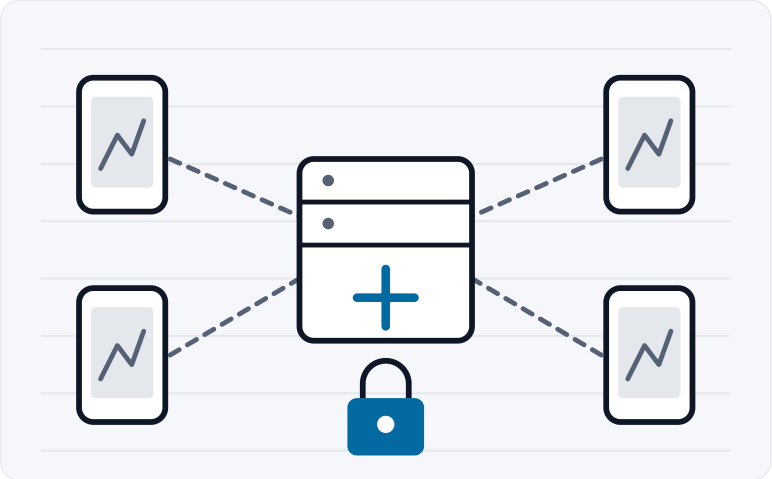
26 September 2024 Overview · Privacy & Crypto · E-Government · 23 min read

MPC-based Secure Aggregation in Federated Learning: Overview, Protocols, & Google's Gboard

Karl W. Koch



Authors Karl W. Koch
Version 2024-07-12
Source https://technology2.a-sit.at/en/publications/mpc-based-secure-aggregation-and-gboard/
Note Machine translation.

Key findings

- Even the ML parameters in federated learning allow conclusions about input data; GradInversion uses them to reconstruct training images.
- With secure aggregation, the server receives only the sum of all users' updated ML parameters.
- MPC-based SecAgg is more flexible than HE, which has a much higher computational cost, but requires trust in the server or participants.
- SAFELearn and SCOTCH deliver the global model only to the participants, but need at least two trusted aggregation servers.
- Gboard combines MPC-based SecAgg in training with differential privacy; most variants use only DP.

As digitalization progresses, it is becoming increasingly attractive to learn from as much data as possible in order to, e.g., continuously improve applications. Such as predicting the next word, possible travel routes, providing UI elements in services such as "Digitales Amt", or improving people care in hospitals.

In order to train a global ML model based on data from many end-user devices and also to preserve user privacy, Google introduced Federated Learning (FL) in 2016/17. With FL, each user trains the corresponding ML model locally and "only" sends the updated ML parameters to a server. However, it has been shown that the ML parameters themselves can also be used to reconstruct the respective input data. For this reason, Secure Aggregation (SecAgg) has been integrated in FL. With SecAgg, the server only receives the sum of the updated ML parameters

from all users. For the concrete instantiation of SecAgg, the cryptographic building block of secure multi-party computation (MPC) has proven to be practicable. In order to bring MPC-based SecAgg in FL further into practice – and thus enable further privacy-preserving ML applications – general approaches and dedicated protocols need to be analyzed, compared and, if necessary, improved.

Therefore, this report first shows general methods for privacy-preserving Federated Learning, and then MPC-based Secure-Aggregation protocols for Federated Learning, and compares them based on their computational and communication complexity, and their security guarantees. Furthermore, a practical example is shown: Google's Gboard, which also integrates an MPC-based protocol.

FL with appropriate extensions, makes it possible to preserve the privacy of participants' training data. For the training phase, for example, the cryptographic building blocks of Homomorphic Encryption (HE) and MPC have proven to be practically relevant. Although MPC-based SecAgg requires trust in the server or a subset of other participants, this method essentially offers more flexibility in practical implementations (especially in scenarios with many participants, who take part in the computation with a rather low-performance device). And even in MPC-based SecAgg, there are numerous protocols that offer different trade-offs. For example, whether (only) the participants receive the resulting global ML model (e.g. SAFELearn, SCOTCH), or primarily only the aggregation server (e.g. SecAgg, SecAgg+, FastSecAgg, LightSecAgg). For the inference phase – in which the ML model is evaluated using "new input" – the method of Differential Privacy (DP) has proven to be practically relevant. However, as with almost all methods, each method provides different trade-offs. Furthermore, (privacy-preserving) FL also has the advantage that it can develop special ML models; e.g., locally-adapted ML models, as shown by Google's Gboard.

The different trade-offs for the respective application scenarios need to be examined in more detail for the further use of (privacy-preserving) FL. For example, how high the degree of privacy is, depending on the number of participants. In addition, it is also necessary to solve the different challenges of FL, such as data heterogeneity on end-user devices.

1. Introduction

Due to advancing digitalization, it is becoming increasingly attractive to learn from as much data as possible in order to, e.g., continuously improve applications. Examples are the prediction of the next word (see 3.2 - Example: Google's Gboard) or improving the care of people in hospitals[1]. To achieve this "improvement" or learning, machine learning (ML) algorithms are used more and more often. For "improvement", a new input - e.g., text - is fed into an ML model and the output is a (new) "state"; e.g., what the next word of a sentence might be. There are different approaches to creating an ML model. In general, a distinction is made between 3 ML learning approaches[2]: (1) supervised[3], (2) unsupervised[4] and (3) reinforcement[5] learning. In supervised learning, there is first a training phase, in which the ML model is created using labeled examples of training data, and then an inference phase, in which the ML model is evaluated with "new" queries (such as the next-word prediction described above). In unsupervised learning, an ML model is created without labeled examples. In reinforcement learning, the ML model interacts dynamically with its environment and aims to maximize the rewards received (in the long term); thus, there are no labeled examples in reinforcement learning either.

Most ML models require a training phase with (ideally) meaningful training data. And collecting meaningful training data is one of the major challenges in ML. Until recently, centralized ML was the most common approach to collecting and storing data for training ML models. In centralized ML, the data is stored on a central server and the ML model is trained there. However, with centralized ML, on the one hand the amount of data to be stored may become large, and on the other hand, much data may from the outset not be wanted or allowed on the central server for reasons of privacy or data protection. The aspect of data protection can then lead to many potentially meaningful data not being used for training ML models; especially when the data concerns individuals/end users.

