ISSN 1831-9424

European Commission

JRC TECHNICAL REPORT

Technological Enablers for Privacy Preserving Data Sharing and Analysis

A comparative study

Daniel Hurtado Ramírez, Luis Porras Díaz, Sepideh Rahimian, Juan Miguel Auñón García, Borja Irigoyen Peña, Yusra Al-Khazraji, Ángel J. Gavín Alarcón, Pablo González Fuente, Josep Soler Garrido, Alexander Kotsev

2023

Joint Research Centre

EUR 31634 EN

Contact information
Name: Alexander Kotsev
Address: via E. Fermi, 2749 – TP 263 - 21027 Ispra (VA), Italy
Email : alexander.kotsev@ec.europa.eu
Tel.: +39 0332 78 9069

EU Science Hub
https://joint-research-centre.ec.europa.eu

JRC134350

EUR 31634 EN

PDF ISBN 978-92-68-06693-5 ISSN 1831-9424 doi:10.2760/427718 KJ-NA-31-634-EN-N

Luxembourg: Publications Office of the European Union, 2023

How to cite this report: Ramírez, D.H., Díaz, L.P., Rahimian, S., García, J.M.A., Peña, B.I., Al-Khazraji, Y., Alarcón, Á.J.G., Fuente, P.G., Soler Garrido, J. and Kotsev, A., *Technological Enablers for Privacy Preserving Data Sharing and Analysis*, Publications Office of the European Union, Luxembourg, 2023, doi:10.2760/427718, JRC134350.

Contents

Abstract

As data becomes more important, so does the need to protect privacy, in the sense of protecting the personal, confidential and/or private information that it contains. Privacy Enhancing Techniques (PETs) are a key enabler technology for ensuring that privacy is maintained while extracting value from the data.

This report has two primary objectives: firstly, to analyse and assess the usability and maturity of various PETs within data sharing scenarios, specifically within common European data spaces; secondly, to demonstrate the practical application of one PET in a collaborative scenario involving multiple entities. To achieve these objectives, the report follows a two-phase approach.

In the first phase, a detailed analysis of state-of-the-art PETs was conducted, evaluating their strengths, limitations, and maturity levels. Based on this assessment, Federated Learning, was selected for further exploration in a collaborative scenario.

The second phase focused on implementing a realistic use case within the healthcare domain, which is highly relevant to the European Health Data Space. Healthcare data can be highly heterogeneous across different healthcare providers. This heterogeneity can make it challenging to apply federated learning algorithms effectively. Using Federated Learning as the selected PET, the report presents the results and evaluates their effectiveness.

By addressing these objectives and highlighting the issued posed by healthcare data, the report provides valuable insights into the usability and applicability of PETs, while showcasing the benefits and challenges of implementing collaborative data sharing scenarios. Overall, this report offers valuable guidance for stakeholders seeking to protect privacy and unlock the potential of data in various domains and data spaces.

Authors

Daniel Hurtado Ramírez, Luis Porras Díaz, Sepideh Rahimian, Juan Miguel Auñón García, Borja Irigoyen Peña, Yusra Al-Khazraji, Ángel J. Gavín Alarcón, Pablo González Fuente, GMV

Josep Soler Garrido, European Commission, Joint Research Centre, T.3. Algorithmic Transparency Unit

Alexander Kotsev, European Commission, Joint Research Centre, T.1. Digital Economy Unit

Executive summary

Policy context

The European Union has put forward an ambitious agenda for the establishment of a single market for data that can facilitate a plethora of different use cases ranging from business innovation to improved public services and environmental protection. Through its European Strategy for Data, the EU is establishing a legal framework for improving the trust and fairness of the data economy. In addition, thematic common European data spaces should ensure that data can be shared at scale in a trustworthy and secure manner. Within that context, improving the availability of data is a core step for harnessing the potential of the digital transition, allowing the economy and society to extract value from data. However, these benefits cannot come at the expense of privacy rights. Privacy Enhancing Techniques (PETs) offer a wide variety of technological solutions to achieve both data utility and privacy, each suited to specific scenarios. This report summarises the following consecutive activities.

1. PET Analysis

The analysis presented here is divided into several phases. First, a landscape analysis of privacy-enhancing techniques (PETs) was conducted, and a wide range of techniques were evaluated, at a high level, for their suitability for privacy-sensitive data sharing scenarios. The results of this analysis are contained in Section 3.

Then, Section 4 goes into further detail and brings together an analysis of the state of the art for each of the selected technologies, and for each of them:

- Evaluates its relevance to data sharing scenarios.
- Evaluates its degree of usability by creators and users of data spaces.
- Lists and analyses real-world use cases enabled by that technique.
- Explains the level of security achieved, enabling data holders to take informed decisions.
- Lists and evaluates existing implementations and references.
- Provides and justifies the technology's readiness level (TRL).

2. Experimental Evaluation of PET

Section 5 shows the design and specification of a selected use case that represents a typical data space scenario. It is consisting of the construction of a federated model for the classification of skin lesions, some of which are of cancerous nature, based on data from 4 medical institutions in different countries. A possible approach from the perspective of common European data spaces is presented and data pre-processing (mainly homogenization), modelling and training are outlined.

Three datasets of skin lesions freely available on the Internet were used. Two of these contained images from a single institution and the third from two. The latter was partitioned to separate the images from one and the other. Finally, the federated learning system was formed with four nodes, each containing data from a single institution. The main problem at this point was to standardize the images to make them compatible with the deep learning architecture of the global model. The developed application, which is installed on each node, transforms the data into a previously agreed common format when loading the data.

As this is an image classification task, it was approached using convolutional neural networks. Two different architectures - both implemented with PyTorch - were experimented with: a lightweight three-layer network and a pre-trained EfficientNet network. Although the latter achieved much better metrics, the former was chosen to reduce the running time of the experiments. Both federated and centralized models were trained to compare the two approaches. In the federated case, the framework chosen to implement the infrastructure was Flower FL, due to its flexibility and ease of use. The model aggregation procedure was federated averaging. The model has been evaluated by comparing it with a centralized version and with local versions; the results are presented in Section 6.

While the selected use case is in the healthcare domain, therefore relevant to the European Health Data space, the findings and technical approaches provided can be reused in other data space contexts relatively easy. The developed code can be reused in cases requiring the utilisation of sensitive personal or private data.

Objectives and target audience

The goal of this report is to explore the potential of PETs in improving the utilisation of existing personal and sensitive private data within the EU policy context. There are two main goals: first, the report analyses and evaluates various PETs, their usability, and maturity, in the context of data sharing scenarios, particularly within common European data spaces as defined in the European strategy for data. The intention of the report is to serve as a reference guide for policymakers and data space stakeholders who are our main target audience. The work presented here would allow them to navigate this complex landscape and to understand the possibilities and opportunities made possible by PETs. Secondly, we demonstrate their application in a practical collaborative scenario between different entities. By running a realistic simulation, we give a practical example that can be used as a template for real-world projects. We intended to encompass every aspect, starting from the initial investigation to model evaluation, addressing practical challenges that are frequently overlooked in scientific publications.

Key conclusions

This report provides a detailed evaluation of each PET with an emphasis on their applicability in a data space context. As a general overview, PETs have reached a level of maturity that makes them suitable for real-world applications. Therefore, they hold the potential to play a crucial role in balancing the need for data utility and user privacy. Although some level of specialized knowledge is usually required to use PETs, their usability and technology readiness level is constantly improving thanks to the growing number of mature open-source implementations.

Furthermore, the implementation of a prototype in this study sheds light on the challenges encountered when dealing with healthcare data. The highly heterogeneous nature of healthcare data, characterized by differences in data collection protocols, formats, and standards across various providers, presents obstacles for the effective application of federated learning algorithms. It becomes evident that addressing this heterogeneity is essential to ensure the successful deployment of PETs in healthcare settings.

Next Steps

Several improvements have been suggested to improve the implemented prototype such as, including more data holders and data subjects, a more prominent role of data intermediaries and data altruism organisations, using different models, considering a better aggregation method for the Federated Learning. In addition, privacy can be improved by integrating another PET like Differential Privacy to ensure that no sensitive data can be extracted from the final model. Another idea is to combine the local evaluations using Secure Multi-Party Computation (SMPC) so that only global metrics are revealed instead of per-node metrics. Furthermore, it is recommended to test the usability and effectiveness of different PETs in realistic environment and to investigate their integration within common European data spaces. Raising awareness and educating data space stakeholders is another crucial step in order to enable the application of PETs. Finally, it is essential to enrich open-source frameworks with further development based on the state-of-the-art advances from academia.

1 Introduction

The European data strategy [1] aims to empower researchers, public administrations, and businesses by creating a single market for data. This is a key step in tackling and benefiting from the digital transition currently taking place. Within that context, while access to data and the ability to use it are essential for innovation and growth, they must not come at the cost of the citizen's privacy rights. The tension between the promotion of data access and the protection of privacy rights is an important policy-relevant problem that must be tackled. *Privacy Enhancing Techniques (PETs)* are defined by the European Union Agency for Cybersecurity (ENISA) as *"a coherent system of information and communications technologies (ICT) measures that protects privacy by eliminating or reducing personal data or by preventing unnecessary and/or undesired processing of personal data; all without losing the functionality of the data system [2]"*. In other words, they aim to extract value from data without compromising data privacy.

The goal of this report is to explore the potential of PETs in addressing this tension and promoting both data utility and privacy, tailored to the EU policy context. There are two main goals: first, we analyse and evaluate various PETs, their usability, and maturity, in the context of data sharing scenarios, particularly within common European data spaces as defined in the European strategy for data. The intention of the report is to serve as a reference guide for policymakers and data space stakeholders, allowing them to navigate this complex landscape and to understand the possibilities and opportunities made possible by PETs. Secondly, we demonstrate their application in a practical collaborative scenario between different entities. By running a realistic simulation, we give a practical example that can be used as a template for real-world projects. We intended to encompass every aspect, starting from the initial investigation to model evaluation, addressing practical challenges that are frequently overlooked in scientific publications.

Following this brief introduction, the report is organised in six consecutive sections. Section 2 briefly introduces the most prominent PET together with their relevance to privacy-sensitive data-sharing scenarios. Section 3 each of those PET are evaluated against a set of predefined criteria including their relevance to common European data spaces. Section 4 is dedicated for the prototype design for our selected use case. Section 5 elucidates the implementation of the prototype and the results. Finally, we draw some conclusions and recommendation for future work.

2 Overview of PET

The purpose of this section is to have a landscape analysis of privacy-enhancing techniques (PET). A high-level set of techniques is reviewed, having a preliminary assessment of their relevance in privacy-sensitive data-sharing scenarios. This section should inform the scoping and implementation of approaches for the reuse of citizen and other sensitive data for the establishment of European Data spaces (as defined by the European Strategy for Data [1]).

2.1 Data Sharing Scenarios

One of the main points that we discuss in this document is the "data-sharing" scenario, that is, there is not a one-size technique that can solve the complex problem of sharing data in a secure way, so every technique covers different use cases and has its own specific advantages and disadvantages.

From a research point of view, data sharing is the practice of making raw (and derivative) data available to other researchers. Data sharing increases the transparency of research by allowing confirmation of the interpretation of the results, maximizing the usefulness of the data by allowing it to be used in other research.

While the idea of data sharing and combined use is feasible, and there are different mechanisms to achieve it, we have already discussed how carrying it out becomes tedious when the data contains sensitive information or other interests intervene. Three roles are clearly identified [3, 4]:

- **Data provider**: Refers to an entity or organization, including both aggregators and individuals, that shares data for collaborative learning purposes while preserving data privacy. Typically, a data provider acts as the custodian or curator of the data and seeks to share various types of data. This may include raw data, such as point-of-sale or time series data, as well as pre-processed, curated data that already contains analytics and insights. In the context of Federated Learning, the role of a data provider can vary depending on the scenario. In cross-silo[1] Federated Learning, it is normally organizations who serve as data providers, while in cross-device[2] Federated Learning, they use to be individuals.
- **Data consumer**: It is the organization that is receiving data from a data provider. The data consumer may be wanting to join the shared data with their own data to derive insights.
- **Data intermediary**: It acts as a trusted entity. This optional role can be viewed as the one that provides mechanisms to help you trust and collaborate on data sharing, for example by providing metadata about what is available, or validation of claims by the other actors.

Related to the following PET techniques, the next figure depicts two data-sharing scenarios, where the main requirement is that the output data must be consumed in a private way, that is, preserving data privacy.

— Figure 2.1 (left). Data providers have some interest in common, that is, the output data consumed is a kind of computation where all the data providers are involved. One example of this scenario is in finance. Banks could collaborate to train a model to detect fraud. Each bank would train its own model on its own data, and then the models would be combined to create a more accurate model. This would allow the banks to fight fraud more effectively without sharing sensitive financial data.

  This scenario allows for collaborative analysis and knowledge sharing among the participating organizations. By combining their datasets and expertise, data providers can collectively tackle complex research questions and potentially achieve a more comprehensive understanding of the data. However, this scenario also presents challenges related to data governance, coordination among providers, and ensuring privacy protection for the shared data.

[1] **Cross-silo federated learning** is used when the data is partitioned into silos, where each silo is owned by a different organization. For example, a hospital might have data on its patients, and a bank might have data on its customers. The data from the hospital and the bank would be stored in separate silos.

[2] Cross-device federated learning is used when the data is partitioned into devices, where each device is owned by a different individual. For example, a phone might have data on the user's location, and a laptop might have data on the user's browsing history. The data from the phone and the laptop would be stored on separate devices.

— Figure 2.1 (right). Data providers are independent organizations that just want to provide their datasets to external consumers. Each data provider operates autonomously, sharing their dataset with the intention of enabling external consumers to access and utilize the data for their own analysis and research purposes.

  For example, a company that collects data on customer behaviour could sell its dataset to a research company. The research company would then use the data to conduct research on new marketing strategies. Another example of this scenario is also in the finance industry: a bank could provide its data to a credit rating agency. The credit rating agency would then use the data to assess the risk of lending money to businesses.

  This scenario offers advantages such as increased access to diverse datasets, fostering innovation and facilitating novel research opportunities. However, it also brings considerations related to data compatibility, standardization, trust between data providers and consumers, and establishing mechanisms for data discovery and access.

The functionality of the data intermediary in these scenarios is not closed, so some examples are detailed in the description of the privacy techniques.

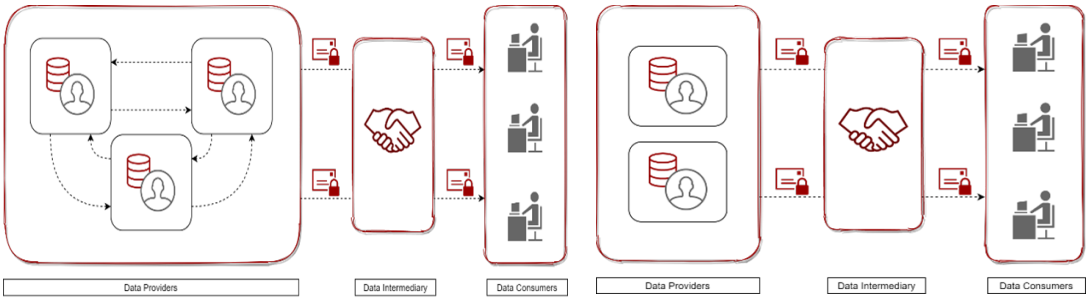

Figure 2. 1 Data-sharing scenarios. Left: Data providers have some interest in common, that is, the output data consumed is a kind of computation where all the data providers are involved. Right: Data providers are independent organizations that they just

2.2 Techniques

In this section, we delve into various privacy-preserving technologies (PETs) and explore their relevance in the context of data-sharing scenarios. The selection of PETs discussed in this section, as well as in the following chapter, is based on their widespread adoption, effectiveness, and relevance to the challenges faced in data-sharing scenarios. From cryptographic techniques to anonymization methods and differential privacy, each technology offers unique advantages and addresses specific privacy concerns. By understanding these technologies and their practical applications, we can navigate the landscape of data sharing while upholding the principles of privacy and data protection.

2.2.1 Secure Multi Party Computation

Secure Multi Party Computation (SMPC) [5] is an umbrella term encompassing techniques in which a set of mutually distrusting data holders want to jointly compute a function without revealing anything beyond the function's output. For example, consider a group of friends who wish to learn the group's average salary without disclosing any of their individual salaries. SMPC provides techniques for achieving this goal, even when not all parties can be trusted. It was originally studied as a theoretical curiosity, but recent advances in SMPC algorithms, network speed, and computing power have made it suitable for real world usage [6].

There are several possible classifications for SMPC techniques:

- **Generality**: Some techniques are optimized for a single type of operation (e.g., Private Set Intersection) while others allow evaluating arbitrary arithmetic or Boolean circuits. This second group usually works by using Secret Sharing to encrypt the input values and then securely evaluating each

gate, transforming shared secrets of the inputs into secret shares of the output [6]. There are other approaches not based on Secret Sharing, such as Yao's Garbled Circuits [7].

- **Adversarial model**: Not all techniques are resistant against all types of attackers. Adversaries might be honest-but-curious, meaning that they will follow the protocol while trying to extract as much information as possible from any intermediate values. Covert adversaries will deviate from the protocol to obtain secret information or compromise the result, while keeping its presence secret. Malicious adversaries are not constrained by trying to appear honest, giving them a wider range of possible attacks [6].
- **Proportion of corrupted parties**: Shamir's secret sharing [8] allows choosing t such that a secret distributed between n parties will need at least t cooperating parties to reveal it. Choosing a small t will make the scheme more resistant to parties going offline, but less resistant against parties becoming corrupted. Beyond secret sharing, some approaches require an honest majority, while others relax this requirement at the cost of having to abort the computation if an adversary is detected.
- **Performance**: Multi Party Computations are usually constrained by network latency and bandwidth, rather than computational cost. Research is focused on lowering the frequency and size of communication. For example, Beaver Triples [9] greatly improve the performance of arithmetic circuit evaluation by shifting the burden of communication to an offline phase.

We will now study a simple example to understand how SMPC works. Going back to the salary example, each of the friends can encrypt their data by splitting it into "additive shares": for example, to encrypt the number 24, pick random numbers that sum to 24, such as {10, 20, -6}. Knowing only two of them gives no information; all three are required to obtain the secret value. Thus, a data owner can distribute two of these numbers while keeping the third one secret. Each friend does this, leaving them with a share of their own data and shares of their friend's data. Operations on the encrypted data work by having each friend perform local operations on their available shares. For example, the sum of all the secret data can be obtained by simply summing the local shares; this will generate a share of the result, which can then be broadcasted to reconstruct the result. Other operations, such as multiplying the data or finding the maximum, are more involved and require intermediate communication rounds since they cannot be fully performed by local operations.

This example can be easily extrapolated, for example, the reuse of patient data, where the data is spread into different health centers. In this case, we can think replace the attribute salary with the diagnosis of a medical treatment to compare the treatment between hospital without sharing the value of it.

SMPC is highly relevant to data-sharing scenarios, since it allows data holders to jointly compute arbitrary functions using their private data, with strong cryptographic guarantees of security. It is suited for data-sharing scenarios such as the one depicted in Figure 2.1 (left), in which the data holders share a common goal. Data providers need to agree beforehand in the exact computation to be performed and the data schema to be used, which might be facilitated by the intervention of the intermediary role. SMPC can be used for simple statistical analysis, such as computing the average, standard deviation, maximum, etc. of a distributed private dataset, as well as for training complex models such as neural networks. Some of these protocols require a trusted dealer that provides trusted cryptographic material [9] to be used for the rest of entities, fitting this role with the aforementioned data-provider.

Notice also that this technology has already been used in the real world, with positive results [6]. SMPC is also suited for encrypted inference, in which a private model is applied to private data without the data owner gaining access to the mode and vice versa.

However, SMPC requires the data holders to perform computations which require online availability and a fast connection, constraining the applications it can be used for. Performance is generally orders of magnitude worse than operating on clear data, which limits the size of models that can benefit from SMPC; still, it might be the right choice for small datasets or models, and/or appropriate infrastructure and connection.

2.2.2 Federated Learning

Federated learning [10] [11] is a form of distributed computing aimed at preserving the privacy and confidentiality of data owned by different organizations when they agree to collaborate to obtain, in a common way, a machine learning model. Under this approach, a model is trained through multiple servers (nodes) each of them containing their own data, without this data leaving where it is stored and, therefore,

without being accessible to the different parties that participate in the training. It is said that in federated learning it is the models that move to where the data is and not the other way around, as in traditional machine learning. The basic idea is that each node builds a local model that periodically sends its parameters to a server, where it is combined with models from other nodes into a global model. This model, once updated, is sent back to the nodes to continue learning in an iterative process that ends after a predetermined number of cycles. The architecture of a federated learning system is shown schematically in Figure 2.2.

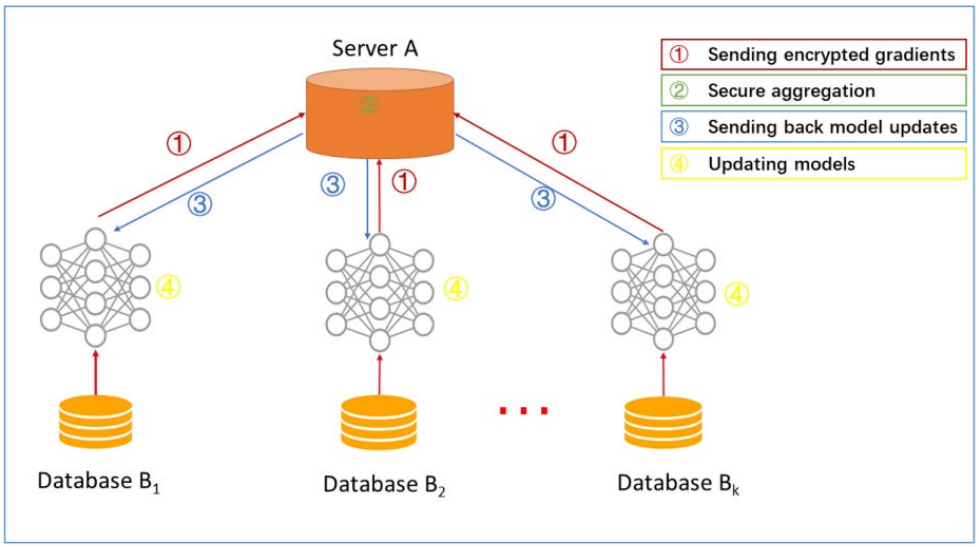

Figure 2. 2 Architecture of a federated learning system [103]

When training a model in a federated way, however, it must be taken into account that the fact that the data does not leave the server where it is stored is not a sufficient guarantee to ensure its confidentiality, since a malicious user can find out a large amount of information from other parties with only local model parameters and their own data, so it is common to use some additional privacy mechanism, such as aggregating local models on a central server trusted by all parties, or encrypting data parameters models, for example using a *secret sharing* technique like in multi-party computation.

