Proceedings Paper:
Brennaf, M.S., Yang, P. and Lanfranchi, V. (2025) Privacy enhanced federated learning in encrypted anonymous personal device domain. In: Alfian, G., Oktiawati, U.Y., Saputra, Y.M. and Pratama, C., (eds.) Engineering Headway. The 10th International Conference on Science and Technology (ICST), 23-24 Oct 2024, Yogyakarta, Indonesia. Trans Tech Publications Ltd, pp. 3-12. ISSN: 2813-8325. EISSN: 2813-8333.

https://doi.org/10.4028/p-erhli5

Privacy Enhanced Federated Learning in Encrypted Anonymous Personal Device Domain

1st Muhammad Senoyodha Brennaf
Department of Computer Science
University of Sheffield
Sheffield, United Kingdom
msbrennaf1@sheffield.ac.uk

2nd Po Yang
Department of Computer Science
University of Sheffield
Sheffield, United Kingdom
po.yang@sheffield.ac.uk

3rd Vitaveska Lanfranchi
Department of Computer Science
University of Sheffield
Sheffield, United Kingdom
v.lanfranchi@sheffield.ac.uk

***Abstract*—The increase in privacy concerns and the introduction of privacy and data protection legislation compel organisations to reevaluate their practices regarding traditional machine learning. The aggregation and management of users' private data on the central server may contravene regulations if not properly administered. Federated learning provides a technique that eliminates the necessity of uploading users' data to the server. It facilitates substantial learning by collaboratively training on each client's devices and pooling the model gradient changes. Federated learning, augmented with a proxy as an intermediary and encrypted model parameters, will enhance anonymity, privacy, and data protection against malicious threats, including membership inference adversaries. Nonetheless, encrypted data incurs costs for customers' communication and data size that exceed twice the original size. Our paper seeks to resolve these issues. We present two secure approaches for effective communication in an anonymous encrypted federated learning framework as our contribution. Additionally, our experiments demonstrated that it is feasible to attain equivalent communication costs as in non-encrypted scenarios. We provide recommendations in the conclusion for the effective implementation of privacy-preserving federated learning in the area of personal devices.**

***Index Terms*—encryption, federated learning, privacy, proxy, anonymous**

I. Introduction

On-device training and inference have gained popularity as a prevailing trend. For instance, utilizing a smartphone or smart gadget to obtain a health inference[1]. Nevertheless, this pattern also gives rise to a research dilemma in which data stored on personal devices are often segregated due to many privacy concerns. Merely uploading a user's data for the purpose of making an inference can be subject to various risks, such as security breaches, violation of privacy regulations (e.g. GDPR [1]), or failure to ensure privacy protection for sensitive information. Nevertheless, by engaging in collaborative federated learning (FL) [2], users' devices can be trained together to generate a superior global model and attain a high level of accuracy. The utilisation of aggregation in FL empowers users to privately do machine learning tasks on their own gadgets, hence obviating the necessity of transmitting private information to a central server.

However, this privacy-preserving FL is not completely secure. The membership inference attack is a specific type of attack that allows the server to ultimately determine the identity of a client by analysing their gradient change [3]. Several privacy enhancement algorithms have been suggested to counteract this detrimental practice. These include 1) Differential privacy [4], which involves altering actual data to protect privacy; 2) Secure aggregation [5], which allows to securely aggregate the clients' model parameters; and 3) Homomorphic encryption [6], which enables calculations to be performed on encrypted model updates. Nevertheless, there are certain limitations associated with them in specific scenarios. For instance, they may exhibit accuracy problems when it comes to differential privacy [7], encounter dropout concerns for secure aggregation [8], and prove to be computationally expensive for HE [9], [10].

This work introduces a novel approach called anonymous FL with proxied privacy enhancement, which serves as an alternate method to enhance privacy in FL environments. This method allows for collaborative learning while maintaining privacy, as it ensures that neither the server nor the aggregator gains knowledge about the origin of the model updates. This is particularly useful for personal devices with unique properties. Our contributions include:

- The development of two approaches for anonymous FL with proxies, one based on designated client selection and the other on voluntary selection. These approaches are secure, cost-effective, and do not compromise accuracy.
- We have conducted an analysis and made improvements on the aspects of communication and computation costs. We have demonstrated that achieving a low cost of communication cost for encoded parameter updates under an unidentified FL environment is feasible based on our tests. Moreover, the compressed data yielded a reduced size compared to the original data, reducing clients' memory usage by up to 60%.

