Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

Privacy-Preserving Federated Learning for Secure Healthcare Data: A Sensitivity-Aware Selective Encryption Framework for Cross-Institutional Clinical Model Training

Dhwani Jain, Bharat Thathera

Department of Artificial Intelligence and Data Science, Poornima Institute of Engineering and Technology, Jaipur, India

Corresponding Author: dhwani16jain@gmail.com

ABSTRACT

Federated learning (FL) lets hospitals train a shared model without pooling patient records, but exchanged updates can still leak sensitive information, and homomorphic encryption (HE) of every parameter is costly for resource-limited clinical sites. This paper proposes SASHE-FL, a sensitivity-aware selective HE framework for cross-silo healthcare FL. Sites use SCAFFOLD-style local training and encrypt, with the CKKS scheme, a jointly agreed top-p fraction of parameters ranked by a label-sensitivity score; the server aggregates ciphertexts without decrypting them. A prototype with real CKKS encryption (TenSEAL) was evaluated on the Wisconsin breast-cancer dataset, split non-IID across five simulated institutions. Accuracy matched plaintext SCAFFOLD at every encrypted fraction (96.5 ± 1.4%). Encrypting 10% of parameters cut per-site upload by 81.5% and cryptographic compute time by 86.3% versus encrypting all parameters. However, a worst-case gradient-inversion probe still recovered inputs (cosine similarity 0.999) with the top 30% encrypted and failed only when the entire first layer was encrypted. SCAFFOLD gave no accuracy gain over FedAvg. Protection covers only encrypted coordinates, against a non-colluding semi-honest server. The results are a small proof of concept, not clinical evidence or a formal privacy guarantee.

Keywords: CKKS; federated learning; gradient leakage; healthcare data; selective homomorphic encryption.

1. INTRODUCTION

The digitisation of clinical workflows has generated huge amounts of digital health records, medical images and sensor data of immense value for diagnostic modelling and clinical decision support. Models trained at one hospital represent a small group of patients, and generally good models need to be trained at multiple hospitals [13,24]. It is in general not feasible to have all this information centralized. There are regulations like the Health Insurance Portability and Accountability Act (HIPAA) and the General Data Protection Regulation (GDPR) that limit the transfer of identifiable patient information, and a central repository pulls together breach risk [5,22,29]. Federated learning (FL) [19] avoids this by keeping raw data at each institution: sites train locally and send only model updates to a server that aggregates them. FL has been applied to diagnostic tasks across multiple clinical sites [18,24]. FL is not, however, a privacy guarantee. Shared gradients can be inverted to approximately reconstruct training inputs [6,32], models and updates can support membership inference [25], and leakage accumulates over communication rounds [30]. A second obstacle is statistical heterogeneity. Hospitals differ in patient mix, disease prevalence, equipment, and labeling practice, which yields non-independent and identically distributed (non-IID) client data that can slow or degrade Federated Averaging (FedAvg) [8,16]. Control-variate methods such as SCAFFOLD (Stochastic Controlled Averaging for Federated Learning) [14] were designed to correct the resulting client drift. Cryptographic protection exists but is costly. Differential privacy (DP) [1,11] adds noise and can reduce accuracy. Secure aggregation and secure multi-party computation

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

(MPC) [3,28] need interactive protocols and online clients. Homomorphic encryption (HE), in particular the Cheon–Kim–Kim–Song (CKKS) scheme [4] for approximate arithmetic on real numbers, lets a server aggregate encrypted updates directly, but encrypting every parameter inflates communication and computation, which burdens hospital IT environments [18,31]. Recent work therefore encrypts only the most privacy-critical parameters, chosen by a sensitivity score [10,12].

This paper studies that idea end to end. We propose SASHE-FL (Sensitivity-Aware Selective Homomorphic Encryption Federated Learning), implement it with real CKKS encryption, and measure what selective encryption buys and what it does not. The contributions are:

1) A protocol that combines SCAFFOLD-style local training with CKKS encryption of a shared, sensitivity-ranked subset of coordinates. The same mask covers model updates and control-variate updates, and the mask is agreed through encrypted aggregation of per-site sensitivity maps, so all sites encrypt the same coordinates, and the server can add ciphertexts.
2) A closed-form, single-pass sensitivity score for sigmoid/cross-entropy networks that equals the label-derivative sensitivity used in [10] exactly (Section 4.4).
3) A working prototype with real CKKS operations, evaluated on a non-IID five-site partition of a public clinical dataset, reporting accuracy, per-round communication, and cryptographic compute time as functions of the encrypted fraction p (Section 6).
4) An empirical privacy probe with a negative finding: ranking-based selection lowers the gradient energy left in plaintext, yet an analytic gradient-inversion attack on single-sample gradients (a worst-case bound) succeeds unless the whole first layer is encrypted. We describe and cost a first-layer guard variant.
5) Transparent reporting of null results, including that SCAFFOLD gave no accuracy gain over FedAvg on this task.

The remainder of the paper is organised as follows. Section 2 defines the notation (Table 1) and the background. Section 3 reviews related work and states the research gap. Section 4 presents SASHE-FL: the architecture (Figure 1), the threat model (Figure 2), heterogeneity-aware local training (Eqs. (3)–(4)), sensitivity scoring and mask agreement (Eqs. (5)–(7)), selective encrypted aggregation (Eqs. (8)–(9), Figure 3), and the protocol (Figure 4, Algorithm 1, Table 3). Section 5 describes the prototype and experimental setup, Section 6 the results (Table 5, Table 6, Table 7, Figures 5–6), Section 7 the security and privacy analysis, Section 8 the limitations and future work, and Section 9 concludes.