Why "Federated Learning" (FL)? In order to train a global ML model based on data from many end-user devices and also to preserve user privacy, Google introduced Federated Learning (FL) in 2016/17 (McMahan, Moore, Ramage, Hampson, & Aguera y Arcas, 2017). With FL, each user trains the corresponding ML model locally and "only" sends the updated ML model parameters - often called weights - back to the server. This process is called an FL epoch. To train such an ML model optimally, several such FL epochs may be required. Figure 1 shows the graphical ML model flow of an FL epoch. It should be noted that although FL distributes the ML model and has no access to the actual training data, it is a modified variant of "classical" distributed ML[6].

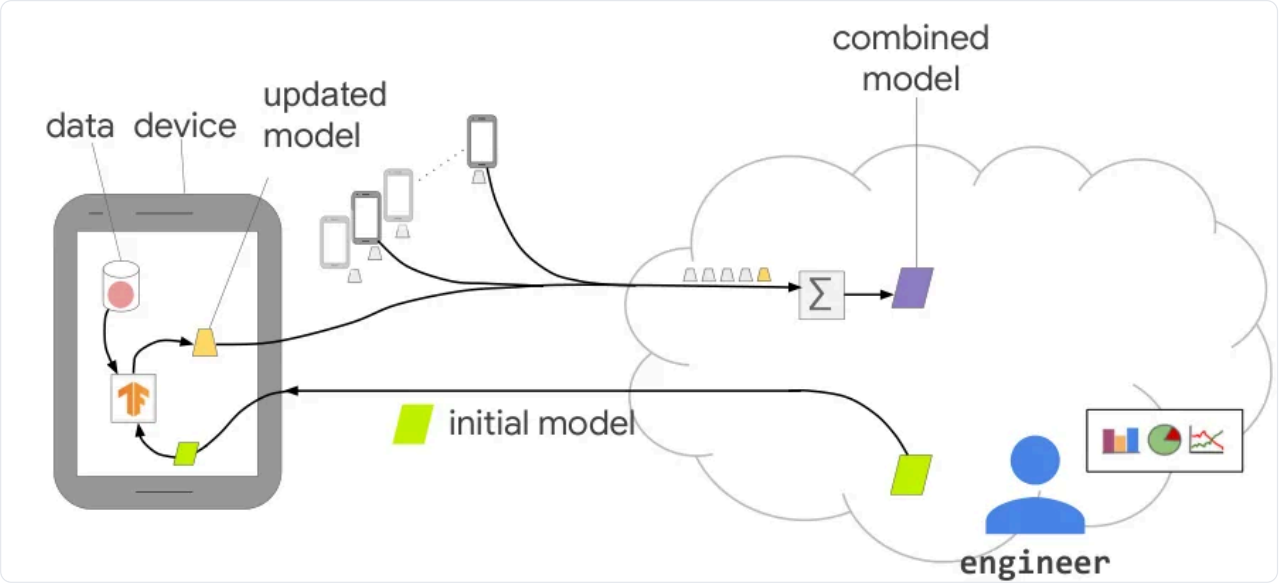


Figure 1: – The ML model flow of an FL epoch. First, the initial ML model is sent to the respective participants, who then train the ML model locally. The resulting ML model is then returned to the server, which aggregates the received updated ML models into a new global ML model.

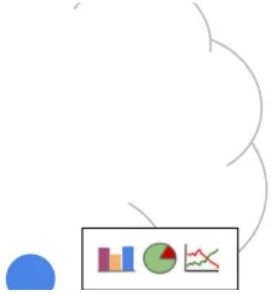


Image source: Federation & Privacy Lectures – PPML @ ITU Copenhagen 2022 by Peter Kairouz (Google) https://cs.au.dk/news-events/events/show-event/artikel/default-d8c512df12

2 types of participants in FL. With regard to the participating users, FL can generally be divided into 2 types: X-device ("cross-device") and X-silo ("cross-silo"). X-device refers to a large number of rather low-performance devices that may also drop out during an FL epoch (e.g., empty battery or no network reception); these are mainly smartphones/tablets/etc. of end users. X-silo refers to a tendency toward few, rather high-performance devices that normally take part in an entire FL epoch; these can be, e.g., "a handful" (~10) of hospitals that want to optimize an ML model together. Figure 2 gives a rough overview of the 2 "FL participant types" X-device and X-silo.

X-device X-silo

vs.

Image source: Federation & Privacy Lectures – PPML @ ITU Copenhagen 2022 by Peter Kairouz (Google) https://cs.au.dk/news-events/events/show-event/artikel/default-d8c512df12

Attacks on standard FL. Although in "standard FL" the respective training data remain on the devices of the participants, it has been found that the ML parameters themselves can also allow conclusions to be drawn about the respective input data. This type of attack on the ML model and parameters is usually called an ML model inversion attack[7]. (Yin, et al., 2021) showed, e.g., an algorithm (GradInversion) to reconstruct training images of an FL-trained ML model using the updated ML parameters. Figure 3 shows a rough schematic representation of GradInversion, including 3 example images. Furthermore, (Phong, Aono, Hayashi, Wang, & Moriai, 2017) and (Geiping, Bauermeister, Droge, & Moeller, 2020), e.g., also show practical attacks on standard FL.



Figure 3: - Rough schematic representation of the GradInversion algorithm for reconstructing training images using the ML parameters returned by the participants. The reconstructed images show the original images more or less clearly - but recognizably (from left to right: sandwich / grasshopper / bird).

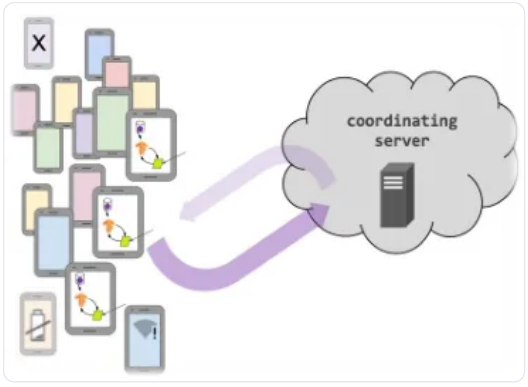


Figure 2: - Overview of the 2 types of FL with regard to the participating users. X-device refers to many rather low-performance devices or participants that may also drop out. X-silo refers to a tendency toward few, rather high-performance devices or participants that are normally involved in an entire round/epoch.

Image source: (Yin, et al., 2021).

In addition to attacks on ML parameters, the resulting (aggregated) ML model itself can also allow conclusions to be drawn about the input data. Especially when very specific training data are (also) learned. E.g., an ML model for next-word prediction can potentially reveal the credit card number, or the place and time of a private meeting. Figure 4 shows this example using a virtual keyboard on a smartphone and text in general.