In general, when talking about federated learning, it is assumed that all participants share the same schema for the data; that is, that the columns -or components- of the data set are the same for everyone, in a collaborative framework in which each participant contributes rows (records) to the global set. This is what is known as federated learning with horizontal partitioning, and it is the most common approach in research articles and the one contemplated by the main frameworks. However, a novel approach within federated learning is vertical partitioning, in which each part contributes columns, rather than rows, to the whole. This approach is technically more complex as it requires aligning the database records of the different parties and is also more prone to errors due to failures that occur when carrying out this alignment, which will hardly be perfect, however, it is a little explored field with great room for improvement that opens the door to innumerable business cases. The assumption of having the same schema can be carried out by the data intermediary role, where the metadata associated to the data can be requested to know if it is possible to perform a federated learning training or not.

There are several scenarios in which federated learning combined with other techniques described here can or must be applied to maintain the privacy of all the parties, all of them related with the data-sharing scenario depicted in Figure 2.1 (right). Some of these techniques are:

- A central party wishes to train a model on the data of several distinct data holders. In addition, the central party wishes to keep their model private. A possible solution would be to encrypt both the data and the model with Homomorphic Encryption and then do the training with Federated Learning.
- Two or more data holders have similar data on different entities, that is, they have different rows of the same dataset (Horizontal Partition). They may wish to jointly train a model on the global dataset as to mitigate the effects of their local biases. They may, for example, use Federated Learning to jointly train a model, and use Differential Privacy to fully ensure that no sensitive data is revealed in any model update.

- Two or more data holders have recorded different features of the same entities (Vertical Partition), that is, they have different columns of the same dataset. They may wish to jointly train a model on the global dataset, with access to all the combined features. They must first align their records; this can be done privately, even when there is not a global ID both datasets share, via private Entity Resolution (ER). A Bloom filter-based approach is normally used to accomplish this task, although some papers have demonstrated that no strict (sample to sample) alignment is required to obtain a good approximation of the optimal model. Once the entities are somehow aligned, the parties can train a model using any of the above techniques.

The Machine Learning models that can be trained are mostly those of traditional Machine Learning. In particular, the following operations are supported:

- **Classification:** Within supervised learning, classification consists in predicting the class, or label, of given data points on the basis of a training data set containing instances whose label is known.
- **Clustering:** Consists in dividing the dataset in several groups of instances with similar features. In this case the instances are not labelled and thus it is a type of unsupervised learning.
- **Regression:** Is the task of modelling the relationship between a dependent variable, or target, and one or more explanatory variables (or independent variables). Like classification, regression is framed within supervised learning, and also requires a training data set to build the model.
- **Dimensionality reduction:** Is the transformation of data from a high dimensional space to a low dimensional space so that the low dimensional representation retains some significant properties of the original data.
- **Anomaly detection:** (or outlier detection) is the identification of rare items, events or observations which raise suspicions by differing significantly from the majority of the data.
- **Neural Networks:** A collection of connected units (artificial neurons) capable of recognizing underlying patterns in a data set. There exist neural networks architectures/algorithms for both supervised and unsupervised learning.

2.2.3 Differential Privacy

Differential Privacy (DP) [12] [13] is a privacy preserving technique in which noise is added to the data to avoid the identification of any individual record when this data is queried through any aggregation function, like mean, sum, variance, etc. This noise is added in such a way that the global dataset retains the maximum of its statistical properties to guarantee a certain degree of privacy for a particular query, which is the key aspect that makes the difference between DP and just "adding noise" to the data. In this sense, differential privacy mathematically guarantees to those who contribute sensitive data to a database that the risk (which will be very low) that their personal information can be discovered through one or successive queries to the database will always remain below a threshold. This guarantee is based on the idea that making queries invariant to the removal of a record from the database makes the data immune to membership and reconstruction attacks, no matter how much side knowledge about a particular individual the adversary has. This is a powerful argument in favour of a user agreeing to share their data in a database, as it ensures that query results will be very similar regardless of whether the user is included in the database or not.

Formally, differential privacy is normally defined as follows: a randomized algorithm K is ε-differentially private if for all data sets D and D' differing on at most one row, and S ⊆ Range(K),

$$\Pr[K(D) \in S] \leq \exp(\varepsilon) \times \Pr[K(D') \in S]$$

While this definition does not create differential privacy, it is a measure of how much privacy is afforded by a query K. Specifically, it is a comparison between running the query K on a database (D) and a parallel database (D'), where parallel databases are defined to be the same as a full database (D) with one entry/person removed.

Thus, this definition says that for all parallel databases, the maximum distance between a query on database (D) and the same query on database (D') will be epsilon.

Intuitively, noise addition masks the differences between the query on the original data set and the set resulting from excluding a record. The main consequence of this is that the result of an analysis that has been carried out using differential privacy is actually an approximation, and not an exact result. Similarly, if the same analysis is performed twice on the same dataset using differential privacy, different results will be obtained, as random noise has been introduced. However, it is usually possible to calculate accuracy bounds for the analysis that measure how much an analysis result is expected to differ from the noiseless result.

In a DP setting, increasing the amount of noise will make the data more private at the cost of the query's accuracy, while decreasing the amount of noise will have the opposite effect. Thus, in DP there is a natural trade-off between privacy and accuracy. This compromise is controlled by the privacy loss parameter (ε) which determines to what extent each individual's information needs to be hidden and therefore how much noise needs to be added. This parameter is considered, in turn, what is called "privacy budget", which will be spent in the successive analyses that are made of the data using differential privacy. If only a single analysis is to be performed, then this analysis can be allowed to exhaust the entire privacy budget, however the most typical scenario is one where multiple analyses are expected to be performed on a dataset, and it is necessary to calculate what total use has been made of the budget through these analyses. Several composition theorems for differential privacy have been developed that state that the composition of two analyses results in a privacy loss that is limited to the sum of the privacy losses of each of them. According to this, if a privacy loss parameter ε = 0.1 has been set, an analyst could perform one analysis consuming 0.1 epsilon, or two analyzes each consuming 0.05, without violating the loss-limiting policy privacy to 0.1.

There are two ways of introducing differential privacy, which refer to the two different places where noise can be added: local DP and global DP. While local DP involves adding noise individually to each of the records in the database before performing a query, global DP incorporates the noise at the end, on the result of an aggregation. In a local DP setting users are more protected, as they do not have to trust the database owner to use their data responsibly. However, if the database operator is trustworthy, global DP leads to more accurate results with the same level of privacy.

A large number of analyses can be performed with differential privacy guarantees. Differentially private algorithms are known to exist for a wide range of statistical analyses such as count queries, histograms, cumulative distribution functions, and linear regression; techniques used in statistics and machine learning such as clustering and classification; and statistical disclosure limitation techniques like synthetic data generation, among many others.

It is worth mentioning that, whereas the roles of data provider and data consumer are clear, the data intermediary is optional (the technique by itself does not require this role). However, one of their functionalities could be to provide a description of the domain of the datasets. In this way, the data consumer can refine the queries in order not to consume the whole budget in one request.

2.2.4 Homomorphic Encryption

In 1978, R.L. Rivest presented a sketch of how to solve the problem of performing operations on encrypted data securely [14] using a loan company as example. In parallel, and after the public key cryptosystem scheme invented by Diffie and Hellman [15], Rivest, Shamir and Adleman revolutionized the world of secure communications by presenting what is commonly known as the RSA cryptosystem, valid both for encrypting messages and for the authentication of them [16].

In these and subsequent investigations, the key concept is to perform operations on encrypted data and not on raw data, and it is precisely this idea that lies behind what mathematicians define as homomorphic encryption.

In a more rigorous way, a scheme is defined as homomorphic encryption if

$$E(m_1) \blacktriangle E(m_2) = E(m_1 \blacktriangle m_2) \; \forall \, m_1, m_2 \in M,$$

where M is the set of messages and E is the encryption algorithm over an operation ▲. This scheme is characterized by four main steps:

- **KeyGen**: This is the process of generating a set of keys that will be used to encrypt and decrypt. It if the encryption and decryption keys are the same the protocol is called symmetric and otherwise asymmetric.
- **Enc**: This the process of transform the input from plaintext to cyphertext. Typically, this is done using the asymmetric public key.
- **Dec**: This is the process of transform the input from cyphertext to plaintext. Typically, this is done using the asymmetric private key.
- **Eval**: This is the most important point. In this step the evaluation is perform over cyphertext and the output must guarantee the preservation of the format, that is, a cyphertext.

A significant effort has been made to find a cryptographic scheme that would allow any type of operation. So much so that it was not until 2009 [18] that it was demonstrated that homomorphic encryption could be valid for any type of operation. Because of this, a taxonomy of homomorphic encryption types has been created:

- *Partially Homomorphic Encryption (PHE)* allows only one type of operation with an unlimited number of times. Example of PHE schemes are:
    - **RSA** [16]: It is homomorphic over multiplication.
    - **ElGamal** [18]: It is homomorphic over multiplication.
    - **Paillier** [19]: It is homomorphic over addition. Additionally, it allows cross-relation between plaintext and cyphertext, i.e., being possible the multiplication between a cyphertext and a plaintext.
- *Somewhat Homomorphic Encryption (SHE)* and *Fully Homomorphic Encryption (FHE)* allows computation over addition and multiplication but a limited (e.g., evaluate a set of circuits with a limited depth) and non-limited number of times, respectively. Recent examples of SHE and FHE are:
    - **CKKS** [20]: This scheme proposes an approximate HE, that is, it does not satisfy the correctness property, meaning that the decrypted result of an operation ▲ is not the same as if we would perform it with plaintexts.
    - **TFHE** [21]: In this work a fast implementation is showed.

    One of the challenges in this type of encryption is the multiplication, and the increase of the dimension of the cyphertext in each iteration. The step to preserve the dimension is known in the bibliography as relinearization.

Nowadays, HE is still a field that is progressing continuously. Namely, one of the concerns of security teams is to know what will happen when quantum computing becomes a reality and what new cryptography schemes will be necessary to guarantee the security of our communications.

2.2.4.1 Relevance to privacy-sensitive data-sharing scenarios

HE works always with the idea of sending encrypted datasets to carry out encrypted operations over them. This idea fits with Figure 2.1 (right) where there is a group of data-consumers that want to make analysis without revealing the raw data. Namely, one the most cited applications is the outsourcing of computational resources.

In this scenario, the data provider and the data consumer may belong to the same organization, but they do not have enough computational resources, so they have to delegate the computation to an external provider such as a cloud service. The point here is that once that data is outside the security controls of the data holders there is a clear risk of information leakage, as many times we do not know what interests are behind these external providers.

2.2.5 Anonymization

Data anonymization [22] [23] [24] is known as the set of models and techniques aimed at protecting personal or private information in a data collection (relational, graph-oriented, etc.) using procedures that alter, delete or encode the identifiers that either directly reveal personal information or allow establishing a relation between the information in the database and specific persons or entities.

Anonymization allows organizations to share data to improve processes or create new business opportunities. For example, in the field of medical research it is an essential procedure for sharing patient information that has led to important scientific advances. Anonymization, while not perfect, minimizes the risk of disclosure of sensitive information in order to comply with strict privacy regulation (like GDPR), at the same time that preserves data credibility and consistency to carry out statistical analyses or build machine learning models.

There are multiple data anonymization techniques, which have the same common principle: to achieve privacy by preserving, on the one hand, the utility of the data, that is, minimizing the loss of information, and on the other, truthfulness, in the sense that each anonymized record corresponds to a single record of the original table.

An important concept in anonymization before going on to review the different techniques is that of quasi-identifier (QID). Quasi-identifiers are attributes, like zip, sex or date of birth that, despite they do not allow to identify a person on their own, when combined can be joined with information obtained from diverse sources (e.g., public voting registration data) in order to reveal the identity behind individual records.

Some techniques of data anonymization are:

- **Generalization**: Replaces QIDs by other less specific values, for example grouping numerical values by ranges or replacing different categorical values by the same label. Although generalization entails a considerable loss of information, which decreases data utility, there are several methods that reduce information loss.
- **Suppression and relocation**: These techniques are used to reduce over-generalization. While suppression consists of eliminating outliers, relocation simply modifies their QIDs, but without eliminating them. In general, the existence of outliers leads to over-generalization, since these points are distant from other records and since there are usually not too many of them, they cannot be grouped in the same equivalence class. With these methods the outliers are eliminated or changed.

  Suppression can be accomplished in two different ways: by deleting the entire record or by deleting only the QIDs that cause a tuple to violate privacy constrains, which reduces information loss.
- **Perturbation**: Consists of replacing sensitive QIDs with modified values. Some specific alteration strategy is usually followed, such as character shuffling, encryption, term, or character substitution, which makes identification by reverse engineering difficult. This technique is useful for preserving the utility of the data since the degree of generalization is limited by the use of fake records instead of equivalence classes like in generalization.
- **Bucketization**: Consists of partitioning the tuples into buckets and then separating the sensitive attribute from the QIDs by randomly permuting the values of the sensitive attributes in each bucket. The sanitized data then consists of buckets with permuted sensitive values. This way better utility than generalization is achieved. However, because bucketization publishes the QID values in their original forms, an adversary can find out whether an individual has a record in the published data or not, and therefore does not prevent membership disclosure.
- **Slicing**: This technique aims at ensuring that highly correlated attributes are grouped together by partitioning the data both horizontally and vertically. First, vertical slicing is performed by grouping attributes into columns based on their correlations. As a result, each column contains a subset of attributes highly correlated. Then performs horizontal slicing by grouping tuples into buckets where each bucket contains a subset of tuples. The main idea is to break the association between columns but retain the association within each column. This way the data dimensionality is reduced while preserving data utility better than bucketization and generalization.
- **Synthetic data**: Consists of the algorithmic creation of artificial data with no direct relation to the records whose privacy is to be preserved, but generated from mathematical models derived from the patterns and statistical properties of the original dataset.

An anonymized database is susceptible to what is known as a re-identification attack, which consists of attempting to trace the supposedly anonymized records to the records of another database or related data source in order to extract confidential information from it.

To address reidentification attacks, the k-anonymity model was proposed: a data set is said to be k-anonymous ($k \geq 1$) if for every record there are at least $k - 1$ other records indistinguishable to it on the quasi-identifier attributes. Records with identical quasi-identifier values constitute an equivalence class, and each

individual remains anonymous within it. The larger the value of k, the better the privacy is protected. k-anonymity is usually implemented through generalization or suppression, which implies information loss. However, because data should retain as much information as possible to remain useful for analytics, a trade-off between privacy and accuracy arises at this point. In addition, it has been demonstrated that a lack of diversity in the sensitive attributes (e.g., all persons suffer from the same disease) leads to severe privacy issues.

Luckily, the concept of l-diversity comes here to the rescue by considering diversity among sensitive attributes. l-diversity prevents uniformity and background knowledge attacks by ensuring that at least l sensitive attributes values are well-represented in each equivalence class (e.g., the probability to associate a tuple with a sensitive attribute value is bounded by 1/l)

In a context where Artificial Intelligence and Machine Learning have gained great importance in recent years, the need to share data to feed algorithms becomes increasingly important. However, the existence of sensitive data does not allow many databases to be shared in raw, making anonymization, due to its relative simplicity and ease of implementation, normally the first choice to prevent the disclosure of sensitive data and comply with GDPR.

In this scenario various algorithms and techniques like classification, clustering, regression, neural networks, association rules, decision trees, genetic algorithm, nearest neighbour method etc., are used for knowledge discovery from anonymized databases. However, due to the relatively high probability of re-identification, especially in high-dimensional databases, and the high cost in terms of information loss incurred to prevent it, other methods presented in this document are usually preferred.

2.2.6 Trusted Execution Environment

Trusted Execution Environment (TEE) is defined as a secure area inside a main processor [25] , that is, the code and data executed within a TEE is guaranteed to be secure against unauthorized entities from altering what is running.

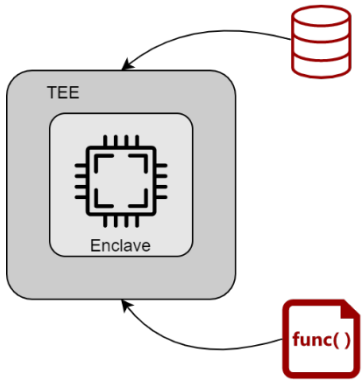

Figure 2. 3 Trusted Execution Environment

To illustrate what a TEE is, let us consider a mobile phone. In these devices there are several applications which provide different functionalities, some of them related with mobile wallets with access to our bank. It is especially important that the information managed by this application is secure, but not only in terms of the application itself, but also in its interaction with the rest of the apps that we have installed. There may be others with malicious intent that are trying to attack and alter their original behaviour. The isolation of this app with the rest of the word is what TEE provides (it can be also found in the bibliography as “enclave”).
An important feature of a TEE is that a remote user can receive attestation that the enclave has not been modified, so what is running is exactly what is expected [26].

We have not entered in detail about the current implementations; however, a TEE is not something that can be developed, it is provided by integrated circuit manufacturers such as ARM Trustzone, AMD Secure Processor, or the best-known Intel Software Guard Extensions.

TEE applies to every data-sharing scenario because it is not a proper technique by itself like the rest of the ones that are explained in this document, it is more a software/hardware solution that tells you that if you run your app in a specific way, what is executed is what you expected.

Seen in this way, it could be said that TEEs provide an additional layer of security to those discussed here, regardless of the privacy techniques that may be implemented and the mathematical protocols behind them. TEE is agnostic to data privacy and what it guarantees is that the data has not been modified by any external agent, without entering whether the data is sensitive or not.

For illustration purposes, let us consider a TEE combined with other PET technique such as HE. We have already addressed that HE is suitable for outsourcing of computation. Whereas this idea is a clear advantage of HE, it is not out of threats because even if our data is encrypted via HE, this encrypted data could be modified by this external entity. If this happens, once that the calculation is carried out, the decrypted result would be wrong. Here is where TEE could appear, providing a secure layer that guarantees that the data has not been modified.

2.2.7 Zero Knowledge Proofs

Zero Knowledge Proofs [27] are a cryptographic method that allows one party (called the prover) to prove the validity of a statement to another party (called the verifier) without revealing the secrets that make this statement true. For example, the prover might wish to know a secret password without actually revealing it. If the prover really knows the password, and both parties follow the protocol, the verifier will be convinced (**Completeness**); if the prover is lying, and the verifier follows the protocol, the verifier will with high probability detect the lie (**Soundness**); finally, the verifier will not learn the password, or any other information that would allow them to impersonate the prover (**Zero-knowledge**).

This is an umbrella term under which covers many related technologies, each with different trade-offs:

- **Supported statements**: Some methods support arbitrary arithmetic circuits, making them highly versatile. Other methods achieve efficiency gains by specializing in specific types of statements, such as the satisfiability of instances of a particular NP problem or the validity of another zero-knowledge proof.
- **Interactive/Non interactive**: The original formulation requires interaction between prover and verifier, in the form of challenge-response messages. Newer advances allow for fully local generation and verification of proofs, at the cost of increased proof size, computation time, and/or trust in third parties [28]
- **Trusted setup**: Some methods require a trusted third party to setup the computation. This requirement can be sidestepped with the use of other technologies such as SMPC, at the cost of lower efficiency.
- **Efficiency**: Both the prover and verifier's complexity must be considered, as well as the communication complexity between them. Each method has its own trade-off between these three quantities, and the choice will depend on the specific use case; for example, if proofs are verified more often than they are checked, the chosen method should shift the computational complexity to the prover.
- **Cryptographic assumptions**: Most methods rely on standard cryptographic assumptions (e.g., hardness of factoring). However, some widely accepted and used methods such as SNARK, require nonstandard assumptions. Some other methods are based on stronger assumptions than usual, making them post-quantum secure.

While all applications are related to user privacy and verifying computations or knowledge, the difference in supported statements gives rise to different technological applications. These have mostly been motivated by blockchain but also are of independent interest:

- Verify that a given data point is included in a dataset, without revealing the dataset. This can be achieved efficiently with the use of Merkle Trees, with the proof size being logarithmic in the dataset size [29]
- Securely prove one's identity (i.e., knowledge of a secret password) without needing to trust the identity checker [30]

- Proving some arithmetic relation over private data without revealing the data itself. For example, a data owner might wish to prove the truth of a given statistical aggregate without revealing their dataset, or a model owner might wish to prove its accuracy on a public dataset without disclosing the model [31]
- Verifiably execute an arbitrary computation. This verification is much faster than executing the program (in some specific cases, such as executing the same program with different inputs to amortize a per-program setup cost), making it a suitable tool for verifying outsourced computations: for example, it could be used to verify that a model has been correctly trained (i.e., no data poisoning or backdoor attacks have been added). However, the cost to the prover is immense, making this technology still unsuitable for realistic use cases [32].

Data sharing scenarios are partly defined by the terms governing which data can be accessed by each party, and which computations are allowed. Zero-Knowledge Proofs offer a technological trust-free way to enforce and verify that contractual obligations are met. This includes user verification, allowing trustless transactions of datasets or models, and verifying that an outsourced computation took place as expected. This verification might be performed by the data holders, or by a trusted intermediary such as the data intermediary role defined.

However, it can add significant overhead, making it unsuitable for large scale datasets or computations. In other cases, it might be rendered unnecessary by other technologies (e.g., some SMPC methods include their own verification schemes).