2. BACKGROUND

2.1 Federated Learning

Consider N institutions, each holding a private dataset D_i, and a coordinating server. FL seeks a global model w that minimizes

$$F(w)=\Sigma_{i=1}^{N}\alpha_i F_i(w), \quad (1)$$

where F_i is the local empirical loss of site i and $\alpha_i = |D_i|/|D|$ [19]. Here $i \in \{1,\ldots,N\}$ indexes the institutions, D is the union of all private datasets (which are never pooled in practice), $w \in \mathbb{R}^M$ is the vector of the M model parameters, and the weights satisfy $\Sigma_i \alpha_i = 1$. In each round the server sends w^t, every site trains locally, and the server aggregates by weighted averaging (FedAvg):

$$w^{t+1}=\Sigma_{i=1}^{N}\alpha_i w_i^{t+1}. \quad (2)$$

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

In Eq. (2), the superscript $t \in \{0,\ldots,T-1\}$ is the communication round index (T is the total number of rounds), w^t is the global model that the server sends at the start of round t, w_i^{t+1} is the local model returned by site i after local training in round t, and FedAvg stands for Federated Averaging [19].

2.2 Non-IID Data and Client Drift

If the clients' data are not independent and identically distributed (non-IID), that is, if their distributions differ, local updates tend to move towards each site's own optimum, which is known as client drift that slows down convergence under plain averaging [8,14]. SCAFFOLD (Stochastic Controlled Averaging for Federated Learning) [14] corrects client drift with control variates, which are vectors of the same dimension M as the model. The server keeps a global control variate $c \in \mathbb{R}^M$, an estimate of the average update direction over all sites. Each site $i \in \{1,\ldots,N\}$ keeps a local control variate $c_i \in \mathbb{R}^M$, an estimate of the update direction of its own loss F_i; both are initialised to zero. During local training, the correction term $(c - c_i)$ is added to every gradient step (Eq. (3)), which pulls each site's trajectory toward the global direction instead of its own optimum [14].

2.3 Privacy Threats

There are three types of attacks to consider. Gradient inversion is used to reconstruct training data through shared gradients [6,32]. Membership inference tests whether a record was used in training [25]. Model reconstruction (or extraction) tries to rebuild a model's parameters or behavior from repeated observations of its outputs or updates. A semi-honest (honest-but-curious) server follows the protocol but tries to learn from everything it sees; because it observes each round, so small leaks can accumulate [30].

2.4 CKKS Homomorphic Encryption

CKKS [4] (Cheon–Kim–Kim–Song) is a homomorphic encryption scheme for encrypting vectors of real numbers that supports approximate addition and multiplication. For a polynomial degree of 8192, there is one ciphertext that has ns = 4096 slots (SIMD, single instruction multiple data, packing) where thousands of parameters are encrypted, added, or multiplied by plaintext constants in one operation [2]. A ciphertext is far larger than the message it encrypts, because each value is encoded at a fixed-point scale (here 2^{40}) into polynomials with a coefficient modulus of about 200 bits. Approximation errors are present in decrypted data. We use the TenSEAL library [2].

Terminology. A *ciphertext* is an encrypted vector; a *slot* is one position in it; the *scale* is the fixed-point factor by which real values are multiplied before encoding; and the *coefficient modulus* bounds the size of the polynomial coefficients and, together with the polynomial degree, sets the security level and the number of multiplications that can be performed. In CKKS, adding two ciphertexts is cheap, while multiplying a ciphertext by a plaintext vector consumes one multiplicative level and is followed by a rescaling. Table 1 lists the notation used in all equations of this paper.

Table 1. Notation Used in This Paper

Symbol	Meaning	Range / size
N, i	number of institutions; site index	$i \in \{1,\ldots,N\}$; $N = 5$
D_i, D	private dataset of site i; union of all datasets	$\alpha_i = \lvert D_i\rvert/\lvert D\rvert$
α_i	aggregation weight of site i (Eqs. (1), (2), (8))	$\Sigma_i\, \alpha_i = 1$
F_i, F	local loss of site i; global objective (Eq. (1))	–
M, m	number of model parameters; parameter index	$m \in \{1,\ldots,M\}$; $M = 40{,}961$
t, T	communication round index; number of rounds	$t \in \{0,\ldots,T-1\}$; $T = 30$

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

Symbol	Meaning	Range / size
w^t	global model at the start of round t	$\mathbb{R}^M$
y_i	local model of site i during local training (starts at w^t)	$\mathbb{R}^M$
τ, η	local steps per round; learning rate	10; 0.05
$g_i(\cdot)$	mini-batch stochastic gradient of F_i (Eq. (3))	$\mathbb{R}^M$
c, c_i	global and local control variates	$\mathbb{R}^M$; start at 0
c_i^+	updated local control variate (Eq. (4))	$\mathbb{R}^M$
Δy_i, Δc_i	model update and control-variate update of site i	$\mathbb{R}^M$
$\Delta\bar{y}$, $\Delta\bar{c}$	averaged updates after aggregation	$\mathbb{R}^M$
q	number of local samples used for scoring	–
k, ξ_k, v_k	sample index; input features; label	$k \in \{1,\ldots,q\}$; $v_k \in \{0,1\}$
ℓ_k, z_k, σ	loss, logit and sigmoid for sample k	–
$S_i(m)$, $\bar{S}(m)$	sensitivity of parameter m at site i; mean over sites (Eqs. (5)–(7))	≥ 0
p	encrypted fraction of parameters	$p \in (0, 1]$
n_l	number of parameters in layer l	–
K, U	encrypted and plaintext index sets	$K \cup U = \{1,\ldots,M\}$; $\lvert K\rvert \approx pM$
TopP$(\cdot, p)$	layer-wise choice of the $\lceil p\, n_l \rceil$ largest entries	–
δ_i	packed vector $[\Delta y_{i,K}; \Delta c_{i,K}]$ (Eq. (8))	$\mathbb{R}^{2\lvert K\rvert}$
ω_i	plaintext weight vector (α_i on updates, $1/N$ on control variates)	$\mathbb{R}^{2\lvert K\rvert}$
Enc$(\cdot)$, Dec$(\cdot)$	CKKS encryption and decryption	uses pk, sk
pk, sk	CKKS public and secret keys	–
$\odot$	element-wise product of a plaintext vector and a ciphertext (Eq. (8))	–
C_K	aggregated ciphertext (Eq. (8))	–
n_s, n_{ct}	slots per ciphertext; ciphertexts per site per round (Eq. (9))	$n_s = 4096$
$\boldsymbol{\Delta}$, δ (Secs. 4.4, 6.4)	back-propagated logit derivatives; hidden-layer error (not the updates Δy, Δc)	–
β	Dirichlet concentration of the partition (Sec. 5)	0.5
C, z (Sec. 5)	clipping bound and noise multiplier of the DP baseline (not the ciphertext C_K)	$C = 1$; $z \in \{1, 4\}$