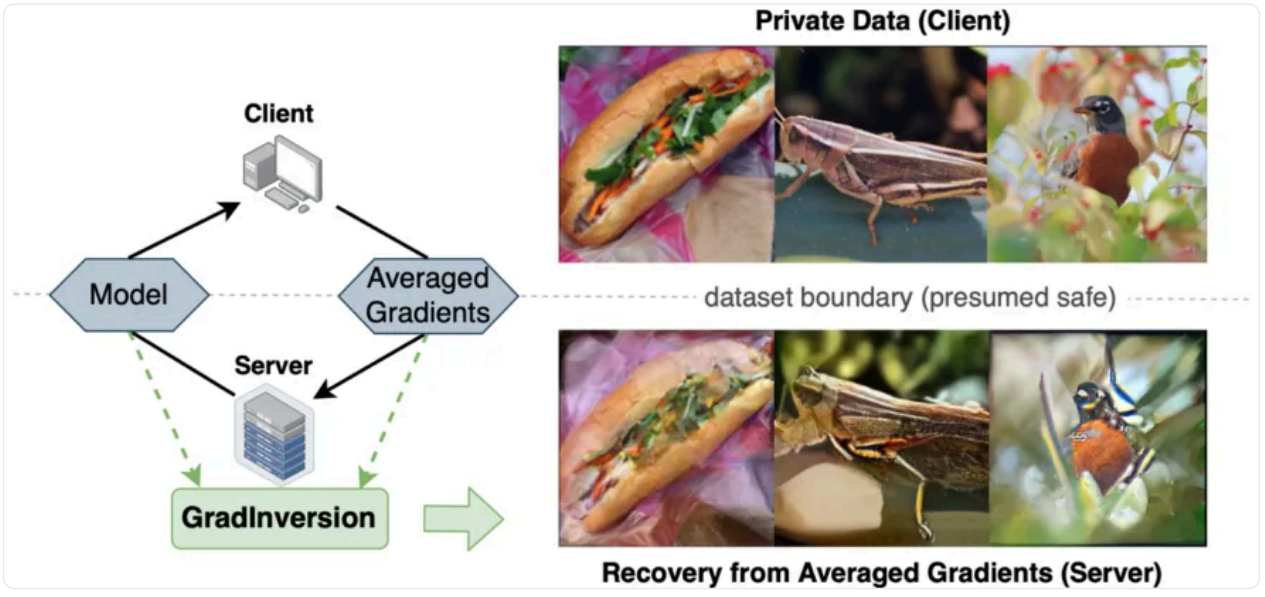
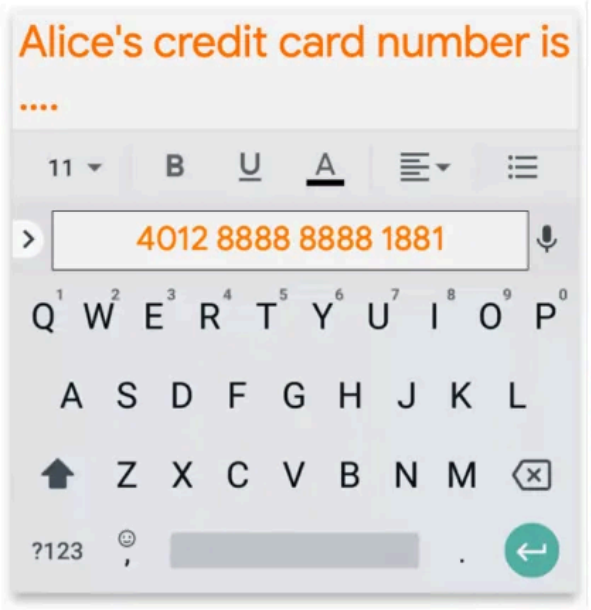


Figure 4: – "Attack"/data leak in the inference phase of an (aggregated) ML model for next-word prediction.





Privacy-preserving FL. To prevent the mentioned and shown attacks on "standard FL" and thus preserve the privacy of the individual participants, corresponding methods have been developed/adapted/used. For protecting the ML parameters in the training phase, several methods have been investigated; the methods with cryptographic building blocks tend to be the most future-oriented because they do not require special hardware. The cryptographic building blocks of Homomorphic Encryption (HE) and Secure Multi-Party Computation (MPC) have proven themselves for FL in recent years. For protecting sensitive data in the inference phase, the method of Differential Privacy has mainly proven itself. However, the field of privacy-preserving FL is currently still in constant development, and it is not always immediately apparent which method or corresponding protocol is best suited to the respective use case; especially in the training phase there are several trade-offs between HE and MPC, and above all between the numerous MPC-based protocols.

Goals & outlook. Therefore, this report first shows (2) general methods for privacy-preserving Federated Learning, and then shows (3) MPC-based Secure Aggregation protocols in Federated Learning and compares them based on their computational and communication complexity and their security guarantees. Furthermore, a practical (3.2) example is shown: Google's Gboard, which, among other things, also integrates an MPC-based protocol. Finally, in (4) Conclusion & Further Work, the findings gained are briefly summarized and potentially interesting further directions are pointed out.

2. Methods for Privacy-Preserving Federated Learning

This section shows methods to transform "standard FL" into "privacy-preserving FL". In an FL framework used in practice, attention must be paid to the security & privacy of the whole cycle. E.g., also whether the current versions of third-party software libraries are used in an Android/iOS app. Since this goes beyond the scope of this report, the focus is placed on ML- and FL-specific aspects. As described in the introduction, the training and inference phases in particular require an extension in order to preserve the privacy of participants' training data.

2.1. Training phase: Secure aggregation of the ML parameters

As shown by numerous ML inversion attacks, the collection of the updated ML parameters in particular requires an extension. The method of Secure Aggregation (SecAgg) has proven itself in this regard in recent years. With SecAgg, the server only receives the sum of the updated ML parameters from all users . The primary goals of SecAgg are (I) correct decoding & summing of the aggregated ML parameters (i.e., the same result as FL without SecAgg) with the guarantee that a certain percentage of participants may drop out during the FL epoch (empty battery, no network connection, etc.) and (II) preserving the privacy of the training data of the individual participants. For the concrete instantiation of SecAgg, there are several variants. In a talk at the "Privacy-Preserving Machine Learning Summer School" 2022, Peter Kairouz (Google) showed that three basic methods of SecAgg can be distinguished: (1) via a trusted third party, (2) via hardware, (3) via cryptographic methods in software. Figure 5 shows the three basic types of SecAgg.

