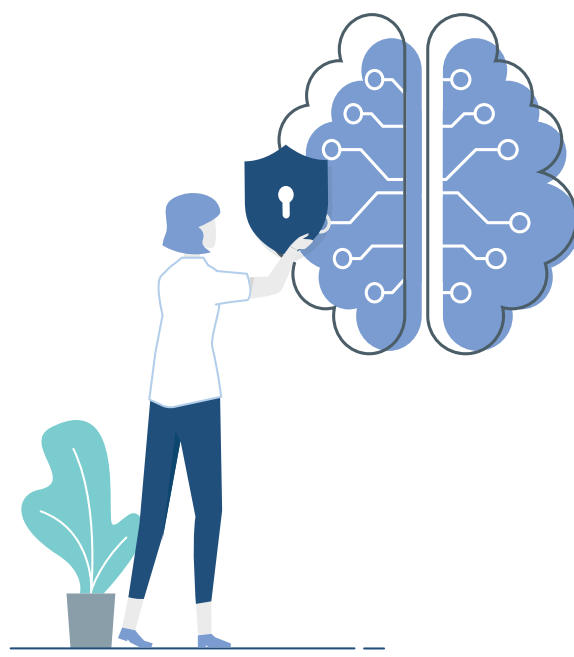
As we can see from the horizontal axis of the graph in Figure 2, the schemes we used range from those that have all classes present in each partition (left, called `IID_balanced`) to those that have only one class present in each partition (right, called `01ccp`). The vertical axis shows the *accuracy* of the models computed by centralized machine learning (blue line) and federated learning (green line), respectively. The accuracy is a statistical measure that computes the proportion of true predictions – both true positives and true negatives – from all the inputs that have been classified by the model.

We can see that the benefits of federated learning are already visible when the data is independent and identically distributed across partitions (`IID_balanced`). Federated learning can do better here since it simply has access to more data, where this access is indirect through the repeated aggregation of locally updated models.

At the other extreme of data partitioning, we see that federated learning and the centralized approach fare equally poorly when the partitions are such that they cannot really distinguish between classes of input. Such artificial partitions are a pathological case for any attempt of learning classifications.

Crucially, we can see that federated learning clearly outperforms the centralized approach when the data is distributed in a very uneven fashion across partitions: for federated learning, the accuracy is consistently higher and deteriorates much more gracefully when subjected to more and more pathological partitioning schemes.

Let us stress that federated learning use cases typically have little or no control over the partitioning scheme, as local datasets cannot be merged or re-partitioned with other datasets prior to learning. We varied these schemes here merely for the sake of an experimental comparison. A real federated learning use case will have a given partitioning whose statistical nature will very likely be that data are neither pathologically distributed nor identical and independently distributed. Therefore, practical use cases are better represented by the partition grades in Figure 2 for which federated learning offers huge benefits.



5. Masked federated learning

We now describe our solution for making our federated learning platform XayNet privacy-preserving. This solution is designed to make it compliant with data privacy regulations that share common principles, such as the EU GDPR or the California Consumer Protection Act (CCPA). Our working definition of “personal data” is consistent with both the GDPR and CCPA: “Data is personal data if it allows inferences about identified or identifiable natural persons to be made with reasonably likely means used.” Based on this definition, we state assumptions we make about use cases running on XayNet:

- AS1** The owner of a local dataset will have a legal basis, if required, to train local AI models.
- AS2** XayNet will treat local AI models as personal data, since training data will typically contain personal data. So XayNet will neither access training data nor local AI models.
- AS3** Use cases can be divided into a few risk types, and each type can be configured for federated learning such that computed global AI models will not be personal data.
- AS4** The owner of a use case may require that XayNet does not learn the values of computed global AI models, for example for the protection of intellectual property.

Private computations in abelian group Given these assumptions, we devise a privacy-preserving solution for federated learning used in XayNet. We recall that local and global models are abstracted as d -dimensional vectors of real numbers; this abstraction already hides information about model structure from XayNet and so helps with achieving overall privacy.

These numbers within vectors stem from bounded numerical types within programming languages in which use cases run. Therefore, we may encode such vectors

into Z_m^d , the abelian group of d -tuples of numbers from 0 up to $m-1$, where the group operation is addition modulo m in each coordinate. The value m will be large enough so that additions of encoded tuples will never require the performance of a “modulo m ” operation; at the same time, m will be small enough so that the range of the encoding is well distributed in that group. The former means that computed results can be correctly decoded into d -dimensional real numbers, concretely as computed global models. The latter reduces the range bias that attackers might attempt to exploit.

Intuition of our PET solution

The intuition of our solution is as follows. At each round i of federated learning, XayNet will publish some information for all participants to see. Based on this information and local state of participants’ devices, participants can determine whether they have been selected for at most one of the following tasks for round i :

- ▷ “update” participants: they train an updated local model, mask it securely, and send the masked local model to XayNet for aggregation into an updated global model,
- ▷ “sum” participants: they compute the sum of masks of those masked local models that XayNet wants to aggregate into a global model and send that sum to XayNet.

Afterwards XayNet aggregates the masked local models and subtracts from that aggregation the sum of masks to obtain the aggregated global model. XayNet sends the latter to a Model Distributor that, in turn, sends this new global model to relevant parties of the use case. A simplified version of this workflow is depicted in Figure 3.

Importantly, the selection of participants is random but transparent and only participants that are selected will know that they are selected and then share this information with XayNet, which can verify whether a participant fulfils the random selection criterion. This creates trust in the protocol and makes it