3. RELATED WORK

3.1 Differential Privacy

DP adds calibrated noise so that any single record has bounded influence on the released output [1]. A client-level DP method for federated medical imaging is presented in [11]. DP adds no ciphertext expansion but trades accuracy for privacy; Lin *et al.* [18] report that this DP baseline reached lower F1 than their encrypted framework on skin-lesion benchmarks, and our own DP baseline shows the same direction (Section 6.1).

3.2 Homomorphic Encryption and Selective Encryption

Lin *et al.* [18] combine SCAFFOLD-style optimization with layer-wise packed CKKS aggregation for skin-lesion classification. They report F1 gains of 2.2 and 4.0 points over baseline FL schemes, communication reductions of 94.85% and 93.48% relative to unpacked HE, and 17.8 s per-sample encrypted inference. Their scheme still encrypts every parameter. Selective schemes encrypt a ranked subset instead: FedML-HE [12] and Jeong and Choi [10], who compress local sensitivity maps to the top fraction (for

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

example 30%) and report 80.67% lower latency and 57.19% lower memory in the map encoding, encryption, and transmission stages on an embedded setup. That study does not report end-to-end FL accuracy or leakage, which is the gap our prototype addresses.

3.3 Secure Aggregation and MPC

Bonawitz *et al.* [3] aggregate updates with secret sharing and tolerate client dropout, at the price of several communication rounds per iteration. MPC-based encrypted training for cancer imaging [28] gives strong guarantees, and Lin *et al.* [18] report training times more than ten times longer than their HE-based framework. Hu *et al.* [9] build aggregation from one-way trapdoor permutations without public-key HE and show lower computation than Paillier-based schemes in simulation; their evidence covers computation cost only, and they note that designs in which all clients share one key cannot resist collusion between the aggregator and a client.

3.4 Committee and Blockchain Mechanisms

Shu *et al.* [26] let a rotating committee score submitted gradients, encrypt uploads with CKKS, and record the process on a blockchain. On MNIST they report better robustness to label-poisoning and gradient-scaling attacks than Multi-Krum, CMFL, and plain averaging. This targets integrity rather than the confidentiality of honest sites' updates.

3.5 Encrypted Matching, Platforms, and Benchmarks

Hsu and Huang [7] use homomorphic comparison and equality tests so that a task initiator can find matching training data without revealing its criteria; on a heart-disease dataset they report 68.7% accuracy with encrypted training versus 69.2% in plaintext. Korkmaz et al. [15] benchmarked five CNN architectures under four federated learning optimizers across five medical image datasets, and found FedAvg to be the most reliable, followed by FedAvgM. APPFLx [17] is an application for cross-silo federated learning as a service that includes federated identity management and privacy settings based on differential privacy. Pawar et al. [22] ran FedAvg on four virtual hospitals with a Pima diabetes dataset to identify differential privacy, homomorphic encryption, and blockchain as directions for future research.

Table 2. Summary of Reviewed Studies

Work	Mechanism	Evidence reported	Limitation relevant here
Lin *et al.* [18]	SCAFFOLD-style optimizer, layer-wise packed CKKS, FHE-friendly CNN	HAM10000, ISIC2019: F1 +2.2 / +4.0 points; communication −94.85 % / −93.48 % vs. unpacked HE; 17.8 s encrypted inference	Encrypts all parameters; upload still tens of MB per round
Jeong and Choi [10]	Selective HE from compressed sensitivity maps (top-p)	−80.67 % latency, −57.19 % memory for sensitivity-map encryption and transmission	No FL training accuracy or leakage reported
Hu *et al.* [9]	Aggregation from one-way trapdoor permutations, no public-key HE	Lower computation than Paillier baselines in simulation	No model-training accuracy; no deep models
Shu *et al.* [26]	Committee scoring, CKKS, blockchain	MNIST: more robust than Multi-Krum, CMFL, plain FL under poisoning	Integrity focus; small benchmark
Hsu and Huang [7]	Encrypted matching and preprocessing, CKKS FL	Heart-disease data: 68.7 % (encrypted) vs. 69.2 % (plain)	Small logistic-regression setting
Korkmaz *et al.* [15]	Benchmark: 5 CNNs × 4 FL optimizers	FedAvg best, then FedAvgM; FedAdam and FedYogi often poor	No cryptographic protection
Li *et al.* [17]	FL as a service, federated identity, DP settings	MNIST demo on 5 heterogeneous sites	No clinical evaluation; updates not encrypted

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

Work	Mechanism	Evidence reported	Limitation relevant here
Pawar *et al.* [22]	FedAvg over 4 virtual hospitals	Pima diabetes; near-centralized accuracy (reported qualitatively)	No privacy mechanism implemented

3.6 Research Gap

The reviewed work shows several trade-offs. Differential privacy (DP) adds no ciphertext expansion but costs some accuracy. Secure aggregation and MPC give strong confidentiality but require interaction among participants and scale poorly. Encrypting all parameters with CKKS is powerful but costly. Selective encryption lowers that cost, but its published evaluations stop at the sensitivity-map stage and do not report end-to-end training accuracy or leakage. Committee-based mechanisms address the integrity of submitted updates rather than the confidentiality of honest sites' updates. To our knowledge, none of the reviewed studies measures accuracy, communication, cryptographic compute time, and a leakage test together for a selectively encrypted FL algorithm. This paper addresses that gap.

