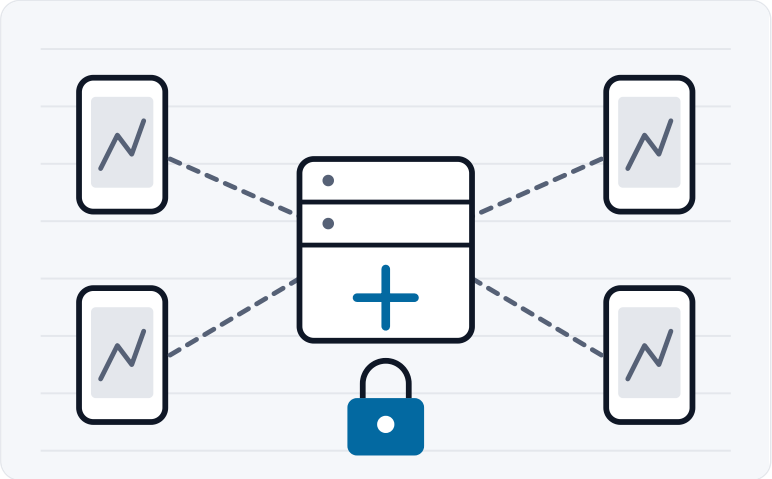
26. September 2024 Überblick · Privatsphäre & Krypto · E-Government · 22 Min. Lesezeit

MPC-basierte Sichere Aggregation in Federated Learning: Überblick, Protokolle, & Google's Gboard

Karl W. Koch



Autorinnen und Autoren	Karl W. Koch
Stand	2024-07-12
Quelle	https://technology2.a-sit.at/veroeffentlichungen/mpc-basierte-sichere-aggregation-und-gboard/

Kernaussagen

- Auch die ML-Parameter in Federated Learning erlauben Rückschlüsse auf Eingabedaten; GradInversion rekonstruiert damit Trainingsbilder.
- Bei Secure Aggregation erhält der Server nur die Summe der aktualisierten ML-Parameter aller Nutzer.
- MPC-basiertes SecAgg ist flexibler als HE, das einen wesentlich höheren Rechenaufwand hat, setzt aber Vertrauen in Server oder Teilnehmer voraus.
- SAFELearn und SCOTCH liefern das globale Modell nur an die Teilnehmer, benötigen dafür aber mindestens zwei vertrauenswürdige Aggregations-Server.
- Gboard kombiniert MPC-basiertes SecAgg im Training mit Differential Privacy; die meisten Varianten nutzen nur DP.

Durch die fortschreitende Digitalisierung wird es immer attraktiver von so vielen Daten wie möglich zu lernen, um, z.B., Anwendungen stetig zu verbessern. Wie die Vorhersage des nächsten Wortes, mögliche Reiserouten, die Bereitstellung von UI-Elementen in Services wie „Digitales Amt", oder der Verbesserung der Personen- Betreuung in Spitälern.

Um ein globales ML-Modell basierend auf Daten von vielen End-Nutzer-Geräten zu trainieren, und auch die Privatsphäre der Nutzer zu bewahren, hat Google 2016/17 föderiertes Lernen („Federated Learning" / FL) ins Leben gerufen. Bei FL trainiert jeder Nutzer lokal das entsprechende ML-Modell, und sendet „lediglich" die aktualisierten ML-Parameter an einen Server. Jedoch wurde festgestellt, dass auch die ML-Parameter an sich Rückschlüsse auf die jeweiligen Eingabe-Daten ziehen lassen können. Deshalb wurde die Sichere Aggregation („Secure Aggregation" / SecAgg) in FL entwickelt. Bei SecAgg erhält der Server nur die Summe der aktualisierten ML-Parameter von allen Nutzern. Für die konkrete Instanziierung von SecAgg, hat sich der kryptografische Baustein der sicheren Mehrparteien-Berechnung („Secure Multi-Party

Computation“ / MPC) als praktikabel erwiesen. Um MPC-basiertes SecAgg in FL weiter in die Praxis zu bringen – und somit weitere privatsphären- schützende ML-Anwendungen zu ermöglichen – sind generelle Ansätze und dedizierte Protokolle zu analysieren, miteinander zu vergleichen, und ggf. zu verbessern.

Deshalb werden in diesem Bericht zuerst generelle (2) Methoden für Privatsphären-bewahrendes Federated Learning gezeigt, und anschließend (3) MPC-basierte Sichere-Aggregations Protokolle in Federated Learning gezeigt, und basierend auf ihrer Berechnungs- und Kommunikations-Komplexität, und deren Sicherheitsgarantien, miteinander verglichen. Weiters wird ein praktisches (3.2) Beispiel: Google’s Gboard gezeigt, welches unter anderem auch ein MPC-basiertes Protokoll integriert.

FL mit entsprechenden Erweiterungen, ermöglicht es die Privatsphäre der Trainings-Daten von Teilnehmern zu bewahren. Für die Trainings-Phase haben sich, z.B., die kryptografischen Bausteine der homomorphen Verschlüsselung („Homomorphic Encryption“ / HE) und MPC bewährt. Wobei MPC-basiertes SecAgg zwar Vertrauen in den Server bzw. einer Teilmenge von anderen Teilnehmern bedingt, bietet diese Methode grundsätzlich mehr Flexibilität in der praktischen Umsetzung (vor allem bei Szenarien mit vielen Teilnehmer:innen, welche mit einem eher leistungsschwachen Gerät an der Berechnung teilnehmen). Und auch in MPC-basiertem SecAgg, gibt es zahlreiche Protokolle, welche unterschiedliche Trade-Offs bieten. Z.B. ob (nur) die Teilnehmer das resultierende globale ML-Modell erhalten (z.B. SAFELearn, SCOTCH), oder primär nur der Aggregations-Server (z.B. SecAgg, SecAgg+, FastSecAgg, LightSecAgg). Für die Inferenz-Phase – in der das ML-Modell mittels „neuem Input“ ausgewertet wird – hat sich die Methode der Differentiellen Privatsphäre („Differential Privacy“ / DP) bewährt. Wobei, wie bei nahezu allen Methoden, jede Methode unterschiedliche Trade-Offs zur Folge hat. Weiters hat (privatsphären-bewahrendes) FL auch den Vorteil, dass es spezielle, z.B., örtlich-angepasste ML-Modelle entwickeln kann; wie von Google’s Gboard gezeigt.

Für den weiteren Einsatz bzw. Verbreitung von privatsphären-bewahrendem FL müssen die unterschiedlichen Trade-Offs für die jeweiligen Anwendungsszenarien näher untersucht werden. Z.B. wie hoch der Grad der Privatsphäre, abhängig von der Anzahl an Teilnehmern, ist. Zudem ist es auch notwendig die unterschiedlichen Herausforderungen von FL zu lösen; wie, z.B., die der Daten-Heterogenität auf End-Nutzer-Geräten.