Given the variety of applications, adoption of this technology must be carefully considered for each scenario. It will be most relevant on scenarios where auditing would otherwise require access to sensitive data.

2.3 Conclusion

We have addressed a set of techniques that guarantees data privacy via different mechanisms, most of them based in cryptography, but also via statistic methods or hardware solutions such as differential privacy and trusted execution environments, respectively.

Different data sharing scenarios have also been addressed, showing the main conclusion: there is no single PET that solves all privacy issues or covers all possible use cases. Therefore, it is crucial to know the data sharing scenario to determine which technique is the most appropriate.

Another conclusion is that privacy does not come for free. It often requires sacrifices, which usually involves having to find a compromise been privacy, accuracy, and performance. These sacrifices can be reduced by combining some of the techniques. For example, SMPC and HE guarantees privacy of the inputs, that is, neither organization learns anything in the computation flow but they do not say anything about the output (the reader can think in a SMPC average aggregation with only two parties, if the output is analysed by one of the two parties, it can learn the input of the other one because SMPC guaranteed the inputs, no what can be learnt from the output, and for specific use cases this is a privacy leak). In this scenario SMPC could be combined with DP to aggregate a statistical noise to the output (sacrificing the accuracy of the result).

3 PET Evaluation

This section analyses the state of the art of privacy-enhancing techniques (PET). For each of the relevant technologies, we:

- Evaluate its relevance to data sharing scenarios.
- Evaluate its degree of usability by creators and users of data spaces.
- List and analyse real-world use cases enabled by that techniques.
- Explain the level of security achieved, enabling data holders to take informed decisions.
- List and evaluate existing implementations and references.
- Provide and justify the technology's readiness level (TRL).

Overall, the section aims to serve as a quick but detailed reference for data holders, data subjects, data analysts and data intermediaries when evaluating a technique's applicability to any data sharing scenario.

There is a wide variety of Privacy Enhancing Techniques. While all of them aim to protect the data subject's privacy, each has its own unique drawbacks and strengths. Understanding the state of the art is a key step in designing and regulating data spaces. Before accepting and adopting data spaces, participants must understand the risks incurred and the potential benefits.

3.1 Secure Multi Party Computation

Secure Multiparty Computation (SMPC) is a technology which allows a group of data holders/subjects to jointly compute a function on their combined private inputs, without revealing anything other than the result. It is a key enabling technology for privacy preserving computations: it allows the group to act as a trusted third party with access to all the group's private data, without any of the individual members seeing any data besides their own [6].

As a toy example, SMPC enables a group of friends to compute the group's average age, or find the oldest one, without any of them revealing their age. Realistic use cases include:

- **Statistical analysis of distributed datasets**: In a healthcare use case involving several hospitals, each hospital has data on their patients but is unable to share it with others to obtain statistical insights form the global dataset. SMPC enables such a collaboration, while keeping the datasets private. Possible computations include:
  - All common statistical analysis, such as the average, standard deviation, maximum, etc.
  - Specialized analysis, such as the Kaplan-Meier survival curve [33].
- **Record linkage**: In the financial domain, fraud is easier to detect when the same fraudulent customer can be traced through several banks. SMPC enables several banks to find which problematic customers they have in common, without revealing any information about any other customer, using Private Set Intersection [34].
- **Encrypted Inference**: SMPC enables a data owner and a model owner to jointly compute the result of applying the model to the data, without either the data or the model's parameters being revealed to the other party. For example, a hospital could detect genomic diseases for a patient, without the patient's genome being revealed or the patient gaining access to the hospital's model [35].

In general, SMPC is useful for all applications in which a group would benefit from pooling their data but have legal barriers or incentives preventing them from sharing it. In particular, SMPC is well suited to the data sharing scenario of Figure 2.1 (left): Data providers have a common interest, and they will actively participate in the computation. In this case, each data provider ensures their own safety: they only need to trust themselves to know the data is secure (assuming strong security parameters).

SMPC can also be used for the other scenario: data providers can outsource their data to a SMPC cluster acting as a trusted third party. While this has the risks associated with moving data, it mitigates them: the data is encrypted before leaving the data provider's premises, and the distributed nature of the SMPC cluster removes the single point of failure inherent in a single-node third party. The technical and computational

costs of running a SMPC node are shifted from the data provider to the data intermediaries running the cluster, at the cost of having to trust the data intermediaries to collude.

3.1.1 Effectiveness & Usability

The main drawback of SMPC technology is the performance overhead: computations will be several orders of magnitude slower than operating on unencrypted data, due to the communication cost inherent in SMPC protocols. This performance cost is non-trivial and must be considered: SMPC is currently not suitable for training large-scale machine learning models, operating on terabyte-sized datasets, or complex real time applications. However, most data-sharing scenarios consist of one-off computations, simple statistical queries, inference of large-scale machine learning models, and analysing gigabyte-sized datasets. For these common applications, SMPC is efficient enough to be used in practice, making it an effective technology.

The effectiveness of SMPC has been shown in several real world uses:

- The **Danish Sugar Beet Auction** was performed in 2008, using SMPC to run a double auction while keeping individual bids private. SMPC proved to be cheaper and more practical than hiring an external consultancy to act as a trusted third party. The computation was performed between 3 parties: the buyer, a group representing all sellers, and the data intermediary responsible for setting up the computation, as in the data sharing scenario of Figure 2.1 (left) [36].
- **IKT erialade tudengite töötamine** (employment of ICT students), a 2015 social study by the Estonian Government, had the goal of finding if there was any relation between the high dropout rate of college students enrolled in ICT and the early hirings of the ICT industry. To answer this question, two different datasets had to be linked: 10 million tax records from the Ministry of Finance, and 600.000 education event records from the Ministry of Education. To circumvent the legal restrictions that prevent these organizations from sharing the data between themselves, the data was encrypted between 3 parties: an association representing ICT universities, the Ministry of Finance, and the SMPC platform owner, which acted as a data intermediary. Note that most of the data preprocessing and transformation was done after encrypting the data, reducing the data provider's technical burden but increasing the computation's complexity. The technical report offers a detailed account of the challenges present when bringing SMPC into practice [37].
- The **Boston Women's Workforce Council 2017 Report** analysed data from 114 companies to assess the gender wage gap in the Boston area. The scope and detail of the study wouldn't have been possible without SMPC, since it required analysing sensitive data from competing companies. There was a strong focus on usability: data holders/subjects encrypted and uploaded their data via the browser, without needing to download or configure anything, or to stay online during computation. Once encrypted, the data was processed by the data intermediary orchestrating the computation and the data analyser which gets the decrypted results. Note that the protocol used only supports additive aggregation, instead of arbitrary operations. This has important practical advantages: its simplicity aids comprehensibility and trust, reduces the technical cost of implementation, and improves performance [38].

  As these examples show, SMPC is an effective technology for enabling privacy preserving analysis in a data sharing environment. The technical reports give us a glimpse into SMPC's usability:

- SMPC is usable without specialized knowledge. Data analysts can use existing frameworks to express their computations in a high-level language, without needing to implement or even understand the underlying complexity inherent in SMPC protocols. Data holders only need to understand the security assumptions behind the chosen protocol, and optionally take part in the computation. Most of the burden falls on the SMPC platform operator, which deploy the technical solution, coordinates all the computation nodes, and optionally acts as an extra node. Data intermediaries might either act as platform operators or as intermediaries between platform operators and the data holders & analysts.
- For realistic use cases, SMPC's performance is not a major roadblock. While computations might take minutes instead of seconds, this cost is dwarfed by the weeks or months required to set up any study, with or without SMPC technology.
- There are legal, bureaucratic, and technical barriers to processing sensitive data. The mathematics behind an SMPC protocol ensure that no data will be revealed. However, this is far from enough to

allow sidestepping the legal barriers protecting private data: the security claims must be confirmed by cryptographic experts, the code implementation must be audited, and legal departments must ensure that data protection laws are respected. This process will hopefully get easier with more awareness of SMPC and more standardized solutions, but it is currently a time-consuming obstacle.

- SMPC expects all data providers to have clean, numerical, preprocessed data, with the same clearly defined scheme. This is rarely the case in practice; one of the key duties of the data intermediary is to enable this standardization. Note that it i's also technically possible to clean or preprocess data once encrypted, thanks to SMPC's computing flexibility. While useful for simple adjustments, such as subtracting the mean value from a column, complex ad hoc transformations should be avoided in practice since they will require operating blindly or revealing data.

3.1.2 Security

SMPC offers strong security guarantees based on standard cryptographic assumptions. In contrast to standard centralized processing, an attacker must compromise a majority of (or all) computing nodes to disrupt the computation or obtain private data. While all protocols offer a baseline level of security, they make different assumptions on the power level of an attacker and the number of expected corrupted nodes. The level of security will depend on these choices:

- **How many data holders will take place in the computation?** A higher number of nodes makes it harder to attack a majority of them, while a lower number makes it easier to audit and trust all nodes. Some protocols are specialized for 2, 3, or 4 parties, while others allow an arbitrary number. Note that 2 data holders can still perform a 3-party protocol, by employing a non-colluding third party to aid as a computation node.
- **How many participants are necessary to complete the computation?** Choosing a number lower than the number of parties will make the computation tolerant to faulty nodes going offline, at the cost of lowering the number of parties an attacker must corrupt to access private data. Realistic use cases usually have a small number of parties and do not require critical availability, so most protocols require all parties to be present to avoid the increased risk of attack.
- **What severity of attack is expected?** Honest participants will follow the protocol and not collude by sharing their private intermediate results. However, an attacker might compromise some of the nodes, gaining access to their data (honest-but-curious attacker) or deviating from the protocol (malicious attack). Protocols range from resisting 1/3 of corrupted nodes, to resisting all nodes but one being corrupted. Resistance against more powerful attacks has a performance cost.
- **Is there a trusted third party that will not collude with any data owner?** Some protocols have a costly pre-processing phase, which improves the performance of the actual computation. A "trusted dealer" can make these protocols more efficient, at the risk of adding a single point of failure.

This flexibility of hyperparameters is a two-edged sword. These choices are specific to each project and require both domain knowledge and a mild degree of cryptography knowledge to answer effectively. While the data intermediary might offer expertise and recommended configurations, data holders should understand these choices before engaging in the computation, since their privacy depends on them.

Note that SMPC only ensures the privacy of the computation; an attacker might still extract sensitive information from the computation's output or manipulate the input data to generate an incorrect result. These attacks aren't specific to SMPC.

3.1.3 Implementations and References

There are several implementations, both open source and proprietary, of SMPC technologies. They range from academic implementations of cryptographic protocols to production-ready applications focused on real world deployment and usage. Overall, the existing implementations make it possible to use SMPC without specialized cryptographic knowledge.

At the time of writing, these are some of the most interesting solutions, roughly ordered from easier to use to more focused on real world deployment. Unless specified otherwise, they are open-source:

- **MPyC (MIT License)** is a Python package suitable for local prototyping SMPC computations. It features a high-level interface mimicking standard python operators, extensive documentation, and several examples of adapting common algorithms to a SMPC setting. It's easy to install and the examples can be tried online, making it a perfect fit for getting familiar with SMPC [39].
- **MP-SPDZ (MIT License)** is a benchmarking-focused academic framework. It features reference implementations of 34 different SMPC protocols, as well as a high-level language based on python for seamlessly switching between them. Protocols cover the whole range of security choices, making this library a valuable resource to benchmark the performance cost of different options without needing specialized cryptographic knowledge [40].
- **CrypTen (MIT License)** and **tf-encrypted (Apache License 2.0)** are two unrelated frameworks with a common focus on usability by machine learning specialists, by mimicking PyTorch and TensorFlow interfaces respectively. While not yet production ready, they are useful tools for prototyping and benchmarking complex machine learning models, while working with a familiar interface [41] [42].
- **JIFF (MIT License**) is an open source, production-ready, JavaScript library for building web-based SMPC applications. It features high level abstractions to avoid direct contact with the SMPC protocols. It's highly usable, since it can be integrated within existing applications, runs in any JavaScript environment (browsers, mobile, & servers), and is resistant to nodes going offline [43]. It has been successfully used in the real world [38].
- **FBPCF (MIT License)** is a SMPC C++ library ready to be deployed on the cloud. Originally created for running randomized controlled trials without revealing user data, it allows custom computations with a moderately abstract layer [44] [45].
- **Sharemind (closed source)** is a production-ready proprietary platform for running secure multiparty computations. Programs are written in a custom C-like language which gets compiled to SMPC protocols. [46]. It has been successfully used in the real world [37].

Due to the nature of the static document, the above list might be outdated; we refer the reader to the living documents of [47] and [48] for updated references.

Two standards exist for SMPC: IEEE 2842-2021 includes a technical framework for SMPC, security levels, and example use cases. ISO/IEC DIS 4922-1 defines all the relevant properties for an SMPC protocol, with the intention of using these to standardize specific protocols.

3.1.4 Conclusion

To sum up, we briefly summarise the upsides of SMPC technology:

- Allows aggregating data from several data holders, even if they distrust each other.
- Offers strong security guarantees, based on traditional cryptographic assumptions.
- Has a high hyperparameter flexibility, allowing to tailor the balance between security and performance.
- Can perform arbitrary computations on the input data, revealing only the end result.
- Has a vibrant development ecosystem with a wide variety of open-source frameworks.
- Does not require data holders or subjects to have a deep knowledge of SMPC.

On the other hand, SMPC has the following limitations:

- Performance will be several orders of magnitude worse (4~6) than operating on unencrypted data.
- Computation nodes require a high bandwidth, low latency connection during the computation.
- All participant nodes must agree beforehand on the exact computation to be performed.
- For most cases, data must be preprocessed, clean, and numerical before uploading.
- Data holders require some degree of specialized knowledge to choose the right privacy settings.
- Only protects the computation itself; sensitive data might still be extracted from the result.

Overall, we consider SMPC to be a key technology for privacy aware data processing. While not yet widely available in production environments, it has been used for real-world use cases, giving it a **Technology Readiness Level of 6~7**.

3.2 Federated Learning

Federated learning (FL), by having a larger set of training samples available from multiple organizations, allows better models to be created than if each federated entity were trained only on its own data. For example, in medical research this is especially advantageous considering that a hospital's patient data and images are obtained from a specific subset of the population, and they are therefore unlikely to have been seen or shared with other hospitals. Moreover, if the hospitals in the federation are located in different geographic regions, patient traits differ substantially: the sex ratio, age distribution and ethnicities of the patient populations may differ, which introduces diversity into the training set and thus increases its generalization capabilities. Federated learning also makes a difference when dealing with atypical patients. If the model is built with data from only one hospital, a few outlier cases will be treated as noise and will not be captured by the model, as it will not have the opportunity to see enough cases, but in a federated environment, where data availability is greater, these rare cases could be integrated into the model and, by generalizing better, increase its diagnostic capability. This type of cooperation therefore enables the advancement of precision medicine. However, the advantages of FL can be extrapolated to other domains. An example is the detection of banking fraud. In general, cases of fraud are exceptional compared to the total volume of operations, and are not sufficient to develop a model, so this problem is usually addressed from an anomaly detection approach, which presents major shortcomings, but in a federated environment where the learning algorithm can see enough cases of fraud can develop a model that captures the fraud in an operation, leading to greater accuracy than the mere detection of anomalies.
However, when referring to federated learning we do not have to think exclusively of organizations that collaborate by providing their data repositories and their servers (what is known as cross-silo FL). Another common way -if not the most common- of doing FL is cross-device FL, where clients are a very large number of mobile or IoT devices. There are several examples of successful applications of cross-device FL in consumer digital products, including the following:

- **Google's Gboard**, a virtual keyboard for smartphones which suggest the next word while typing a text thanks to a recurrent neural network model trained in millions of phones with a miniature version of TensorFlow. The training only happens when the device is idle, plugged in on a wireless connection to avoid any impact on the phone's performance.
- **Messages** (formerly known as Android Messages) is an SMS, RCS, and instant messaging application developed by Google for its Android mobile operating system. In addition to using FL to improve its features, Google also claims to use secure aggregation to protect users' contributions to the global model.
- **Now Playing** feature on Google's Pixel phones -a tool that shows what song is played-, when recognizes a song, records the track name into the on-device Now Playing history, where users can see recently recognized songs and add them to a music app's playlist. Later, when the phone is idle, plugged in, and connected to WiFi, Google's federated learning and analytics server may invite the phone to join a "round" of federated analytics computation, along with several hundred other phones. Each phone in the round computes the recognition rate for the songs in its Now Playing History, and uses the secure aggregation protocol to encrypt the results. The encrypted rates are sent to the federated analytics server, which does not have the keys to decrypt them individually. But when combined with the encrypted counts from the other phones in the round, the final tally of all song counts (and nothing else) can be decrypted by the server. The result enables Google engineers to improve the song database (for example, by making sure the database contains truly popular songs), without any phone revealing which songs were heard.
- Apple also uses cross-device FL, in this case together with DP, in iOS 13, for applications like the **QuickType** keyboard and the vocal classifier for **Siri**.

Also, edge federated transfer learning methods can be applied to personal health measurement devices [49]. Some personal healthcare devices, such as blood pressure monitors and activity recognition devices, are used to observe health conditions and raise health alarms, which plays an important role in smart health systems. For users, it is necessary to have a pre-trained model at the beginning and train a personalized model (fine tune) updated by their physical conditions in real time.

3.2.1 Effectiveness & Usability

A few aspects to consider when implementing a FL system are listed below:

IID data
Having Independent and Identically Distributed (IID) data is always desirable in a FL environment. Formally, IID at the clients means that each mini-batch of data used for a client's local update is statistically identical to a uniformly drawn sample (with replacement) from the entire training dataset (the union of all local datasets at the clients). However, since the clients independently collect their own training data, which vary in both size and distribution, and these data are not shared with other clients or the central node, the IID assumption almost never holds in practice. It turns out that the federated averaging (FedAvg) algorithm which is the most common technique for aggregating local models into a global model, fails to achieve a satisfactory model and system performance when the datasets produced by different clients are not independent and identically distributed (Non-IID) and the communication cost is high, so challenges arise when training federated models from data that is not identically distributed across devices, both in terms of modelling the data and analysing the convergence behaviour of associated training procedures.

Efficient Communication across the federated network
In cross-device FL, communication is a bottleneck, and it is necessary to use efficient communication methods that minimize information transfer. To achieve this, two possibilities arise: (1) reducing the total number of communication rounds, or (2) reducing the size of transmitted messages at each round. Local updating methods allow for a variable number of local updates to be applied on each machine in parallel at each communication round, which in turn reduces the total number of communication rounds. Model compression schemes can significantly reduce the size of messages communicated at each update round. Also, decentralized topologies are an alternative when communication to the server becomes a bottleneck, especially when operating in low bandwidth or high latency networks.

Systems Heterogeneity
Due to variability in hardware (CPU, memory), network connectivity (3G, 4G, 5G, WiFi) and power supply (battery level), differences in storage, computing and communication capabilities of devices across the federated network may occur. Moreover, only a few of them are available at a time and, even then, not all of them are reliable, as it is not uncommon for an edge device to go down due to connectivity or power issues.
For this reason, it is essential for a federated learning platform to be fault tolerant, as participating devices may drop out before completing a given training iteration. Therefore, federated learning methods have to be developed in such a way that they (1) anticipate a low amount of participation, (2) tolerate heterogeneous hardware, and (3) are robust against devices dropping out of the network. For example, asynchronous communication can be used to enable parallelization of iterative optimization algorithms, which will prevent slow devices from blocking an entire training round.

3.2.2 Security

FL protocols may contain vulnerabilities for both (1) the (potentially malicious) server, who can observe individual updates over time, tamper with the training process and control the view of the participants on the global parameters; and (2) any participant who can observe the global parameter and control its parameter uploads. For example, malicious participants can deliberately alter their inputs or introduce stealthy backdoors into the global model [43]. This means that an adversary might attempt to prevent a model from being learned at all, or they might attempt to bias the model to produce inferences that are preferable to the adversary (poisoning attacks). On the other hand, observations of model updates can be used to infer a significant amount of private information, such as class representatives, membership as well as properties associated with a subset of the training data. Even worse, an attacker can infer labels from the shared gradients and recover the original training samples without requiring any prior knowledge about the training set.
To address poisoning attacks differential privacy [12] can be used, as in [50] and [51]. Data poisoning can be thought of as a failure of a learning algorithm to be robust: a few attacked training examples may strongly affect the learned model. Thus, one natural way to defend against these attacks is to make the learning algorithm differentially private, improving robustness. Intuitively, an adversary who is only able to modify a few training examples cannot cause a large change in the distribution over learned models. While differential privacy is a flexible defense against data poisoning, it also has some drawbacks. The main weakness is that noise must be injected into the learning procedure. While this is not necessarily a problem—common learning

algorithms like stochastic gradient descent already inject noise—the added noise can hurt the performance of the learned model.
Client-level differential privacy can also be used to prevent any client from trying to reconstruct the private data of another client by exploiting the global model [52]. Client-level differential privacy is achieved by adding random Gaussian noise on the aggregated global model that is enough to hide any single client's update. In addition, we propose another defense technique: secure aggregation. It is a functionality for n clients and a server. It enables each client to submit a value (often a vector or tensor in the FL setting), such that the server learns just an aggregate function of the clients' values, typically the sum. Multi-party computation SMPC [6] can be used to carry out this aggregation, in such a way that the server aggregates *shares* of each client's contribution to the model instead of their clear values.