4. PROPOSED SASHE-FL FRAMEWORK

4.1 Architecture

SASHE-FL involves three roles (Figure 1): (1) participating institutions that keep their data local, train, rank parameters by sensitivity, and encrypt; (2) a semi-honest aggregation server that adds ciphertexts and averages plaintext coordinates; and (3) a key management authority (KMA) that generates a CKKS key pair, gives the public key to the server, and gives the secret key to the institutions only.

Figure 1 shows the information flow of one training round; the steps below follow its arrows. (1) The KMA generates one CKKS key pair (pk, sk), sends pk to the server, and sends sk to the N institutions only. (2) Each institution $i \in \{1,\ldots,N\}$ trains on its private dataset D_i, which never leaves the site, using the corrected local steps of Eqs. (3) and (4) (Section 4.3). (3) It splits its update Δ_i into an encrypted part $\Delta_{i,K}$, indexed by the set K, and a plaintext part $\Delta_{i,U}$, indexed by the set U, and uploads $\mathrm{Enc}(\Delta_{i,K})$ together with $\Delta_{i,U}$ (solid arrows). Here Δ_i stands for the model update Δy_i and the control-variate update Δc_i of Section 4.3. (4) The server forms the weighted ciphertext sum $C_K = \Sigma_i\, \alpha_i\, \mathrm{Enc}(\Delta_{i,K})$ of Eq. (8) and the plaintext weighted average $\Delta_U = \Sigma_i\, \alpha_i\, \Delta_{i,U}$; it holds no secret key. (5) The server broadcasts C_K and Δ_U (dashed arrows); each institution decrypts C_K locally with sk and merges it with Δ_U to update the model (Section 4.5).

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

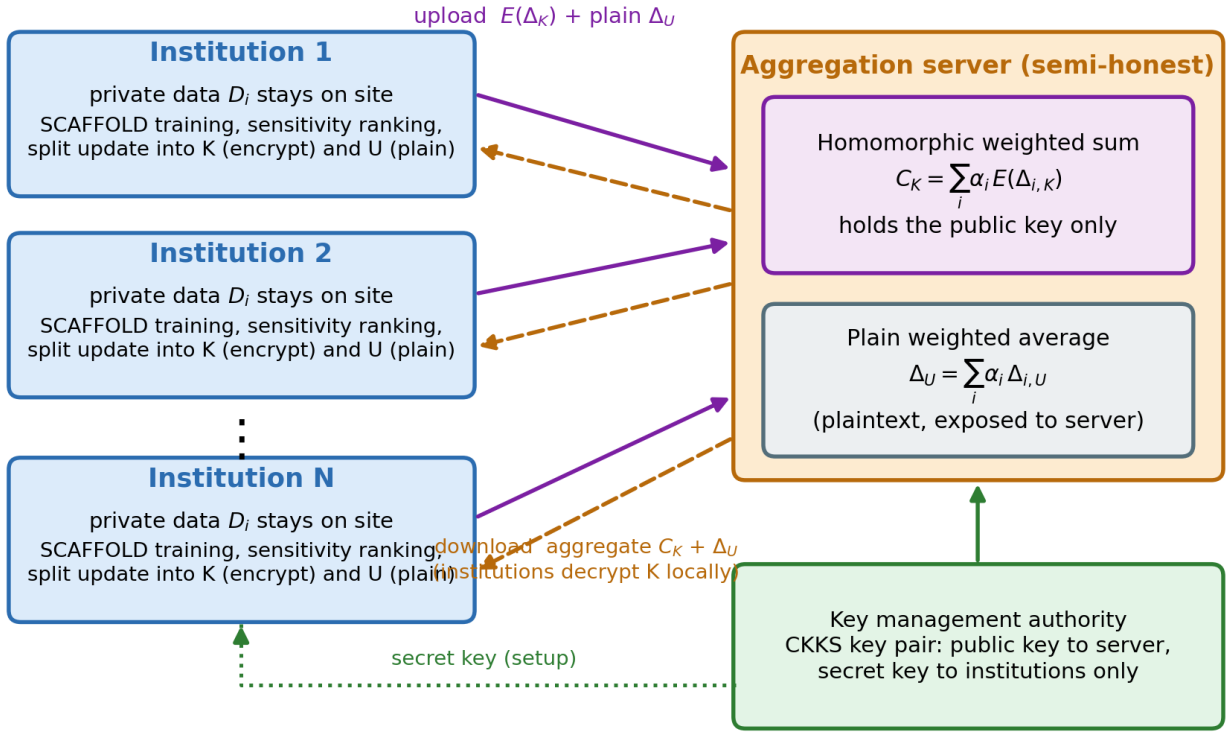


Figure 1. Architecture of SASHE-FL. Each site splits its update into an encrypted subset K and a plaintext subset U. The server holds only the public key.

4.2 Threat Model

The server is semi-honest: it follows the protocol but tries to learn from what it observes. An external eavesdropper may read channel traffic; we assume authenticated, Transport Layer Security (TLS)-protected links so that messages cannot be modified undetected. Institutions are honest-but-curious toward one another. All institutions share one CKKS secret key, generated by a key management authority (KMA) during a trusted setup. We therefore assume explicitly that (i) the KMA is trusted and deletes its copy of the secret key after distributing it, and (ii) the server does not collude with any institution. If the server colludes with even one institution, it can decrypt every site's encrypted update and the protection of the encrypted subset is lost; likewise, any institution that obtains another site's ciphertext can decrypt it (Section 7 discusses threshold key generation as a mitigation). Protection is limited to the encrypted subset K against the server and eavesdroppers: the plaintext subset U, the mask, and the released global model are not protected. Byzantine institutions that submit corrupted updates are out of scope; committee-based verification [26] addresses them. Figure 2 summarizes the adversaries, attacks, and protections.