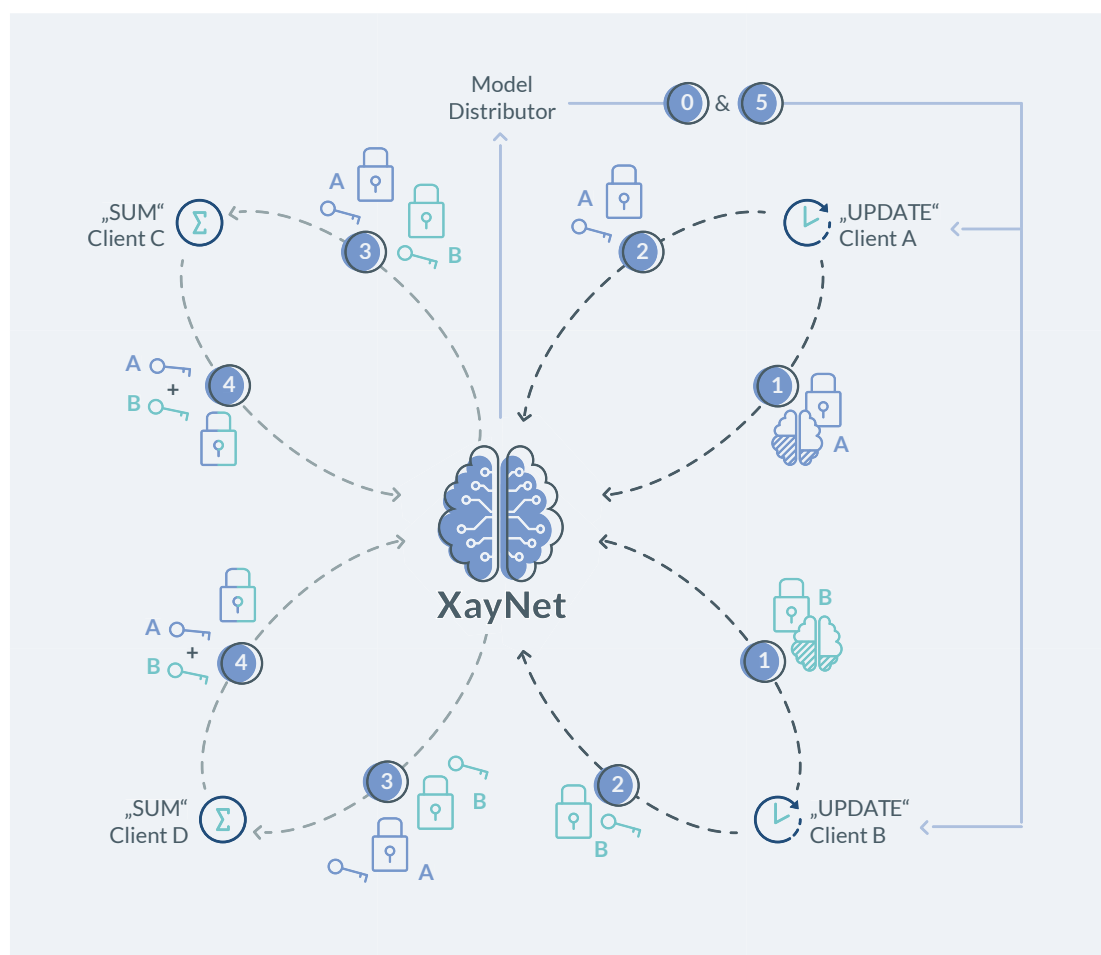


Figure 3: Our PET solution for federated learning: (1) “update” participants send their masked local models to XayNet and their seed of the corresponding mask – via XayNet in (2) and (3) – to “sum” participants. XayNet receives the sum of masks from “sum” participants (4), uses that to compute the aggregation of masked local models, and (0 & 5) sends this new global model to a Model Distributor which passes it on to relevant parties of the use case.

resilient against malicious participants. Uses cases also have a *policy* for task selection that must be met in addition to the random selection. For example, a phone may only be selected as “update” participant if it is charging, connected to WLAN or 5G, and has a global model installed that is deemed to be recent enough.

Rational numbers u_{i-1} and s_{i-1} in the interval $[0,1]$ are part of what XayNet publishes. Numbers u_i and s_i are the fraction of participants that are expected to be selected as “update” and “sum” participants in round i , respectively. The actual values of selected participants may be lower, depending on how constraining policies will be.

Once participants have determined that they are selected for one of these tasks, they contact XayNet with proof of this selection, a simple digital signature;³ participants that

are selected for neither of these tasks will sleep during this round. “Update” participants also upload their masked local models to XayNet. “Sum” participants will share with XayNet fresh public keys that XayNet will pass on to “update” participants. The latter will encrypt the (seeds of) masks that they used to mask local models and send these ciphertexts to XayNet which will pass them on to “sum” participants. The latter decrypt these ciphertexts and can thus compute the masks and then their corresponding sum.

The “sum” participants then send the sum of masks to XayNet which then first aggregates the corresponding masked models and then subtracts in the abelian group Z_m^d from this addition result the sum of masks, obtaining the updated global model. If the IP of the global model needs to be protected from XayNet, then the sum of masks will instead

be sent to an unmasking component hosted on the client side that performs this unmasking step.

The resulting global model can then be decoded from Z_m^d into the numerical datatype used by the AI application. A Model Distributor component then sees to it that this updated global model reaches specified locations, be it participants, decision support systems, and so forth.

Let us now describe the protocol in more technical detail.

5.1 PET PROTOCOL SPECIFICATION

This protocol controls round i and is repeated at each round. Public keys are elements g^x of a cyclic group G where x is the corresponding secret key. The protocol consists of the following steps, S1-S11, not necessarily performed in that order:

- S1** XayNet publishes a random bit-string Q_{i-1} , rationals u_{i-1} and s_{i-1} from $[0,1]$, and its public round key pk_i^x and perhaps other information.
- S2** Participants somehow learn these values and determine, without interaction, whether they are selected for a task.

Implementations need to ensure that no participant is selected for tasks “update” and “sum” in the same round. Each participant k has a public key pk_k that may or may not change across rounds. The round key pk_i^x only needs to be known by XayNet and participants of the use case; similarly, public keys pk_k only play a role within this use case. Then k may determine whether she is selected for task “sum”, e.g., by the following local evaluation:

- ▶ If $0.\text{SHA-3}(\text{SIGN}(sk_k, Q_{i-1} || pk_i^x || \text{“sum”})) < s_{i-1}$ is false, k determines that she is not selected for task “sum”.
- ▶ Otherwise, she evaluates a policy, shared by all participants but depending on the

local state as well. If the policy is also true, she determines to be selected for the “sum” task.

Expression $\text{SIGN}(sk_k, m)$ denotes the digital signature of message m with the secret key sk_k associated with public key pk_k . The expression $0.\text{SHA-3}(\text{SIGN}(sk_k, Q_{i-1} || pk_i^x || \text{“sum”}))$ is a rational number in $[0,1]$ where the hash bitstring $\text{SHA-3}(\dots)$ represents a natural number⁴ that is then written in decimal form as the fractional part of the number on the left-hand side of the inequality:

$$0.\text{SHA-3}(\text{SIGN}(sk_k, Q_{i-1} || pk_i^x || \text{“sum”})) < s_{i-1} \quad (1)$$

Given the properties of cryptographic hash functions, an expected s_{i-1} fraction of all participants will satisfy this inequality in (1). This technique is known as *Cryptographic Sortition* [CM16]. Participant k can send the digital signature $\text{SIGN}(sk_k, Q_{i-1} || pk_i^x || \text{“sum”})$ to XayNet who can then verify that participant k satisfies the random selection. XayNet will not verify that k meets the policy, as XayNet cannot and must not get access to the participant’s device. The first message from participants to XayNet is this:

- S3** “Sum” participants k send a first of two round messages to XayNet. From this message, authenticated through the public keys, XayNet will learn the public key pk_k of participant k . Using pk_k , Q_{i-1} , and s_{i-1} , XayNet can verify that “sum” participant k satisfies the random oracle, i.e. that the inequality in (1) holds. If that oracle is satisfied, XayNet will agree with k on the shared session key s_k^x which XayNet computes from its secret round key sk_i^x and pk_k .⁵ XayNet will also learn from this message a fresh public key pk_k^s that k generated for task “sum”.