[1]https://foodmarble.com/

- We have also provided solutions to address various concerns, such as the possibility of collusion between the server and proxy. We suggest enhancing the privacy and data security measures under an FL system by using an anonymiser such as a proxy and implementing a combination of asymmetric encryption and compression. This approach offers advantages such as reduced communication cost, less memory utilisation, and enhanced security. However, this approach is suggested for tasks where quick inference is not the main objective.

The work is structured into six chapters: Chapter 1 introduces the topic, Chapter 2 discusses the background research, Chapter 3 outlines the technique, Chapter 4 describes the investigation setup, Chapter 5 evaluates the results, and Chapter 6 wraps up the tests and offers suggestions.

II. Background Research

A. Federated Learning

Google introduced federated learning (FL) [2], [11], [12] as an innovative method for addressing issues related to data security and privacy. FL is the process of training a shared model using a centralised server, where the data is distributed over a wide variety of users and the models are aggregated. FL is a decentralised approach where the users work together to address an ML task [3], [13].

Federated learning operates in a collaborative manner, with the training and testing procedures taking place on the client's side rather than on the server's side. Initially, the server will select a subset of clients and allocate them a global model for a certain task. Subsequently, each individual client proceeds to train the model by utilising their respective confidential information. Subsequently, the client transmits the generated model update, not for the client's private data, to the server. The server consolidates the incoming parameter updates and then generates a new global model. Following that, the process is iterated for another cycle until it achieves a specific threshold or fulfils the requirements, such as achieving high accuracy.

B. Additional Privacy Guarantee Through Anonymity

Oblivious DNS over HTTPS (ODoH) is a cutting-edge internet technology that aims to safeguard user privacy while maintaining optimal performance [14]. FL with proxy, drawing inspiration from ODoH, holds the potential to provide clients with enhanced anonymity and stronger privacy assurances. In contrast to earlier privacy enhancement methods, this method is anticipated to be devoid of any accuracy drawbacks, dropout problems, or computationally burdensome disadvantages. Furthermore, it is projected to work effectively in a variety of client configurations, whether few or numerous.

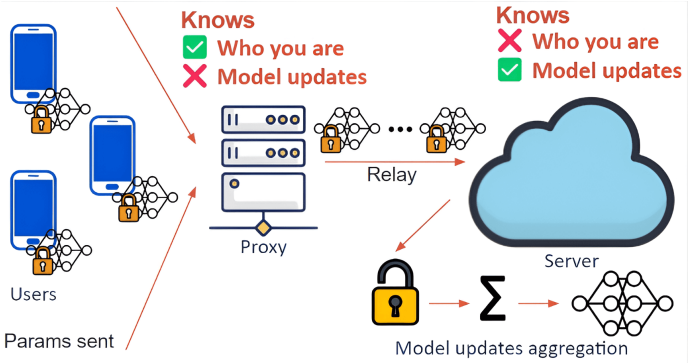

Fig. 1: High concept of anonymous model updates through a proxy

Several studies have been conducted on the implementation of anonymisation techniques in the field of ML. [14] devised a technique for obfuscating user identity during internet access by utilising a fortified intermediary server within an encrypted transaction. [15] presented a method for pseudonymisation in Recommendation-as-a-Service (RaaS) that involves using two layers of proxy. ProxyFL [16] facilitates the exchange of models via proxy in a decentralised FL system while also including differential privacy to enhance privacy. [17] offers FedKAD, an innovative framework for FL that utilises local Knowledge Aggregation on high-level feature maps and Knowledge Distillation. Finally, [18] shows disparities between modern web browser in personal FL setups. Our research proposes two secure approaches in anonymous encrypted FL without compromising accuracy, bandwidth, and privacy.