Figure 2 states, for each of the three adversaries, what it observes, the attack it can mount, and the protection that applies. (a) The semi-honest server sees every upload and holds only the public key. It can run gradient inversion on the plaintext subset U, but it cannot decrypt the encrypted subset K, because CKKS ciphertexts are IND-CPA secure under the Ring Learning With Errors assumption (Section 7). (b) An external eavesdropper sees all channel traffic and holds no key; TLS on every link together with the encryption of K means that it learns no more than the server does. (c) A curious institution holds the secret key and sees the broadcast aggregate; it can decrypt whatever it receives and can infer information from the released global model, which is why sites receive only the aggregate and why collusion between the server and a site is excluded by assumption.

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

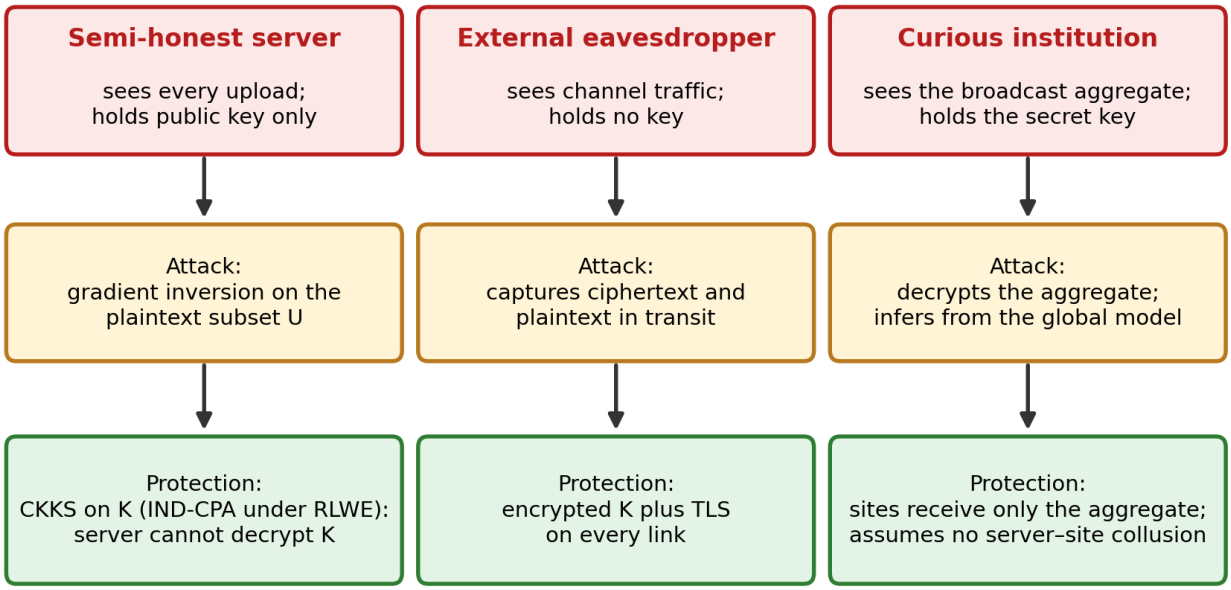


Figure 2. Threat model. Protection applies to the encrypted subset K; the plaintext subset U remains exposed to the server.

4.3 Heterogeneity-Aware Local Training

Each site starts from the global model w^t and runs τ local steps with the control-variate correction of SCAFFOLD [14]:

$$y_i \leftarrow y_i - \eta(g_i(y_i) - c_i + c), \tag{3}$$

where $i \in \{1,\ldots,N\}$ is the site index, $y_i \in \mathbb{R}^M$ is the local model of site i (initialised to w^t at the start of the round), τ is the number of local steps, $g_i(y_i)$ is a mini-batch stochastic gradient of the local loss F_i at y_i, η is the learning rate, c_i and c are the local and global control variates of Section 2.2, and $\leftarrow$ denotes assignment. The site then forms $\Delta y_i = y_i - w^t$ and updates its control variate:

$$c_i^+ = c_i - c + (w_t - y_i)/(\tau\eta), \tag{4}$$

with $\Delta c_i = c_i^+ - c_i$. Here c_i^+ is the updated local control variate of site i, c_i its value before the update, $\Delta y_i = y_i - w^t$ is the model update, and $(w^t - y_i)/(\tau\eta)$ is the average gradient correction over the τ local steps. Because Δc_i is a function of Δy_i, it leaks the same information and is protected by the same mask as Δy_i.

4.4 Sensitivity Scoring and Mask Agreement

Following [10], and using the sensitivity notion studied in [21], site i scores every parameter w_m on q local samples (ξ_k, v_k) with loss ℓ_k:

$$S_i(m) = (1)/(q)\Sigma_{k=1}^{q} |(\partial)/(\partial v_k)((\partial \ell_k)/(\partial w_m))|. \tag{5}$$

In Eq. (5), $m \in \{1,\ldots,M\}$ indexes the model parameters, $k \in \{1,\ldots,q\}$ indexes the local samples used for scoring, q is the number of such samples, ξ_k is the input feature vector and $v_k \in \{0,1\}$ the label of sample k, ℓ_k is the loss of sample k, and $S_i(m) \geq 0$ is the sensitivity of parameter w_m at site i, that is, the average magnitude with which the label v_k changes the gradient of w_m. A large value means that the gradient of w_m depends strongly on the label.

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

For a network with logit z_k and sigmoid/binary cross-entropy loss, $\partial \ell_k/\partial w_m = (\sigma(z_k) - v_k)\, \partial z_k/\partial w_m$ is linear in v_k, so the derivative in Eq. (5) equals $-\partial z_k/\partial w_m$ and

$$S_i(m)=(1)/(q)\Sigma_{k=1}{}^{q}|(\partial z_k)/(\partial w_m)|. \quad (6)$$

For a fully connected ReLU layer this is evaluated for all its weights at once as $(|\mathbf{A}|^T|\mathbf{\Delta}|)/q$, where $\mathbf{A}$ holds the layer's input activations and $\mathbf{\Delta}$ the back-propagated logit derivatives, since $|a_{ki}\delta_{kj}| = |a_{ki}||\delta_{kj}|$. We checked numerically that this equals the finite label difference of the per-sample gradient. Here $|\cdot|$ denotes the element-wise absolute value, $(\cdot)^T$ the matrix transpose, A has one row per sample and one column per layer input, and the bold Δ in this paragraph denotes the back-propagated logit derivatives; it is unrelated to the updates Δy_i and Δc_i.