Similarly, “update” participants send their first of two round messages to XayNet:

- S4** From the authenticated message that “update” participant k sends,

⁴Leading zeros of the hash matter in this, otherwise this would create bias.

⁵Since these keys are from a cyclic group, k can compute the same key from her secret key sk_k and pk_i^x : if $sk_k = x$ and $pk_i^x = g^x$, then s_k^x equals $(g^x)^x$, which k can compute as she knows g^x and x .

XayNet learns the public key pk_k of participant k , verifies that k satisfies the corresponding random oracle, i.e. the inequality in (2):

$$(2) 0.\text{SHA-3}(\text{SIGN}(sk_k, Q_{i-1} || pk_k || \text{"update"})) < u_{i-1}$$

and does not satisfy the inequality in (1). Then XayNet computes the shared session key s_k^x from sk_i^x and pk_k , and learns the masked local model of participant k and perhaps other relevant metadata.

The masked local model and other data are sent as ciphertext encrypted with AES (or another suitable symmetric encryption scheme) under key s_k^x .

S4 Next, XayNet builds a dictionary D_s whose keys are public keys pk_k from received and verified “sum” participants of protocol step S3. Its lookup satisfies $D_s.pk_k = pk_k^s$.

Then K_1 denotes the set of “update” participants whose message XayNet received and verified in protocol step S4. Set K_1 and dictionary D_s are built incrementally and frozen prior to the execution of the next protocol step S5:

S5 XayNet broadcasts D_s to all participants (ideally, to all in K_1).

S6 Each participant k' in K_1 who receives dictionary D_s creates a dictionary $ELG_{k'}$ satisfying:

- ▶ The keys of dictionaries $ELG_{k'}$ and D_s are the same.
- ▶ $ELG_{k'}.pk_k$ is the ElGamal encryption of random seed s_k , with public key pk_k^s .

S7 Each “update” participant k' that computes $ELG_{k'}$ sends $ELG_{k'}$ to XayNet.

The seed s_k was used by k' to generate mask r_k , used in protocol step S4 to convert the weighted local model x_k into the masked lo-

cal model $y_k = x_k + r_k$, where addition is in the abelian group Z_m^d . Possession of $ELG_{k'}$ enables “sum” participant pk_k to recover the seeds s_k , of “update” participants. ElGamal encryption is chosen as it is semantically secure (and other semantically secure schemes are possible), the significance of this is discussed in the next section.

S8 Next, XayNet does the following: XayNet collects received dictionaries $ELG_{k'}$ from k' in K_1 and freezes the set of such participants as the subset K_2 of K_1 .⁶ XayNet resorts entries (ElGamal encryptions of seeds) of all these dictionaries into dictionaries C_k for each “sum” participant k (i.e. where pk_k is a key in D_s). XayNet then sends or broadcasts the dictionaries C_k to those “sum” participants.

The dictionary C_k has as keys the public keys pk_k of participants in K_2 where $C_k.pk_k$ equals the ElGamal encryption of seed s_k under public key pk_k^s .

S10 “Sum” participant k receives C_k from XayNet and does the following:

- ▶ k decrypts all ElGamal ciphertexts in C_k with secret key s_k^s to get seed s_k , for all k' in K_2 .
- ▶ k applies the shared pseudorandom number generator on s_k , to compute mask r_k , that k' used to mask her local model for protocol step S4.
- ▶ k computes σ as the sum of all r_k^s in the abelian group Z_m^d where k' ranges over K_2 .
- ▶ k sends σ as its second round message to XayNet (or to some unmasking system outside of XayNet, if desired).

We note that K_2 and D_s need to be of sufficient size, otherwise XayNet will declare the round attempt a failure, will log the failure and will restart the round.

S11 XayNet (or unmasking system outside of XayNet, if desired) receives values σ for the sum of masks from “sum” participants and does the following:

- ▶ XayNet runs an algorithm that votes for one such sum of mask value or declares the round a failure (if there is disagreement about the value of σ , for example).
- ▶ If XayNet accepts a sum of mask value σ , that value is subtracted in Z_m^d from the aggregation of masked local models for all k' in K_2 to retrieve the global model.
- ▶ That global model is communicated to the Model Distributor (which may then decode it into the numerical domain of the AI app).

XayNet then determines new values Q_i , u_i , and s_i whose publication initiate the next round. The threshold values u_i and s_i may be adjusted across rounds based on meta-data collected during these rounds. For example, one such metric for a round is the difference between the number of “update” participants expected by Cryptographic Sortition and the actual number of “update” participants that contributed a local model in the aggregation of that round.

The value Q_i should be computed by XayNet in a manner that is transparent, verifiable, but also not predictable before the previous round has completed. For example, one may set

$$(3) \quad Q_i = \text{SHA-3}(Q_{i-1} || pk_i^X || u_{i-1} || s_{i-1} || \langle fo \rangle)$$

where $\langle fo \rangle$ is some data that XayNet could not produce before the current round has begun and that is verifiable by participants. For example, this could be the first fresh public key of a “sum” participant k listed in the dictionary D_s and the corresponding public key pk_k .

5.2 EXAMPLE RUN OF OUR PET PROTOCOL

Let us illustrate this round protocol with an example. We have 200,000 participants and $u_0 = 0.0025$, $s_0 = 0.00005$. This gives us 500 expected “update” participants and 10 expected “sum” participants for round 1. In round 1, we may observe that 509 “update” participants send a first round message, of which 475 send the second round message as well. In round 1, then 8 of the expected 10 “sum” participants may send a first round message, of which 6 also send the second round message. The value Q_0 is a 128-bit string. ElGamal, Hash functions, pseudorandom number generators and so forth are configured with sufficient security.

In terms of communications, XayNet first publishes 128 random bits and two rationals. Then 8 “sum” participants send a short message to XayNet. Also, 509 “update” participants send a long message (or a short one if masked local models are instead uploaded to an S3 bucket) to XayNet. XayNet broadcasts a dictionary D_s of 8 fresh public keys (to 475 of the 509 “update” participants). These 475 “update” participants send, each, a dictionary of 8 ElGamal encryptions of their short seed to XayNet. XayNet sends or broadcasts 8 dictionaries of 475 ElGamal encryptions of short seeds (one to each of the 8 “sum” participants). Then 6 of the 8 “sum” participants send a short message (authenticated encryption of a rational number) to XayNet.

Let us use this example to also illustrate how frequently participants get selected by the random oracles (1) or (2), where policies may lower these frequencies further. For the 200,000 participants, let us say that federated learning uses 500 rounds, and that $u_i = 0.0025$ and $s_i = 0.00005$ for all rounds. How often is a participant expected to be selected over all 500 rounds? We may assume that the random oracle selection is independent across rounds. Therefore, a participant is expected to be selected $500 \cdot 0.00005 = 0.025$ times as a “sum” participant and $500 \cdot 0.0025 = 1.25$ times as an “update” participant across all 500 rounds.