1. Einleitung

Durch die fortschreitende Digitalisierung wird es immer attraktiver von so vielen Daten wie möglich zu lernen, um, z.B., Anwendungen stetig zu verbessern. Wie die Vorhersage des nächsten Wortes (siehe 3.2 - Beispiel: Google’s Gboard) oder der Verbesserung der Personen-Betreuung in Spitälern[1]. Um dieses „Verbessern“ bzw. Lernen zu erreichen, kommen immer häufiger Algorithmen des maschinellen Lernens („Machine Learning“ / ML) zum Einsatz. Beim „Verbessern“ wird eine neue Eingabe („Input“) - z.B. Text – bei einem ML-Modell eingegeben und als Ausgabe („Output“) erhält man einen (neuen) „Status“; z.B. was das nächste Wort eines Satzes sein könnte. Dabei gibt es verschiedene Ansätze um ein ML-Modell zu kreieren. Grundsätzlich unterscheidet man zwischen 3 ML-Lern-Ansätzen[2]: (1) überwacht[3] („supervised“), (2) unüberwacht[4] („unsupervised“) und (3) bestärkend[5] („reinforced“) Lernen. Beim überwachten Lernen gibt es zuerst eine Trainingsphase, in der das ML-Modell mittels gekennzeichneten („labeled“) Beispielen von Trainingsdaten kreiert wird, und dann eine Inferenzphase, in der das ML-Modell mit „neuen“ Anfragen ausgewertet wird (wie z.B. die zuvor beschriebene Vorhersage des nächsten Wortes). Beim unüberwachten Lernen wird ein ML-Modell ohne gekennzeichnete („unlabeled“) Beispiele kreiert. Beim bestärkenden Lernen interagiert das ML-Modell dynamisch mit der jeweiligen Umwelt, und zielt darauf ab die erhaltenen Belohnungen (langfristig) zu maximieren; somit gibt es auch bei bestärkendem Lernen keine gekennzeichneten Beispiele.

Die meisten ML-Modelle bedingen eine Trainingsphase mit (idealerweise) aussagekräftigen Trainingsdaten. Und das Sammeln von aussagekräftigen Trainingsdaten ist eine der großen Herausforderungen in ML. Bis vor kurzem war das zentralisierte ML („centralized ML“) der gängigste Ansatz um Daten für das Trainieren von ML-Modellen zu sammeln bzw. zu speichern. Beim zentralisierten ML werden die Daten auf einem zentralen Server gespeichert, und dort das ML-Modell trainiert. Jedoch kann es beim zentralisierten ML sein, dass einerseits die zu speichernde Datenmenge groß wird, und andererseits, dass viele Daten aus Gründen der Privatsphäre bzw. des Datenschutzes schon von vornherein nicht auf dem zentralen Server gespeichert werden wollen bzw. dürfen. Der Aspekt des Datenschutzes kann dann dazu führen, dass viele potentiell-aussagekräftige Daten nicht für das Training von ML-Modellen verwendet werden; vor allem wenn es Daten von Individuen/End-Nutzern betrifft.

Warum „Federated Learning“ (FL)? Um aber ein globales ML-Modell basierend auf Daten von vielen End-Nutzer-Geräten zu trainieren, und auch die Privatsphäre der Nutzer zu bewahren, hat Google 2016/17 föderiertes Lernen („Federated Learning“ / FL) ins Leben gerufen (McMahan, Moore, Ramage, Hampson, & Aguera y Arcas, 2017). Bei FL trainiert jeder Nutzer lokal das entsprechende ML-Modell, und sendet „lediglich“ die aktualisierten ML-Modell-Parameter – oft Gewichte genannt – zurück an den Server. Dieser Prozess bezeichnet eine FL-Epoche. Um solch ein ML-Modell optimal zu trainieren, kann es mehrerer solcher FL-Epochen bedingen. Abbildung 1 zeigt den grafischen ML-Modell-Fluss einer FL-Epoche. Dabei ist anzumerken, dass obwohl FL das ML-Modell verteilt und keinen Zugriff auf die eigentlichen Trainingsdaten hat, es eine modifizierte Variante des „klassischen“ verteilten („distributed“) MLs ist[6].

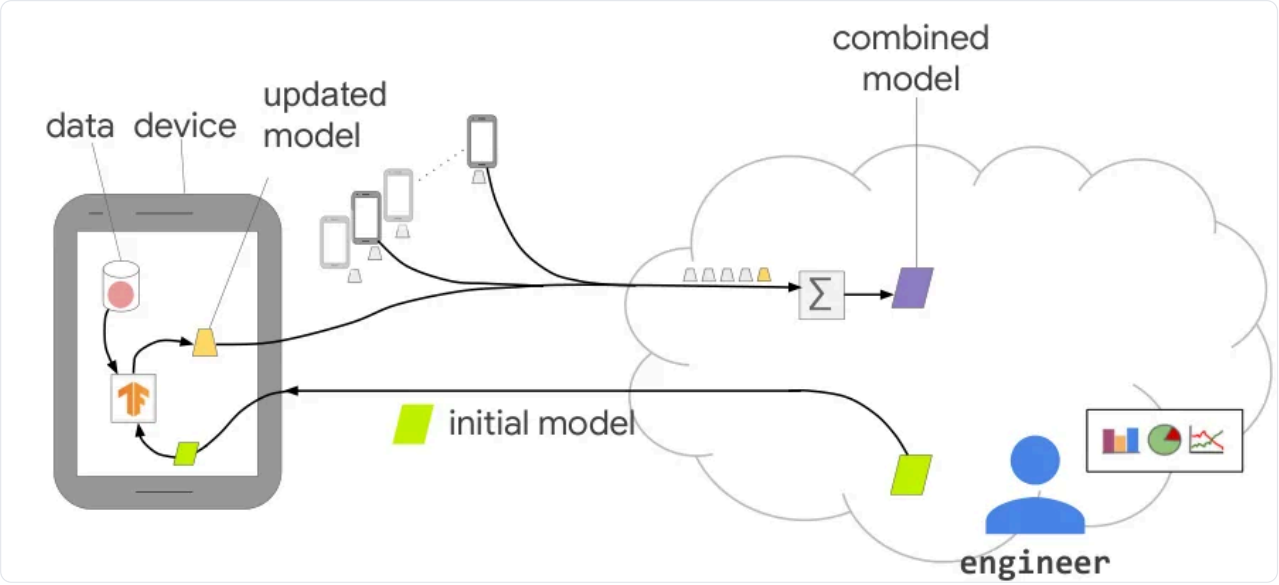
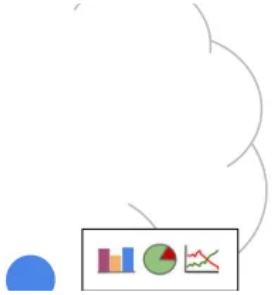


Abbildung 1: – Der ML-Modell-Fluss einer FL-Epoche. Zuerst wird das initiale ML-Modell an die jeweiligen Teilnehmer:innen gesendet, welche dann lokal das ML-Modell trainieren. Das resultierende ML-Modell wird anschließend an den Server retourniert, welcher die erhaltenen aktualisierten ML-Modelle zu einem neuen globalen ML-Modell aggregiert.