III. Methodology

For the purpose of enhancing privacy, we utilise a reliable intermediary proxy in our experiments to facilitate anonymous FL. To bolster the safety of information and prevent the proxy from accessing the content, we utilise encryption techniques such as symmetric and asymmetric encryption to protect the transferred data. Symmetric encryption utilises one key for both the encryption and decryption processes. Conversely, asymmetric encryption utilises a pair of keys consisting of a public key and also a private one. A compression technique may be employed to enhance the efficiency of communication cost, particularly for personal devices with restricted bandwidth. This work introduces two approaches: Two-Stage Communication (2SC), which follows a server-client-server pattern, and Three Stage Communication (3SC), which follows a client-server-client-server design.

A. Two-stage Communication

The approaches are described in Fig. 2 and Algorithm 1. The aggregate function utilised here is based on the original FedAvg algorithm [11].

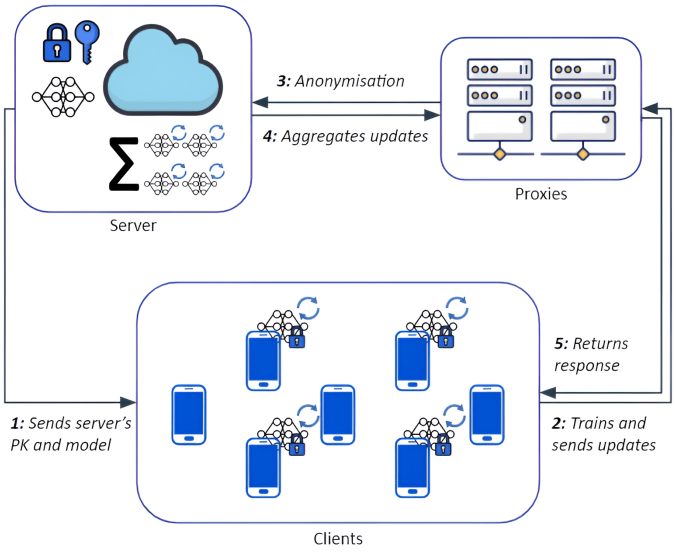

Fig. 2: Two-stage communication anonymised FL

Algorithm 1 `Two-Stage Communication`. The users are denoted by U, indexed by u, and anonymised into x. The server determines the selection of the aggregation function and the specifications for model training.

Run on the server:
initialise gm_0, PK, and SK
for each round i = 1, 2, ... **do**
  $U_i \leftarrow$ (random set of users)
  for each user $u \in U_i$ **in parallel do**
    $e^x_{i+1} \leftarrow \text{UserTrain}(u, gm_i, PK)$
    $gm^x_{i+1} \leftarrow \texttt{decrypt}(e^x_{i+1}, SK)$
  end for
  $gm_{i+1} \leftarrow$ run an aggregate function for all gm^x_{i+1}
end for

UserTrain(u, gm, PK)**:** *// Run on client* u
$gm \leftarrow$ run a model training locally
$e \leftarrow \texttt{encrypt}(gm, PK)$
$P \leftarrow$ (an arbitrary proxy)
$S \leftarrow$ (the server's address)
ProxyForward(P, e, S)

ProxyForward(P, e, S)**:** *// Run on proxy* P
remove source's identity
forward e to S

In general, the server generates a public key (PK) and a secret key (SK). The server selects certain clients for the training round and sends the PK along with the global model to the clients. The client trains the model on their local device. The client encrypts their model update (gm) using the provided PK. The client then chooses a random proxy and sends the encrypted gm along with the server's address to the proxy. The proxy authenticates the request, strips the client's identity, and forwards the gm to the server. Subsequently, the server authenticates the request and decrypts the content using the secret key (SK). The server then aggregates gm to create a new global model or to get ready for the next training phase. The server responses are ultimately transmitted to the client through the proxy.

Users are accountable for choosing a reliable or arbitrary proxy to minimise the chances of collusion between the proxy and server. Validation of each request is required by both the server and proxy. Subsequently, a response is expected from them to indicate the success or failure of a request.