These expected values are independent of the total number of participants, here 200,000. These frequencies apply to any sufficiently random selection process and so are not dependent on choosing (1) and (2) as random oracles. These expected values also suggest that there is no point in keeping any 1-1 channels open across rounds, and not doing that also helps with achieving overall privacy. But one may keep channels open between two round messages that participants send in a round when selected for a particular task.

5.3 PRIVACY OF OUR PET PROTOCOL

Let us discuss the privacy of this PET protocol. In our case, the privacy goal is not a standard one from Cryptography, such as computational indistinguishability under chosen ciphertext attacks. Rather, our goal is that parties (XayNet and participants) will neither get exposed to nor will be able to access personal data for whose control they have no legal basis. We make the following security assumptions:

- AS5** No participant will send a local model to any other party (participant or XayNet).
- AS6** No participant will send a masked local model to any other participant.

If these assumptions were not met, then some parties would be in breach of their own data protection obligations. Controlling such behavior is not in the scope of our PET protocol.

Let us first consider the case when all parties are honest-but-curious and so follow the rules of our PET protocol. To be conservative, we assume that XayNet could reconstruct the AI models from the flat vector in Z_m^d that they are abstracted to, whenever XayNet knows the values of such flat vectors. This is a reasonable and cautionary assumption to make, as companies may run its own AI apps on XayNet.

Masked local models as non-personal data

The masking of local models acts as a one-time pad and so XayNet won't be able to determine any information about the local model

with reasonably likely means used: the shared pseudo-random number generator ensures that the masked local model is computationally indistinguishable from a random element in the group Z_m^d . So even if the local model would allow for inferences to be made about personal data, XayNet has no information gain from the masked local model. XayNet might as well test random elements in that group, transform these elements back into the structure of local models, and attempt on these *random* models any known attacks for making inferences about personal data.

Moreover, suppose XayNet was given an element κ of that group Z_m^d with a claim that this element unmasks the given random element c into a genuine local model as $c - \kappa$ in Z_m^d . Then XayNet would have no means of checking whether that claim is true as the unmasking will produce what looks like a random vector. This is in contrast to decrypting ciphertext of plaintext that is understandable by humans, such as *"We will attack at dawn."* In such cases, decryption with the claimed key can verify (for conventional encryption schemes such as AES) that the correct key was used; this is so since the wrong key will compute a non-comprehensible string of characters. This is perhaps why such ciphertext is deemed to be personal data as well, in the spirit of Article 4(5) of the GDPR.

But, for the reasons given and expanded below, we argue that masked local models should not qualify as "pseudonymized personal data".

Attacks that attempt to make inferences about personal data from local models are involved and, from the perspective of XayNet or any attacker, done here on random AI models. So any such inferences appear random to XayNet or another attacker as well. To be specific, this randomness does not refer to the inference statement itself, which might say *"With likelihood 0.74, this patient record belongs to someone who has HIV."* Rather, it refers to the fact that such a quantitative statement is made based on having performed a successful attack on one, putative, local model. But the genuine model may be any other one from

billions or more possible elements in Z_m^d . An attacker can, therefore, have no confidence whatsoever in the veracity of the inferred statement.

An attacker, therefore, would have to significantly increase such confidence in order to make actual inferences about personal data. XayNet, or another putative attacker, may attempt to boost such confidence by performing and succeeding with such attacks on many random elements of Z_m^d as putative local models, to see whether the same “inferences” occur more frequently across such models. But this is not a reasonably likely means used as it would seem to require successful such attacks on a vast number of random elements in Z_m^d – what is known as *brute-force* methods. To summarize, this suggests considering masked local models as non-personal data from a legal point of view, even though they are conceptually encryptions of what might, in principle, allow inferences about personal data.

Privacy of seeds and masks

Another concern is that XayNet gets exposed to any information about the seeds s_k . Like any party, XayNet can compute the corresponding mask r_k if it knows the value of s_k . Therefore, any inference that XayNet could make about s_k would allow XayNet to make inferences about r_k . Since XayNet knows the masked local model y_k and that $y_k = x_k + r_k$ in the abelian group Z_m^d , this would allow XayNet to make inferences about the local model x_k .

But ElGamal encryptions are semantically secure when the underlying cyclic group satisfies the Decisional Diffie-Hellman assumption. Semantic security means that, given a ciphertext, any party can learn about the plaintext no more information than what she could already learn about the plaintext from knowing its length. In our case, the length is the length of seeds and a public system parameter. And there is nothing one can learn from the length of a random seed about the value of that seed. This means that XayNet, by receiving ElGamal ciphertexts of seeds, does not get exposed to any information that might allow it to gain information

from a masked local model about its unmasked version.

By assumption As5, participants will neither share their local models nor any information about them with any party.⁷ So in our protocol, “update” participants will not be exposed to personal data for whose control they have no legal basis. And “sum” participants will learn the values of seeds and, indirectly, the masks used by “update” participants to mask local models. However, these seeds are generated from entropy that is completely independent of the local model and is, therefore, random information. This means that “sum” participants will not be able to make inferences about personal data that may reside in local models.

Next, we consider what information might be inferred from the sum of masks. If all masks used in the computation of the sum of masks σ are random, then nothing can be inferred about the value of any of these masks from that sum σ . This argument seems to persist when perfect randomness is replaced with *pseudo-randomness* in implementations. The absolute guarantee of perfect randomness, that no information flows from that sum of masks, then turns into “no information flows from that sum with reasonably likely means used”. Therefore, when XayNet learns the value of σ , it has no reasonably likely means used to infer any information about any of the summands that was used in the computation of σ .⁸ To XayNet or any other putative attacker, σ appears like a random element in the group Z_m^d . We reassess this below when parties may act maliciously.

Pseudonyms and metadata

For data privacy, all public keys used in our PET solution have to be pseudonyms that cannot allow inferences about identified or identifiable natural persons. In particular, public keys would not be those already used for identification, for example in a company’s intranet. Our PET protocol uses pseudorandom numbers as secret keys $s k_k$, which also makes public keys $p k_k$ and $p k_k^X$ pseudorandom (although related to their secret keys). Refreshing the key for XayNet for each round adds security, for example, it prevents replay

attacks of messages. Messages sent in a round may also contain metadata. For example, “update” participants may send values of quality metrics about their local models along with the masked local model to XayNet. Such metric information itself does not allow inferences about the local model or personal data, for example, an F1 score. But we will need to consider holistically the impact on data protection of data communicated, processed, stored, and deleted in our PET solution, and in XayNet overall. In particular, our production-grade version of XayNet would only log privacy or security-relevant events and do this securely; and masked local models and metadata would not be kept in storage once a round has completed successfully.

Data privacy of the global model

Let us assess whether the updated global model may contain personal data. This seems not to be the case when all local models used in its aggregation do not contain personal data. So let us consider the case when some local models contain personal data; for example, this may occur when each local model is trained on a specific person’s voice and speech characteristics.