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

2 Arten von Teilnehmern bei FL. Im Hinblick auf die teilnehmenden Nutzer:innen, kann man bei FL generell zwischen 2 Arten unterscheiden: X-Gerät („cross-device“) und X-Silo („cross-silo“). X-Gerät bezeichnet eine Vielzahl an eher leistungsschwachen Geräten, welche während einer FL-Epoche auch ausfallen können (z.B. Akku leer oder kein Netzwerk-Empfang); dies sind vor allem Smartphones/Tablets/etc. von End-Nutzer:innen. X-Silo bezeichnet tendenziell wenige, eher leistungsstarke, Geräte, welche normal an einer ganzen FL-Epoche teilnehmen; dies können z.B. „eine Handvoll“ (~10) von Spitälern sein, welche gemeinsam ein ML-Modell optimieren wollen. Abbildung 2 gibt einen groben Überblick von den 2 „FL-Teilnehmer-Arten“ X-Gerät und X-Silo.

X-Gerät X-Silo

vs.

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

Angriffe auf Standard-FL. Zwar bleiben beim „Standard-FL“ die jeweiligen Trainingsdaten auf den Geräten der Teilnehmer:innen, jedoch wurde festgestellt, dass auch die ML-Parameter an sich Rückschlüsse auf die jeweiligen Eingabe-Daten ziehen lassen können. Diese Art von Angriff auf das ML-Modell und -Paramater wird üblicherweise als ML-Modell-Invertierungs-Angriff bezeichnet[7] („ML Model Inversion Attack“). (Yin, et al., 2021) zeigten, z.B., einen Algorithmus (GradInversion) um Trainings-Bilder eines FL-trainierten ML-Modells, mithilfe der aktualisierten ML-Parameter, zu rekonstruieren. Abbildung 3 zeigt eine grob schematische Darstellung von GradInversion, inklusive 3 Beispielbildern. Weiters zeigen, z.B., auch (Phong, Aono, Hayashi, Wang, & Moriai, 2017) und (Geiping, Bauermeister, Droge, & Moeller, 2020) praktische Angriffe auf Standard-FL.



Abbildung 3: - Grobe schematische Darstellung des GradInversion-Algorithmus, um Trainings-Bilder mithilfe der retournierten ML-Parameter der Teilnehmer:innen zu rekonstruieren. Die rekonstruierten Bilder zeigen mehr oder weniger klar - aber doch ersichtlich - die Ursprungsbilder (von links nach rechts: Sandwich / Heuschrecke / Vogel).

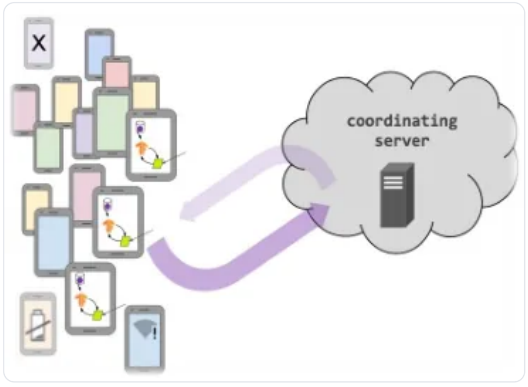


Abbildung 2: - Überblick der 2 Arten von FL in Bezug auf die teilnehmenden Nutzer:innen. X-Gerät bezeichnet viele eher leistungsschwache Geräte bzw. Teilnehmer, welche auch ausfallen können. X-Silo bezeichnet tendenziell wenige, eher leistungsstarke, Geräte bzw. Teilnehmer, welche normal eine ganze Runde/Epoche beteiligt sind.

Bildquelle: (Yin, et al., 2021).

Neben den Angriffen auf ML-Parameter, kann auch das resultierende (aggregierte) ML-Modell an sich Rückschlüsse auf die Eingabedaten ziehen lassen. Vor allem wenn sehr spezifische Trainingsdaten (mit)gelernt werden. Z.B. kann ein ML-Modell für die Vorhersage des nächsten Wortes potentiell die Kreditkarten-Nummer preisgeben, oder den Ort und Zeitpunkt eines privaten Treffens. Abbildung 4 zeigt dieses Beispiel anhand einer virtuellen Tastatur auf einem Smartphone bzw. dem Text generell.

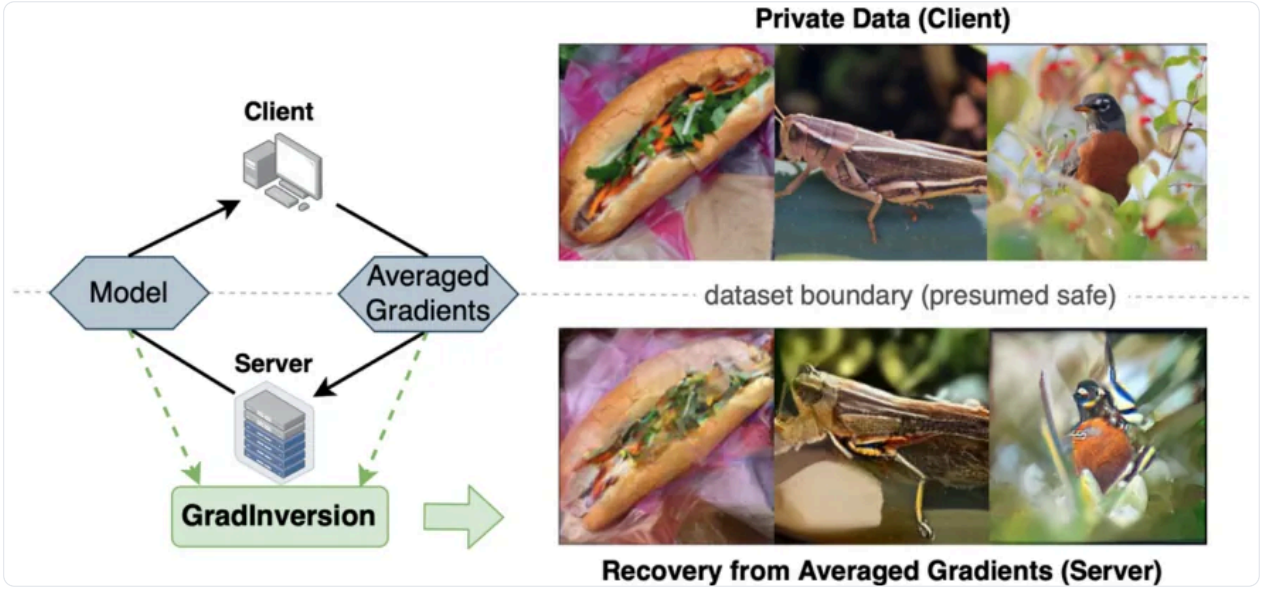
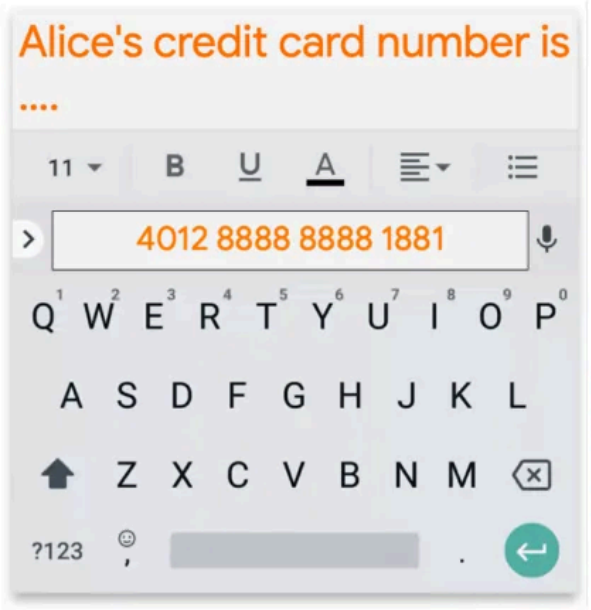