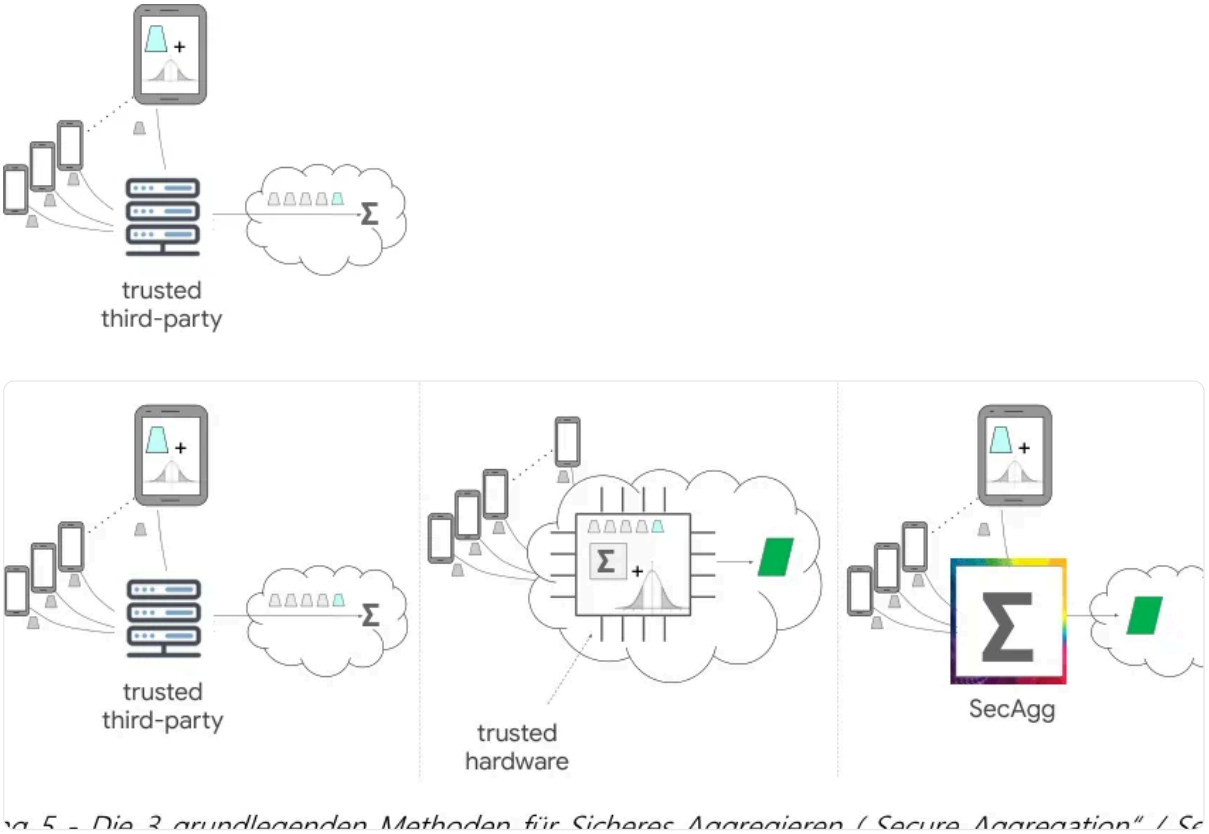


Figure 5 – The 3 basic methods for secure aggregation ("Secure Aggregation" / SecAgg) of ML parameters of the participants of an FL epoch. Via (1) a trusted third party, (2) trusted hardware, or (3) trusted software (= proven cryptographic algorithms (e.g., via MPC (as shown here) or also via homomorphic encryption).





Image source: Federation & Privacy Lectures – PPML @ ITU Copenhagen 2022 by Peter Kairouz (Google) https://cs.au.dk/news-events/events/show-event/artikel/default-d8c512df12

With (1) - the third party - a further entity is added to the FL framework, which must be trusted. With (2) - hardware - dedicated hardware components are needed to enable SecAgg. Here, the server must be trusted to implement this correctly. With (3) - software via cryptography - the ML parameters are already "encrypted" locally on the respective end-user device in such a way that the server (practically speaking) only receives the sum of the ML parameters of all participants. Since the cryptographic building blocks require no further entity or special hardware, the remainder focuses on this method as an extension of the FL training phase.

For the concrete instantiation of cryptography-based SecAgg, the cryptographic building blocks of Secure Multi-Party Computation (MPC) and Homomorphic Encryption (HE) have proven to be practicable. MPC is by now a practicable privacy-preserving cryptographic building block. With the help of MPC, numerous data analyses or computations in general can be carried out in such a way that the input data of the participants remain secret and only the final result becomes known either to a dedicated party or to all participants. In MPC-based SecAgg, the aspect of Secret Sharing (SeSh) comes into play. HE is likewise a privacy-preserving cryptographic building block that is becoming steadily more practicable. The basic idea of HE is that operations are carried out on, e.g., 2 ciphertexts in such a way that the decrypted product contains the result as if the operation had been carried out on the 2 plaintexts. E.g., C1(2) * C2(12) = C(24) (the encrypted number 2 multiplied by the encrypted number 12 yields the encryption of the number 24).

The basic advantage of HE is that one "only" has to trust the encryption or the cryptographic algorithm - and its implementation. With MPC - which is based on SeSh - one must additionally trust that the secret is correctly split among the respective recipients and that the recipients do not collaborate to reconstruct the secret. It should be noted here that with certain MPC or SeSh protocols - "malicious security" - all parts of the secret are needed for the reconstruction (see also the A-SIT report on MPC in general[8]). The basic disadvantage of HE is the considerably higher computational effort; therefore, for dedicated use cases a special form of HE that is less computationally intensive is often used. Here, MPC is fundamentally more flexible and, for the possibility of general computations, tends to be less computationally intensive. Due to the fundamentally higher flexibility of MPC, the remainder of the report focuses on MPC-based Secure Aggregation protocols in Federated Learning.

2.2. Inference phase: Targeted noising of the data/ML parameters

In order to protect potentially sensitive training data during the ML inference phase, the method of "noising" has proven itself in recent years. With "noising", e.g., either the training data directly or the updated/new ML parameters are "distorted" or changed in such a way that the resulting ML model is still meaningful but does not learn "too specific" inputs (like the credit card number example in the introduction). For the concrete instantiation of "noising", the method of Differential Privacy[9] (DP) has proven itself. With DP, a fundamental distinction is made between the approach in which the aggregating server adds the noise (= Central DP) and the approach in which the respective participants add the noise locally (= Local DP).