B. Three-stage Communication

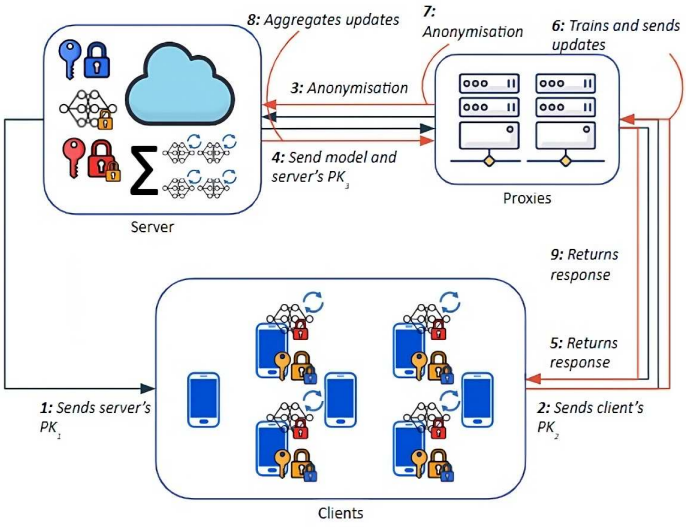

Fig. 3: Three-stage communication anonymised FL

Unlike the 2SC approach, the 3SC scheme initiates with a request from voluntary clients to the server. Fig. 3 and algorithm 2 provide an end-to-end explanation of the procedures.

The primary objective of the 3SC technique is to enhance privacy protection by using stronger encryption measures, which can potentially impact performance and result in higher computational expenses. It is imperative not to reuse any of the generated key pairs to maintain enhanced security. To further improve authentication and safeguard against unauthorised modification and fabrication of the transmitted model update, the server can generate a distinct key pair for each client during the model training phase.

Algorithm 2 `Three-Stage Communication`. The users are denoted by U, indexed by u, and anonymised into x. The server determines the selection of the aggregation function and the specifications for model training.

Run on the server:
initialise gm_0, PK_s, and SK_s
for each round i = 1, 2, ... **do**
initialise PK_i, and SK_i
$U_i \leftarrow$ (random set of voluntary users)
for each user $u \in U_i$ **in parallel do**
$ePK_u \leftarrow$ RequestModel(u, PK_s)
$PK_u \leftarrow \texttt{decrypt}(ePK_u, SK_s)$
$eGM_i^u \leftarrow \texttt{encrypt}(gm_i, PK_u)$
$e_{i+1}^x \leftarrow$ UserTrain(u, eGM_i^u, PK_i)
$gm_{i+1}^x \leftarrow \texttt{decrypt}(e_{i+1}^x, SK_i)$
end for
$gm_{i+1} \leftarrow$ run an aggregate function for all gm_{i+1}^x
end for

RequestModel(u, PK_s)**:** *// Run on client* u
initialise PK_u and SK_u
$e \leftarrow \texttt{encrypt}(PK_u, PK_s)$
$P \leftarrow$ (an arbitrary proxy)
$S \leftarrow$ (the server's address)
ProxyForward(P, e, S)

UserTrain(u, eGM_u, PK_i)**:** *// Run on client* u
$gm \leftarrow \texttt{decrypt}(eGM_u, SK_u)$
$gm_2 \leftarrow$ run a model training locally using gm
$e \leftarrow \texttt{encrypt}(gm_2, PK_i)$
$P \leftarrow$ (an arbitrary proxy)
$S \leftarrow$ (the server's address)
ProxyForward(P, e, S)

ProxyForward(P, e, S)**:** *// Run on proxy* P
remove source's identity
forward e to S

IV. Experiments

The experiments are performed on modern web browsers that are supported in order to ensure compatibility and interoperability across different personal devices. Utilising web browsers for learning has numerous benefits, such as convenient plug-and-play functionality without the need for installations or drivers, extensive compatibility across various platforms including PCs, laptops, and handheld devices, and sufficient access for browsers to utilise the machine's resources, such as memory, sensors, and GPU.