Abbildung 4: – „Angriff"/Datenleck in der Inferenz-Phase bei einem (aggregierten) ML-Modell für die Vorhersage des nächsten Wortes.





Privatsphären-bewahrendes FL. Um die erwähnten bzw. gezeigten Angriffe auf „Standard FL" zu verhindern, und somit die Privatsphäre der einzelnen Teilnehmer:innen zu bewahren, wurden dementsprechende Methoden entwickelt/adaptiert/verwendet. Für das Schützen der ML-Parameter in der Trainings-Phase wurden mehrere Methoden untersucht; wobei die Methoden mit kryptografischen Bausteinen tendenziell am zukunftsorientiertesten sind, weil sie keiner speziellen Hardware bedingen. Die kryptografischen Bausteine der Homomorphen Verschlüsselung („Homomorphic Encryption" / HE) und Sicherer Mehrparteien-Berechnung („Secure Multi-Party Computation" / MPC) hat sich in den letzten Jahren für FL bewährt. Für das Schützen sensitiver Daten in der Inferenz-Phase hat sich hauptsächlich die Methode des differentiellen Lernens („Differential Privacy") bewährt. Jedoch ist der Bereich des privatsphären-bewahrenden FL aktuell noch in stetiger Entwicklung, und es ist nicht immer sofort einsehbar, welche Methode bzw. dementsprechendes Protokoll am besten für den jeweiligen Anwendungsfall passt; vor allem bei der Trainings-Phase gibt es einige Trade-Offs zwischen HE und MPC, und vor allem auch zwischen den zahlreichen MPC-basierten Protokollen.

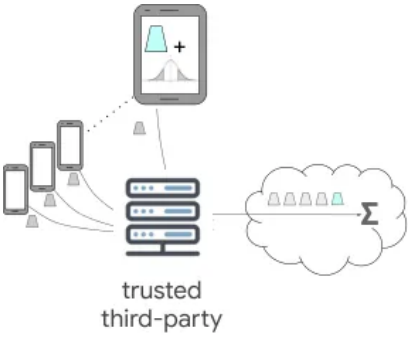
Ziele & Ausblick. Deshalb werden in diesem Bericht zuerst generelle (2) Methoden für Privatsphärenbewahrendes Federated Learning gezeigt, und anschließend (3) MPC-basierte Sichere-Aggregations Protokolle in Federated Learning gezeigt, und basierend auf ihrer Berechnungs- und Kommunikations-Komplexität, und deren Sicherheitsgarantien, miteinander verglichen. Weiters wird ein praktisches (3.2) Beispiel: Google's Gboard gezeigt, welches unter anderem auch ein MPC-basiertes Protokoll integriert. Abschließend werden im Rahmen einer (4) Conclusio & Weiterführende Arbeiten, die gewonnenen Erkenntnisse kurz zusammengefasst und potentiell-interessante weiterführende Richtungen aufgezeigt.

2. Methoden für Privatsphären-bewahrendes Federated Learning

In diesem Abschnitt werden Methoden gezeigt, um „Standard FL" zu einem „privatsphären-bewahrenden FL" zu transformieren. Bei einem FL-Framework, welches in der Praxis eingesetzt wird, muss man auf Sicherheit & Privatsphäre des ganzen Zyklus achten. Z.B. auch ob bei einer Android/iOS-App die aktuellen Versionen von eingesetzten Software-Bibliotheken von Drittanbietern verwendet werden. Da dies über den Rahmen dieses Berichts hinausgeht, ist der Fokus auf ML- bzw. FL-spezifische Aspekte gelegt. Wie in der Einleitung beschrieben, bedingen vor allem die Trainings- und Inferenz-Phase eine Erweiterung um die Privatsphäre der Trainings-Daten von Teilnehmer:innen zu bewahren.

2.1. Trainings-Phase: Sicheres Aggregieren der ML-Parameter

Wie bei zahlreichen ML-Invertierungs-Angriffen gezeigt, bedingt vor allem das Sammeln der aktualisierten ML-Parameter eine Erweiterung. Die Methode des Sicheren Aggregierens („Secure Aggregation" / SecAgg) hat sich in den letzten Jahren diesbezüglich bewährt. Bei SecAgg erhält der Server nur die Summe der aktualisierten ML-Parameter von allen Nutzern . Primäre Ziele von SecAgg sind (I) korrektes Dekodieren & Aufsummieren der aggregierten ML-Parameter (d.h., gleiches Ergebnis wie bei FL ohne SecAgg) mit der Garantie, dass ein gewisser Prozentsatz an Teilnehmern während der FL-Epoche ausfallen darf (Akku leer, keine Netzwerkverbindung, etc.) und (II) Erhalten der Privatsphäre der Trainingsdaten von den individuellen Teilnehmern Für die konkrete Instanziierung von SecAgg, gibt es mehrere Varianten. Peter Kairouz (Google) hat bei einem Vortrag auf der „Privacy-Preserving Machine Learning Summer School" 2022 gezeigt, dass man grundlegend zwischen 3 Methoden von SecAgg unterscheiden kann: (1) via einer vertrauenswürdigen Drittpartei, (2) via Hardware, (3) via kryptografischer Methoden in Software. Abbildung 5 zeigt die 3 grundlegenden Arten von SecAgg.