Since with Central DP the server must be (fully) trusted to add noise, the Local DP approach is advantageous with regard to privacy. However, Local DP has the disadvantage that the accuracy of the aggregated model tends to be lower, and thus possibly more examples or training epochs are needed. Therefore, a model between Central and Local DP has also become established: Shuffle DP[10] [11] [12]. In the Shuffle DP approach, the data are generally not given as much local noise because they are then sent to a trusted "curator". The "curator" then shuffles the received data in such a way that the (aggregating) server cannot draw conclusions about which participant sent which data. This makes it harder for a potential attacker to draw conclusions about the original training data. However, with the Shuffle DP model there must be a further trusted entity. The advantage of, e.g., MPC-based SecAgg is that the aggregating server only receives the sum from all participants and in practice cannot trace which participant sent which ML parameters anyway. This is why, with MPC-based SecAgg, the additional "curator" is, so to speak, already "built in".

Due to these trade-offs of privacy and accuracy in the DP models, approaches of Local - and also Shuffle - DP in FL have been studied in particular in recent years[13] [14] [15] [16]. Local or Shuffle DP is also of particular interest because it can be combined with, e.g., MPC-based SecAgg. Figure 6 shows the example of Local DP during an FL epoch, applied to the ML parameters.

Image source: Federation & Privacy Lectures – PPML @ ITU Copenhagen 2022 by Peter Kairouz (Google) https://cs.au.dk/news-events/events/show-event/artikel/default-d8c512df12

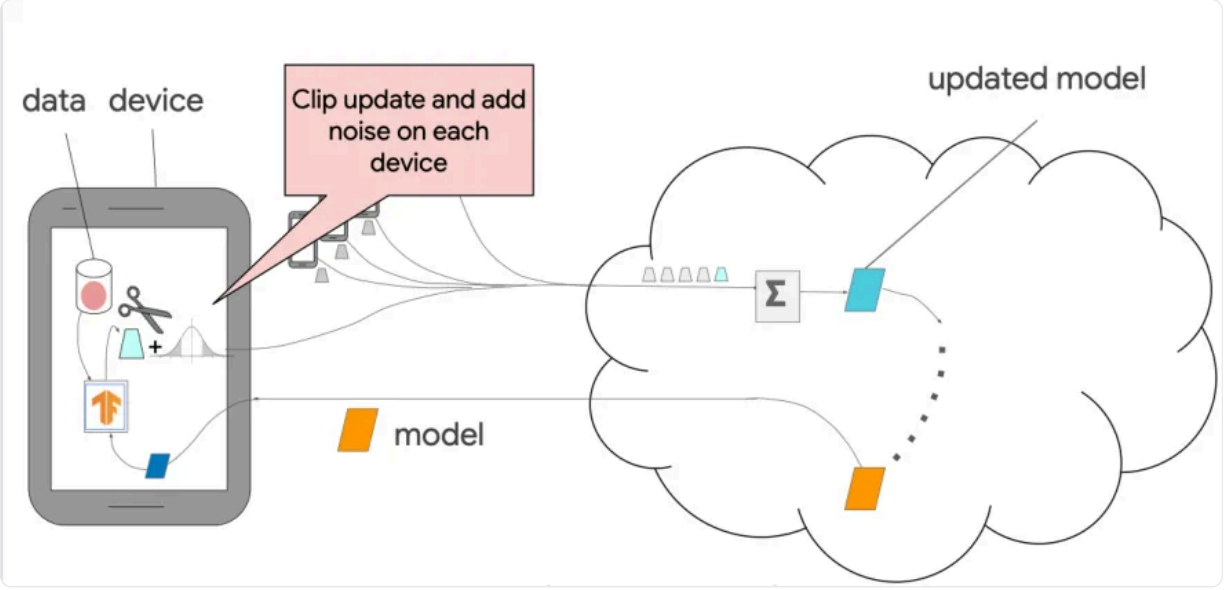


Figure 6: - The method of Local Differential Privacy ("Local DP") during an FL epoch; applied to the ML parameters. Before the updated ML parameters are sent to the server, noise is added.

3. MPC-based Secure Aggregation Protocols in Federated Learning

Since Google's introduction of FL in 2016/17, numerous methods have been developed to make SecAgg in FL ever more practicable. And MPC-based SecAgg is one of the most common practically relevant instantiations of SecAgg. Therefore, this section lists selected MPC-based SecAgg protocols and compares them based on their theoretical computational and communication effort. Furthermore, an example of MPC-based SecAgg used in practice, in combination with DP, is shown: Google's Gboard.

3.1. Overview & comparison

Since the introduction of FL in 2016/17, numerous MPC-based SecAgg protocols have been developed. It is always a trade-off between computational effort, communication effort, and how many entities or aggregators are trusted. Table 1 gives a complexity overview of the following protocols, which were selected from a mix of introductory protocol, successors and decentralization approaches in order to show the fundamental variations of MPC-based SecAgg:

- 2017 SecAgg

  "Practical Secure Aggregation for Privacy-Preserving Machine Learning" (Bonawitz, et al., 2017) CCS ¨ The beginning of practically relevant MPC-based SecAgg. Uses pairwise masks, which cancel out during aggregation/summation.

- 2020 SecAgg+

  "Secure Single-Server Aggregation with (Poly)Logarithmic Overhead" (Bell, Bonawitz, Gascón, Lepoint, & Raykova) CCS ¨ Further development of SecAgg. Instead of splitting the ML parameters for all other participants (=n), the ML parameters are "only" split for log(n) other participants; which means a communication reduction of ~log().

- 2020 FastSecAgg

  "Scalable Secure Aggregation for Privacy-Preserving Federated Learning" (Kadhe, Rajaraman, Koyluoglu, & Ramchandran, 2020) arXiv ¨ Instead of "classical" secret sharing based on Shamir's protocol, the secret shares are generated or reconstructed based on the Fast Fourier Transformation[17] (FFT).

  [17] en.wikipedia.org/wiki/Fast_Fourier_transform

- 2021 SAFELearn

  „Secure Aggregation for private FEderated Learning" (Fereidooni, et al., 2021) S&P Workshops ¨ Generic framework for fundamentally decentralized FL, in which only the participants receive the resulting global ML model. For SecAgg, e.g., the cryptographic building blocks HE and also 2/MPC can be selected.