Sensitivity maps are encrypted by each site and added by the server; the key holder decrypts the sum, forms the mean map, and selects the encrypted index set:

$$\bar{S}(m)=(1)/(N)\Sigma_{i=1}{}^{N}S_i(m), \quad K=\mathrm{TopP}(\bar{S},p). \quad (7)$$

In Eq. (7), $\bar{S}(m)$ is the mean sensitivity of parameter m over the N sites; $\mathrm{TopP}(\bar{S}, p)$ returns, in every layer l, the indices of the $\lceil p\, n_l \rceil$ largest entries of $\bar{S}$ (n_l is the number of parameters of layer l and $p \in (0, 1]$ the encrypted fraction); K is the resulting encrypted index set, and its complement U is the plaintext set.

Ranking is done within each layer, so that $\lceil p\, n_l \rceil$ parameters are chosen from layer l of size n_l and every layer is represented; we fixed this scope in advance to mirror layer-wise packing [18]. The remaining indices form the plaintext set U, and $|K| \approx pM$ for a model with M parameters. The mask is computed once at the initial model, as in [10].

4.5 Selective Encrypted Aggregation

Site i concatenates its selected entries into $\delta_i = [\Delta y_{i,K}; \Delta c_{i,K}]$, packs them into 4096-slot vectors, and encrypts them with the public key. The server multiplies each ciphertext by a plaintext weight vector ω_i (α_i on the model entries and $1/N$ on the control-variate entries) and adds:

$$C_K=\Sigma_{i=1}{}^{N}\omega_i \odot \mathrm{Enc}(\delta_i,K). \quad (8)$$

In Eq. (8), $\delta_i \in \mathbb{R}^{2|K|}$ is the packed vector of selected entries of site i (a data vector, unrelated to the back-propagated error δ of Section 4.4); $\mathrm{Enc}(\cdot)$ is CKKS encryption under the public key; $\omega_i \in \mathbb{R}^{2|K|}$ is a plaintext weight vector (α_i on the model-update entries and $1/N$ on the control-variate entries); $\odot$ is the element-wise (Hadamard) product of a plaintext vector with a ciphertext, applied slot by slot, which in CKKS consumes one multiplicative level and is followed by a rescaling; Σ is homomorphic ciphertext addition; and C_K is the aggregated ciphertext, which only holders of the secret key can decrypt. It is the weighted average of the encrypted entries of all sites.

Plaintext coordinates in U are averaged in the clear. Sites decrypt C_K, merge it with the plaintext aggregate into the averaged updates $\Delta\bar{y}$ and $\Delta\bar{c}$, and set $w^{t+1} = w^t + \Delta\bar{y}$ and $c^{t+1} = c^t + \Delta\bar{c}$. Figure 3 illustrates this flow. Only addition and one plaintext multiplication are needed, so no bootstrapping is used. The number of ciphertexts per site per round is

$$n_{ct}=\lceil (2|K|)/(n_s) \rceil. \quad (9)$$

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

In Eq. (9), n_{ct} is the number of ciphertexts per site per round, $n_s = 4096$ is the number of slots per ciphertext, $\lceil \cdot \rceil$ rounds up to the next integer, $|K|$ is the number of encrypted coordinates, and the factor 2 counts the two vectors $\Delta y_{i,K}$ and $\Delta c_{i,K}$. In the preceding paragraph, $\Delta\bar{y}$ and $\Delta\bar{c}$ are the averaged model and control-variate updates, each obtained by merging the decrypted C_K with the plaintext average over U, and w^{t+1} and c^{t+1} are the global model and global control variate after round t. For example, at $p = 10$ %, $|K| = 4{,}098$, so $n_{ct} = \lceil 8{,}196/4{,}096 \rceil = 3$ (Table 6).

Figure 3 shows the aggregation step by step. (1) The update of each site is split by the mask into the K part and the U part. (2) The K part, $[\Delta y_{i,K}; \Delta c_{i,K}]$, is packed into $\lceil 2|K|/n_s \rceil$ vectors of n_s slots (Eq. (9)) and encrypted with the public key. (3) The server multiplies every ciphertext by ω_i and adds the results of the N sites (Eq. (8)). (4) The sum C_K and the plaintext average over U are broadcast. (5) Each site decrypts C_K and merges the two parts into $\Delta\bar{y}$ and $\Delta\bar{c}$.