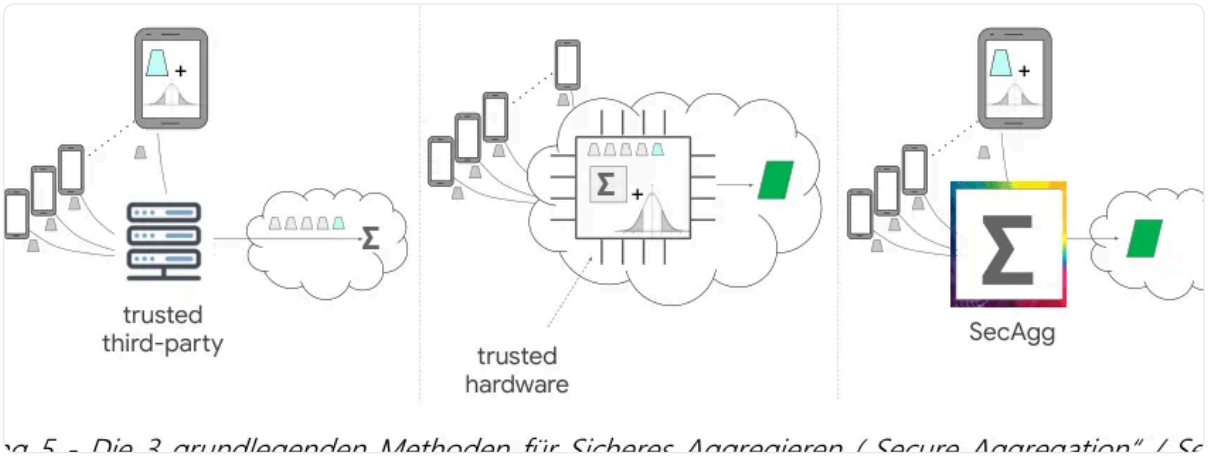


Abbildung 5 - Die 3 grundlegenden Methoden für Sicheres Aggregieren („Secure Aggregation" / SecAgg) von ML-Parametern der Teilnehmer:innen einer FL-Epoche. Via (1) einer vertrauenswürdigen Drittpartei, (2) vertrauenswürdiger Hardware, oder (3) vertrauenswürdiger Software (= erprobte kryptografische Algorithmen (z.B. via MPC (wie hier gezeigt) oder auch via homomorpher Verschlüsselung).





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

Bei (1) – der Drittpartei – wird eine weitere Entität im FL-Framework hinzugefügt, welcher vertraut werden muss. Bei (2) – Hardware – benötigt man dedizierte Hardware-Bausteine um SecAgg zu ermöglichen. Dabei muss man dem Server vertrauen, dass dies korrekt umgesetzt wird. Bei (3) – Software via Kryptografie – werden bereits lokal am jeweiligen End-Nutzer:innen-Gerät die ML-Parameter so „verschlüsselt", sodass der Server (praktisch gesehen) nur die ML-Parameter-Summe von allen Teilnehmer:innen erhält. Da man bei den kryptografischen Bausteinen keine weitere Entität oder spezielle Hardware benötigt, fokussiert sich der weitere Teil auf diese Methode als Erweiterung der FL-Trainings-Phase.

Für die konkrete Instanziierung von kryptografisch-basiertem SecAgg, hat sich der kryptografische Baustein der Sicheren Mehrparteien-Berechnung („Secure Multi-Party Computation" / MPC) und der homomorphen Verschlüsselung („Homomorphic Encryption" / HE) als praktikabel erwiesen. MPC ist ein mittlerweile praktikabler privatsphären-bewahrender kryptografischer Baustein. Mithilfe von MPC können zahlreiche Daten-Analysen bzw. Berechnungen im Allgemeinen so durchgeführt werden, dass die Eingabe-Daten der Teilnehmer:innen geheim bleiben, und nur das Endergebnis entweder einer dedizierten Partei oder allen Teilnehmern bekannt wird. Bei MPC-basiertem SecAgg kommt der Aspekt des Geheimen Teilens („Secret Sharing" / SeSh) zum Tragen. HE ist ebenfalls ein privatsphären-bewahrender kryptografischer Baustein, welcher stetig praktikabler wird. Die Grundidee

von HE basiert darauf, dass man Operationen auf, z.B., 2 Ciphertexten so ausführt, dass das entschlüsselte Produkt das Ergebnis enthält, als hätte man die Operation auf den 2 Klartexten ausgeführt. Z.B. C1(2) * C2(12) = C(24) (die verschlüsselte Zahl 2 multipliziert mit der verschlüsselten Zahl 12 ergibt die Verschlüsselung der Zahl 24).

Grundlegend ist der Vorteil von HE, dass man „nur" der Verschlüsselung bzw. dem kryptografischen Algorithmus – und dessen Implementierung – vertrauten muss. Bei MPC – welches ja auf SeSh basiert – muss zusätzlich darauf vertraut werden, dass der Geheimtext richtig an die jeweiligen Empfänger aufgeteilt wird, und dass die Empfänger nicht zusammenarbeiten um den Geheimtext zu rekonstruieren. Wobei man hierbei anmerken muss, dass bei gewissen MPC- bzw. SeSh-Protokollen – „Malicous Security" - alle Teile des Geheimtextes für die Rekonstruktion benötigt werden (siehe auch beim ASIT-Bericht über MPC generell[8]). Der grundlegende Nachteil von HE ist der wesentlich höhere Rechenaufwand; weshalb oftmals bei dedizierten Anwendungsfällen eine spezielle Form von HE verwendet wird, welche weniger rechenintensiv ist. Hierbei ist MPC grundlegend flexibler und für die Möglichkeit der allgemeinen Berechnungen tendenziell weniger rechenintensiv. Durch die grundlegend höhere Flexibilität von MPC fokussiert sich der weitere Bericht auf MPCbasierte Sichere-Aggregations Protokolle in Federated Learning.

2.2. Inferenz-Phase: Gezieltes Rauschen der Daten/ML-Parameter

Um potentiell-sensitive Trainings-Daten während der ML-Inferenz-Phase zu schützen, hat sich in den letzten Jahren die Methode des „Rauschens" bewährt. Beim „Rauschen" werden, z.B., entweder direkt die Trainings-Daten oder die aktualisierten/neuen ML-Parameter so „verzerrt" bzw. verändert, dass das resultierende ML-Modell noch aussagekräftig ist, aber keine „zu speziellen" Eingaben gelernt werden (wie das Beispiel der Kreditkartennummer in der Einleitung). Für die konkrete Instanziierung des „Rauschens" hat sich die Methode der Differentiellen Privatsphäre[9] („Differential Privacy" / DP) bewährt. Bei DP unterscheidet man grundsätzlich zwischen dem Ansatz, dass der aggregierende Server das Rauschen hinzufügt (= Zentrale DP / „Central DP"), oder dass die jeweiligen Teilnehmer lokal das Rauschen hinzufügen (= Lokale DP / „Local DP").