- 2022 LightSecAgg

  "Lightweight and Versatile Design for Secure Aggregation in Federated Learning" (So, et al., 2020) PMLS ¨ The basic idea is to improve SecAgg or SecAgg+ with a modified masking approach that reduces the number of communication rounds.

- 2022 SCOTCH

  "Efficient Secure Computation Framework for Secure Aggregation" (More, et al.) arXiv ¨ Based on additive secret sharing and uses m trusted aggregation servers, so that fundamentally only the participants receive the resulting global ML model; which corresponds, so to speak, to a decentralized approach for FL. Figure 7 shows the basic components and their communication flow in SCOTCH.

Image source: (More, et al.)http://arxiv.org/abs/2201.07730

Trade-offs. Each of the protocols fundamentally preserves the privacy of the participants' training data. However, each protocol entails different trade-offs. E.g., SAFELearn and SCOTCH focus on the "decentralized FL" approach, in which only the participants receive the resulting global ML model. The aggregation server(s) "only" aggregate the ML parameters of the participants in such a way that only the participants can decrypt or reconstruct the aggregation. However, these 2 protocols require ≥2 trusted aggregation servers. All other protocols focus on the "centralized FL" approach, in which fundamentally only the aggregation server receives the resulting global ML model and then, where applicable, sends it on to the respective participants.

FastSecAgg fundamentally offers a good computational effort and also uses a novel method to split the ML parameters (Fast Fourier Transformation / FFT). LightSecAgg also fundamentally offers a good computational effort by extending the masking approach of SecAgg/SecAgg+. Furthermore, both protocols - FastSecAgg & LightSecAgg - offer a relatively low number of communication rounds; however "only" in the passive attacker model. SecAgg and its successor SecAgg+, by contrast, both also offer a variant for the active attacker model. Furthermore, SecAgg+, through special selection of "masking neighbors" in a graph, additionally offers a fundamentally good/comparable communication effort.

Attacker models. In FL, a fundamental distinction is made between 2 attacker models: Passive or semi- honest ("semi-honest") and semi-active or "semi-malicious" ("semi-malicious"). In the passive model, the server and participants follow the protocol but try to learn as much as possible from the information received. It may also be that the server and a subset of the participants collaborate to derive more information from it. In the semi-active model, the server is trusted during the setup phase to, e.g., send out a "correct" ML model and, where applicable, to forward asymmetric key material correctly. However, during the actual SecAgg protocol the server may not follow the protocol and, e.g., give some participants incorrect information about which participants are still active in the respective FL epoch and which have become inactive ("dropout"). In a purely active model ("fully-malicious"), the server would not necessarily have to be trusted even during the setup phase.

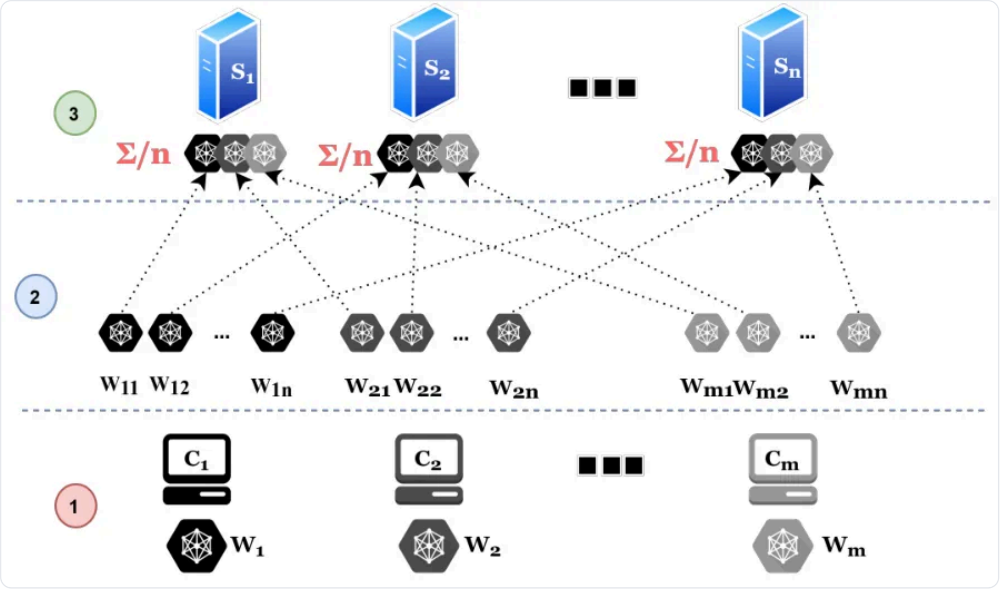


Figure 7 - Overview of the components and their communication flows of the decentralized MPC-based SecAgg protocol SCOTCH.

MPC- based

SecAgg- protocol

variant

or attacker-

model

Computational cost *("Computational Cost")*

Number of communication-

rounds for aggregation (participant←

Communication cost

***("Communication Cost")* Aggregation**

>server)

Server Participant Server Participant

Table 1 - Overview of MPC-based SecAgg protocols for FL, with a focus on aggregation complexity. n...number of participants; L...length or number of ML parameters; for LightSecAgg: U...min. number of participants for reconstruction + T...max. number of corrupt participants; for SCOTCH: m...number of (trusted aggregation servers.

3.2. Example: Google's Gboard

An example of an FL-trained ML model already in practical use is Google's Gboard. Gboard is a virtual keyboard for Android[17] - and iOS[18] devices. Google has also written a scientific paper on the security & privacy, efficiency and accuracy of Gboard (Xu, et al., 2023).

"Standard FL"		$O(Ln)$	$O(L)$	1	$O(Ln)$	$O(L)$	1 Server
SecAgg	Passiv / Semi-Aktiv	$O(Ln^2 + n^2)$	$O(Ln + n^2)$	3 / 4	$O(Ln + n^2)$	$O(L + n)$	1 Server
SecAgg+	Passiv / Semi-Aktiv	$O(Ln\log(n) + n\log^2(n))$ $O(L\log^2(n) + n\log^2(n))$	$O(L\log(n) + \log^2(n))$	3 / 5	$O(Ln + n\log(n))$ $O(Ln + \log^2(n))$	$O(L + log(n))$ $O(L + \log^2(n))$	1 Server
FastSecAgg	Passiv	$O(L\log(n))$	$O(L\log(n))$	2	$O(Ln + n^2)$	$O(L + n)$	1 Server
SAFELearn	Passiv 2/MPC	$O(Ln)$	$O(L)$	1	$O(Ln)$	$O(L)$	≥2 vertrauens-würdige Server
LightSecAgg	Passiv	$O(L^*(U\log(U)/(U-T)))$	$O(L^*(n\log(n)/(U-T)))$	2	$O(Ln + LU/(U-T))$	$O(L + L/(U-T) + Ln/(U-T))$	1 Server
SCOTCH	Passiv	$O(mn)$	$O(2mn)$	1.5	$O(n)$	$O(m)$	m vertrauens-würdige Server

Figure 8: - Logo of Google's Gboard (virtual keyboard app).