The intuition is that inferences about personal data from the global model alone are no longer possible once it is the aggregation of sufficiently many local models. This intuition is guided by our common understanding of how aggregations such as averaging work, and this is also suggested in the legal opinion in [Par14]. For example, if someone tells XayNet that the average height of a group of people is 172cm, then XayNet can infer from this only that at least one person in that group has height at least 172cm. If a participant gets such information and also is told that she belongs to that group, more information can be obtained. For example, if the participant knows her height of 175cm, then she can infer that at least one other member of that group has height less than 175cm. Since she also knows the difference between her height and the average height, she can make further such inferences. But neither that participant nor XayNet will be able to make any

such inferences about a specific individual in that group.

However, this “safety in numbers” makes some implicit assumptions that may not hold for certain types of learning running in certain configurations of federated learning. We will discuss this in more detail in the next two subsections. Essentially, these risks, that the computed global model allows for inferences about personal data, need to be prevented or mitigated. Prevention may mean not to execute certain use cases at all. Mitigation may be achieved by constraining the values of parameters for federated learning to minimize such risks to a level at which “reasonably likely means used” will not allow for inferences about personal data.

5.4 SECURITY OF OUR PET PROTOCOL

We now study the security of our PET protocol, with a focus on how security breaches could also breach data privacy. Let us first study malicious “sum” participants. Suppose a participant k' has access to all masks s_k where k ranges over K_2 and k' is in K_2 as well. If k' also were a “sum” participant, she could send $\sigma + x_{k'}$ as value of sum of masks (instead of the correct σ) where $x_{k'}$ is the weighted local model of participant k' . Note that XayNet would not be able to detect this behavior since $\sigma + x_{k'}$ and σ will look like random elements in Z_m^d .

The global model, unmasked with $\sigma + x_{k'}$, would then not be the sum of all weighted local models but only of those weighted local model x_k where k is in $K_2 \setminus \{k'\}$. When K_2 has exactly 2 elements, then the unmasked global model will be the other local model, exposing potentially personal data.

More generally, “sum” participants could collude to further reduce the aggregation scope (by adding more local models $x_{k''}$ to the sum $\sigma + x_{k'}$) so that the unmasking would produce a sum of local models from a smaller subset of K_2 . This is why we do not allow participants to take on both tasks, “sum” and “update”, in the same round and why aggregations must have at least three summands in our protocol implementation. In a cross-device use case,

we may insist on a much higher number of summands, say 100, to sufficiently reduce any residual risks.

Another security concern is that “sum” or “update” participants could send to XayNet random data, e.g. a correct masking but of a random element in Z_m^d instead of a local model. XayNet could not detect such randomness as such, due to the privacy-preserving nature of the protocol. This is a trade-off that is well recognized in federated learning [KMA⁺19]. Similarly, “update” participants could send correctly masked local models but where their local models are “poisoned” to enable an attack vector about personal data that may reside in other participants’ local models.

The security of the mechanism by which XayNet selects the sum of masks can also be hardened with voting algorithms. For example, if “sum” participants submit disagreeing values for the sum of masks, this algorithm may vote for the one that was submitted most. The criterion of the algorithm for declaring a round attempt to be a failure needs to be resilient to malicious behavior. For example, if a round attempt is declared a failure as soon as not all “sum” participants send the same sum of masks to XayNet, a malicious such “sum” participant can force a round attempt failure by sending an incorrect value as the sum of masks. We here point out how our PET protocol can mitigate against such threats:

Security with scale amplification

Our random oracles in (1) and (2) are transparent and cannot be manipulated by participants. Their randomness also means that an attacker needs to compromise l percent of *all* participants if she wants to compromise l percent of a group of selected participants. For example, if there are 10 expected “sum” participants and she wants to control at least 5 of them to compromise a voting algorithm for sums of masks, she will have to compromise at least half of all participants, regardless of how many participants there are.

For an expected number of 500 “update” participants, an attacker would need to control

about 0.2 percent of all participants to ensure that at least one selected “update” participant is controlled by her to launch a “data poisoning” or “model poisoning” attack. For two million participants, this means that the attacker needs to control at least 10,000 participants to achieve this on average.

This security, therefore, scales in the number of all participants and allows us to control and potentially contain the probability that a malicious participant will be allowed to submit either a sum of masks or a masked local model to XayNet. In this reasoning, we should not forget that a malicious participant could also leak local models or indeed raw personal data to other parties outside of the PET protocol. So we have to make assumptions about the security of the end devices of participants in any event.

XayNet also needs to be able to verify that public keys are legitimate. How this is done may depend on the type of use cases. For example, a XayNet instance may run in a company intranet where trust is created within the intranet. In other use cases, participants may have a certificate of their key, included in messages for XayNet to verify. Other use cases may provide trust through access tokens.

ElGamal encryption is malleable, so one may transform a ciphertext into a ciphertext of a related message such as a multiple of the plaintext. Our solution uses this scheme only for the encryption of random seeds whose sums will be voted on by XayNet. Malleability, therefore, does not seem to increase the capabilities of attackers and is irrelevant when participants and XayNet are honest-but-curious.

5.5 ATTACKS ON FEDERATED LEARNING

Attacks on federated learning may compromise security, privacy, accuracy of models or other aspects. While our focus here is on attacks that may allow for inferences on personal data, we begin with discussing an attack that illustrates how malicious attackers may exploit a privacy-preservation mechanism for federated learning, such as our PET

solution, to create *semantic backdoors* that allow an attacker to poison their local model so that certain inputs to the global model will produce outputs that the attacker controls. In [BVH⁺18], such attacks are developed and their feasibility is demonstrated. To put this work into the context of our technology, we take the liberty of discussing their work using our terminology.

Backdoor attacks of federated learning [BVH⁺18]

A malicious “update” participant aims to compute an updated local model so that its masked local model will influence the aggregation into a global model such that two objectives are met:

- ▶ The global model has, on attacker-desired inputs, output behavior given by the attacker.
- ▶ The global model has the intended and benign output behavior on all other input.

For the attacker, this is essentially a two-task type of learning that may be continued across federated learning rounds. Insertion of such a backdoor local model by a “sum” participant as an attacker during the convergence phase of federated learning, as opposed to in earlier rounds, tends to lead to higher persistence of that backdoor in continuously modified global models. This is a powerful attack since backdoors also get introduced right after the aggregation step and basically cannot be detected at aggregation time due to the PET mechanism that governs aggregation. Making federated learning resilient to such attacks is an important and open problem, especially in cross-device use cases in which many devices may act maliciously. The threat of this attack may be mitigated in use cases in which devices or their apps are under the control of an organization and insider attacks are unlikely to occur.

On the positive side, backdoor attacks do not seem to have any implications for data privacy, though, if we discard pathological attacks as described next. For example, a use case

may learn a global model that completes trigger sentences with a word. Backdoors would then control the completion word for certain trigger sentences, where these words would presumably be personal data that the attacker has.