Da man bei der Zentralen DP dem Server (ganz) vertrauen muss, dass dieser ein Rauschen hinzufügt, ist im Hinblick der Privatsphäre der Ansatz der Lokalen DP vorteilhafter. Jedoch hat die Lokale DP den Nachteil, dass die Genauigkeit des aggregierten Modells tendenziell geringer ist, und somit eventuell mehr Beispiele bzw. Trainings-Epochen benötigt werden. Deshalb hat sich, weiters, ein Modell zwischen der Zentralen und Lokalen DP etabliert: Misch DP[10] [11] [12] („Shuffle DP"). Beim Ansatz der Misch DP werden die Daten generell nicht so stark mit lokalem Rauschen versehen, weil sie dann zu einem vertrauenswürdigen „Kurator" gesendet werden. Der „Kurator" vermischt dann die erhaltenen Daten so durch, dass der (aggregierende) Server keine Rückschlüsse darauf ziehen kann, welche:r Teilnehmer:in welche Daten gesendet hat. Dadurch wird es für einen potentiell Angreifer schwieriger, Rückschlüsse auf die ursprünglichen Trainings-Daten zu ziehen. Allerdings muss es beim Modell des Misch DP eine weitere vertrauenswürdige Entität geben. Der Vorteil bei, z.B., MPC-basierter SecAgg ist, dass der aggregierende Server nur die Summe von allen Teilnehmern erhält, und ohnehin praktisch nicht nachvollziehen kann, welche:r Teilnehmer:in welche ML-Parameter gesendet hat. Weshalb bei MPC-basierter SecAgg der zusätzliche „Kurator" quasi schon „mitintegriert" ist.

Durch diese Trade-Offs an Privatsphäre und Genauigkeit bei den Modellen von DP, wurden in den letzten Jahren speziell Ansätze der Lokalen – und auch Misch - DP in FL untersucht[13] [14] [15] [16]. Lokale bzw. Misch DP ist vor allem auch deshalb interessant, weil es mit, z.B., MPC-basierter SecAgg kombinierbar ist. Abbildung 6 zeigt das Beispiel der Lokalen DP während einer FL-Epoche, angewandt auf die ML-Parameter.

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

![- Die Methode der Lokalen Differentiellen Privatsphäre (“Local DP”) während einer FL-Epoche; angewandt auf die ML-Parameter. Bevor die aktualisierten ML-Parameter an den Server gesendet werden, wird ein Rauschen hinzugefügt.](abbildung-11.webp “Abbildung 6: - Die Methode der Lokalen Differentiellen Privatsphäre (“Local DP”) während einer FL-Epoche; angewandt auf die ML-Parameter. Bevor die aktualisierten ML-Parameter an den Server gesendet werden, wird ein Rauschen hinzugefügt.”)

3. MPC-basierte Sichere-Aggregations Protokolle in Federated Learning

Seit der FL-Einführung von Google im Jahre 2016/17, wurden zahlreiche Methoden entwickelt um SecAg in FL immer praktischer zu gestalten. Und MPC-basiertes SecAgg ist eine der gängigsten praktisch-relevanten Instanziierungen von SecAgg. Deshalb werden in diesem Abschnitt ausgewählte MPC-basierte SecAgg-Protokolle aufgezählt und basierend auf ihrem theoretischen Rechen- und Kommunikationsaufwand miteinander verglichen. Weiters wird ein in der Praxis eingesetztes Beispiel von MPC-basiertem SecAgg, in Kombination mit DP, gezeigt: Google’s Gboard.

3.1. Überblick & Vergleich

Seit der Einführung von FL in 2016/17, wurden zahlreiche MPC-basierte SecAgg-Protokolle entwickelt. Dabei ist es stets ein Trade-Off von Rechenaufwand, Kommunikationsaufwand, und wie vielen Entitäten bzw. Aggregatoren vertraut wird. Tabelle 1 gibt einen Komplexitäts-Überblick über folgende Protokolle, welche aus einem Mix aus Einführungs-Protokoll, Nachfolger und Dezentralisierungs-Ansätze ausgewählt wurden, um die grundlegenden Variationen von MPC-basiertem SecAgg zu zeigen:

- 2017 SecAgg

  “Practical Secure Aggregation for Privacy-Preserving Machine Learning” (Bonawitz, et al., 2017) CCS ¨ Der Beginn von praktisch-relevantem MPC-basiertem SecAgg. Verwendet paarweise Masken, welche beim Aggregieren/Aufsummieren wegfallen.

- 2020 SecAgg+

"Secure Single-Server Aggregation with (Poly)Logarithmic Overhead" (Bell, Bonawitz, Gascón, Lepoint, & Raykova) CCS ¨ Weiterentwicklung von SecAgg. Anstatt die ML-Parameter für alle anderen Teilnehmer aufzuteilen (=n), werden die Ml-Parameter „nur" für log(n) andere Teilnehmer aufgeteilt; was eine Kommunikations-Reduktion von ~log() bedeutet.

- 2020 FastSecAgg

  "Scalable Secure Aggregation for Privacy-Preserving Federated Learning" (Kadhe, Rajaraman, Koyluoglu, & Ramchandran, 2020) arXiv ¨ Anstatt des „klassischen" Geheimen Aufteilens basierend auf dem Protokoll von Shamir, werden die geheimen Teile basierend auf der Schnellen Fourier-Transformation[17] („Fast Fourier Transformation" / FFT) generiert bzw. rekonstruiert.

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

- 2021 SAFELearn

  „Secure Aggregation for private FEderated Learning" (Fereidooni, et al., 2021) S&P Workshops ¨ Generisches Framework für grundlegend dezentralisiertes FL, in dem das resultierende globale ML-Modell nur die Teilnehmer erhalten. Für SecAgg können z.B. die kryptografischen Bausteine HE und auch 2/MPC ausgewählt werden.

- 2022 LightSecAgg

  "Lightweight and Versatile Design for Secure Aggregation in Federated Learning" (So, et al., 2020) PMLS ¨ Grundlegende Idee ist SecAgg bzw. SecAgg+ mit einem modifizierten Maskierungs-Ansatz zu verbessern, welcher die Anzahl der Kommunikationsrunden reduziert.

- 2022 SCOTCH

  "Efficient Secure Computation Framework for Secure Aggregation" (More, et al.) arXiv ¨ Basiert auf additivem Geheimen Teilen und verwendet m vertrauenswürdige Aggregationsserver, sodass grundsätzlich nur die Teilnehmer das resultierende globale ML-Modell erhalten; was sozusagen einem dezentralisierten Ansatz für FL entspricht. Abbildung 7 zeigt die grundlegenden Komponenten und deren Kommunikationsfluss von SCOTCH.

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

Trade-Offs. Jedes der Protokolle bewahrt grundsätzlich die Privatsphäre der Trainings-Daten der Teilnehmer. Wobei jedes Protokoll unterschiedliche Trade-Offs mit sich bringt. Z.B. fokussieren sich SAFELearn und SCOTCH auf den Ansatz des „dezentralisierten FL", indem nur die Teilnehmer das resultierende globale ML-Modell erhalten. Der bzw. die Aggregations-Server aggregieren „nur" die ML-Parameter der Teilnehmer, in einer Art und Weise, dass nur die Teilnehmer die Aggregation entschlüsseln bzw. rekonstruieren können. Jedoch benötigen diese 2 Protokolle ≥2 vertrauenswürdige Aggregations-Server. Alle anderen Protokolle fokussieren sich auf den Ansatz des „zentralisierten FL", indem grundlegend nur der Aggregations-Server das resultierende globale ML-Modell erhält, und dann gegebenenfalls an die jeweiligen Teilnehmer sendet.