Image source: https://en.wikipedia.org/wiki/Gboard

With regard to ML or FL, Gboard integrates ML language models for, e.g., next-word prediction ("Next-Word Prediction") or writing assistance while typing a word ("Smart Compose"). Figure 9 shows an example of each of these 2 ML models for end users. Gboard's ML model is built on a neural network with 670 "hidden neurons"[19]. The ML model for active writing assistance, e.g., requires a vocabulary of approx. 30,000 words, with approx. 6.4 million ML parameters.

Image source: (Xu, et al., 2023)https://aclanthology.org/2023.acl-industry.60

To preserve privacy during the training and inference phases, Gboard combines the methods of MPC-based SecAgg (training phase) and DP (inference phase). Most variants of Gboard use only DP for both phases. Figure 10 shows Gboard's ML model flow of an FL epoch. On the one hand, this preserves the privacy of the participants' training data, and on the other hand, FL makes it possible to address locally specific requirements. E.g., with Gboard's approach it is possible to develop, with participants from Austria/Germany/Switzerland, an ML model specially suited to the German language; or, e.g., a dedicated ML model for participants from Brazil.



Figure 10: - ML model flow of an FL epoch in Google's Gboard. To preserve privacy, DP and MPC-based SecAgg are used.

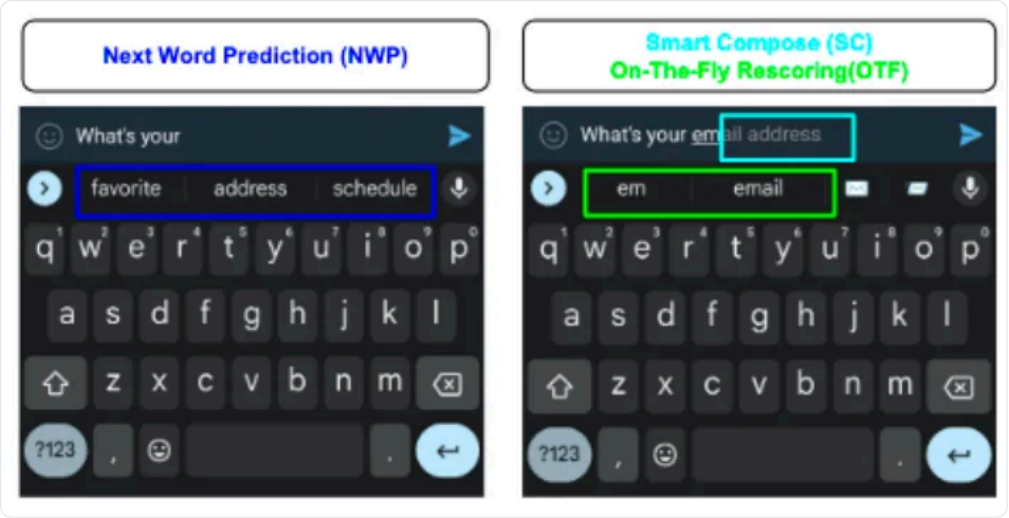


Figure 9 - An example each of next-word prediction (left) and writing assistance while typing a word (right) in the virtual keyboard Gboard (by Google), using FL-trained ML language models.

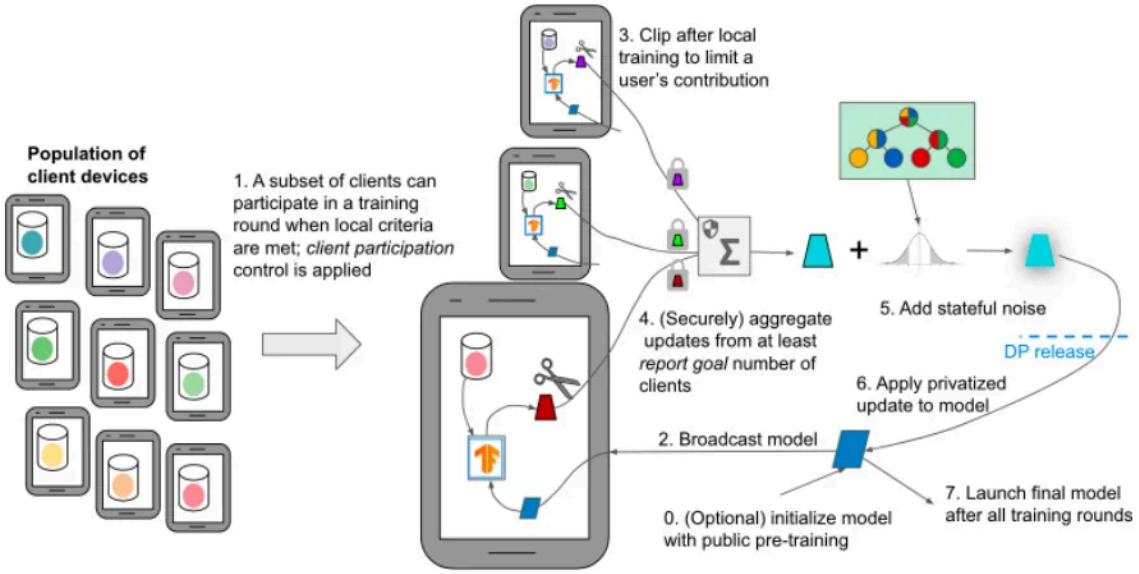


Image source: Federation & Privacy Lectures – PPML @ ITU Copenhagen 2022 by Peter Kairouz (Google) https://cs.au.dk/news-events/events/show-event/artikel/default-d8c512df12

4. Conclusion & Further Work

This section first concludes the report and then points out potential directions for further work.