The tests are executed within a single local network, involving several clients, a proxy, and a server. The proxy and server will operate on a personal computer, while the clients will operate on a smartphone and a personal computer. Both devices represent personal devices. Clients will be represented by web browser instances. In order to expedite the process and optimise the performance, we enable supporting GPU-settings on the browsers. The studies utilise TensorFlow.js and DISCO.js [19] for the learning platform backbone, FedAVG [11] for the aggregation algorithm, and an optimised CNN model by Chris's [20] with MNIST from Kaggle dataset for the training rounds.[2]

Regarding the initial experiment, our objective is to analyse the utilisation of a secure proxy in FL, employing the 3SC approach. We perform experiments on all browsers across three diverse environments: a PC with only a CPU, a PC with a GPU, and a mobile phone. We conduct three distinct experiments: the initial FL without a proxy, a proxy employing Crypto's AES symmetric message encryption, and a proxy utilising TweetNaCl's key pair asymmetric message encryption. When it comes to web browsers, we commonly utilise popular options such as Chrome, Edge, and Firefox on personal computers. Regarding the mobile phone, we utilise Chrome Mobile, Edge Mobile, Firefox Mobile, Firefox Nightly Mobile, and Samsung Mobile. We seek to assess the performance and efficiency of the setup. Regarding mobile phone browsers, we utilise a smaller set of data to generate faster results.

Strengthening the model updates with an additional layer of encryption provides heightened security. Nevertheless, the size of the data would increase as a result of the data translation. Thus, the second tests seek to facilitate size compression on encoded parameter updates in order to reach an equivalent communication cost as unencrypted updates. This would help conserve bandwidth for the users. We utilise LZ-String for the compression technique in this phase. The assessment will prioritise data size efficiency, followed by an examination of memory use and time consumption.

V. Result

A. Anonymous Federated Learning and Encryption

TABLE I: Comparison between the original and the proxied federated learning on PC browsers

PC	Trn.	Val.	Test	Time	Comm.
Original (CPU)	97.9%	99.5%	99.6%	657s	236.2MB
Original (+GPU)	97.7%	99.2%	99.6%	294s	236.1MB
AES (CPU)	98.1%	99.4%	89.9%	676s	472.4MB
AES (+GPU)	97.6%	99.3%	99.6%	330s	472.4MB
Key Pair (CPU)	98.0%	99.2%	99.6%	683s	557.8MB
Key Pair (+GPU)	97.5%	99.1%	99.6%	456s	557.8MB

Based on the data shown in Tables I and II, The accuracies are mainly comparable, with a standard deviation of 3.62%,

[2]https://gist.github.com/senoyodha/9e964ec18155ebecef14ea21a539430f

TABLE II: Comparison between the original and the proxied federated learning on mobile browsers

Mobile	Trn.	Val.	Test	Time	Comm.
Original	94.7%	99.2%	95.2%	1,375s	75.0MB
AES	96.5%	99.3%	98.6%	1,441s	149.9MB
Key Pair	96.8%	99.3%	98.9%	1,431s	177.0MB

since the proxy and encryption did not modify the data or introduce any noise. The amount of time required for Original, AES, and Key Pair operations is generally similar, except on GPU tests. On PC, the order of speed from fastest to slowest is Original, AES, and then Key Pair. However, on mobile devices, the order is Original, Key Pair, and then AES.

Regarding PC performance, AES and Key Pair operations exhibit a 2% and 6% decrease in speed compared to Original, respectively. On the other hand, in the mobile scenario, AES and Key Pair operations are 5% and 4% slower than Original, respectively. In terms of communication cost in PC tests, AES and Key Pair have a cost that is 1.5 times and two times higher than Original, respectively. Conversely, the expenses associated with mobile tests for AES and Key Pair are twice and 2.4 times higher than Original, respectively.

In summary, disregarding the cost of communication, utilising an asymmetric Key Pair would be the optimal choice for a PC due to its superior security compared to symmetric AES. Nevertheless, in the case of mobile cases with limited bandwidth, it may be more suitable to employ AES or a less complex encoding method, as long as the user's security and privacy are upheld. Alternatively, employing more robust encryption methods, such as asymmetric key pairs, along with compression, can provide enhanced security and efficiency.

B. Measurement of Memory Usage

TABLE III: Comparison of accuracy and communication cost under several federated learning setups

Mode	Trn.	Val.	Test	Comm.
Original	97.7%	98.9%	99.4%	236.2MB
Symm. AES	97.8%	99.3%	99.4%	472.4MB
Asym. PK	98.2%	99.4%	99.5%	557.7MB
Symm. AES+LZ	98.2%	99.3%	99.6%	264.0MB
Asym. PK+LZ	98.8%	99.6%	99.5%	231.7MB

As depicted in Table III, all accuracy results exhibit similarity and consistency across modes, having less than 0.7% standard deviation. As for the communication cost, both encryptions bloated the data size to more than twice the original. Nevertheless, the LZ-string reduced half of the communication cost back to a comparable size to the original one, even smaller in size in asymmetric PK.