FastSecAgg bietet grundlegend einen guten Berechnungs-Aufwand und verwendet auch eine neuartige Methode um die ML-Parameter aufzuteilen (Schnelle Fourier Transformation / FFT). Auch LightSecAgg bietet grundlegend einen guten Berechnungs-Aufwand, indem es den Maskierungs-Ansatz von SecAgg/SecAgg+ erweitert. Weiters bieten beide Protokolle – FastSecAgg & LightSecAgg – eine relativ-geringe Anzahl an Kommunikationsrunden; jedoch „nur" im passiven Angreifer-Modell. SecAgg bzw. dessen Nachfolger SecAgg+, gegenüber, bieten beide auch eine Variante für das aktive Angreifer-Modell. Weiters bietet SecAgg+, durch spezielles Auswählen an „Maskierungs-Nachbarn" in einem Graphen, zudem einen grundlegend guten/vergleichbaren Kommunikationsaufwand.

Angreifer-Modelle. Bei FL unterscheidet man grundsätzlich zwischen 2 Angreifer-Modellen: Passiv bzw. halb ehrlich ("semi-honest") und Semi-Aktiv bzw. bzw. „halb-bösartig" („semi-malicious"). Beim passiven Modell, folgen der Server und Teilnehmer dem Protokoll, versuchen aber aus den erhaltenen Informationen so viel wie möglich zu lernen. Dabei kann es auch sein, dass der Server und eine Teilmenge der Teilnehmer zusammenarbeiten, um daraus mehr Informationen abzuleiten. Beim semi-aktiven Modell, wird dem Server bei der Einrichtungsphase vertraut, dass er z.B. ein „korrektes" ML-Modell aussendet und gegebenenfalls asymmetrisches Schlüsselmaterial korrekt weiterleitet. Jedoch kann es während dem eigentlichen SecAgg-Protokoll sein, dass der Server dem Protokoll nicht folgt, und z.B. manchen Teilnehmern inkorrekte Informationen darüber gibt, welche Teilnehmer noch aktiv in der jeweiligen FL-Epoche sind und welche inaktiv geworden sind („droput"). Bei einem rein aktiven Modell („fully-malicious"), würde man dem Server auch bei der Einrichtungsphase nicht zwingend vertrauen müssen.

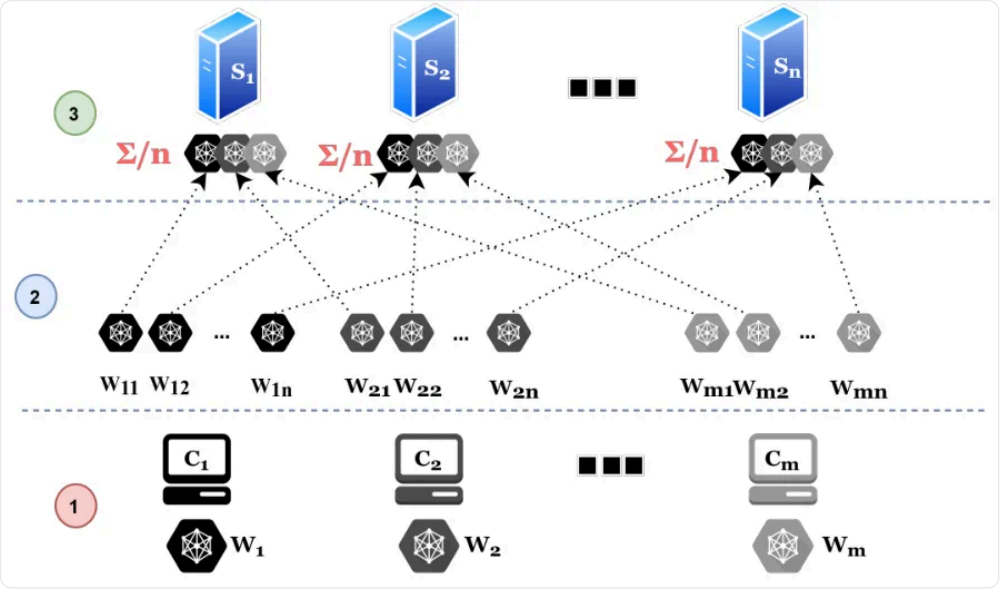


Abbildung 7 - Überblick der Komponenten und deren Kommunikationsflüsse des dezentralisierten MPC-basierten SecAgg-Protokolls SCOTCH.

MPC- basiertes

SecAgg- Protokoll

Variante

bzw. Angreifer-

Modell

Rechenaufwand *(“Computational Cost”)*

Anzahl Kommunikations-

Runden für Aggregieren (Teilnehmer←

Kommunikations-Kosten

***(“Communication Cost”)* Aggregation**

>Server)

Server Teilnehmer Server Teilnehmer

Tabelle 1 - Überblick von MPC-basierten SecAgg-Protokollen für FL, mit einem Fokus auf die Aggregations-Komplexität. n...Anzahl der Teilnehmer; L... Länge bzw. Anzahl der ML-Parameter; bei LightSecAgg: U...min. Anzahl an Teilnehmern für Rekonstruktion + T...max. Anzahl von korrupten Teilnehmern; bei SCOTCH: m...Anzahl an (vertrauenswürdigen Aggregations-Servern.

3.2. Beispiel: Google’s Gboard

Ein Beispiel für ein bereits praktisch eingesetztes FL-gelerntes ML-Modell ist Google’s Gboard. Gboard ist eine virtuelle Tastatur für Android[17] - und iOS[18] -Geräte. Google hat auch einen wissenschaftlichen Artikel über die Sicherheit & Privatsphäre, Effizienz und Genauigkeit von Gboard verfasst (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

Abbildung 8: - Logo von Google's Gboard (App für virtuelle Tastatur).

Bildquelle: https://en.wikipedia.org/wiki/Gboard

Im Hinblick auf ML bzw. FL, integriert Gboard ML-Sprach-Modelle für, z.B., die Vorhersage des nächsten Wortes („Next-Word Prediction") oder der Schreib-Unterstützung während dem Eintippen eines Wortes („Smart Compose"). Abbildung 9 zeigt jeweils ein Beispiel dieser 2 ML-Modelle für End-Nutzer. Gboard's ML-Modell baut auf ein neuronales Netzwerk mit 670 „Versteckte Neuronen" („Hidden Neurons") auf[19]. Das ML-Modell für die aktive Schreib-Unterstützung, z.B., benötigt ein Vokabular von ca. 30.000 Wörtern, mit einer Anzahl von ca. 6,4 Millionen ML-Parametern.

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

Um die Privatsphäre während der Trainings- und Inferenz-Phase zu bewahren, kombiniert Gboard die Methoden von MPC-basiertem SecAgg (Trainings-Phase) und DP (Inferenz-Phase). Wobei die meisten Varianten von Gboard nur DP für beide Phasen verwenden. Abbildung 10 zeigt Gboard's ML-Modell-Fluss einer FL-Epoche. Einerseits wird dadurch die Privatsphäre der Trainings-Daten der Teilnehmer bewahrt, und andererseits kann durch den Einsatz von FL auf lokal-spezifische Anforderungen eingegangen werden. Z.B. ist es mit dem Ansatz von Gboard möglich mit Teilnehmern von Österreich/Deutschland/Schweis ein speziell für die deutsche Sprache geeignetes ML-Modell zu entwickeln; oder, z.B., ein dediziertes ML-Modell für Teilnehmer aus Brasilien.