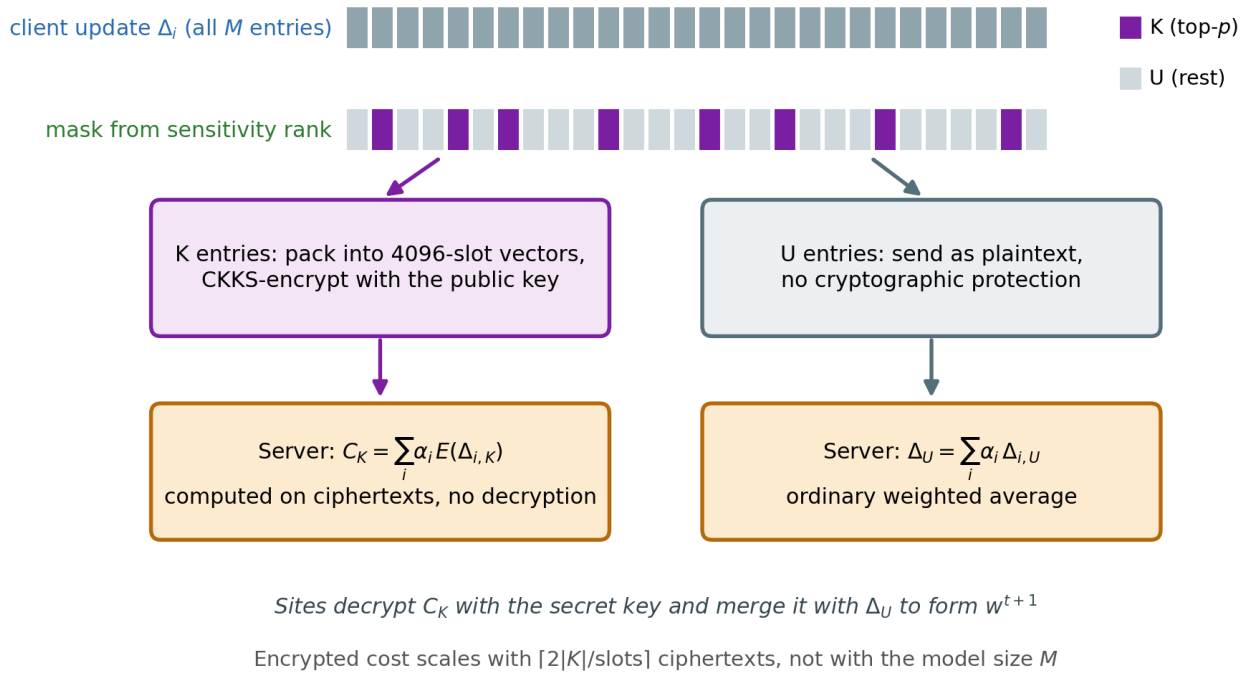


Figure 3. Selective encrypted aggregation. The update is split by the sensitivity mask; only the K part is encrypted and summed homomorphically.

4.6 Protocol

Figure 4 shows the one-time setup (key generation and mask agreement) and the per-round loop. Algorithm 1 gives the same steps in pseudocode.

The steps of Figure 4 are as follows. *Setup (once):* (a) key generation: the KMA creates (pk, sk) and distributes them (Algorithm 1, line 1); (b) sensitivity maps: every site computes S_i (Eq. (6)) at the initial model w^0, encrypts it, and uploads it (line 2); (c) mask agreement: the server adds the ciphertexts, the key holder decrypts the sum, forms $\bar{S}$ and $K = \mathrm{TopP}(\bar{S}, p)$ (Eq. (7)), and broadcasts the mask (line 3). *Training round $t = 0,\ldots,T-1$:* (d) local SCAFFOLD training with Eqs. (3) and (4) (lines 4–5); (e) encrypted upload of Δ_K and plaintext upload of Δ_U (line 6); (f) server aggregation by Eq. (8) and broadcast (line 7); (g) decryption and update of w^{t+1} and c^{t+1} (line 8). Table 3 maps every line of Algorithm 1 to the figures, equations, and sections of this paper.

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

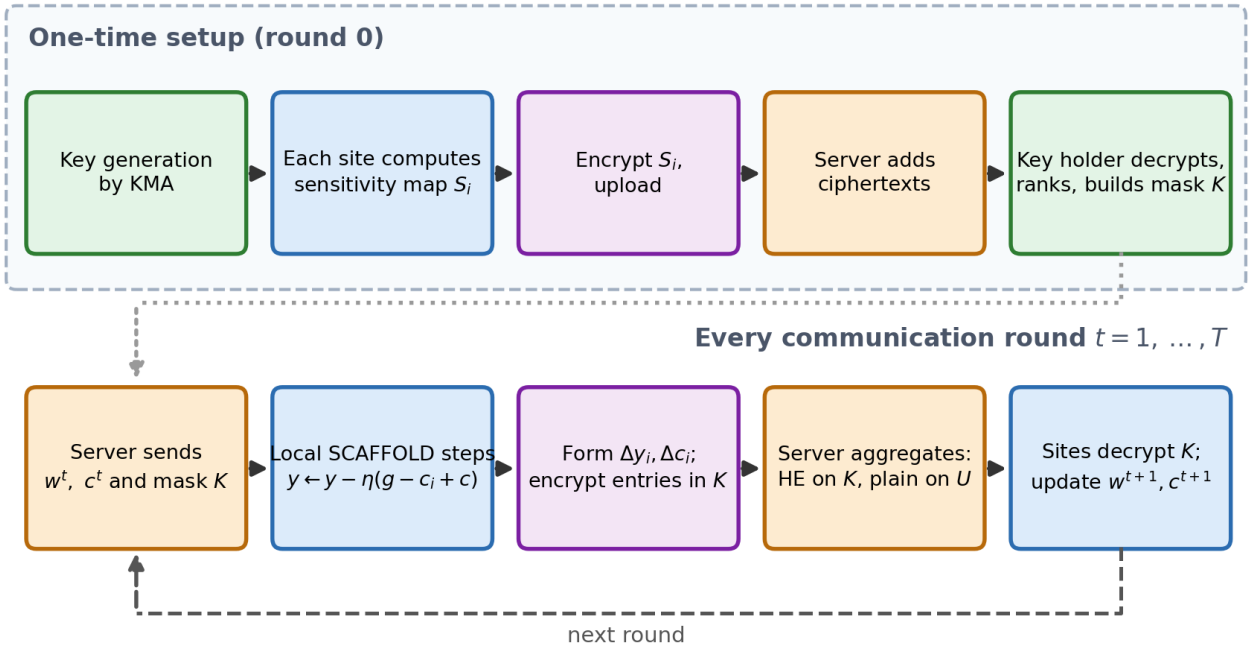


Figure 4. Workflow of SASHE-FL: one-time mask agreement followed by repeated training rounds.