3.2.3 Implementations & References

There are a variety of federated learning frameworks that provide the basic algorithms and infrastructure required to perform PPML. Some that stand out from the rest are:

- **PySyft/PyGrid (Apache License 2.0)** is an open source developed by the OpenMined community. PySyft is a library that supports federated and secure learning on PyTorch, with support for Deep Learning algorithms, while PyGrid is the platform that allows to set up a peer-to-peer network to exploit PySyft's functionality (https://github.com/OpenMined/PySyft).
- **TensorFlow Federated (TFF) (Apache License 2.0)** is an open-source framework, developed by Google, for machine learning and other calculations on de-centralized data. TFF has been developed to facilitate open research and experimentation with federated learning (https://www.tensorflow.org/federated).
- **FATE (Federated AI Technology Enabler) (Apache License 2.0)** is an open-source project initiated by Webank's AI Department to provide a secure computing framework to support the federated AI ecosystem. It implements secure computation protocols based on homomorphic encryption and multi-party computation (MPC). It supports federated learning architectures and secure computation of various machine learning algorithms, including logistic regression, tree-based algorithms, deep learning and transfer learning (https://github.com/FederatedAI/FATE).
- **uTile PET (closed source)** a secure calculation and federated learning platform based on the PySyft/PyGrid framework. uTile allows for training a machine learning model in a collaborative manner between various parties without the data leaving where it is stored, additionally guaranteeing data privacy by means of multi-party computation and/or differential privacy techniques. uTile enables PySyft/PyGrid for production environment and enhances it with a data loading utility and allows the users to incorporate schemas to properly handle dataset feature headers across the participating entities. While it is mainly focused in horizontal federated learning at this moment, uTile can carry out clustering on vertically partitioned datasets, and it is planned to incorporate soon more algorithms to the vertical federated learning setting (https://www.gmv.com/en-es/products/utile).
- **Flower (Apache-2.0 license)** is an open-source project agnostic to which library/framework is used for training, focusing on the aggregation strategies of the local models. It also provides an interface for training on devices (https://github.com/adap/flower).
- **Fed-ML (Apache-2.0 license)** is very similar to Flower, focusing on the deployment in production environments (https://github.com/FedML-AI/FedML).

3.2.4 Conclusion

Federated Learning is of great interest in research and new techniques and algorithms are constantly being presented, as well as new frameworks and use cases in different application domains. However, while FL on devices has proven to be an indisputably successful technique, which has been commercially exploited, in its so-called cross-silo version (including vertical FL) the results do not seem to have been transferred to production environments with the same ease. Although there are a variety of possible causes, we dare to guess the following:

1) These types of FL deployments have not had the support of the technology giants such as Google or Apple, who have preferred to focus on FL on devices.
2) The difficulty of homogenizing data between different organizations poses a problem that cross-device FL does not have, as Google/Apple create the applications that collect/contain the data.
3) The reluctance of organizations with large volumes of data whose privacy is vital (think of medical records or bank account movements) towards a novel technology that "promises" not to reveal private information, but which is not impregnable, particularly if it is not complemented by other PETs such as SMPC or DP.

A **TRL of 8-9** can be assumed, since at least in its cross-device mode this technology has not only gone beyond mere prototypes but has even been commercially exploited.

3.3 Differential Privacy

Differential Privacy (DP) addresses some of the limitations of previous approaches like k-anonymity. In the words of its authors [12], Differential Privacy promises to protect individuals from any additional harm that they might face due to their data being in the private database x that they would not have faced had their data not been part of x. For example, this may contribute to the advancement of medical science to the extent that it facilitates the availability of datasets (which would otherwise be private, and thus, not accessible) as well as encourages patients to participate in clinical studies. In addition, DP can help companies comply with data privacy regulations, such as the GDPR and CCPA, without undermining their ability to analyze their customers' behavior, and therefore generate value from data insights that would otherwise be unfeasible without a privacy protection technique.
However, the application domain of DP is not limited to statistical queries. Sometimes the private dataset is used to train a ML model. In this case DP plays a fundamental role to protect privacy by adding noise while ensuring that the model still gains insight into the overall population, and this way provides predictions that are accurate enough to be useful, while makes it tough for the adversary to make any sense from the data queried. This applies not only to centralised learning; in fact, Differential Privacy is an essential complement to enhance the privacy of other PET technologies such as Federated Learning.
To this day, companies such as Google, Apple and Uber, as well as government agencies such as the US Census Bureau, have implemented various real-world implementations of differential privacy.

3.3.1 Effectiveness & Usability

There are a few aspects to consider when implementing a DP system, which are explained in the following:

Utility

Utility evaluates the quality of a query output. One popular approach to measure utility in the context of differential privacy is (α, δ)-usefulness, defined as follows:

A mechanism is (α, δ)-useful if every query output is within α of the correct output with a probability of at least 1-delta, i.e.:

$$\Pr[|K_f(D) - f(D)| \leq \alpha] \geq 1 - \delta$$

Where K is a randomized function used to respond to query f.

Setting epsilon

Setting a value for epsilon is not an easy task, as there is no way for ordinary data holders to figure out the exact level of privacy protection of a dataset given by a specific epsilon value. Furthermore, this question is not adequately covered in the Differential Privacy literature.
While there exists no experimental evaluation to guide the user on choosing an appropriate epsilon value, authors normally suggest values in the range (0,1), sometimes greater than 1, but always lower than 10. In a recent attempt at finding general guidelines for setting appropriate epsilon values, the authors in [53] found that epsilon cannot be defined in general but will always depend on the dataset in question. Similarly, the authors in [54], [55], state that, given an epsilon value, the probability of re-identification is not fixed, it rather depends on data values in the data set and even on data for individuals outside the data set.

Sensitivity

In addition to epsilon, sensitivity is another parameter that comes into play to regulate the amount of noise introduced when querying with differential privacy. To determine sensitivity, the maximum of possible change in the result of a query when removing an individual from the database needs to be calculated. While properly set sensitivity guarantees that the query meets the ε-DP requirements, adhering strictly to this theoretical concept (whose value can many times be infinite) can undermine the usefulness of a query. For this reason, DP frameworks usually offer the possibility to introduce upper

and lower bounds (i.e., a reasonable range of possible values in the queried data) which they use to perform a sensitivity estimation.

A relaxation of DP: (ε-δ) differential privacy

A relaxed version of differential privacy described in [56], (ε-δ)-differential privacy, only requires that ε-differential privacy is satisfied with a probability of at least 1-δ, in other words, ε-differential privacy can be violated for some tuples, however the probability of that occurring is bounded by δ. Unlike the canonical version of DP where the noise is drawn from a Laplace distribution, here a Gaussian distribution is used. The major advantage of using a Gaussian mechanism is that in some applications it allows adding much less noise, due to a different calculation of sensitivity, although there is a chance of failing at protecting privacy.

3.3.2 Security

Differential Privacy is supported by a rich and rapidly advancing theory that allows for mathematically rigorous reasoning about privacy risk. Systems that adhere to the strong formal definitions of differential privacy provide protection that is robust to a wide range of potential privacy attacks, including re-identification, record linkage, and differencing attacks, but also attacks that are unknown at the time of deployment. This means that an analyst using differentially private tools need not anticipate particular types of privacy attacks, as the guarantees of differential privacy hold regardless of the attack method that may be used.
Also, Differential Privacy provides provable privacy guarantees with respect to the cumulative risk from successive data releases. The limit set by the privacy budget makes it possible to keep track of the privacy leakage of each query to prevent a potential attacker from inferring information from successive queries. So far, it is the only existing approach to privacy that provides such a guarantee [13].

3.3.3 Implementations & References

- **Microsoft's OpenDP (MIT License):** OpenDP is a suite of open-source tools developed by Microsoft and Harvard. OpenDP was developed to provide a privacy-protective analysis of sensitive personal data. The project is focused on algorithms for generating differentially private statistical releases.
- **IBM's Diffprivlib (MIT License):** Developed by IBM, Diffprivlib is a general-purpose library. Developers can experiment, investigate and develop DP applications using this library.
- **Google's Differential Privacy library (Apache License 2.0):** Google released its open-source library last year to meet the needs of developers. It supports most common data science operations. It can be used to compute counts, sums, averages, medians and percentiles, which are widely used techniques for differential privacy.

3.3.4 Conclusion

Although the security guarantees offered by Differential Privacy on a theoretical level are beyond any doubt, the practical aspects when building a system that implements DP raise all kinds of concerns, since when it comes to aspects such as which epsilon to choose, establishing a privacy budget or what bounds to set for estimating sensitivity, there are more questions than answers. For example, how can we explore or experiment with DP-protected data without exhausting the budget? What if there are several users accessing the same dataset? Do we assume that they don't know each other and assign a budget to each one or do we assume that they can collude and therefore assign a budget to the whole dataset? There is also the question of estimating sensitivity, which, depending on the domain of the data, can be a difficult task requiring a number of assumptions about the boundaries of the dataset that can lead to a privacy leak or reduce the utility of the results.
As stated in [57], when implementing a DP setting the challenge is to find a differentially private mechanism that can (a) answer random queries (any type of queries), (b) answer a large number of queries while providing non-trivial utility for each of the queries, (c) be efficient, (d) achieve ε-differential privacy (the stronger privacy guarantee), and (e) answer queries adaptively. However, most algorithms attempt to achieve some of the previous requirements but fail in others.
Despite this, while the general DP framework does not specify how to implement various practical issues, DP as a core technology has already been used in several commercial/institutional applications, giving us a **TRL of at least 8**.

3.4 Homomorphic Encryption

Homomorphic Encryption (HE) allows to perform computation on encrypted data without revealing the input data or the output. The only way to decrypt the result is to have access to a secret/private key (usually, the data holders store this secret key in a safe place). Some use cases related to HE are:

- **Outsourcing of data storage and computing resources**. In general, small users and large companies are tending to store their data on external servers known as Cloud Providers. When we

are working with sensitive data, such as health data, sending this data outside my security network can have serious consequences, so HE would allow encryption and analytics on it.
An example of this outsourcing related with healthcare data would the following. I am a Hospital and I need to store my cancer patients' data in a cloud provider because I do not have space on my premises, but I want to be able to query/analyze this data, guaranteeing tits privacy and integrity. Then, HE would be used to encrypt patients' data and upload to an external database. Afterwards, the doctors and researchers could perform secure remote analytics (SELECT age, COUNT(*) FROM breast_cancer GROUP BY age), being them the only one who could see (decrypt) the result.

- **Encrypted Inference**. This idea can be viewed as a subcase of the previous one. There are many Machine Learning Services (Machine Learning as a Service) with pre-trained models that can be used in a service way, that is, the ML model is hosted in a remove service, and you have to send a request with your data to get the inference. The problem here is again that we have to send our data, so HE can help to send our data encrypted and perform computations on it, revealing the value only once I get the response from the service.
  An example would be the following: I am a hospital and I want to provide a machine learning service for cancer prediction without giving the uses direct access to the model and without making them share their patients' data. He would be used to send encrypted data to the service and performing the computation on the remote server, without revealing the input in any case but the data owner.

In general, HE applications are related with the possibility to do operations in a secret way and this can be applied to different domains e.g. health, financial, etc. The encryption mechanism is based on Public-key cryptography where a public key is used to encrypt your data, and can be shared without restrictions, but only the owner of the private key can decrypt it.

3.4.1 Effectiveness & Usability

Homomorphic Encryption is a very powerful technique, but this is not free. The main drawback, and one of the main reasons is not widely adopted, is the computation overhead. HE, as any PET technique, is an extra layer of security and this layer has cost and some limitations.
The limitations are given by the type of operation can be carried out. The theory stablishes that HE can perform any computation [17] (this is known as Fully Homomorphic Encryption), but there other HE schemes, such as Somewhat of Partial HE where the number and the type operations (sum, multiplication) are limited. In general, the latter are usually quicker to execute but they are not so flexible.
In 2013 Rass and Slamanig [58] stated that "It must be emphasized that homomorphy is a theoretical achievement that merely lets us arithmetically add and multiply plaintexts encapsulated inside a ciphertext. In theory, this allows the execution of any algorithm complex manipulations like text replacements or similar, but putting this to practice requires the design (compilation) of a specific circuit representation for the algorithm at hand. This may be a nontrivial task."

In next sections, the implementations will be discussed; however, it is important to have an order of magnitude about the performance of HE. Then, next figure shows some of the main important libraries used at this moment, and the computation time of some operators such as addition, subtraction, and multiplication.

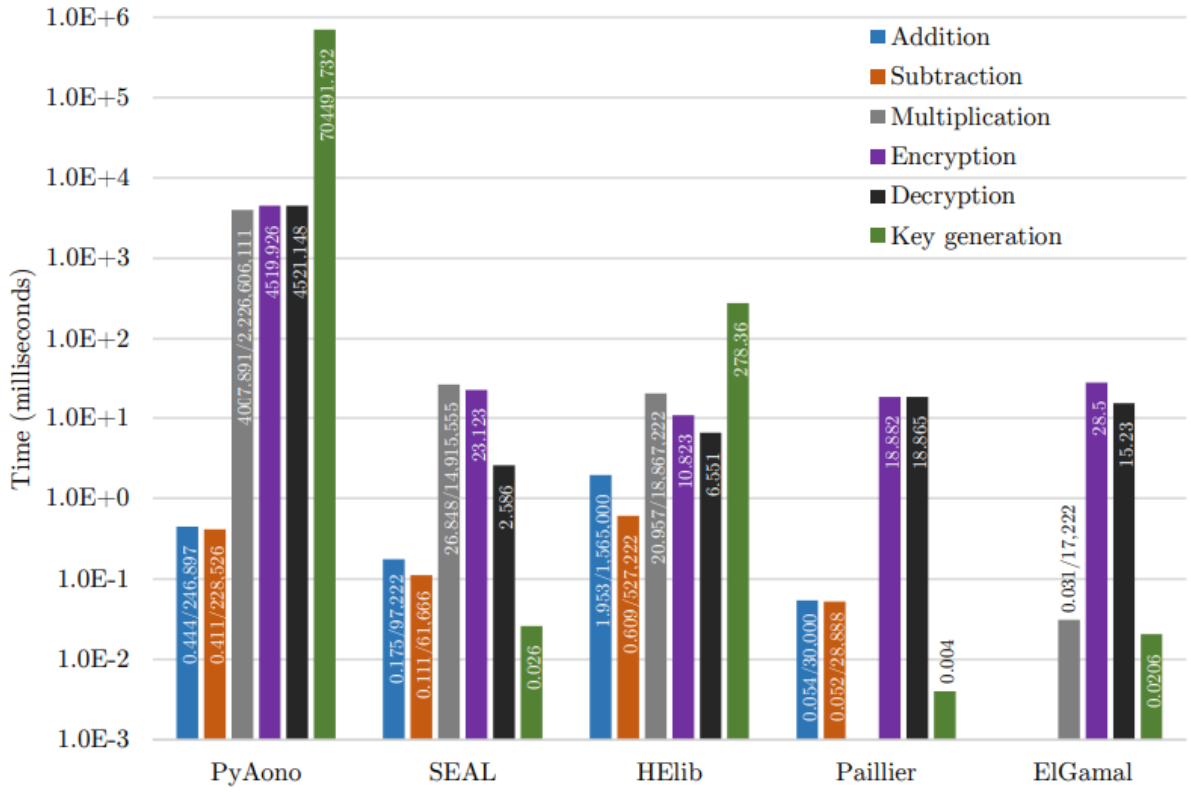

Figure 3. 1 Overall times for some operations and libraries. For operations of + − ×, values are in form t/r, where t is time in ms, and r is the ratio of t and the time execution of the same operation took over plaintexts. E.g. SEAL' addition is 97,2 times slower than plaintext addition. Some of these implementations will be discussed in the following sections. From [59].

In general, the figure shows that HE is 2-3 orders of magnitude slower than its plaintext counterpart. Partial HE (PHE) such as Paillier and ElGamal schemes are faster than Fully HE (SEAL and HElib) but Paillier does not allow the multiplication and ElGamal does not allow the addition.
This performance issue is considered the main reason why HE is almost limited to research projects (see [60] and references therein). However, it is worth mentioning how the well-known Apple company tries to apply this technology. Private Set Intersection (PSI) allows to perform an inner join between two databases, then Apple PSI uses this secure computation into its "password monitoring system" [61]. In its own words: "Password Monitoring is a feature that matches passwords stored in the user's Password AutoFill keychain against a continuously updated and curated list of passwords known to have been exposed in leaks from different online organizations. If the feature is turned on, the monitoring protocol continuously matches the user's Password AutoFill keychain passwords against the curated list."

3.4.2 Security

Public-key cryptography deal with problems that take so much time to solve that there is no practical solution. The classic example is to decompose a number into prime factors, for example, we know that 2*3 = 6, and the inverse operation is easy. However, if we take two large prime numbers and multiply them, trying to decompose it again is time-consuming. Without entering in detail, new cryptography algorithms are based on the ring learning-with-error (RLWE) problem, which is the basis of the public-key algorithms to protect even against quantum computers.
HE is considered the holy grail of cryptography but that does not mean it cannot be attacked. In general, the security leaks are more related with the improper use of the protocols (for example, a bad selection of the needed parameters) or a poor implementation than the protocols itself.
Whereas it is difficult to demonstrate mathematically how secure HE is, we will show a particular HE scheme such as the famous RSA [Reference] for illustrative purposes. RSA scheme is famous because it allows to sign messages, but RSA is also a PHE that holds the multiplicative property:

$$E(m_1)\cdot E(m_2)=E(m_1\cdot m_2)$$

where the Encryption is performed using a public key PubKey and the decryption using a private key PrivKey. The PubKey is composed by the tuple (n, e) where n=p•q (being p and q prime numbers) and 1<e<n.

A classical attack deals with the factorization of n, that is, find an algorithm that given the number n, we can obtain p and q.
In 1999 was demonstrated that using an integer factorization algorithm such as General Number Field Sieve (GNFS) is possible to break a 512-bit system, being needed to generate larger PubKeys.

Parameter	Value
p	10,970,103,190,600,829,290,247,338,335,211,965,420,780,388,872,078,099,817,126,416,270,434,061,669,554,410,207,492,801,377,661,305,902,226,501,029,112,866,903,671,523,834,035,826,355,543,324,388,124,920,973
q	13,164,852,671,587,485,455,965,094,125,779,564,950,272,221,281,419,381,826,998,888,259,126,654,909,845,464,491,695,166,162,180,575,248,237,990,023,090,913,097,495,624,605,196,693,623,810,189,612,824,914,243
$n = p \cdot q$	144,419,792,296,371,725,651,441,968,400,358,744,461,728,782,901,326,239,991,405,917,679,201,034,382,007,748,881,971,474,426,385,183,967,758,037,953,533,411,198,072,902,015,684,446,787,555,458,034,414,740,963,576,780,520,697,356,780,486,494,558,406,350,578,084,209,743,407,242,917,060,532,002,122,918,673,464,703,294,445,336,808,056,050,381,062,260,909,405,854,189,571,030,915,935,592,600,979,039,849,877,118,439

Figure 3. 2 Example of public key with 51 bits. From https://asecuritysite.com/encryption/random3?val=512

Just to give some numbers, breaking a 512-bit RSA key took around 7 months (the whole project) and 300 workstations (a total of 35.7 years of CPU-time was consumed). In 2015 researchers from Pensylvalnia published Factoring as a service [62], where they demonstrated that a RSA 512-bit system can be factorized in just ~4h and $75.

3.4.3 Implementations & References

Now we address different implementations and libraries for HE computations. In the following list, academic implementations, i.e., developed for demonstration purposes will be omitted, listing those with a minimum of a mature:

- **HElib** is a FHE C++ library developed by IBM which implements Brakerski-Gentry-Vaikuntanathan (BGV)[1] and approximate of Cheon-Kim-Kim-Song (CKKS) schemes. Licensed under Apache License v2.0
- **SEAL** is a FHE C++ library developed by Microsoft which implements BGV, CKKS and Brakerski-Fan-Vercauteren (BFV) schems. License under MIT.
- **TFHE** is a FHE C++ library that works with Boolean gates. This implementation is under the hood of commercial companies such as Enveil or Inpher. https://tfhe.github.io/tfhe/ https://eprint.iacr.org/2018/421.pdf
- **Python-Paillier** is a PHE python library that implements paillier scheme (Homormorphic under aggregation). License under Apache License v2.0 and GPLv2. https://github.com/data61/python-paillier
- **PyFhel** is a FHE python library. It is a python binding of SEAL. Licensed under GPLv3.
- **Palisade/OpenFHE**. Palisade, recently migrated to OpenFHE, is an open-source project that provides efficient extensible implementations of the leading post-quantum Fully Homomorphic Encryption (FHE) schemes (BGV, BFV and CKKS). Licensed under BSD-2.

- **Concrete** is a Rust implementation of TFHE. It also provides a Python client via the numpy library. It is licensed under BSD-3

The reader can observe that most of the implementations are done in a low-level programming language such as C++, trying to get the maximum performance out of the hardware. This also means that these libraries have not reached a more general public and therefore their expansion is limited.

3.4.4 Conclusion

We showed how useful HE is, making emphasis that solves the big problem of secure computation. To sum up, HE:

- Can perform arbitrary computations (from a theoretical point of view).
- Offers strong security guarantees, providing protection against quantum computers.
- Tools are under active development.

However, HE has the following limitations:

- Performance is still a challenge; the existing implementations cannot cover all the possible use cases.
- Great knowledge about cryptography is required. There are several cryptographic schemes, and the user has to know which one is suitable for a given use-case.
- HE is secure (nobody but you can see your data), but if encrypted data is accessible by third parties, they cannot see the values, but they could alter/modify the encrypted one. This can be solved providing an authorization system or adding an extra layer of security such as Trusted Execution Environment.
- Only protects the computation itself; sensitive data might still be extracted from the result.

To conclude, HE is one of the more powerful existing PET, the theory is well-stablished, but it is still not ready to cover an arbitrary number real-world use cases. It has been used for limited prototypes, so this gives us a **Technology Readiness Level of 5~6**.

3.5 Anonymization

Data anonymization aims to ensure every data subject's anonymity in a dataset, while keeping the dataset's utility. This is achieved by removing or altering any personal information, as well as any attributes that, when paired together or with other data sources, might allow a data subject's identification.
Ideally, anonymization enables all data use cases by removing privacy concerns: a data owner could anonymize their data and then publish it without any further concerns, as in the second data sharing scenario.

3.5.1 Effectiveness & Usability

Data anonymisation is by far the most mature technology explored on this document and the simplest at a theoretical and technical level. It is widely accepted and used in the real world: it is routine to share anonymized datasets. It does require some specialized knowledge, since "anonymisation processes need to be tailored to the nature, scope, context and purposes of processing as well as the risks of varying likelihood and severity for the rights and freedoms of natural person" [65]. Existing legislation takes anonymisation into account, greatly reducing the legal and bureaucratic barriers when working with anonymised data. To sum up, data anonymisation has a high degree of usability.