Federated Learning (FL), with extensions in the training and inference phases, makes it possible to preserve the privacy of participants' training data. For the training phase, the cryptographic building blocks HE and MPC have proven themselves. Although MPC-based SecAgg requires trust in the server or a subset of other participants, this method fundamentally offers more flexibility in practical implementation (especially for X-device FL). And even in MPC-based SecAgg, there are numerous protocols that offer different trade-offs. E.g., whether (only) the participants receive the resulting global ML model (e.g., SAFELearn, SCOTCH), or primarily only the aggregation server (e.g., SecAgg, SecAgg+, FastSecAgg, LightSecAgg). For the inference phase, the method of Differential Privacy (DP) has proven itself. However, as with nearly all methods, each method entails different trade-offs. Furthermore, (privacy-preserving) FL also has the advantage that it can develop special, e.g., locally adapted ML models; as shown by Google's Gboard.

For the further use and dissemination of privacy-preserving FL, the different trade-offs for the respective application scenarios must be examined more closely. E.g., how high the degree of privacy is, depending on the number of participants[20]. In addition, it is also necessary to solve the different challenges of FL; such as, e.g., that of data heterogeneity on end-user devices[21].

References

McMahan, B., Moore, E., Ramage, D., Hampson, S., & Aguera y Arcas, B. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. International Conference on Artifical Intelligence and Statistics.

Geiping, J., Bauermeister, H., Droge, H., & Moeller, M. (2020). Inverting Gradients - How easy is it to break privacy in federated learning? Neural Information Processing Systems (NeurIPS). https://proceedings.neurips.cc/paper/2020/hash/c4ede56bbd98819ae6112b20ac6bf145-Abstract.html.

Yin, H., Mallya, A., Vahdat, A., Alvarez, J. M., Kautz, J., & Molchanov, P. (2021). See through Gradients: Image Batch Recovery via GradInversion. IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://openaccess.thecvf.com/content/CVPR2021/html/Yin_See_Through_Gradients_Image_Batch_Recovery_via_GradInversion_CVPR_2021_paper.html.

Phong, L. T., Aono, Y., Hayashi, T., Wang, L., & Moriai, S. (2017). Privacy-preserving deep learning: Revisited and enhanced. Applications and Techniques in Information Security (ATIS). https://link.springer.com/chapter/10.1007/978-981-10-5421-1_9.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., . . . Seth, K. (2017). Practical Secure Aggregation for Privacy-Preserving Machine Learning. CCS. https://dl.acm.org/doi/10.1145/3133956.3133982.

Bell, J. H., Bonawitz, K. A., Gascón, A., Lepoint, T., & Raykova, M. (n.d.). Secure Single-Server Aggregation with (Poly)Logarithmic Overhead. CCS. 2020: https://dl.acm.org/doi/10.1145/3372297.3417885.

Kadhe, S., Rajaraman, N., Koyluoglu, O. O., & Ramchandran, K. (2020). Scalable Secure Aggregation for Privacy Preserving Federated Learning. arXiv. http://arxiv.org/abs/2009.11248.

So, J., He, C., Yang, C.-S., Li, S., Yu, Q., E. Ali, R., . . . Avestimehr, S. (2020). Lightweight and Versatile Design for Secure Aggregation in Federated Learning. PMLS. https://proceedings.mlsys.org/paper_files/paper/2022/hash/6c44dc73014d66ba49b28d483a8f8b0d-Abstract.html.

More, Y., Ramachandran, P., Panda, P., Mondal, A., Virk, H., & Gupta, D. (n.d.). SCOTCH: An Efficient Secure Computation Framework for Secure Aggregation. arXiv. http://arxiv.org/abs/2201.07730 .

Fereidooni, H., Marchal, S., Miettinen, M., Mirhoseini, A., Möllering, H., Nguyen, T. D., . . . Zeitouni, S. (2021). Secure Aggregation for private FEderated Learning. S&P Workshops. https://ieeexplore.ieee.org/abstract/document/9474309.

Xu, Z., Zhang, Y., Andrew, G., Choquette, C., Kairouz, P., Mcmahan, B., . . . Zhang, Y. (2023). Federated Learning of Gboard Language Models with Differential Privacy. ACL. aclanthology.org/2023.acl-industry.60 .

---

1. predicting-health.at ↩
2. en.wikipedia.org/wiki/Machine_learning - Approaches ↩
3. en.wikipedia.org/wiki/Supervised_learning ↩
4. en.wikipedia.org/wiki/Unsupervised_learning ↩
5. en.wikipedia.org/wiki/Reinforcement_learning ↩
6. en.wikipedia.org/wiki/Distributed_artificial_intelligence ↩
7. owasp.org/www-project-machine-learning-security-top-10/docs/ML03_2023-Model_Inversion_Attack ↩
8. technology.a-sit.at/evaluierung-von-mpcs-stand-der-technik ↩
9. en.wikipedia.org/wiki/Differential_privacy ↩
10. Bittau et al.: “Prochlo: Strong Privacy for Analytics in the Crowd” @ SOSP 2017: https://dl.acm.org/doi/10.1145/3132747.3132769 ↩
11. Albert Cheu, Adam D. Smith, Jonathan Ullman, David Zeber, Maxim Zhilyaev: “Distributed differential privacy via mixnets” @ arXiv 2019: http://arxiv.org/abs/1808.01394 ↩
12. Albert Cheu: “Differential Privacy in the Shuffle Model: A Survey of Separations” @ arXiv 2022: ↩
13. Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, Brendan McMahan: “cpSGD: Communication-efficient and differentially private distributed SGD” @ NeurIPS 2018: proceedings.neurips.cc/paper/2018/hash/21ce689121e39821d07d04faab328370-Abstract.html ↩
14. Borja Balle, James Bell, Adrià Gascón, Kobbi Nissim: “The privacy blanket of the shuffle model” @ Crypto 2019: link.springer.com/chapter/10.1007/978-3-030-26951-7_22 ↩
15. Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, Adam Smith: “What can we learn privately?” @ SIAM Journal on Computing 2011: epubs.siam.org/doi/10.1137/090756090 ↩
16. “Secure Single-Server Aggregation with (Poly)Logarithmic Overhead” @ CCS 2020: ↩
17. play.google.com/store/apps/details?id=com.google.android.inputmethod.latin ↩
18. apps.apple.com/de/app/gboard-die-google-tastatur/id1091700242 ↩
19. arxiv.org/abs/1811.03604 ↩

20. Elkordy et al.: *"How Much Privacy Does Federated Learning with Secure Aggregation Guarantee?"* : petsymposium.org/popets/2023/popets-2023-0030.php ↩

21. Kim et al.: *"HeteroSwitch: Characterizing and Taming System-Induced Data Heterogeneity in Federated Learning":* proceedings.mlsys.org/paper_files/paper/2024/hash/0badcb4e95306df76a719409155e46e8- ↩