It seems difficult to perceive this as a threat to privacy though. For one, the information channel of model outputs is limited, in this example a sole word, and in classification problems a class name. Second, that personal data would occur in a context that may not violate privacy and rather produce fake news – for example in a trigger sentence “My favorite actor is” and the completion word (if the information channel had this much capacity) “Joanne Dow from 15 Lawn Mews in Arlington”. However, it seems prudent to monitor the advance of such attack technologies in terms of their capabilities of compromising privacy.

Membership inference and property inference attacks

In the above attack, an attacker uses two-task learning so that federated learning would basically also learn both tasks without realizing this. This approach can also be applied in privacy attacks on collaborative learning, including federated learning, as developed and demonstrated in [MSCS18]. In this paper, it is shown how an attacker can exploit the control of its local model and knowledge of the values of global models across rounds to influence the creation of global models that learn a “hidden task”. The latter may be

- ▶ a *membership inference* attack, to decide whether a certain data point belonged to a certain batch of local training data, or
- ▶ a *property inference* attack, to decide whether all or specific members of a class enjoy a specific property.

These are instances of *model inversion* attacks, that aim to make inferences about the training data, given the trained model. For deep learning, such attacks can exploit the structure of deep learning models and how errors of loss functions get propagated through the layers

of these networks. When federated learning uses Stochastic Gradient Descent, active inference attacks can improve their effectiveness by isolating participants from the aggregation server [NSH18]. Our PET solution, therefore, needs to ensure that attackers cannot impersonate XayNet across communication links.

Since XayNet will not allow rounds with only two participants, we focus on the experiments in [MSCS18] that had more than two participants (which were between 4 and 30). They show the feasibility of such attacks across a range of benchmarks. While the accuracy of the “hidden task” seems to deteriorate with more participants in rounds, at least one of their experiments had authors who write such individual text that these authors were identified through the “hidden task” in the global model even with larger (but not larger than 30) numbers of participants.

But such strong signals for re-identification may deteriorate when the number of participants reaches hundreds or even thousands, and not tens. In any event, these research results suggest a periodic review of the state of the art of such attacks, to assess whether they give cause to updating XayNet and its configuration space for use-case execution.

Generally, we can say that in XayNet no party gets access to local models of other parties. This restricts any attempted model inversion attacks in XayNet to the global model as a target.

Also, attacks that are specific to federated learning, such as the one in [WSZ⁺18], and that aim to identify to which participant a made inference about personal data is attributable will be ineffective against XayNet. This is so since they assume that the attacker (in the paper, the aggregation server, i.e. XayNet) has full access to unmasked local models.

5.6 PRIVACY IN DEPTH

Given that nascent attack techniques may extract personal data from global models in certain use contexts, this raises the question of whether and when AI models would be de-

med to be personal data themselves. We refer to [VBE18] for a nice discussion of this question and implications of its answer in terms of the rights and obligations of data subjects and data controllers. As advocated in [VBE18], we believe that regulation and legal opinions around the data-privacy status of AI models need to reflect the governance of AI models more holistically.

Let us explain this through an example of two AI models M and M' , used for different applications. Suppose the AI model M can be argued to not contain personal data (with unclear confidence in that argument), but it lacks transparency in the provenance of its training data, is used in automated decision making concerning individuals, and is traded in non-transparent ways in marketplaces outside the EU or its spheres of influence. Whereas the AI model M' might be judged to contain personal data because of a theoretical attack such as the one in [MSCS18], but has transparent data provenance, is only used as a decision-support component in a sole system within the EU, is not stored longterm, and is replaced with an updated model once a month.

Which of these two models M and M' presents a stronger need for regulation and checks regarding the protection of the interests of data subjects?

While this may be a rhetorical question in terms of social values, it does suggest one sense of what we call *Privacy in Depth*: the consideration of the environment in which AI models, data subjects, and data controllers, and potentially other parties interact and where governance needs to take all such interactions into account to offer privacy in AI.

A second sense of Privacy in Depth concerns the design of platforms such as XayNet.

Our design and implementation of XayNet are based on the belief that the insertion of privacy-enhancing mechanisms throughout the layers of its technology stack, including storage, will help with preserving the privacy of federated learning. Our PET solution, in

combination with random pseudonyms, deletion of data once it serves no further purpose in processing for federated learning, and constraints on the types of learning and parameter choices for model aggregation can – together – make it unreasonably hard for an attacker to extract personal data from the global model and messages that XayNet would communicate to other parties.

5.7 POST-QUANTUM CRYPTOGRAPHY

Our PET solution uses primitives from classical cryptography, some of which are susceptible to attacks with future, scalable quantum computers. These primitives include public-key cryptography based on the discrete logarithm and may include the digital signature scheme used.

We can make our PET solution post-quantum ready by exchanging those primitives with post-quantum alternatives that offer similar data protection assurances, e.g. the semantic security of encrypted seeds that XayNet receives. The architecture of XayNet will facilitate such a transition, which we will initiate if and when our customers want to see such a change.



6. Open-source and fully managed service approaches

Our federated learning project XayNet is open-source under: <https://github.com/xaynetwork/xaynet>.

These components are predominantly written in Rust for scalability, security and robustness reasons. Parts are written in Python to allow for easy integration into any kind of AI framework, such as TensorFlow or PyTorch. All of these packages come with extensive documentation.

The Coordinator may be run locally and also as a Docker image. The Coordinator needs a storage service that provides an AWS S3 API. For development, we use MinIO. We provide docker-compose files that start a Coordinator container along with a MinIO container, and pre-populate the appropriate storage buckets. The open-source SDK is also installed as a Python package on the devices of participants. There is support for registering your AI app and device to the Coordinator. We also offer utility methods to aid the implementation of your `train_round()` method.

Furthermore, we provide a closed source SDK in Dart that can be integrated to cross-platform mobile devices (Android and iOS) or compiled to JavaScript for Browser applications. In principle, you may use this open-source software to build and host your own use cases for federated learning. Alternatively, XayNet is a hosted version of this open-source project that offers several, significant benefits over your use of a self-hosted solution:

- ▶ Most importantly, XayNet supports cross-device mobile, browser or IoT use cases. Our closed source SDK is required to run on mobile devices or other machines, e.g. cars.
- ▶ The cross-device SDK also enables on-device training functionalities, e.g. for training voice recognition models directly on the machine.
- ▶ XayNet gives you predictable costs for running a federated learning use case and provides service level guarantees.

- ▶ XayNet does not require pre-existing federated learning skills from your AI engineers.
- ▶ XayNet runs our PET protocol for privacy-preserving federated learning. This comes with GDPR warranties for compliance of the aggregation and computed AI models.
- ▶ XayNet allows you to choose whether XayNet gets to know the value of the global AI model or whether you want to protect this information, e.g. if it is valuable IP.
- ▶ XayNet offers a dashboard that shows metrics and other diagnostics for AI engineers and DevOps alike and supports project life-cycle management.