3.5.2 Security

There are two main drawbacks when using data anonymization:

- **Risk of re-identification**: By cross-referencing the anonymized dataset with other data sources, an attacker might be able to re-identify the original data subjects. [63] This is not a merely theoretical concern; the strength of these attacks has been proven in real-world cases [64]
- **Loss of utility**: When removing or altering too many attributes, the dataset's statistical utility diminishes. For example, the data subject's address or date of birth might be useful to the data analyst, but unavailable due to anonymization.

There is a fundamental and subjective trade-off between both drawbacks: removing more data will reduce the effectivity of re-identification attacks, at the cost of lower data accuracy. Anonymization techniques try to get the best of both worlds. For example, keeping only the ZIP code or year of birth will keep most of the utility while keeping most of the privacy.

Regarding anonymization, there is an ongoing discussion over whether its techniques are sufficient to protect the data subject's privacy: on one hand, there are multiple real-world examples of successful reidentification attacks [66]; on the other, the attacks were performed on poorly anonymized data, required a non-trivial amount of external data about the data subjects, or had low success rates [63]. Proper data anonymization greatly reduces the risk of re-identification, but the risk remains, especially for high-dimensional datasets.
Unlike the other technologies explored in this document, data anonymisation relies on not having been proved to be insecure, rather than being proved to be secure. Data anonymization algorithms, and the privacy measures they provide (k-anonymity, l-diversity, t-closeness) are purely heuristic: they lack solid theoretical grounding that ensure the level of protection or utility achieved. There are some efforts to translate data anonymization concepts into the language of Differential Privacy [67], showing there is limited value in them.
Existing standards include:

- Section 164.514(a) of the HIPAA Privacy Rule, tailored to protecting medical information
- ISO/IEC 20889:2018 specifies terminology and techniques for data anonymisation
- ITU-T Rec. X.1148 describes a versatile de-identification process framework

3.5.3 Conclusion

Overall, data anonymization is suitable as a first step for data holders but should not be used on its own due to the privacy risk and information loss it entails. We give it a **Technology Readiness Level of 9**.

3.6 Trusted Execution Environment

Trusted Execution Environments (TEE) is an environment in which the executed code and the data that is accessed are physically isolated and confidentially protected so that no one without integrity can access the data or change the code or its behaviour. A trusted execution environment (TEE) is a portion of the main processor device that is separate from the system and the main operating system (OS) [68]. It ensures that the stored and processed data is protected in a secure environment. It provides protection for connected things such as Trusted Applications (TA) and enables isolated cryptographic electronic structures to provide end-to-end security. Some existing providers are:

Intel SGX [69] is the instruction ser architecture that enables the creation of enclave.
ARM TrustZone [70] is a security extension to the ARM architecture with modifications.
AMD Secure Processor [71] is a microcontroller coprocessor integrated within chipsets of AMD.

3.6.1 Effectiveness & Usability

The main concern while using TEE is compatibility with current systems. Since the currently popular process-based model requires existing applications that are supposed to run need to be partly rewritten for TEEs, the virtual machine-based model does not need refactored applications and could be more readily adopted. The first research to improve the compatibility gap has been undertaken, as an abstraction layer on top of the process model to reduce refactoring needs. As a drawback of an abstraction layer, it was noted that an adversary or malware within a TEE cannot be detected by current standard security measures. A solution for this problem is currently not known and is a topic for further research [73].
Major hardware vendors are also focusing on closing the compatibility gap with current systems in their upcoming releases of AMD's Secure Nested Paging and Intel's Trusted Domain Extension. The research community could therefore help to improve organizational understanding of TEEs.

3.6.2 Security

One of the biggest concerns is the security of confidential corporate data and databases. To overcome these security risk, TEE assumes that everything is isolated. However, TEE are not always isolated in practice, and as a result, it is possible to release information from the environment. From the ongoing technical discussion surrounding TEEs, it is evident there are several possible attacks undermining TEE security guarantees [72]. These attacks rely on the assumption that the attacker has full control over the platform. Successful mitigations for these attacks implement additional security primitives to guard memory and I/O accesses.

3.6.3 Conclusion

Overall, the TEE establishes an isolated execution environment that runs parallel to a standard operating system. It is becoming a new form of computing in today's hardware, but there are not many real-world use cases. We give it a **Technology Readiness Level of 6~7**.

3.7 Zero Knowledge Proofs

Zero Knowledge Proofs (ZKP) allow proving statements about private data, without revealing anything else about the data. It allows one party (the Prover) to convince another party (the Verifier) that they know the data, or that it includes a certain element, or obeys a given constraint, without the Verifier having access to the data or having to trust the Prover. [74]. They are a powerful tool for auditing a system without having to break its privacy. Some of the uses of ZKP relevant to data sharing scenarios are:

- **Data auditin**g: ZKP enable external auditors to verify the integrity of a dataset without giving them access to the sensitive data it contains. This includes the ability to check if the dataset contains information for a specific data subject (and thus enabling the subject's right to erasure), if the data for that subject is correct (enabling the subject's right to rectification), or verifying the whole dataset has not been tampered with (eliminating the risk of faulty or malicious cloud storage) [75] [76].
- **Verifiable Model Accuracy:** When using MLaaS, data holders send their data and get the inference results. However, they have no guarantee that the model used is the model they are paying for (e.g., the model provider might be using a smaller model with lower accuracy to reduce inference cost). ZKP allows the model owner to prove the inference was performed with a model with a given accuracy on a public dataset, without revealing any further details about the model [77] [78].
- **Verifiable Computing:** Generalizing the previous point, a client might wish to outsource any generic computation without fully trusting the cloud provider. ZKP allows the computing party to generate a proof of the computation's correctness, which can be checked by the client in less time that it would take them to run the original computation. Note that this technology is not yet considered practical due to the high computational burden imposed on the prover. In addition, it's only useful for doing the same computation with different outputs [79].
- **Identity Verification:** ZKP allow a user to prove knowledge of a secret password without sharing it. Beyond this, they can be used for more involved checks: proving they meet a certain age requirement without disclosing their age, or that they appear on a public list of trusted elements without disclosing any further information. These applications are closely related to the concept of Self-Sovereign Identity, which aims to create a decentralised identity management system while respecting user's privacy, using blockchain technology [80].
- **Privacy Preserving Blockchain:** Blockchain offers a decentralized alternative for health data storage, creating an immutable record of patient data while giving them control over it via smart contracts. However, all operations performed on a blockchain are public and traceable, limiting their utility for this use case [81]. ZKP can be combined with this technology to create privacy-oriented blockchains, whose correctness can be verified without access to any sensitive data [82].
- **Trustless transactions of datasets:** When conducting online business, a trusted third party is required to ensure that both parties get the agreed goods. Blockchain-based smart contracts offer a trust-free alternative to ensure that the payment is tied to the delivery. For the specific case of selling data, ZKP can offer an extra layer of security and modularity: the seller can prove to the buyer the integrity of the dataset, as well as allow per-query payment by proving the correctness of specific query outputs [83].

ZKP are usually interactive, requiring communication between Prover and Verifier. In this situation, data holders must remain available even after providing their data. However, non-interactive ZKP also exist, removing this constraint. As the above list shows, ZKP have a wide range of possible uses, making them potentially relevant to both data sharing scenarios. The last three use cases are based on blockchain and thus require a high degree of cooperation between data holders, making them relevant to the first data sharing scenario. On the other hand, the first three cases only deal with a single data owner, making them a particular case of the second data sharing scenario.

3.7.1 Effectiveness & Usability

The use of ZKP adds computation and storage overhead. For the use cases listed above, existing protocols are optimized enough to be used for practical applications (except for verifiable arbitrary computations). This is especially true for one-off uses or low traffic services: a verified model inference might take seconds instead of milliseconds, but for a medical diagnosing application this might be an acceptable trade-off.
While ZKP promise a wide range of possible applications, actual real-world implementations are scarce. At the time of writing, there are no existing deployed services for most of the listed applications. This limits the available information on ZKP's real-world usability. Still, they have been successfully used in these projects:

- **Zcash** is a cryptocurrency based on ZKP technology. Traditional blockchains are fully public: all transactions and users can be traced, to allow decentralised auditing of the system. In contrast, Zcash transactions are private. Each transaction is represented via a proof ensuring its correctness (sender had enough funds, recipient correctly got the fund, etc.) which can be efficiently verified by any network node, without having access to the transaction's details [82]. While Zcash is purely a financial product and not relevant to data sharing scenarios, the underlying technology could be employed for creating privacy preserving blockchain-based data storage [81].
- **ZKAttest** created and deployed by Cloudflare, is a privacy-preserving identity verification scheme. Traditional hardware-based authentication allows the user to prove their identity by interacting with a trusted USB security key. During this process, the hardware's model information is revealed to the identity verifier. While not too sensitive, this information might be paired with other public information to identify users. To avoid this, the ZKAttest protocol allows the user to prove that the hardware's model is included in the public list of trusted models, without revealing any further information. The code runs directly in the browser, without requiring any extra effort from the user [84].

These projects' technical reports, along with the contrast between the high number of theoretical uses and the lack of actual working applications, show us the main takeaways for ZKP usability:

- **ZKP are considerably more complex than other cryptographic technologies** [85]. While standardization efforts are underway, both at the usability level [86] and the mathematical level [87], there is no single theoretical or technical framework that captures all the different flavours of ZKP. Applying ZKP to new use cases requires new ad hoc protocols, which currently require significant specialized cryptography knowledge to develop and implement.
- Most aspects of the technology are hindered by its novelty: for example, auditing-based products can't benefit from existing standards and widespread trust in the technology; there are no specialized companies that can be used to outsource the development of new ZKP applications; etc. We expect this situation to improve over time, as data spaces gain awareness of the technology.
- ZKP are inherently two party, requiring a certain degree of cooperation between prover and verifier (or adherence to a common standard).

3.7.2 Security

The security of a given ZKP must be analysed by both the prover and the verifier. The prover must ensure that they aren't sending any sensitive information to the potentially malicious verifier; the verifier must ensure that the potentially malicious prover isn't producing a fake proof. While proving these properties requires a specialized mathematical background, this work has already been done for all commonly used ZKP, removing this burden from the users.
Due to the probabilistic nature of ZKP, a malicious prover has a small chance of success. To mitigate this, the ZKP is run multiple times, exponentially reducing this possibility. There is a natural trade-off between performance and security: more iterations reduce the possibility of failure, at the cost of a higher computing cost.
ZKP derive their security from traditional cryptographic assumptions. As such, they are only vulnerable to brute force attacks, which can be mitigated by selecting appropriate security parameters, such as a high number of bits for secret keys. Other security risks include those suffered by all other technologies: faulty or malicious implementation, malware, phishing for secret keys, denial of service, etc.

3.7.3 Implementations & References

At the time of writing, ZNP implementations are still in an early state. Except for some cryptocurrency-related projects, implementations require a high degree of specialized cryptographic knowledge and are far from production ready. This list gives a representative element for each use case and level of abstraction:

- **Libsnark (custom permissive license)** is a low-level library for generating and verifying non-interactive proofs. It offers flexibility and performance, but requires specialized knowledge and manual labour for expressing and implementing protocols in the low-level language expected by the library. It should be considered only when implementing a higher-level library: most of the other entries in this list use it as a backend [88].
- **Zksk (MIT License)** is a medium-level Python library for creating and running interactive ZKP. It includes a library of common building blocks, the tools to define new ones, and allows composing them to create complex protocols. It is well documented, but still requires specialized knowledge to use effectively [89].
- **Pysnark (custom license)** is a high-level library for automatically converting python functions to non-interactive zero knowledge proofs. This substantially lowers the barrier of entry to developing new applications based on ZKP. It is actively maintained and supports several low-level backends, improving its interoperability with other libraries [90].
- **Pequin (BSD-style license)** is a toolchain which enables verified computation by generating zero knowledge proofs for arbitrary C programs. This enables a data owner to outsource computations to an untrusted cloud provider, guaranteeing the correctness of the output. Note that the cloud provider will incur in a x1000 performance cost, heavily limiting the practical application of this technology until better ZKP are developed [91].
- **Merkletreejs (MIT License)** is a high-level library for set membership ZKP. It allows the prover to efficiently show that a given element is included in a set, without revealing the set's contents. The library can be integrated with smart contracts, to allow blockchain applications such as verifying that a transaction comes from a given private list of accounts [92].
- **ZkLedger (open source, unclear)** is a distributed ledger enabling versatile auditing of private data. Each data owner stores their data privately, only publishing a "commitment" to the ledger that contains no sensitive information. An auditor might ask the data owner to reveal some statistical property of their private data (for example, the exact value of the mean, or that the standard deviation is below a certain threshold). After computing these values, the data owner proves their correctness using the ledger's commitments. While the applications are promising, the actual code is currently in an early stage and lacks documentation [93].

Since this is a static document, the above list might be outdated; we refer the reader to [94] and [95] for updated references. At the time of writing, there are no official standards for ZKP, but there is an ongoing effort to create it [86].

3.7.4 Conclusion

To sum up, ZKP:

- Has a wide variety of applications, mainly focused on auditing private data or computations
- Offer strong security guarantees, based on traditional cryptographic assumptions
- Has an acceptable performance cost for most applications.

On the other hand, ZKP:

- Require a specialized background to understand and deploy, due to the lack of high-level frameworks
- Are almost untested in real-world settings

Overall, we consider ZKP to be a promising technology with plenty of future applications. However, few of the theoretical uses have been developed at the time of writing, giving it a **Technology Readiness Level of 2~3.**

We hope that this section would serve as a guide for data spaces stakeholders and participants to navigate the complex landscape of PETs. The key takeaway is that there is no single PET that solves all privacy issues

or covers all possible use cases. Therefore, knowledge of the available PETs and the specific data sharing scenario is crucial for choosing the right approach.

4 Prototype Design

The purpose of this section is to describe a realistic use case, related to the health domain, where sharing the knowledge of different institutions in a privacy-preserving way is critical. The selection of the health use case is motivated by the inherent importance of privacy protection when dealing with sensitive and personal healthcare data and the fact that the European health data space is currently being rolled out within the context of the European strategy for Data. By focusing on the health domain, we can highlight the specific challenges and considerations that arise in this context, such as regulatory requirements (e.g., HIPAA, GDPR) and ethical concerns surrounding patient data.

The choice of the health use case also allows us to illustrate the importance of privacy-enhancing technologies, such as federated learning, in facilitating collaboration and advancements in medical research, disease diagnosis, and treatment. This use case demonstrates how privacy-preserving techniques enable knowledge sharing among different institutions while safeguarding individual privacy. Whereas we have restricted ourselves to this domain, it is worth mentioning that most of the procedures (data preparation, data sharing, etc.) can be widely used in other domains and their respective data spaces. The intention is to provide a concrete illustration that can be extrapolated and adapted to different contexts, emphasizing the broader applicability and relevance of privacy-preserving techniques across industries. For simplicity reasons, the use case is based on the International Data Spaces (IDS) reference architecture as reference [4][3]. Then, we will show how the privacy enhancing technologies fit in this design. This section deals as well with all the steps involved in this kind of data sharing scenarios.

4.1 Problem Statement

In the following sub-sections, we elucidate the pertinent problem encountered from both a functional perspective (i.e., clarifying the nature of the problem itself) and a data perspective, encompassing aspects of data governance, utilization, and sharing.

We are aware that our data is distributed in different organizations and domains (banking, health, insurance, etc.). In addition, these data have an owner and are protected under different regimes defined on the national and European level that restrict their use and sharing. This implies that if one wants to use this data, there are many steps required before this can happen. This seems to not be fully aligned with to the overarching vision data economy and what is known as a data-driven organization, where the data is the value, and the applications are responsible for using it properly.

Under this scenario, and without loss of generality, we focus here on the health domain. Specifically, we work with skin diseases, where the problem of the use and sharing of data will be reflected in the fact that the data is distributed in different countries.

The specific problem is to classify skin lesions into one of eight possible types, some of them of cancerous nature, by developing artificial intelligence models. Skin lesions image datasets collected by health care institutions are increasingly used to train machine learning. However, by using training data from often too specific populations, machine learning algorithms are susceptible to over-fitting, as their generalization capability is heavily influenced by the participants and images used for training, which are prone to selection bias. This way, algorithms used for skin lesion classification frequently underperform when tested on independent datasets. Several examples of the susceptibility of machine learning algorithms to bias by clinical factors such as age, gender, ethnicity, and socio-economic status have been reported in various areas of healthcare and artificial intelligence [96] [97] [98] [99] [100].

An aggregated training set consisting of samples from different sources that introduces diversity in aspects such as patient age, sex, ethnicity as well as lesion location and even imaging devices, could largely correct this problem by achieving more robust models in terms of generalisability. However, a scenario in which several hospitals agree to contribute their data to a centralised database external to their systems is often infeasible due to strict privacy regulations and ethical considerations. On the contrary, a collaborative scenario based on federated learning technologies [101] [11] is more feasible, as this results in a (global) model that aggregates several locally trained models, without the data leaving the organisations where they are stored.

[3] There are other third-party initiatives such as `Gaia-X` [104], and `Simpl` [105] from the European Commission. During this document IDS is used because of the availability of a mature reference architecture and existing technical building blocks.

Yet, other difficulties arise, like the need to harmonize/standardize data, since different hospitals might name the same features, values or diseases differently, or use different scales for numerical values; on the other hand, it is necessary to deal with the probable non iid-ness (non-Independent and Identically Distributed) of data. Also, model updates should be securitized, as they may be subject to adversarial attacks.

4.2 Data Spaces and PET

We aim to overcome the challenges defined above in the context of a common data space through the use of PETs.
Data Spaces aim, in accordance with the European strategy for data [1], at *creating a single market for data that will ensure Europe's global competitiveness and data sovereignty*. They are structures that provide trust and security context for the sharing of data between various actors in a homogeneous manner through combined governance, organisational, legal and technical mechanisms. Data spaces facilitate interoperability for accessing or transferring data and enable their efficient and legitimate re-use in a context of sovereignty and control for the parties over their own data. The availability of Data Spaces facilitates their exploitation and the extraction of value from them through services based on Artificial Intelligence or Big Data.
Within that context, many prominent initiatives are being established in the EU. Among those, the International Data Spaces (IDS) initiative [102] proposes a Reference Architecture Model for this aim and related aspects, including requirements for secure and trusted data exchange in business ecosystems. The Business Layer of the Reference Architecture Model defines and categorizes the different roles the participants in the International Data Spaces may assume. In our specific use case, the following roles are applicable:

Data owner: Is the entity or individual who has the legal rights and control over the data and its usage. They are responsible for managing, accessing, and sharing the data in accordance with the relevant legal frameworks. The Data owner is crucial in ensuring the security and integrity of the data, as well as determining its availability and accessibility for other users within the international data space.

Data provider: Makes data available for being exchanged between a Data Owner and a Data Consumer. In most cases it is identical to the Data Owner, but not necessarily.

Data consumer: Receives data from a data provider. From a business process modelling perspective, the data consumer is the mirror entity of the data provider; the activities performed by the data consumer are therefore similar to the activities performed by the data provider. Before connecting to a data provider, the data consumer can search for existing datasets by making an inquiry at a Broker Service Provider. The Broker Service Provider then provides the required metadata for the Data Consumer to connect to a Data Provider. Alternatively, the Data Consumer can establish a connection with a Data Provider directly (i.e., without involving a Broker Service Provider).

Data user: Like the data owner being the legal entity that has the legal control over its data, the data user is the legal entity that has the legal right to use the data of a data owner as specified by the usage policy. In most cases, the data user is identical to the data consumer. However, there may be scenarios in which these roles are assumed by different participants.

Broker Service Provider/Data Intermediary: An intermediary that stores and manages information about the data sources available in the International Data Spaces. The activities of the Broker Service Provider mainly focus on receiving and providing metadata. The Broker Service Provider must provide an interface for Data Providers to send their metadata. The metadata should be stored in an internal repository for being queried by Data Consumers in a structured manner.

In addition to the above roles, there is a relevant IDS component in the scope of our solution: the **Connector**. As a technological building block of the IDS, the Connector ensures that participants maintain sovereignty over the data. At the same time, it functions as an interface between the internal systems of the IDS participants and the IDS ecosystem itself.