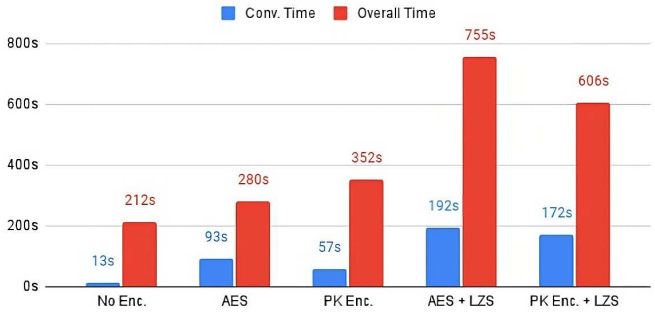

Fig. 4: Comparison of conversion time

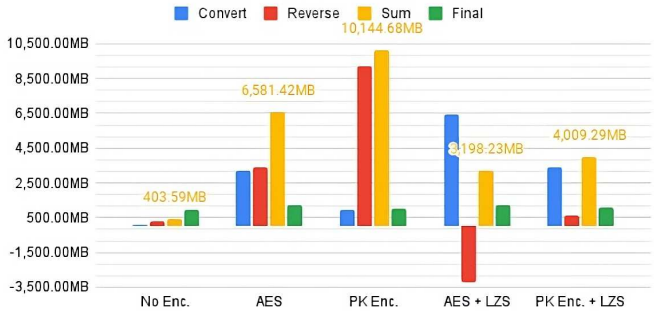

Fig. 5: Comparison of memory heap size

Although applying compression can save half of the bandwidth, it prolongs the overall training time by three times as shown in Fig. 4. It is mainly due to compression procedure that takes longer time than just an encryption function. The whole experiments were done in under one local network. Thus, the transmitted data size, which is shown in Table I, will not affect significantly and is not directly proportional to overall time in Fig 4. Please note that additional data transmutations from JSON object to a data stream of bytes, and vice versa, happened on AES with LZ-string during the whole training process, resulting in being the slowest among the five setups.

Fig. 5 displays measurements of the memory utilised during the conversion phase (data transmutation, encryption, and compression), reversion phase (decryption and decompression), the sum of both phases and upon completion of the entire round. Overall, symmetric AES consumes less memory than asymmetric PK, while compression utilises more memory in the conversion phases than in the reversion phase. It is expected that applying encryption will increase memory usage compared to the original one. Nevertheless, applying compression can help to cut the memory heap usage by more than half.

Remarkably, in the compression scenario, the AES encryption process resulted in a reduction in memory usage during the reverse operation, indicating that the combined encryption and compression tasks first consumed significant memory but then freed up some of it upon reversal. Combining conversion

and reversing heap sizes, applying encryption and compression used memory that was relatively similar, differing by 25.3%. Ultimately, the memory utilisation at the conclusion of the training for all five setups is comparable, having 10% as the standard deviation.

The whole test report can be accessed through: https://s.id/icstexp.

VI. Conclusion

From our experiments and analysis, we can make inferences. Firstly, privacy-preserving techniques that maintain accuracy are an essential objective, particularly on personal devices with limited capabilities, followed by efficiency in computation and communication. Our proposed methodology has demonstrated effectiveness in meeting the specified criteria, although further enhancements are required to optimise computational efficiency.

Secondly, our tests demonstrate that it is feasible to achieve equivalent communication costs between our suggested encrypted anonymous privacy enhancement and the original FL through the utilisation of compression techniques on the model updates. Clients can enjoy enhanced privacy and data protection with anonymity and encryption without compromising their bandwidth. Additionally, using compression when applying encryption can reduce memory usage by up to 60% compared to setups without compression. However, the drawback is that the overall training duration increases when compression is implemented, causing a slowdown of up to 1.7 times compared to setups without compression.