7. Representative use cases

Federated learning trains an algorithm across multiple decentralized devices or servers that hold and train on local data samples. This process happens without any exchange or transfer of these data samples. Federated learning thus can address critical issues such as data privacy, data security, data access rights. In a privacy-preserving form, federated learning can be very trustworthy, making it ideally suited for many of today's applications of machine learning in enterprise or consumer-tech areas.

To illustrate the power of privacy-preserving federated learning, this section will showcase two use cases that represent the generally established differentiation of *cross-device* and *cross-silo* applications, respectively. *Cross-device* applications commonly come with significantly more complexity, as such use cases often involve millions of users or devices. Whereas *cross-silo* applications typically consist of up to hundreds of silos in which local training takes place.

Furthermore, *cross-device* applications face the challenge that the frameworks that support model training on such devices are less mature and also more complex. For example, at present TensorFlow Lite only supports the personalization of the last model layer. Moreover, the development of such cases needs to cope with hard resource limitations of mobile or even IoT environments [BEG⁺19]. For example, such a device may train only when sufficient power is available.

Currently, XayNet supports *cross-silo* use cases while being completely framework agnostic. In its current development, XayNet is strongly focused on supporting *cross-device* applications, which we see as the major driver of Xayn's business model. For *cross-device* applications, we also work on providing the training on mobiles as part of our SDK. Herby, we concentrate especially on training voice recognition models as well on uses cases around search personalization and image classification. This also supports algorithms such as Convolutional Neural Networks (Conv1D), or Multi-armed bandit as a reinforcement learning layer.

7.1 CROSS-SILO USE CASES

Cross-silo use cases are often driven by either a technical or compliance limitation that prevents aggregation of all data into a single database for training. These use cases may be internal in that a larger corporate group tries to consolidate AI efforts over its multiple sub-sidiaries. Or they may be external when several companies want to collaborate without sharing their valuable training data. Other settings face potential mistrust between parties alongside compliance issues. For example, in the internal case, the fact that subsidiaries may have the same shareholders does not necessarily guarantee the right to easily share data across these subsidiaries – thus the creation of a sole aggregated data source for training may not be possible.

Consider, for example, a corporate group that wants to roll-out a new AI-based enhancement of its accounting software across its subsidiaries. Each subsidiary is independently operating its own Enterprise Resource Planning system. So there are technical hurdles for aggregating data across subsidiaries, in addition to potential compliance issues that prevent such aggregation. Using privacy-preserving federated learning, however, each subsidiary can integrate a federated learning SDK into their machine learning pipelines, such as TensorFlow, to consolidate their AI models for faster initial training and continuous learning across multiple subsidiaries. For the AI enhancement to be effective, the accounting processes should be similar across these subsidiaries.

7.2 CROSS-DEVICE USE CASES

As already stated, Xayn's strategic focus is on cross-device use cases within the consumer tech industry. In that space, several AI use cases train on highly sensitive user data and therefore require further protection mechanisms. Increasingly, mobile devices become the predominant interfaces for digital interactions and experiences. Therefore, a platform for federated learning that delivers privacy by design is a compelling enabler for many mobile use cases.

We illustrate this through a closer look at a voice-assistant use case. This may contain the identification of particular voices, the recognition of recordings, as well as the transcription of phonetic conversational input into instructions such as those that assist diary management. Preservation of privacy is, therefore, a highly critical requirement for voice-assistant use cases. Generally, voice assistants only act as an interface to collect user conversations. The latter are then labeled centrally via so-called data associates [BVDT19].

This practice strongly violates user privacy, even though users may have given their consent to such processing. Such consent may not be informed as users are not aware of this scope of processing, and voice conversations may also contain data from other users who did not give consent but who happen to be in the recording environment. For example, recordings in a car or in front of a TV typically involve voice input from other people as well.

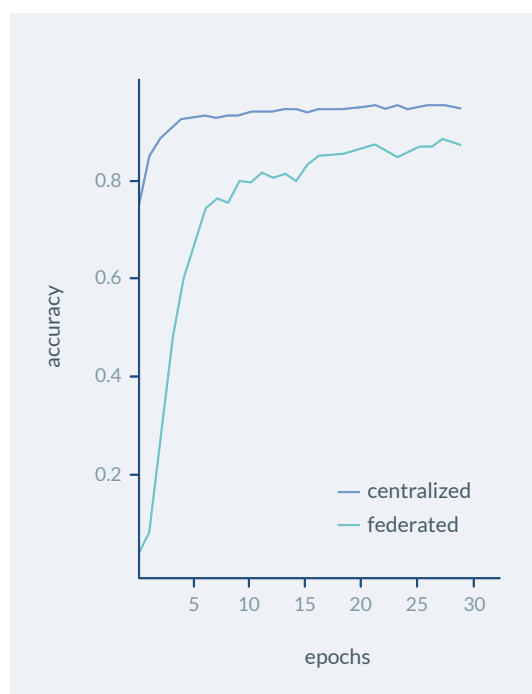


Figure 4: Comparison of centralized and federated learning approaches for Voice Recognition with the TF Speech Recognition Data Set. The number of training epochs and the model accuracy are shown on the x-axis and y-axis, respectively. The training uses the convolutional neural network algorithm Conv1D followed by two LSTMs. Model accuracy of the federated approach converges quickly to that of the centralized approach.

Federated learning can act as a game-changer in this case, as it allows the AI model to be trained directly at the device level, which no longer requires to upload the raw training data to central servers and thus makes the aforementioned problematic practice obsolete.

To corroborate the potential of federated learning for voice-assistant use cases, we used the TensorFlow Speech Recognition dataset and applied the convolutional neural network algorithm Conv1D followed by two LSTMs (long short-term memory), a form of recurrent neural network. In this setup, we compared a central Keras-based TensorFlow training method against a decentralized approach that uses our XayNet federated learning framework.

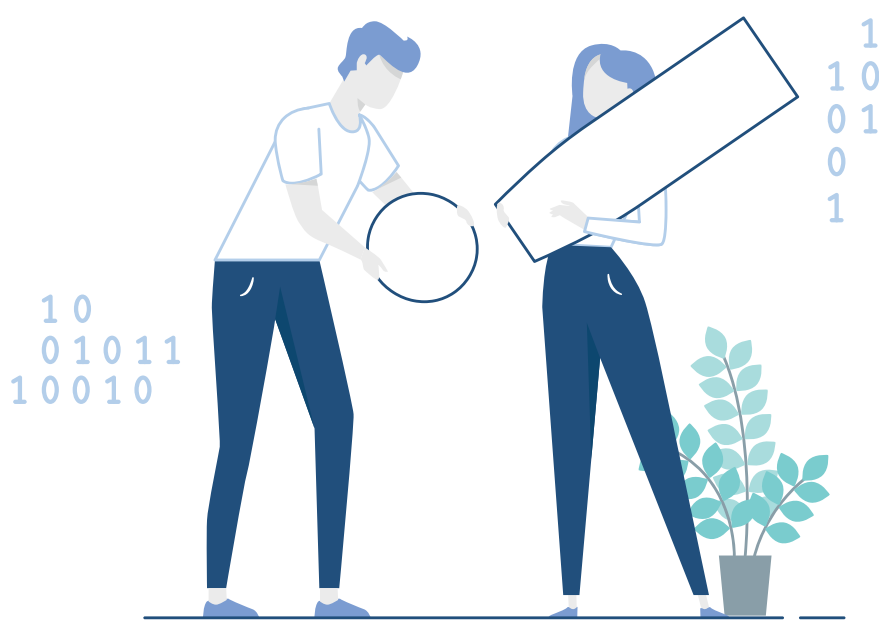
As we can see in Figure 4, the model accuracy of the federated learning approach converges very fast with that of the central variant and achieves a similar performance already after ten training epochs. This benchmark demonstrates two interesting points: firstly, that the global model can learn words many local models may never have been directly trained on; secondly, that the accuracy of the global model tends to converge towards the accuracy of the model trained in the centralized approach. Thus, we expect that – especially once further optimization is added – the performance of the models trained in both approaches is relatively comparable. Crucially, the training with the federated approach preserves the users' privacy.