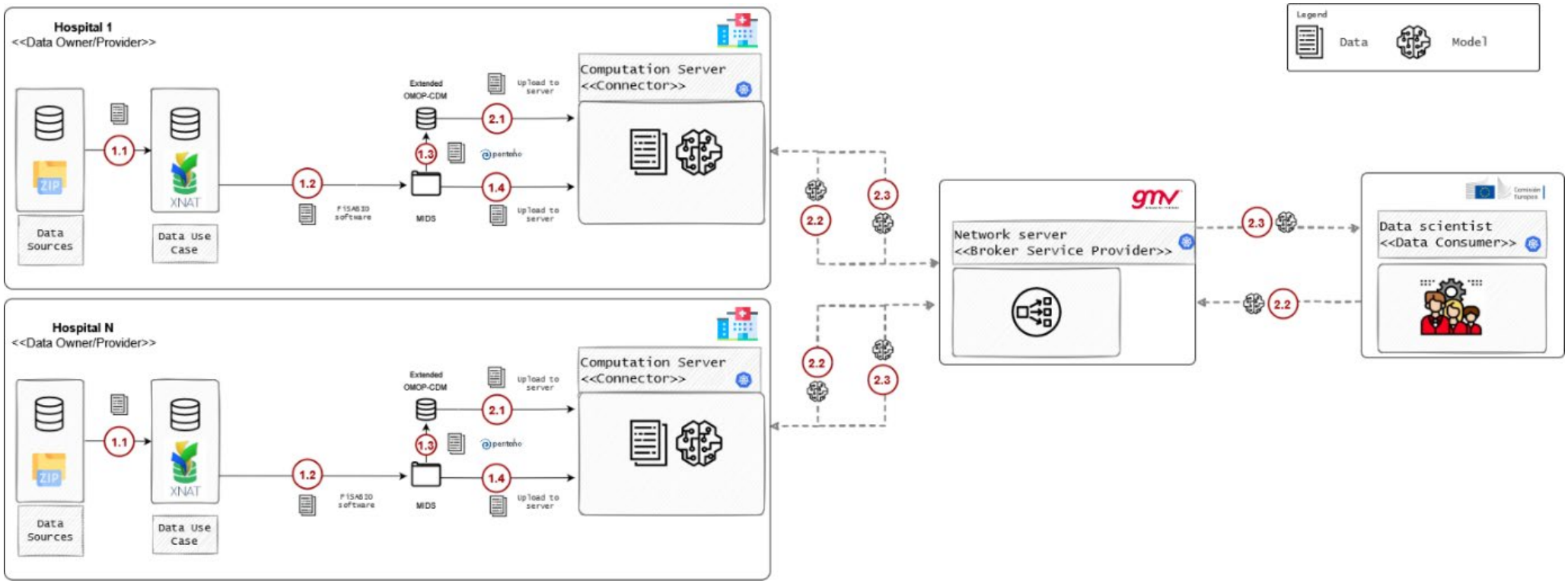

Figure 4. 1 Data Flow. Source: Authors own elaboration

Figure 4.1 depicts the data flow of the proposed solution within the frame of Data Spaces. This data flow can be summarized as follows:

(1.1) The process begins by uploading zip files of skin lesions and metadata to an imaging software platform.
(1.2) The images with their metadata, stored in XNAT[4] will be transformed into the MIDS[5] folder standard.
(1.3) A mapping of images metadata stored in the MIDS directory to extended OMOP-CDM is performed.
(1.4) The harmonized images, stored in the MIDS directory, are delivered to the computation server via a Python API where they are prepared for use within the federated network.
(2.1) OMOP compliant data is uploaded to the server.
(2.2) The data consumer (data scientist) defines the ML model to be trained. As the data do not leave the organizations, it is the model that moves to them. The network server is the component in charge of distributing the model among the computation nodes.
(2.3) The model is trained locally in each organization's computation node. Once this operation is done, the model (containing only the weights) is sent back to the users' node through the network server.

Privacy Enhancing Technologies (PETs) are tools and techniques designed to protect data privacy, without losing the functionality of the data System. In this report, we will focus on those that enable processing private data and creating value from it without breaching the data subject's privacy. Data Spaces have a built-in concept of security and data sovereignty: data holders can set policies dictating how their data can be accessed and used. PETs enable a wider range of policies: in this use case, for example, they enable data holders to allow their data to be used for a collaborative model without needing them to also allow access to the raw data. In the context of data spaces, PETs are a key enabler that support the sharing and analysis of sensitive data.

4.2.1 Data Preparation

Before the raw data can be used, it must be cleaned, transformed, and organized to support further processing and analysis. This homogenization process, often overlooked in academic studies, can be one of the most time-consuming steps in a real-world machine learning project. This is especially true for distributed data: different organizations often store different measures, in different formats, with different labels.
In addition, data privacy must be preserved during data preparation. This introduces challenges unique to the distributed data scenario. For example, if two institutions have data on the same patients, how can they align their data without revealing the identities of the patients they don't have in common? One solution would be to incorporate a third party for this task, i.e., give access to every data source in every organization with the goal of aligning records, however this may raise issues with data access and the associated policies.
Another challenge that we face, and although it is explained in more detail later in the following sections, is that the data is hosted in different hospitals that correspond to different countries, with the legal implications that this implies. Fortunately, PETs can help surmount these challenges.

4.2.1.1 Communication

The naïve solution is for data holders to directly communicate with each other, manually reviewing each variable and agreeing on a common representation for it. This method will produce excellent results and provide a common schema for the distributed datasets, but at the cost of requiring a lot of time and many iterations by domain experts. The result will also be single use, requiring more work if a new data owner joins

[4] XNAT is an open source imaging informatics platform developed by the Neuroinformatics Research Group at Washington University. See https://www.xnat.org/about/

[5] Medical Imaging Data Structure (MIDS) [106] was therefore conceived with the objective of extending Brain Imaging Data Structure (BIDS) methodology to other anatomical regions and other types of imaging systems in these areas.

or if the use case changes. Therefore, such an approach is not scalable. For these reasons, most use cases should use an existing common specification relevant to the domain.

4.2.1.2 Data Harmonisation

Since different organisations may, in their databases, assign different names to variables or values that are conceptually equivalent, a standardisation process is necessary across the different nodes of the federated network in order to develop a common model. To this end, we rely on the OMOP Common Data Model, as it is a widely used infrastructure aimed at standardising the format and content of observational data in the medical field.

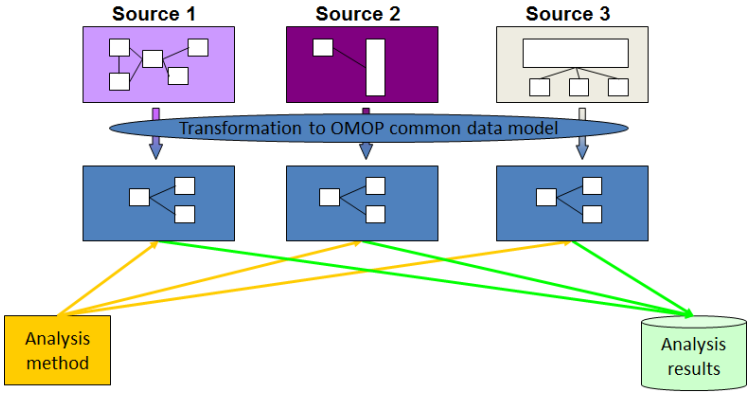

Figure 4. 2 OMOP operation scheme: Once a database has been converted to the OMOP CDM, evidence can be generated using standardized analytics tools. (Source: https://www.ohdsi.org/data-standardization/)

OMOP-CDM is composed of 37 tables of which 17 contain clinical information including the patients table. Another ten tables are used to represent the standardised vocabularies, and the rest contain miscellaneous information such as locations, providers, costs, cohorts or metadata. The clinical tables contain unique identifiers and allow storing together with the standardised data (e.g., the OMOP standard code of a diagnosis or procedure) the original value (e.g., a local coding). The following figure contains the tables that constitute OMOP-CDM.

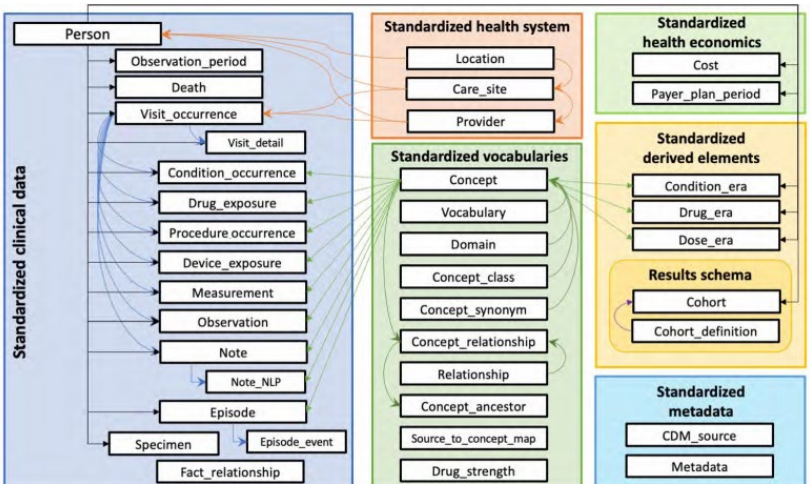

Figure 4. 3 OMOP CMD Tables (Source: https://www.ohdsi.org/data-standardization/)

4.2.1.3 PETs for Homogenization

Pre-processing centralized datasets has no or few privacy concerns: the data never leaves the data owner's control and can be freely accessed to perform any required transformation. The distributed data scenario introduces new privacy challenges. We now examine some of the most common problems and how PETs can solve them.

4.2.1.4 Record Linking

Consider a vertically partitioned distributed dataset. In other words, each data owner has different attributes or labels over the same population of data subjects. Assuming all datasets have a common ID, it's easy to align them in the centralized setting, by simply matching the rows with the same ID. However, on a distributed dataset, this would breach the data subject's privacy: all IDs will be revealed to all parties, even if they are present in a single dataset.
Private Set Intersection (PSI) enables different data holders to filter the IDs appearing in all databases, without revealing any other ID. This enables record linking without any privacy loss (beyond the fact that other data holders have records with the same ID). To use this technique, data holders must be capable of performing SMPC, by having each a computation node. [See Annex A]
Variants of this technique can be used if there is no clearly defined ID, by applying entity resolution techniques before running PSI. For example, the data subject's personal identifiers can be encoded into a Bloom filter and used as an ID; distance between filters can be computed with SMPC, allowing fuzzy set intersection.
In our use case, the data sources are in different countries, so we will not use these techniques since there is no significant intersection between the datasets.

4.2.1.5 Data Normalization

Common normalization techniques (such as scaling all columns to have the same mean and standard deviations or filling missing values with an average value) require statistics computed on the global dataset. These are not available when working with a distributed dataset. If the data is not Identically Distributed, local statistics can't be used to approximate the global properties. Two solutions exist to securely compute global statistics:

- Use SMPC to compute them directly. This is specially recommended if the modelling phase will use SMPC data, since in that case the pre-processing step will not add much complexity.
- Aggregate each data owner's local statistics. For example, to compute the global mean, take the average of each dataset's mean (weighted by the number of rows). This reveals some private data: to avoid it, apply Differential Privacy to each local mean before sharing them for aggregation.

4.2.1.6 Data Auditing

Before jointly training a model, data users might want to verify that the input dataset has valid and complete data. To perform this audit without direct access to the private data, PETs are required. Some options are:

- Privately compute global statistics to verify the dataset's validity, using the techniques from 4.3.2. These global statistics can verify that the data obeys some assumed distribution, ensure it falls into the given bounds, quantify the presence of outliers in the data or bias in the labels, etc.
- Use ZKP techniques to ensure the dataset's completeness and integrity. This can be achieved by generating extra metadata when storing a dataset; when retrieving it, that metadata can be used to ensure that the dataset has not been modified while in storage.

4.2.1.7 Anonymization

As an additional step, regardless of the technique(s) used, the raw data can be anonymized as a first step. Note that this does not fully remove the risk of reidentification, so it should not be taken as a substitute of other Privacy Enhancing Techniques.

4.2.1.8 Conclusion

Using the OMOP Common Data Model (OMOP-CMD) can help to uniformize and standardize different datasets of skin lesion that may have different formats and structures. By using OMOP-CMD, it is possible to transform the data into a consistent format, which can make it easier to integrate and analyse the data in a federated environment. In addition, using the MIDS folder standard can help to organize and store the data in a standardized way, making it easier to access and use. Overall, the use of OMOP-CMD and the MIDS folder standard can help to improve the consistency and interoperability of different datasets.

4.2.2 Modelling & Training

Once each data owner's raw data has been homogenized, it is ready to be used as a single distributed dataset. Such a dataset has many potential applications: in this specific use case, we are focusing on training a neural network for classification, but other applications include collecting statistics, training other types of machine learning models, or generating synthetic data for further reuse.
Traditional techniques can only perform these operations with a centralized dataset. However, in this case (and other realistic cases enabled by data spaces), the data is distributed between several silos which it cannot leave. PETs are a key piece to enable collaboration between data holders without compromising privacy.
In the following subsections the different techniques will be discussed, describing the advantages and disadvantages for this specific scenario.

4.2.2.1 Centralized Solution

Being able to collect the whole dataset in a single location would enable all traditional techniques, making it a valuable target. Can it be achieved without breaching privacy? Two PETs can do it: Anonymization and TEE.
With anonymization, each data owner would remove all personally identifying attributes from their data, before sending it to a centralized location. At a first glance, this seems like the perfect solution. However, it has two drawbacks: the anonymization process removes valuable information from the data, limiting the utility of the dataset; and there is a risk of re-identification, undermining the privacy guarantees.
Another option is for all data holders to send their data to a centralized Trusted Execution Environment (TEE) running the training algorithm. The algorithm will only output the resulting model, without copying or distributing the private data. This technique has some drawbacks too: it's more complex than traditional centralized training, since the algorithm must be fully specified in advance and there is no possibility of data exploration; and the security offered is based on specific hardware instead of general cryptographic protocols.
Both techniques have another drawback: regardless of the level of security they offer, the data will technically be leaving their silos, which automatically activates regulatory restrictions, adding complexity to the process.

4.2.2.2 Federated Learning

Federated learning is perfectly suited to our use case: training a neural network across horizontally distributed data. To train a model, the participants should be connected as specified in Figure 2.1 (right). The neural network architecture is specified by the data consumer, and a copy is sent to each data owner. Each data owner trains the model on their data, and the resulting models are then aggregated by the data consumer. In our experience, most European hospitals either have the required infrastructure or can obtain it as a routine cost of the research project.
The base technique admits some variations:

- By using a Split Neural Network architecture, it can be adapted to work with vertically partitioned data.
- A malicious data owner can perform a data poisoning attack, to add a backdoor into the model or sabotage its performance. These can be mitigated by performing data auditing before training, and by using defensive techniques on the aggregation step.
- The data consumer might extract private data during the aggregation step by measuring the variation between different data owner's updates. This can be mitigated by using SMPC to perform the aggregation step, removing the need for a centralized aggregator. This diminishes the possibility of extracting individual data from the model's updates, and prevents the data being traced to any of the data holders.
- Another option is for the data holders to apply Differential Privacy in each training step. This removes the risk of extracting data from the model's updates, as well as the risk of extracting data from the trained model (which is not specific to Federated Learning). However, it will reduce the trained model's accuracy.

Federated Learning has a computational cost and runtime comparable to centralized training and has already been tested in real world applications. While it is somewhat vulnerable to privacy and poisoning attacks, these require data holders or consumers to be malicious, which can be discouraged with contractual tools. If this is not possible, we have shown how to combine Federated Learning with other PETs to improve security.

4.2.2.3 Multi-Party Computation

SMPC allows a group of data holders to jointly compute any function on their private outputs. This technique can be used for any use case, including ours: training a neural network on distributed data. However, in this specific use case, it offers no advantages over Federated Learning, only adding a high computational and runtime cost, and increasing the computational demands from data holders (see the architecture in Appendix B). For this reason, we will not be considering SMPC for our use case. However, it enables cases not covered by Federated Learning:

- Performing queries to obtain statistics from the global dataset.
- Training other types of models (such as Random Forest or Kaplan-Meier).

4.2.2.4 Homomorphic Encryption

Federated Learning requires, at the very least, as many computational resources as training a traditional model. While this usually feasible, it would be valuable to outsource the computation to a specialized third party, without compromising privacy in the process. Homomorphic encryption makes this possible: the hospital uses it to encrypt its data, which is then sent to a third party which operates on it (for example, training a model) without decrypting it. The computation's result is the sent back to the hospital, which uses its private key to decrypt the result.
However, currently the technology is not mature enough for a real-world use case: the computations are slowed by several orders of magnitude, making them infeasible for any useful model. For this reason, we will not use this technology for our use case.

4.2.2.5 Differential Privacy

All the techniques presented so far focus on preserving the data subject's privacy while training a model. However, this is not enough: a skilled attacker might be able to extract individual data from the final model. This can be remedied with the use of Differential Privacy: during training, random noise is carefully added to the data, masking the impact of any individual's contribution. While Differential Privacy cannot be used on its own to train a model, it can be combined with any other technique to increase the output's security.

4.2.2.6 Conclusion

For our use case, we will train a neural network using Federated Learning, using the architecture described in Appendix A.

4.2.3 Model Evaluation & Deployment

Once the model is trained, only two tasks remain: verify it works as expected and deploy it for real world usage. When training a traditional model on a centralized dataset, there are plenty of existing solutions for both. In our case, however, new technical and theoretical challenges emerge due to the distributed nature of the data.
Evaluation must not breach the privacy of the test data being used, or the model being tested. In a distributed training setting, this can be achieved with the help of PETs.
Deployment also requires special consideration in our use case. The trained model could contain potentially sensitive data or be considered valuable intellectual property. For these reasons, the model owner(s) might wish to control and limit access to it, for example by allowing inference without allowing free access to the model. This raises privacy challenges which again can be solved by PETs.

4.2.3.1 Accuracy and bias

Before training, one or several data holders must reserve some data for testing. The model can be evaluated by applying it to the unseen data and extracting statistics from this result, such as the average accuracy or the bias in the model's output. The same techniques from section 4.2.1.5 can be used:

- In the case of a SMPC model, distributed among several data holders, inference results will be distributed SMPC values; their accuracy or bias can be collaboratively computed and revealed.
- For a federated model, each data owner can directly apply it to their test data and extract the relevant local statistics. These can then be combined directly (optionally perturbing them with DP) or via SMPC.

For our use case, we will measure the model's performance by directly combining each data owner's local performance metrics.

4.2.3.2 Encrypted Inference

The goal of the trained model is to perform inference on new data. If the new data is public, it can be sent to the model's location; if the model is public, it can be sent to the data's location. However, if both the model and the data are private, both options will breach privacy. There are two PETs which can be used to solve the issue:

- Homomorphic Encryption can be used if there's a single model owner. To perform inference on new data, the data owner encrypts it using homomorphic encryption and sends it to the model owner; the model is used to compute an encrypted prediction, which can only be decrypted by the data owner. This technology is not mature enough.
- SMPC is suitable when there are multiple model holders. The model is shared between all of them, using additive secret sharing; it can then be used to perform inference on new data points, without any individual model owner having access to the new data. This technique can be combined with SMPC training, removing the need to ever reveal the model.

Note that regardless of the technique used, the model's privacy might be compromised by a reconstruction attack, which uses inference results to extract internal information from the model.
In our use case, the training is performed with Federated Learning, revealing the model to all participants in each communication round. For this reason, we will consider the final model to be public.

4.2.3.3 Verifiable MLaaS

When using Machine Learning as a Service (MLaaS), data is sent to the model owner which then returns the inference results, allowing the model owner to maintain the model's privacy. However, in commercial applications, the model owner might not be trustworthy: they might be providing results with a smaller, less accurate model, to reduce inference costs. This can be prevented using Zero Knowledge Proofs, which would allow the consumer to verify that the model performing the inference reaches a certain accuracy on a public dataset, without giving them any further information about the model.
In our use case there are no commercial interests and the data is private, making it a bad fit for a MLaaS deployment.

5 Prototype Implementation

This section contains the summary and main learnings from the conducted experiments. We distinguish two types of experiments:

- Local Training, where a Convolutional Neural Network (CNN) model is only trained and evaluated on a single dataset.
- Federated Learning, where the CNN model is trained and evaluated using the distributed datasets.

As for the goal of providing realistic use cases on data sharing scenarios in a data space setup while using privacy-preserving enhancing technologies in the healthcare domain. The following experiments have been implemented.

For this scenario, three different types of experiments were considered, to make the comparison and show the reliability of using privacy-enhancing technologies vs. traditional solutions. The final goal of each experiment is to use machine learning (ML) to classify skin lesions based on the image datasets.

5.1 Preparing for Experiments

The following steps are common for all performed experiments.

5.1.1 Data Collection

The data for the experiment were downloaded from the public datasets of the three different papers related to skin lesions classification.

Dataset	Paper	Country of origin	Imaging modality	N. patients	N. images	Lesions
PAD-UFES-20	PAD-UFES-20: A skin lesion dataset composed of patient data and clinical images collected from smartphones	Brazil	Macroscopic	1373	2298	**Three skin cancer:** - BCC Basal cell carcinoma - MEL Melanoma - SCC Squamous cell carcinoma **Three skin diseases:** - ACK Actinic keratosis - NEV Nevus - SEK Seborrheic keratosis
MED-NODE	MED-NODE: a computer-assisted melanoma diagnosis system using non-dermoscopic images	Netherlands	Macroscopic	Not reported	170	- Melanoma (folder) - Naevus (folder)
HAM10000	The HAM10000 dataset, a large collection of multi-sources dermatoscopic images of common pigmented skin lesions	Australia /Austria	Dermoscopic	Not reported	10015	- akiec: Actinic Keratoses - bcc: Basal cell carcinoma - bkl: Benign keratosis - df: Dermatofibroma - nv: Melanocytic nevi - mel: Melanoma - vasc: Vascular skin lesions

Table 5.1 Information regarding each dataset

5.1.2 Pre-processing

The following pre-processing steps have been performed:

- Image transformation (resize to 224 x 224)
- Image transformation (normalization of the pixel values using mean and standard deviation)
- Splitting the dataset into the training set and test set (70% for training set, 30% for test set)

Note: All the experiments have used the same preprocessing steps.

5.1.3 ML Algorithm for Image Classification (CNN)

The architecture design of the CNN model:

- **Input Layer:** Takes color images of the size of (224 x 224 x 3), the third dimension 3 refers to the three color channels: red, green, and blue.
- **Convolutional Layers:** Identifies features from the input images (feature map) using the kernel size of 5x5, such as textures, corners, and edges.
- **Activation Function:** After each convolutional layer, a ReLU activation function has been used to remove the negative values from the feature map and introduce nonlinearity into the network.
- **Pooling Layers:** Summarizes the results of convolutional layers and reduces the size of feature maps. Max pooling was used to calculate the max value in each patch of the feature map.
- **Fully Connected Layers:** After applying the mentioned convolution and extracting features two flattened layers have been used. The first layer converts the output tensor of the pooled feature map to a one-dimensional layer which will be passed to the second flatten layer to perform the final classification of the image and map the input data into a set of output classes.
- **Output Layer:** Produces the final classification using Softmax Activation Function to normalize the output of the network via the probability distribution on the predicted output classes.

Note: All the experiments have used the same CNN model.

5.1.4 Model Training

After the configuration of the model architecture, the model will be trained over the preprocessed data using the following hyperparameters and strategies:

Hyperparameters:

- Number of hidden layers: In total there are four, two for convolutional layers and the other two are for fully connected layers.
- Learning rate: To define the step size of updating the model parameters according to the optimization function during back propagation process, the leaning rate was set to 0.005.
- Momentum: To speed up the process of updating the parameters during the training process and improve network convergence, it was set to 0.5.
- Minibatch size: For feeding the data to the CNN model with smaller sample size instead of the entire datasets at once the batch size of 32 was considered for both training set and test set.
- Epochs: As for the number of times of training the whole dataset, 50 was set for the number of epochs.