Thirdly, when implementing compression procedure, we suggest utilising Public-key asymmetric encryption instead of AES symmetric encryption for various advantages: 1) Reduce communication expenses and data volume; 2) Accelerate training iterations; and 3) Enhance security and safety. Compression is advised for personal devices to conserve bandwidth and memory when quick inference is not the main need.

Finally, the suggested methodology provides a scalable and cost-effective privacy-preserving solution that maintains accuracy. We strongly believe that this research deserves further exploration and study due to its introduction of a novel approach for conducting anonymous FL via proxy on personal devices.

References

[1] European Union, "What is GDPR, the EU"s new data protection law? - GDPR.eu", May-2018. [Online]. Available: https://gdpr.eu/what-is-gdpr/.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data", in *Artificial intelligence and statistics*, 2017, pp. 1273–1282.

[3] P. Kairouz *et al.*, "Advances and open problems in federated learning", *Foundations and Trends® in Machine Learning*, vol. 14, no. 1--2, pp. 1–210, 2021.

[4] C. Dwork, A. Roth, and Others, "The algorithmic foundations of differential privacy", *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3–4, pp. 211–407, 2014.

[5] K. Bonawitz *et al.*, "Practical secure aggregation for privacy-preserving machine learning", in *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[6] C. Gentry, "Fully homomorphic encryption using ideal lattices", in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 2009, pp. 169–178.

[7] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective", *arXiv preprint arXiv:1712. 07557*, 2017.

[8] S. Kadhe, N. Rajaraman, O. O. Koyluoglu, and K. Ramchandran, "Fastsecagg: Scalable secure aggregation for privacy-preserving federated learning", *arXiv preprint arXiv:2009. 11248*, 2020.

[9] C. Dilmegan, "What is Homomorphic Encryption? Benefits & Challenges [2022]", Apr. 2022. [Online]. Available: https://research.aimultiple.com/homomorphic-encryption/.

[10] W. Wang, Y. Hu, L. Chen, X. Huang, and B. Sunar, "Exploring the feasibility of fully homomorphic encryption", *IEEE Transactions on Computers*, vol. 64, no. 3, pp. 698–706, 2013.

[11] J. Konečn\'y, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency", *arXiv preprint arXiv:1610. 05492*, 2016.

[12] J. Konečn\'y, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence", *arXiv preprint arXiv:1610. 02527*, 2016.

[13] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning", *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 13, no. 3, pp. 1–207, 2019.

[14] E. Kinnear, P. McManus, T. Pauly, T. Verma, and C. A. Wood, "RFC 9230 - Oblivious DNS over HTTPS", *IETF*, Apr. 2022. [Online]. Available: https://datatracker.ietf.org/doc/rfc9230/.

[15] G. Rosinosky, S. Da Silva, S. Ben Mokhtar, D. Négru, L. Réveillère, and E. Rivière, "PProx: efficient privacy for recommendation-as-a-service", in *Proceedings of the 22nd International Middleware Conference*, 2021, pp. 14–26.

[16] S. Kalra, J. Wen, J. C. Cresswell, M. Volkovs, and H. R. Tizhoosh, "ProxyFL: Decentralized Federated Learning through Proxy Model Sharing", *arXiv preprint arXiv:2111. 11343*, 2021.

[17] H. Shi, V. Radu, and P. Yang, "Distributed Training for Speech Recognition using Local Knowledge Aggregation and Knowledge Distillation in Heterogeneous Systems", in *Proceedings of the 3rd Workshop on Machine Learning and Systems*, 2023, pp. 64–70.

[18] M. Brennaf, P. Yang, and V. Lanfranchi, "A Comparative Analysis of Federated Learning Techniques on On-Demand Platforms in Supporting Modern Web Browser Applications", in *2023 IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2023, pp. 2601–2606.

[19] Epfl, "epfml/disco: Decentralized & federated privacy-preserving ML training, using p2p networking, in JS", Aug-2022. [Online]. Available: https://github.com/epfml/disco.

[20] C. Deotte, "25 Million Images! [0.99757] MNIST Kaggle", Aug-2021. [Online]. Available: https://www.kaggle.com/code/cdeotte/25-million-images-0-99757-mnist.