Abbildung 10: - ML-Modell-Fluss einer FL-Epoche bei Google's Gboard. Um die Privatsphäre zu bewahren werden DP und MPC-basiertes SecAgg verwendet.

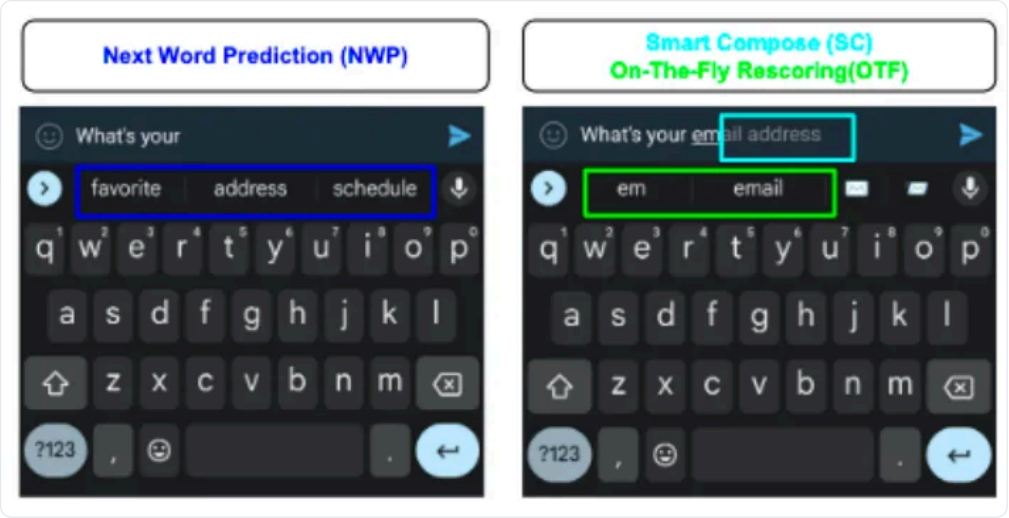
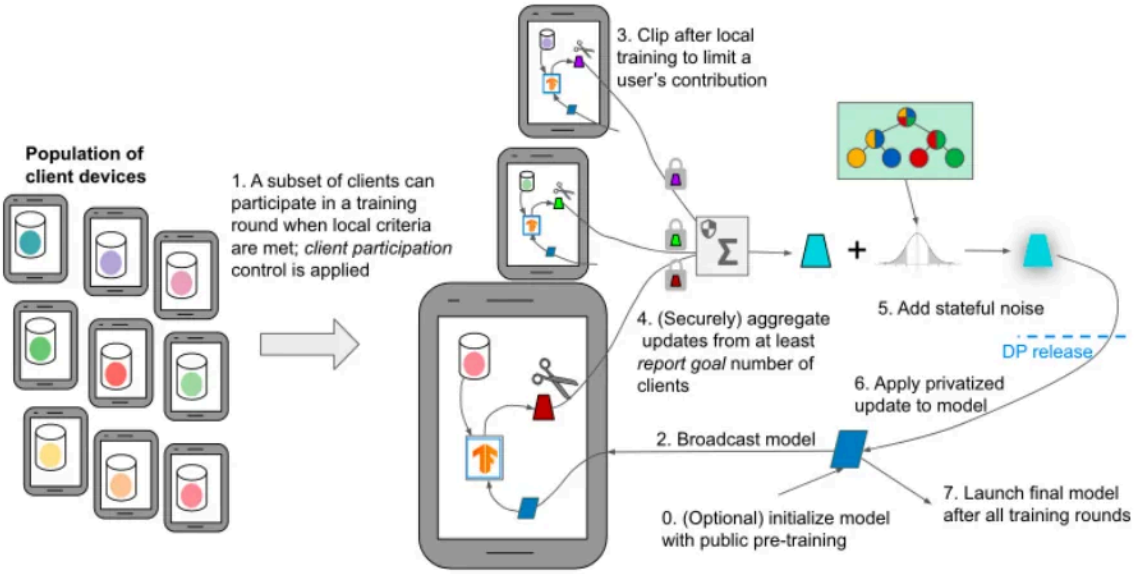


Abbildung 9 - Jeweils ein Beispiel von der Vorhersage des nächsten Wortes (links) und der Schreib-Unterstützung während dem Eintippen eines Wortes (rechts), bei der virtuellen Tastatur Gboard (von Google), mithilfe von FLtrainierten ML-Sprach-Modellen.



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

4. Conclusio & Weiterführende Arbeiten

In diesem Abschnitt wird zuerst der Bericht konkludiert, und im Anschluss daran Potentiale für fortführende Richtungen bzw. weiterführende Arbeiten aufgezeigt.

Föderiertes Lernen („Federated Learning" / FL), mit Erweiterungen in der Trainings- und Inferenz-Phase, ermöglicht es die Privatsphäre der Trainings-Daten von Teilnehmern zu bewahren. Für die Trainings-Phase haben sich die kryptografischen Bausteine HE und MPC bewährt. Wobei MPC-basiertes SecAgg zwar Vertrauen in den Server bzw. einer Teilmenge von andere Teilnehmer bedingt, bietet diese Methode grundsätzlich mehr Flexibilität in der praktischen Umsetzung (von vor allem X-Gerät-FL). Und auch in MPC-basiertem SecAgg, gibt es zahlreiche Protokolle, welche unterschiedliche Trade-Offs bieten. Z.B. ob (nur) die Teilnehmer das resultierende globale ML-Modell erhalten (z.B. SAFELearn, SCOTCH), oder primär nur der Aggregations-Server (z.B. SecAgg, SecAgg+, FastSecAgg, LightSecAgg). Für die Inferenz-Phase hat sich die Methode der Differentiellen Privatsphäre („Differential Privacy" / DP) bewährt. Wobei, wie bei nahezu allen Methoden, jede Methode unterschiedliche Trade-Offs zur Folge hat. Weiters hat (privatsphären-bewahrendes) FL auch den Vorteil, dass es spezielle, z.B., örtlich-angepasste ML-Modelle entwickeln kann; wie von Google's Gboard gezeigt.

Für den weiteren Einsatz bzw. Verbreitung von privatsphären-bewahrendem FL müssen die unterschiedlichen Trade-Offs für die jeweiligen Anwendungsszenarien näher untersucht werden. Z.B. wie hoch der Grad der Privatsphäre, abhängig von der Anzahl an Teilnehmern, ist[20]. Zudem ist es auch notwendig die unterschiedlichen Herausforderungen von FL zu lösen; wie, z.B., die der Daten-Heterogenität auf End-Nutzer-Geräten[21].

Literaturverzeichnis

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- ↩