Strategies:

- Optimization algorithm: In order to minimize the loss function, Stochastic Gradient Descent (SGD) as the optimization algorithm has been used. This algorithm selects the random subset of the training data at each iteration to compute its gradient instead of using the entire training set.
- Criterion function: For the calculation of the loss function between the actual output and the predicted output, the cross-entropy loss function has been used which is common for multi-class classification problems.

5.1.5 Model Evaluation

In order to calculate the performance of the model, evaluation metrics such as loss, accuracy, and confusion matrix have been used on both the training set and test set.

Evaluation metrics:

- **Confusion matrix:** It represents a summary of the prediction of the ML model that shows the number of correct and incorrect predictions per class. Which is displayed as the following:

		Actual class	
		P	N
predicted class	P	TP	FP
	N	FN	TN

- **Accuracy:** It measures the percentage of correctly classified samples:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

- **Recall:** It measures the percentage of correctly predicted positive samples out of all actual positive samples.

$$\text{Recall} = TP / (TP + FN)$$

Precision: It measures the percentage of correctly predicted positive samples out of all samples which were predicted positively.

$$\text{Precision} = TP / (TP + FP)$$

F1 Score: It is calculated by the harmonic mean of precision and recall.

$$F1\ \text{score} = 2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$$

Multi-Class F-1 Score (Micro Average):

- Calculation of the TP, FP, and FN values of all the classes.
- Calculation of F1 Score based on the following formula.

$$F1\ \text{Score} = TP / (TP + ½(FP + FN))$$

5.2 Performing Experiments

5.2.1 First Experiment: Local Training

Each dataset was downloaded and trained on its local environment. The following steps were performed:

- Data preparation on each dataset using the same steps for all datasets.
- Training the CNN model on each prepared dataset separately
- Calculation of the final performance of each local training

5.2.2 Second Experiment: Centralized Training

All the datasets were merged and trained on the central host. The following steps were performed:

- Data preparation on the merged datasets (using the same steps as the first experiment)
- Training the CNN model on the merged datasets
- Calculation of the final performance of each local training

5.2.3 Third Experiment: Federated Learning

Each dataset stayed in its own local machine considering each machine as a data owner and the server does not have access to each machine dataset. Then the CNN model was provided by the server. The following steps were performed:

1. Data preparation: General steps for data preprocessing have been sent to each data owner to be used over the data of their own machine.

2. Model initialization: A base CNN model has been sent to all clients from the server. Each model was trained on each client's dataset separately.

3. Local model training: After each client received the CNN model, it was trained locally over their respective data. The training is done in multiple rounds while the model is getting updated and downloaded after each round.
4. Model aggregation: Aggregation on model parameters using Federated Averaging (FedAvg) algorithm on the server side and improve the model.
5. Model evaluation: The aggregated model is then evaluated on each dataset to measure its performance.
6. Repetition the steps from 2 to 4 until the model reaches its global convergence.

5.3 Model Evaluation

As for the model evaluation we have used accuracy for each epoch on both training and test sets. Also, the confusion matrix for the final evaluation of the model was calculated in each experiment.
Our comparison of federated learning, centralized training, and local training is based on two distinct configurations.

In the first configuration, we utilized a low number of epochs (2 and 5) for both centralized and local training, while using a high number of rounds (25 and 10) for the FL model.

Results: We divide our findings depending on the kind of label:

- When the labels are presented in all four datasets (labels 0 & 1):
  - The local training model was unable to learn all the labels, whereas both the centralized training and the FL model were able to learn all the labels within this group.
  - FL demonstrates F1 scores that are nearly identical to those achieved through centralized training, also FL on each machine surpasses the F1 scores achieved through local training.

- When the labels are only present in some nodes (labels 2-7):
  - With 2 epochs and 25 rounds, FL achieve higher F1 scores than centralized training in 2 epochs.
  - FL shows similar F1 scores (5 epochs and 10 rounds) (except for one class) to centralized in 5 epochs.
  - FL was able to learn all labels except one, whereas centralized training failed to learn two labels.

Note: All models, including local training, were unable to learn the labels of class 7.

Dataset	nev	mel	ack	bcc	sec	df	vasc	scc
Brazil	0.05	0.0	0.13	0.54	0.0	null	null	0.0
Australia	0.59	0.04	0.38	0.28	0.42	0.0	null	null
Austria	0.89	0.29	0.0	0.27	0.10	0.0	0.33	null
Netherlands	0.76	0.5	null	null	null	null	null	null

Table 5.2 F1_score on 5 epochs for each local machine

nev	mel	ack	bcc	sec	df	vasc	scc
0.84	0.24	0.53	0.51	0.39	0.0	0.21	0.0

Table 5.3 F1_score on 5 epochs for the centralized training

Dataset	nev	mel	ack	bcc	sec	df	vasc	scc
Node Brazil	0.258	0.120	0.188	0.467	0.189	nan	nan	0.000
Node Australia	0.609	0.256	0.092	0.351	0.377	0.154	nan	nan
Node Austria	0.879	0.414	0.000	0.437	0.401	0.069	0.261	nan
Node Netherlands	0.743	0.581	nan	nan	nan	nan	nan	nan
Global	0.793	0.349	0.154	0.437	0.351	0.087	0.203	0.000

Table 5.4 F1_score on 5 epochs and 10 rounds for the Federated learning on each machine and in global

Dataset	nev	mel	ack	bcc	sec	df	vasc	scc
Brazil	0.0	0.0	0.0	0.0	0.17	null	null	0.0
Australia	0.54	0.0	0.0	0.0	0.12	0.0	null	null
Austria	0.88	0.16	0.0	0.0	0.06	0.0	0.0	null
Netherlands	0.78	0.21	null	null	null	null	null	null

Table 5.5 F1_score on 2 epochs for each local machine

nev	mel	ack	bcc	sec	df	vasc	scc
0.80	0.32	0.49	0.25	0.27	0.0	0.0	0.0

Table 5.6 F1_score on 2 epochs for the centralized training

Datasets	nev	mel	ack	bcc	sec	df	vasc	scc
Node Brazil	0.331	0.061	0.370	0.492	0.232	nan	nan	0.000
Node Australia	0.615	0.203	0.291	0.371	0.278	0.000	nan	nan
Node Austria	0.896	0.371	0.000	0.406	0.397	0.074	0.528	nan
Node Netherlands	0.811	0.581	nan	nan	nan	nan	nan	nan
Global	0.815	0.317	0.331	0.448	0.325	0.051	0.431	0.000

Table 5.7 F1_score on 2 epochs and 25 rounds for the Federated learning on each machine and in global

In the second configuration, we utilized a high number of epochs (20, 50) for both centralized and local training, while using a low number of rounds (6) for the FL model.

- When the labels are presented in all four datasets (labels 0 & 1):
  - All three experiments were able to learn all labels.
  - Both centralized and FL show similar results. However, when FL was trained on each machine compared to local training, it exhibited lower F1 scores.

- When the labels are only present in some nodes (labels 2-7):
  - FL showed weaker results compared to centralized training.
  - FL exhibited weaker results on each machine, in comparison to local training.
  - FL was unable to learn label 7, while the centralized model was able to detect some labels with a shallow score of 0.1 and .03 with 20 and 50 epochs.

Local Training	nev	mel	ack	bcc	sec	df	vasc	scc
Node Brazil	0.39	0.0	0.56	0.24	0.32	null	null	0.0
Node Australia	0.68	0.13	0.40	0.50	0.39	0.0	null	null
Node Austria	0.89	0.15	0.0	0.21	0.33	0.26	0.56	null
Node Netherlands	0.8	0.43	null	null	null	null	null	null

Table 5.8 F1_score on 20 epochs for each local machine

Local Training	nev	mel	ack	bcc	sec	df	vasc	scc
Node Brazil	0.56	0.31	0.54	0.6	0.48	null	null	0.03
Node Australia	0.69	0.35	0.4	0.49	0.51	0.28	null	null
Node Austria	0.91	0.44	0.4	0.41	0.47	0.34	0.63	null
Node Netherlands	0.87	0.78	null	null	null	null	null	null

Table 5.9 F1_score on 50 epochs for each local machine

nev	mel	ack	bcc	sec	df	vasc	scc
0.84	0.43	0.55	0.51	0.36	0.22	0.5	0.1

Table 5.10 F1_score on 20 epochs for the centralized training

nev	mel	ack	bcc	sec	df	vasc	scc
0.86	0.38	0.51	0.55	0.40	0.47	0.51	0.03

Table 5.11 F1_score on 50 epochs for the centralized training

Dataset	nev	mel	ack	bcc	sec	df	vasc	Scc
Node Brazil	0.376	0.045	0.259	0.450	0.305	nan	nan	0.000
Node Australia	0.640	0.214	0.353	0.478	0.364	0.211	nan	nan
Node Austria	0.869	0.296	0.000	0.301	0.366	0.250	0.560	nan
Node Netherlands	0.550	0.261	nan	nan	nan	nan	nan	nan
Global	0.810	0.248	0.275	0.420	0.350	0.179	0.459	0.000

Table 5.12 F1_score on 20 epochs and 6 rounds for the Federated learning on each machine and in global

For all the experiments we have used the following hyperparameters:

- batch_size: 32
- lr: 0.005
- min_fit_clients: 4

5.4 Conclusions and Future Steps

For some labels and nodes, the federated model performs better than a local model trained on local data. Having access to more data improves the model, providing an incentive to collaborate on federated training.

However, for some other labels, the federated model performs worse. This might be caused by a variety of reasons:

- Insufficient training time: the federated model needs more time to converge.
- Poor strategy: the local models are combined via averaging; less naïve strategies might yield better results.
- Diverse data: some classes might be too different between nodes, "poisoning" the model's understanding.

- Insufficient capacity: only 2 convolutional layers might not be enough to learn all the different data.
- FL needs a good trade-off between the number of epochs and rounds to outperform the Centralized model.

This prototype could be extended in several ways, to better explore the PET space:

- Use a more sophisticated technique for aggregating the federated model
- Use Differential Privacy during training to fully ensure that no model inversion attacks are possible
- Use Secure Multi Party Computation to compute the global metrics, to avoid revealing any individual node's metrics.

6 Final Conclusions

In this JRC technical report, we have provided the motivation for the use of Privacy Enhancing Technologies in a data spaces context, including an overview of their current state, and a practical evaluation for real-world usage. We have shown that PETs are a key technology for resolving the tension between data utility and data privacy, enabling new use cases but also increasing security for existing ones.

Among these technologies, Secure Multi-Party Computation (SMPC), Federated Learning (FL), Differential Privacy (DP), and Anonymization are notable for their high technological readiness level. These methods offer robust solutions for maintaining data privacy while still enabling valuable analysis and insights. SMPC enables computation across multiple parties where the raw data is never revealed to any of the parties involved. Federated Learning allows machine learning models to be trained across numerous devices or servers holding local data samples, without exchanging them. Differential Privacy introduces statistical noise to data or queries to provide plausible deniability, which helps to protect individual's data during analysis. However, Anonymization, despite its widespread use and technological readiness, has significant shortcomings. While it involves removing identifying information from data to protect individuals' privacy, it's increasingly being seen as insufficient due to the growing risk of re-identification. Advances in technology and the availability of diverse datasets have made it possible, in some cases, to re-identify individuals from data that was thought to be anonymous. Therefore, while Anonymization can still be a useful tool in certain contexts, it may not provide the level of privacy protection required in scenarios with high privacy risks.

On the other hand, Homomorphic Encryption (HE), Trusted Execution Environments (TEE) and Zero Knowledge Proofs (ZKP), while offering innovative ways to preserve privacy, are not as mature or widely adopted in practical applications yet. For instance, Homomorphic Encryption is currently computationally expensive and slower than traditional methods, which poses a challenge for its widespread adoption. Similarly, TEEs require specific hardware, and their security heavily relies on the physical protection of the hardware. Zero-Knowledge Proofs, although powerful in theory, are complex to implement and understand, which can pose a barrier to their practical use.

The key takeaway from the analysis presented in this report is that there is no silver bullet: each PET is adapted to different data sharing scenarios, and has different trade-offs (performance, complexity, security, utility). Understanding the strengths and limitations of these technologies is essential for data space actors to select the most appropriate solutions for their specific needs. As always, the choice of PET should be guided by a thorough understanding of the data context, the privacy requirements, and the specific use case. Before choosing a PET, data analysts should know what analysis it enables and data holders should know what protections it offers.

While PETs offer robust solutions for maintaining data privacy, their robustness should be assessed within the context of specific use cases. The effectiveness and resilience of PETs in protecting data privacy depend on various factors such as the implementation, configuration, and adherence to best practices. Robustness should be evaluated through rigorous testing, validation, and continuous monitoring to ensure that the chosen PETs meet the required privacy standards and can withstand potential attacks or vulnerabilities.

Some possible future steps to facilitate the uptake and use of PETs within the context of common European Data Spaces are:

- **Investigate and test the integration of PETs within different sector-specific European Data Spaces**: It is critical to explore how PETs can be integrated into European Data Spaces to maximize both data utility and privacy. Practical implementation of PETs in such environments could allow for secure sharing and analysis of data, opening new possibilities for innovation and research. Pilot projects and real-world testing scenarios will be vital in understanding the challenges and benefits of this integration.
- **Raise awareness of PETs among policymakers**: It's important to bring PETs to the forefront of policy discussions about data privacy and security. Policymakers should be educated on the capabilities and limitations of different PETs, so they can make informed decisions when drafting privacy laws. Ensuring that privacy laws take into account the latest advancements in PETs could lead to more robust and effective legislation that protects individual privacy while enabling data utility.
- **Support academic research on PETs and open-source projects that implement them**: Encouraging and funding research into PETs can help push the boundaries of what's currently possible, leading to the development of more advanced and effective technologies. Open-source

projects are particularly valuable, as they allow for community collaboration and widespread adoption. Providing financial and technical support for these projects can accelerate the development and refinement of PETs.

- **Promote collaboration between academia, industry, and government**: Cross-sector partnerships can lead to more rapid advancement and adoption of PETs. Collaboration could lead to the development of standardized protocols and best practices for using PETs, making it easier for organizations to implement these technologies.
- **Implement training and education programs**: Due to the complexity of PETs, it is crucial to provide comprehensive training and education programs to data analysts, IT professionals, and other data space stakeholders. These programs will equip individuals with the necessary skills and knowledge to utilize PETs effectively and responsibly in their respective domains. By investing in education and promoting a strong understanding of PETs, organizations can maximize their benefits while mitigating risks and ensuring ethical use..
- **Prioritize the development of user-friendly PETs**: To encourage widespread adoption, it is essential to prioritize the development of user-friendly PETs. PETs should be accessible and easy to use, with clear instructions and adequate support. By simplifying the user experience and reducing barriers to adoption, organizations can increase the successful implementation of PETs and promote their broader usage.
- **Establish a clear ethical framework for the use of PETs**: Given the sensitive nature of the data handled by PETs, it is critical to establish a clear ethical framework. This framework should address important aspects such as consent, transparency, accountability, and fairness. By adhering to ethical guidelines, organizations can ensure the responsible and trustworthy use of PETs while safeguarding individual rights and promoting public trust in data sharing initiatives

Appendix A. FL Architecture

This is a Federated Learning simplified architecture of the solution proposed in section **Error! Reference source not found.**. All the tools or software related to data processing have been omitted in order to focus on the federated part. This platform is a client-server architecture, where the client is a software application that in general is installed on-premise on each Data Provider along with its own connector. On the server side, resides the central part of the platform that communicates with all the connectors and acts as a coordinator for all operations, following the guidelines of Data Spaces. Next figure includes the following items:

- The data provider prepares the data (according with the other parties for a common data model) for training.
- The aggregation server (aggregates) the results from the different parties and ships the consolidated results back to the parties. This can go through multiple rounds until a termination criterion is reached.
- The data consumer/data scientist can make a computation request for training a model and receive a computation result from the server.

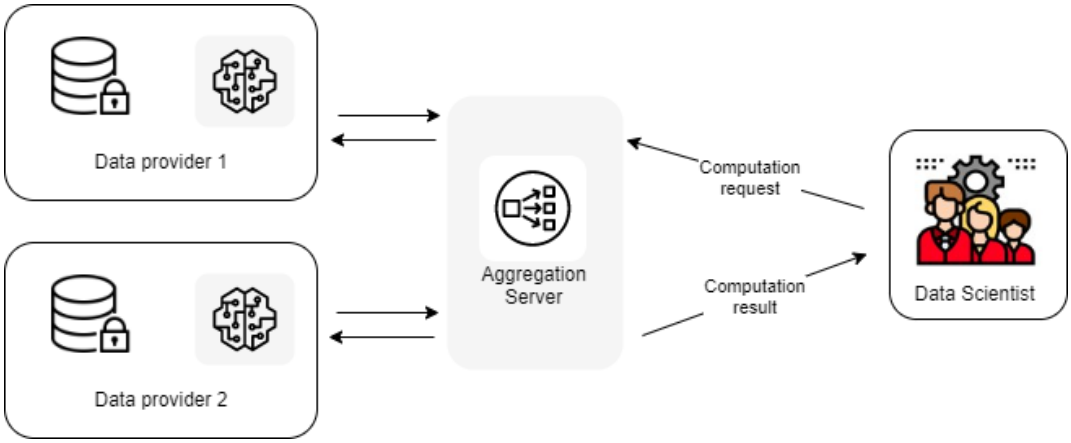

Figure 7. 1 An overview of federated learning

All the components rely on micro-services technology to containerize code, runtime, systems tools, libraries, and settings. The infrastructure, for all participants in the model training, must be configured as a client, which has access to the data. For the sever, must be configured as a server, and needs to have a connection to the clients to share model weights or errors. The central server aggregates the feedback from the participants, and based on predefined criteria, updates the global model. The predefined criteria allow the model to evaluate the quality of the feedback and therefore to only incorporate that which is value-adding.

Appendix B. SMPC Architecture

This is a schematic representation of the secure multiparty computation scenario. All the tools or software related to data processing have been omitted in order to focus on the computation part. This platform consists of several computing machines installed on-premises at each Data Provider. Next figures include the following items:

- The data provider prepares the data (according with the other parties for a common format) for computation.
- The computation server on which the operations are performed.
- The data consumer / data scientist can make a computation request for an operation.

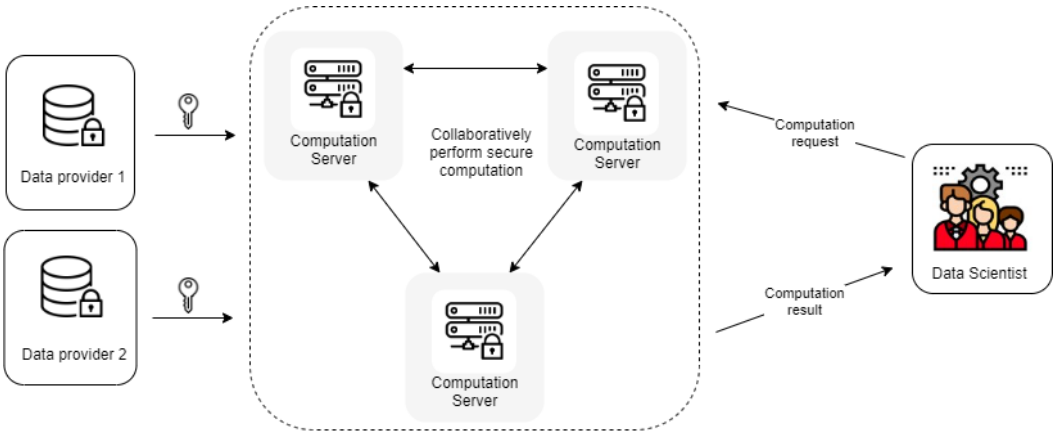

Figure 7. 2 An overview of SMPC

All the components rely on micro-services technology to containerize code, runtime, systems tools, libraries, and settings. The infrastructure, for all participants in the computation, must have a computation server with the data loaded for secret share to the rest of computation servers. The data scientist obtains the results, combining the individual secret shares without reveal any information about the inputs.

Appendix C. DP Architecture

This is a schematic representation of the differential privacy scenario. All the tools or software related to data processing have been omitted in order to focus on the security computation part. This platform consists of piece of software that adds noise to the original input datasets provided by the Data Providers. Next figure includes the following items:

- The data provider give access to the databases/datasets on which to perform queries.
- The aggregator noise is a piece of software that add noise to the output.
- The data analyst can make a query to the databases/datasets with a noise.

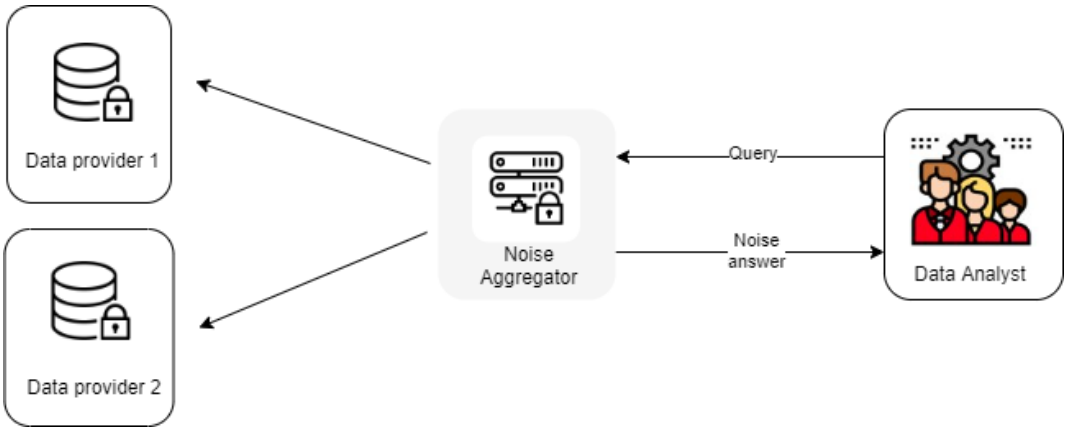

Figure 7. 3 An overview of DP

All the components rely on micro-services technology to containerize code, runtime, systems tools, libraries, and settings. The infrastructure must contain a piece of software, in this case called noise aggregator, which is in charge of executing the algorithm to add noise to the output. The Data Analyst obtain the results with the noise that has been added to obtain privacy on the input data.