8. Conclusion

In this Whitepaper, we described our open-source project on federated learning, XayNet. This project incorporates expertise from AI, law, cryptography, and platform architecture to develop a platform in which privacy is designed into all layers and the execution of federated learning use cases. We specified here in detail our privacy-preserving protocol for federated learning. It combines anonymization for federated learning of global models with pseudonymization of local models prior to their aggregation. The experiments descri-

bed in this Whitepaper demonstrated that federated learning can maintain privacy while also securing high accuracy of AI models, including in voice-assistant use cases.


To follow updates on this work in progress, especially its support for cross-device use-cases for smart-phones, please follow us on Twitter (@XAINAG), LinkedIn, and visit our website xaynet.dev. For expressions of interests and queries, please send us an email at engineering@xaynet.dev.



References

- [ACG+16]
Martin Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. In *ACM Conference on Computer and Communications Security*, pages 308–318. ACM, 2016.
- [AH19]
Patrick Ah-Fat and Michael Huth. Optimal accuracy-privacy trade-off for secure computations. *IEEE Trans. Information Theory*, 65(5): 3165–3182, 2019.
- [BDR19]
Steven M. Bellovin, Preetam K. Dutta, and Nathan Reiter. Privacy and Synthetic Datasets. *Stanford Technology Law Review*, 22, 2019.
- [BEG+19]
Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konecny, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. *CoRR*, abs/1902.01046, 2019.
- [BVDT19]
Mark Bergen, Gerrit De Vynck, Natalia Drozdak, and Giles Turner. Silicon valley is listening to your most intimate moments. *Bloomberg Businessweek*, 2019.
- [BVH+18]
Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. *CoRR*, abs/1807.00459, 2018.29
- [CGDS+20]
Olivia Choudhury, Aris Gkoulalas-Divanis, Theodoros Salonidis, Issa Sylla, Yoonyoung Park, Grace Hsu, and Amar Das. Anonymizing data for privacy-preserving federated learning, 2020.
- [CM16]
Jing Chen and Silvio Micali. ALGORAND. *CoRR*, abs/1607.01341, 2016.
- [DR14]
Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.
- [Dwo06]
Cynthia Dwork. Differential Privacy. In *ICALP (2)*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.
- [ea18]
Philip Gerbert et al. The big leap towards ai at scale. Online, 2018.
- [GPM+14]
Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014*, December 8–13 2014, Montreal, Quebec, Canada, pages 2672–2680, 2014.
- [GZW17]
Suyog Gupta, Wei Zhang, and Fei Wang. Model accuracy and runtime tradeoff in distributed deep learning: A systematic study. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19–25, 2017*, pages 4854–4858, 2017.
- [KJA18]
Sarit Khirirat, Mikael Johansson, and Dan Alishtarh. Gradient compression for communication-limited convex optimization. In *57th IEEE Conference on Decision and Control, CDC 2018, Miami, FL, USA, December 17–19, 2018*, pages 166–171, 2018.
- [KMA+19]
Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurelien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D’Oliveira, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adria Gascon, Badi Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyu-an Huo, Ben Hutchinson, Justin Hsu, Martin

- Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecny, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrede Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Ozgur, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramer, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning, 2019.
- [MMR⁺17]
Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agueray 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, pages 1273–1282, 2017.
- [MMRyA16]
H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agueray Arcas. Federated learning of deep networks using model averaging. CoRR, abs/1602.05629, 2016.30
- [MSCS18]
Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Inference attacks against collaborative learning. CoRR, abs/1805.04049, 2018.
- [NSH18]
Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Stand-alone and federated learning under passive and active white-box inference attacks, 2018.
- [oCCA19]
The Association of Chartered Certified Accountants. Machine learning - more science than fiction. Online, 2019.
- [Par14]
Article 29 Working Party. Opinion 05/2014 on Anonymisation Techniques. Legal Opinion by the Article 29 Data Protection Working Party, April 2014. 0829/14/ENWP216.
- [RAD78]
R L Rivest, L Adleman, and M L Dertouzos. On data banks and privacy homomorphisms. Foundations of Secure Computation, Academia Press, pages 169–179, 1978.
- [Res19]
IDG Research. Studie machine learning / deep learning 2019. Downloadable Report, 2019.
- [RHdM19]
L. Rocher, J. M. Hendrickx, and Y.-A. de Montjoye. Estimating the success of re-identifications in incomplete datasets using generative models. Nature Communications, 2069(10), 2019.
- [VBE18]
Michael Veale, Reuben Binns, and Lillian Edwards. Algorithms that remember: model inversion attacks and data protection law. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 376, 2018.
- [WSZ⁺18]
Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. CoRR, abs/1812.00535, 2018.
- [XLW⁺18]
Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially Private Generative Adversarial Network. CoRR, abs/1802.06739, 2018
- [yA18]
Blaise Agueray Arcas. Decentralized machine learning. In IEEE International Conference on Big Data, Big Data 2018, Seattle, WA, USA, December 10-13, 2018, page 1, 2018.
- [YLCT19]
Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. ACM Trans. Intell. Syst. Technol., 10(2):12:1–12:19, January 2019.
- [YTR⁺19]
Chen Yu, Hanlin Tang, Cedric Renggli, Simon Kassing, Ankit Singla, Dan Alistarh, Ce Zhang,



and Ji Liu. Distributed learning over unreliable networks. In Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, pages 7202–7212, 2019.

[ZZZ⁺19]

Chuan Zhao, Shengnan Zhao, Minghao Zhao, Zhenxiang Chen, Chong-Zhi Gao, Hongwei Li, and Yu-an Tan. Secure Multi-Party Computation: Theory, practice and applications. Inf. Sci., 476:357–372, 2019.





1
0
1 0
1 0
1

OUR MISSION

Putting privacy first
while making AI work
for you



XAIN AG

Unter den Linden 42
10117 Berlin, Germany



engineering@xaynet.dev



[@XAIN_AG](https://twitter.com/XAIN_AG)



<https://xaynet.dev>