Appendix D. HE Architecture

This is a schematic representation of the homomorphic encryption scenario. All the tools or software related to data processing have been omitted in order to focus on the security computation part. This platform consists of a secure server which performs secure computation on encrypted data and returns to the data providers/Data consumers the encrypted result of the operation. Next figure includes the following items:

- The data provider encrypts the data with a private key to be able to perform a computation in a secure server.
- The secure server is the one that performs the computation on the encrypted values received, returning the encrypted results to the different computation participants.
- The data consumer can send a secure operation to the secure server using the encrypted data and can decrypt the result of the operation.

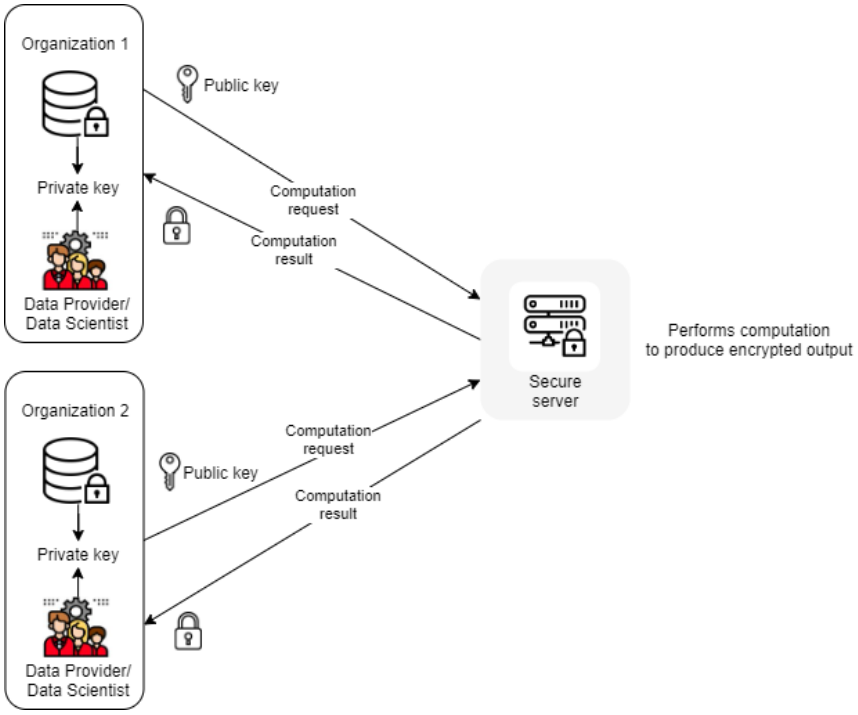

Figure 7. 4 An overview of HE

All the components rely on micro-services technology to containerize code, runtime, systems tools, libraries, and settings. The infrastructure must contain a server where the secure computation is performed on the encrypted data, applying the algorithm/model/computation to them. The data consumer obtains the results encrypted and can decrypt the results with the private key.

References

1. European Commission, «European Strategy for Data, Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions» COM(2020) 66 final, European Commission Brussels, Belgium, 2020.
2. «European Data Protection Supervisor» [Online]. Available:https://edps.europa.eu/data-protection/data-protection/glossary/p_en#:~:text=handful%20of%20entries.-,PETs, The%20acronym%20%27PETs.
3. «Azure Data Share terminology. Retrieved 8 July 2022, from,» Azure, 1 June 2022. [En línea]. Available: https://docs.microsoft.com/en-us/azure/data-share/terminology. [Accessed: 8 July 2022].
4. B. Otto, S. Steinbuss y S. Lohmann, «IDS Reference Architecture Model,» Apr 2019. [En línea]. Available: https://doi.org/10.5281/zenodo.5105529. [Accessed: 03 Aug 2022].
5. D. Evans, V. Kolesnikov y M. Rosulek, «A Pragmatic Introduction to Secure Multi-Party Computation,» Foundations and Trends® in Privacy and Security, vol. 2, 2018.
6. Y. Lindell, «Secure Multipary Computation (MPC),» Communications of the ACM, vol. 64, nº 1, 2021, pp. 86-96,.
7. S. Yakoubov, «A gentle introduction to Yao's Garbled circuits,», 2017, [Online]. Available: https://web. mit. edu/sonka89/www/papers/2017ygc. pdf, 2017.
8. A. Shamir, «How to share a secret,» Communications of the ACM, vol. 22, nº 11, p. 612–613, 1979.
9. D. Beaver, «Efficient Multiparty Protocols Using Circuit Randomization,» de Advances in Cryptology --- CRYPTO '91", Springer Berlin Heidelberg, 1992, pp. 420--432.
10. P. Kairouz, «Advances and Open Problems in Federated Learning,» Foundations and Trends® in Machine Learning, vol. 14, 2021.
11. Hardy y Henecka, «Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption,» arXiv preprint arXiv:1711.10677.
12. C. Dwork y A. Roth, «The Algorithmic Foundations of Differential Privacy,» Foundations and Trends® in Theoretical Computer Science, Volume 9, Issue 3–4, vol. 9, nº 3-4, 2014, pp. 211-407.
13. K. Nissim, «Differential privacy: A primer for a non-technical audience,» Vanderbilt Journal of Entertainment & Technology Law, vol. 21, nº 1, 2018, pp. 209-275.
14. R. Rivest, L. Adleman y M. L. Dertouzos, «On data banks and privacy homomorphism,» Foundations of secure computation, vol. 4, nº 11, 1978, pp. 169-180.
15. W. Diffie y M. E. Hellman, «New directions in cryptography,» ieee transaction on information theory, vol. 22, nº 6, 1976.
16. R. L. Rivest, A. Shamir y L. Adleman, «A method for obtaining digital signatures and public-key cryptosystems,» Communications of the ACM, vol. 21, nº 2, 1978, pp. 120-126.
17. C. Gentry, A fully homomorphic encryption scheme, Stanford University, 2009.
18. T. ElGamal, «A public key cryptosystem and a signature scheme based on discrete logarithms,» IEEE transactions on information theory, vol. 31, nº 4, 1985, pp. 469-472.
19. P. Paillier, «Public-key cryptosystems based on composite degree residuosity classes,» In International conference on the theory and applications of cryptographic techniques, Springer, 1999, pp. 223-238.
20. J. H. Cheon, A. Kim, M. Kim y Y. Song, «Homomorphic encryption for arithmetic of approximate numbers,» 2017.
21. I. Chillotti, N. Gama, M. Georgieva y M. Izabachène, «TFHE: fast fully homomorphic encryption over the torus,» Journal of Cryptology, vol. 33, nº 1, 2020, pp. 34-91.
22. Mogre, Agarwal y Patil, «A review on data anonymization technique for Data publishing,» International Journal of Engineering Research & Technology, 2012.

23. I. E. Olatunji, J. Rauch, M. Katzensteiner y M. Khosla, «A review of anonymization for healthcare data,» Big Data, 2022.
24. Ghinita, «Fast data anonymization with low information loss,» In Proceedings of the 33rd international conference on Very large data bases, 2007.
25. M. Sabt, M. Achemlal y A. Bouabdallah, «Trusted execution environment: what it is, and what it is not,» IEEE Trustcom/BigDataSE/ISPA, vol. 1, 2015, pp. 57-64.
26. J. Ménétrey, C. Göttel, M. Pasin, P. Felbel y V. Schiavoni, «An Exploratory Study of Attestation Mechanisms for Trusted Execution Environments,» arXiv preprint arXiv:2204.06790, 2022.
27. S. M. C. R. Shafi Goldwasser, «The knowledge complexity of interactive proof-systems,» SIAM Journal on computing, 1989, pp. 186-208.
28. F. M. Blum, «Non-interactive zero-knowledge and its applications,» de STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing, 1988.
29. [En línea]. Available: https://transparency.dev/verifiable-data-structures/.
30. F. F. Shamir, «Zero-Knowledge Proofs of Identity,» In STOC '87: Proceedings of the nineteenth annual ACM symposium on Theory of computing, 1987.
31. C. Yun, "Programmable Constraint Systems for Bulletproofs," 19 Nov 2018. [Online]. Available: https://medium.com/interstellar/programmable-constraint-systems-for-bulletproofs-365b9feb92f7. [Accessed 4 Aug 2022].
32. "A system for verifying outsourced computations, and applying SNARKs. Simplified release of the main Pepper codebase," [Online]. Available: https://github.com/pepper-project/pequin. [Accessed 4 Aug 2022].
33. M. von Maltitz, H. Ballhausen, D. Kaul, D. Fleischmann, M. Niyazi, C. Belka y G. Carle, «A Privacy-Preserving Log-Rank Test for the Kaplan-Meier Estimator With Secure Multiparty Computation: Algorithm Development and Validation,» JMIR Med Inform, 2021.
34. Y. Lindell y B. Pinkas, «Secure Multiparty Computation for Privacy-Preserving Data Mining,» IACR Cryptology ePrint Archive, 2008.
35. N. Kumar, M. Rathee, N. Chandran, D. Gupta, A. Rastogi y R. Sharma, «CrypTFlow: Secure TensorFlow Inference,» IEEE Symposium on Security and Privacy, 2020.
36. P. Bogetoft, «Secure Multiparty Computation Goes Live,» Financial Cryptography and Data Security, 2009.
37. D. Bogdanov, L. Kamm, B. Kubo, R. Rebane, V. Sokk y R. Talviste, «Students and Taxes: a Privacy-Preserving Social Study Using Secure Computation,» IACR Cryptol. ePrint Arch., 2015, p. 1159.
38. A. Lapets, F. Jansen, K. D. Albab, R. Issa, L. Qin, M. Varia y A. Bestavros, «Accessible Privacy-Preserving Web-Based Data Analysis for Assessing and Addressing Economic Inequalities,» Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies, 2018.
39. "https://www.win.tue.nl/~berry/mpyc/," [Online]. Available: https://www.win.tue.nl/~berry/mpyc/.
40. "https://github.com/data61/MP-SPDZ," [Online]. Available: https://github.com/data61/MP-SPDZ.
41. "https://crypten.ai/," [Online]. Available: https://crypten.ai/.
42. "https://tf-encrypted.io/," [Online]. Available: https://tf-encrypted.io/.
43. "https://github.com/multiparty/jiff," [Online]. Available: https://github.com/multiparty/jiff.
44. "https://github.com/facebookresearch/fbpcf,"[Online]. Available:https://github.com/facebookresearch/fbpcf.
45. M. Movahedi, B. M. Case, A. Knox, J. Honaker, L. Li, Y. P. Li, S. Saravanan, S. Sengupta y E. Taubeneck, «Privacy-Preserving Randomized Controlled Trials: A Protocol for Industry Scale Deployment,» 2021.
46. "https://sharemind.cyber.ee/," [Online]. Available: https://sharemind.cyber.ee/.
47. "https://github.com/rdragos/awesome-mpc," [Online]. Available: https://github.com/rdragos/awesome-mpc.

48. “https://github.com/MPC-SoK/frameworks,” [Online]. Available: https://github.com/MPC-SoK/frameworks.
49. Q. Xia, W. Ye, Z. Tao, J. Wu y Q. & Li, «A survey of federated learning for edge computing: Research problems and solutions,» High-Confidence Computing, 2021.
50. Y. Ma, X. Zhu y J. Hsu, «Data poisoning against differentially-private learners: Attacks and defences,» International Joint Conference on Artificial Intelligence (IJCAI), 2019.
51. Geyer, Klein y Nabi, «Differentially private federated learning: A client level perspective,» 2017.
52. McMahan, Ramage, Talwar y Zhang, «Learning differentially private recurrent language model,» International Conference on Learning Representations (ICLR), 2018.
53. Lee y Clifton, «How Much Is Enough? Choosing epsilon for Differential Privacy,» Information Security, 2011.
54. Kasiviswanathan y Rudelson, «The price of privately releasing contingency tables and the spectra of random matrices with correlated rows,» Proceedings of the 42nd ACM symposium on Theory of computing, 2010.
55. C. Lee, «Differential identifiability,» Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, 2012.
56. Kasiviswanathan y Smith, «A Note on Differential Privacy: Defining Resistance to Arbitrary Side Information,» CoRR abs/0803.3946, 2008.
57. Dankar y Khaled, «Practicing differential privacy in health care: A review,» Trans. Data Priv., 2013.
58. S. Rass y D. Slamanig, Cryptography for Security and Privacy in Cloud Computing, USA: Artech House, Inc., 2013.
59. V. Sidorov, . E. Y. F. Wei y W. K. Ng, «Comprehensive Performance Analysis of Homomorphic Cryptosystems for Practical Data Processing,» arXiv, 2022.
60. A. Chatterjee y . K. M. M. Aung,, Fully homomorphic encryption in real world applications, Springer, 2019.
61. Apple, “Password Monitoring,” Apple, 18 February 2021. [Online]. Available: https://support.apple.com/guide/security/password-monitoring-sec78e79fc3b/web. [Accessed 13 October 2022].
62. L. Valenta, S. Cohney, A. Liao, J. Fried, S. Bodduluri y N. Heninger, «Factoring as a service,» de International Conference on Financial Cryptography and Data Security, 2016.
63. K. El Emam, E. Jonker, L. Arbuckle y B. Malin, «A Systematic Review of Re-Identification Attacks on Health Data,» 2011.
64. A. Narayanan y V. Shmatikov, «How To Break Anonymity of the Netflix Prize Dataset,» arXiv, 2006.
65. E. AEPD, «AEPD-EDPS joint paper on 10 misunderstandings related to anonymisation,» 2021.
66. P. Ohm, «Broken Promises of Privacy: Responding to the Surprising Failure of Anonymization,» UCLA Law Review, vol. 57, 2010, p. 1701.
67. N. Li, W. Qardaji y D. Su, «On Sampling, Anonymization, and Differential Privacy: Or, k-Anonymization Meets Differential Privacy,» 2011.
68. M. A. A. B. Mohamed Sabt, «Trusted Execution Environment: What It is, and What It is Not.,» 4th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, 2015.
69. “Intel Software Guard Extensions (Intel SGX),” Intel, [Online]. Available: https://www.intel.com/content/www/us/en/developer/tools/software-guard-extensions/overview.html.
70. “TrustZone,” [Online]. Available: https://www.arm.com/technologies/trustzone-for-cortex-m.
71. «AMD Pro Security,» [En línea]. Available: https://www.amd.com/en/technologies/pro-security.
72. P. N. B. J. B. Alexander Nilsson, «A Survey of Published Attacks on Intel SGX,» arXiv, 2020.
73. S. W. D. G. Michael Schwarz, «Practical Enclave Malware with Intel SGX,» arXiv, 2019.

74. O. Goldreich y Y. Oren, «Definitions and properties of zero-knowledge proof systems,» Journal of Cryptology, 1994.

75. F. Zhang, X. Fan, P. Zhou y W. Zhou, «Zero knowledge proofs for cloud storage integrity checking,» arXiv.

76. Y. Yong, L. Yannan, M. H. Au, W. Susilo y C. K.-K. Raymond, «Public Cloud Data Auditing with Practical Key Update and Zero Knowledge Privacy,» Liu, J., Steinfeld, R. (eds) Information Security and Privacy. ACISP 2016. Lecture Notes in Computer Science, vol. 9722, 2016.

77. J. Zhang, Z. Fang, Y. Zhang y D. Song, «Zero Knowledge Proofs for Decision Tree Predictions and Accuracy,» Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, 2020, p. 2039–2053.

78. T. Liu, X. Xie y Y. Zhang, «zkCNN: Zero Knowledge Proofs for Convolutional Neural Network Predictions and Accuracy,» Cryptology ePrint Archive, 2021.

79. B. Parno, J. Howell, C. Gentry y M. Raykova, «Pinocchio: Nearly Practical Verifiable Computation,» 2013 IEEE Symposium on Security and Privacy, 2013, pp. 238-252.

80. E. Brickell, J. Camenisch y L. Chen, «Direct Anonymous Attestation,» Proceedings of the 11th ACM Conference on Computer and Communications Security, 2004, p. 132–145.

81. N. P. X. Z. W. F. Dubovitskaya A, «Applications of Blockchain Technology for Data-Sharing in Oncology: Results from a Systematic Literature Review,» 2020.

82. E. Ben Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer y M. Virza, «Zerocash: Decentralized Anonymous Payments from Bitcoin,» 2014 IEEE Symposium on Security and Privacy, 2014, pp. 459-474.

83. "https://github.com/sec-bit/zkPoD-node," [Online]. Available: https://github.com/sec-bit/zkPoD-node.

84. A. Faz-Hernández, W. Ladd y D. Maram, «ZKAttest: Ring and Group Signatures for Existing ECDSA Keys,» Cryptology ePrint Archive, 2021.

85. J. B. Almeida, E. Bangerter, M. Barbosa, S. Krenn, A.-R. Sadeghi y T. Schneider, «A Certifying Compiler for Zero-Knowledge Proofs of Knowledge Based on Σ-Protocols,» Cryptology ePrint Archive, 2010.

86. "https://zkproof.org/," [Online]. Available: https://zkproof.org/.

87. U. Maurer, «Unifying Zero-knowledge Proofs of Knowledge,» Advances in Cryptology - AfricaCrypt 2009, 2009.

88. "https://github.com/scipr-lab/libsnark," [Online]. Available: https://github.com/scipr-lab/libsnark.

89. "https://github.com/spring-epfl/zksk," [Online]. Available: https://github.com/spring-epfl/zksk.

90. "https://github.com/meilof/pysnark," [Online]. Available: https://github.com/meilof/pysnark.

91. "https://github.com/pepper-project/pequin," [Online]. Available: https://github.com/pepper-project/pequin.

92. "https://github.com/miguelmota/merkletreejs," [Online]. Available: https://github.com/miguelmota/merkletreejs.

93. "https://github.com/mit-dci/zkledger," [Online]. Available: https://github.com/mit-dci/zkledger.

94. "https://github.com/matter-labs/awesome-zero-knowledge-proofs," [Online]. Available: https://github.com/matter-labs/awesome-zero-knowledge-proofs.

95. "https://zkp.science/," [Online]. Available: https://zkp.science/.

96. Hwang, Park y Jin, «Development and validation of a deep learning-based automated detection algorithm for major thoracic diseases on chest radiographs.,» JAMA Netw Open, 2019.

97. Chen, Szolovits y Ghassemi, «Can AI help reduce disparities in general medical and mental health care?,» AMA J Ethics, 2019.

98. Obermeyer, Powers, Vogeli y Mullainathan, «Dissecting racial bias in an algorithm used to manage the health of populations.,» Science, 2019.

99. Buolamwini y Gebru, «Gender shades: intersectional accuracy disparities in commercial gender classification,» PMLR, 2018.

100. Seyyed-Kalantari, Liu, McDermott, Chen y Ghassemi, «CheXclusion: fairness gaps in deep chest X-ray classifiers,» Pac Symp Biocomput, 2021.

101. P. Kairouz, «Advances and Open Problems in Federated Learning,» Foundations and Trends® in Machine Learning, vol. 14, 2021.

102. O. B., S. S., T. A. y L. S. e. al., «IDS Reference Architecture Model (Version 3.0),» 2019.

103. Yang, Qiang, et al. "Federated machine learning: Concept and applications." ACM Transactions on Intelligent Systems and Technology (TIST) 10.2 (2019): 1-19.

104. «Gaia-X: A Federated Secure Data Infrastructure,» 2022. Available: https://gaia-x.eu/. [Accessed: 2023 February 7].

105. European Comission, «Simpl: cloud-to-edge federations and data spaces made simple. (2022).,» European Comission, 30 May 2022. [En línea]. Available: https://digital-strategy.ec.europa.eu/en/news/simpl-cloudedge-federations-and-data-spaces-made-simple. [Accessed: 7 February 2023].

106. J. M. Saborit-Torres, “Medical imaging data structure extended to multiple modalities and anatomical regions,” arXiv, 2020.

List of Abbreviations

CNN	Convolutional Neural network
FL	Federated Learning
IID	Independent and Identically Distributed (data)
PETs	Privacy Enhancing Technologies
PSI	Private Set Intersection
SMPC	Secure Multi-Party Computation
ZKP	Zero-Knowledge Proof

List of Figures

List of Tables

GETTING IN TOUCH WITH THE EU

In person

All over the European Union there are hundreds of Europe Direct centres. You can find the address of the centre nearest you online (european-union.europa.eu/contact-eu/meet-us_en).

On the phone or in writing

Europe Direct is a service that answers your questions about the European Union. You can contact this service:

— by freephone: 00 800 6 7 8 9 10 11 (certain operators may charge for these calls),

— at the following standard number: +32 22999696,

— via the following form: european-union.europa.eu/contact-eu/write-us_en.

FINDING INFORMATION ABOUT THE EU

Online

Information about the European Union in all the official languages of the EU is available on the Europa website (european-union.europa.eu).

EU publications

You can view or order EU publications at op.europa.eu/en/publications. Multiple copies of free publications can be obtained by contacting Europe Direct or your local documentation centre (european-union.europa.eu/contact-eu/meet-us_en).

EU law and related documents

For access to legal information from the EU, including all EU law since 1951 in all the official language versions, go to EUR-Lex (eur-lex.europa.eu).

Open data from the EU

The portal data.europa.eu provides access to open datasets from the EU institutions, bodies and agencies. These can be downloaded and reused for free, for both commercial and non-commercial purposes. The portal also provides access to a wealth of datasets from European countries.

Science for policy

The Joint Research Centre (JRC) provides independent, evidence-based knowledge and science, supporting EU policies to positively impact society

EU Science Hub
joint-research-centre.ec.europa.eu

@EU_ScienceHub
EU Science Hub - Joint Research Centre
EU Science, Research and Innovation
EU Science Hub
@eu_science

Publications Office
of the European Union