Algorithm 1. SASHE-FL

Input: sites 1..N, encrypted fraction p, rounds T, local steps τ, rate η

Setup

1. KMA generates CKKS keys; public key → server, secret key → sites; the KMA then discards its copy
2. Site i computes S_i (Eq. 6) at w^0, encrypts and uploads it
3. Server adds the ciphertexts; key holder decrypts, forms $\bar{S}$, selects K (layer-wise top-p), broadcasts the mask

For $t = 0..T-1$:

4. Site i: $y_i \leftarrow w^t$; τ steps of Eq. (3)
5. Compute Δy_i, c_i^+ (Eq. 4), Δc_i
6. Upload Enc($\Delta y_{i,K}$, $\Delta c_{i,K}$) and plaintext ($\Delta y_{i,U}$, $\Delta c_{i,U}$)
7. Server: C_K by Eq. (8); plain weighted average on U; broadcast both
8. Sites decrypt C_K; update w^{t+1}, c^{t+1}

Table 3. Mapping of Algorithm 1 to the Paper

Line	Step of Algorithm 1	Figure / equation	Sec.
1	KMA generates CKKS keys; distributes pk and sk	Figure 1 (bottom bar); Figure 4 (setup)	4.1, 4.7
2	Sites compute and encrypt sensitivity maps S_i	Figure 4 (setup); Eqs. (5), (6)	4.4
3	Server adds maps; key holder decrypts, selects K, broadcasts mask	Figure 4 (setup); Eq. (7)	4.4
4	Local SCAFFOLD steps from w^t	Figure 1 (institution boxes); Eq. (3)	4.3
5	Compute Δy_i, c_i^+, Δc_i	Eq. (4)	4.3
6	Encrypt the K part, upload with the plaintext U part	Figure 1 (solid arrows); Figure 3; Eq. (9)	4.5
7	Server: ciphertext sum on K, plain average on U, broadcast	Figure 1 (server); Figure 3; Eq. (8)	4.5
8	Sites decrypt C_K; update w^{t+1}, c^{t+1}	Figure 1 (dashed arrows); Figure 3 (step 5)	4.5

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

Line	Step of Algorithm 1	Figure / equation	Sec.
–	Trust assumptions behind all lines	Figure 2 (threat model)	4.2, 7

4.7 Key Management and Cost Model

KMA engages in a trusted setup. It shouldn't be on the aggregation server, and any site with the secret key can decrypt what it sees. The KMA is a single point of trust: it holds the full secret key at generation time and must delete it after distribution. Per site and round, the encrypted payload is n_{ct} ciphertexts (Eq. 9) plus $8|U|$ bytes of plaintext (two float32 vectors), so cost falls in steps as p decreases rather than exactly in proportion to p. The single KMA and the shared secret key are the main trust weakness of this design (Section 4.2). Multiparty or threshold CKKS [20] would remove it: the sites run a distributed key generation so that each holds only a share of the secret key, the server encrypts and aggregates under a collective public key exactly as before, and decryption of the aggregate requires partial-decryption shares from a quorum of sites. No party, including the KMA, would ever hold the full key, so the server together with fewer than the threshold number of sites could not decrypt any individual update (Section 7). The prototype uses a single key; threshold key generation is future work.

5. PROTOTYPE AND EXPERIMENTAL SETUP

We implemented SASHE-FL in Python (NumPy for the network, TenSEAL [2] for CKKS) and ran all sites in one process. The experiments below were executed for this paper; every number in Section 6 comes from these runs. The code is publicly available athttps://github.com/dhwani16jain/SASHE. Table 4 lists the configuration.

Data. We use the Wisconsin diagnostic breast-cancer dataset [27] (569 samples, 30 numeric features) as distributed with scikit-learn [23], with a stratified 75/25 train/test split (426/143 samples) and standardization fitted on the training part. The positive class for the F1 score (the harmonic mean of precision and recall) is benign (label 1 in that copy). AUC denotes the area under the receiver operating characteristic (ROC) curve. This tabular benchmark is a stand-in for the imaging datasets used in [18]; it is small and nearly separable, which limits what it can show (Section 8). The experiments are a proof of concept for the protocol and its cost model, not a clinical validation.

Non-IID partition. Training samples are split over $N = 5$ simulated institutions with a class-wise Dirichlet(0.5) (concentration parameter $\beta = 0.5$, named to avoid confusion with the weights α_i) allocation, giving unequal sizes (27–193 samples across seeds) and skewed benign fractions (from 0.01 to 0.99 at individual sites).

Model and training. A multilayer perceptron (MLP) with layers 30–256–128–1 (rectified linear unit (ReLU) hidden activations, sigmoid output, $M = 40{,}961$ parameters) is trained with plain stochastic gradient descent (SGD). Each round uses $\tau = 10$ local steps, batch size 32, and $\eta = 0.05$, for $T = 30$ rounds. A centralized reference uses the same total number of SGD steps on pooled data.

Encryption. CKKS with polynomial degree 8192, coefficient moduli of 60/40/40/60 bits (200 bits in total, within the 128-bit-security bound of the HE standard tables), scale 2^{40}, and 4096 slots. The server context holds no secret key. Payload sizes are measured from TenSEAL's serialized ciphertexts plus 4 bytes per plaintext value.

Baselines. Plain FedAvg and plain SCAFFOLD; full-parameter HE ($p = 100\%$) with the same protocol; and FedAvg with client-update clipping ($C = 1$) plus Gaussian noise on the aggregate (noise multipliers $z = 1$ and 4) as a DP-style baseline, for which we did not compute (ε, δ). MPC was not implemented; MPC statements in this paper come from the literature.

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

Privacy probe. At the final SCAFFOLD model, for 60 random training samples per seed, we compute the single-sample gradient. This is a deliberately worst-case setting: the protocol itself transmits multi-step updates computed over mini-batches, not single-sample gradients. The first-layer gradient of one sample is rank one, $\partial \ell / \partial \mathbf{W}_1 = \xi \delta^T$, with ξ the input and δ the back-propagated hidden-layer error. An attacker who sees only plaintext entries (those outside K) fits a rank-one model to the visible entries by alternating least squares and takes the fitted vector as the input estimate. We report the absolute cosine similarity to ξ and the fraction of squared gradient norm lying in plaintext coordinates. Masks compared: random, layer-wise sensitivity from the initial model (the protocol), layer-wise sensitivity refreshed at the final model, global-rank sensitivity, and a first-layer guard that encrypts all of $\mathbf{W}_1$ and $\mathbf{b}_1$ plus the layer-wise top-p elsewhere. We added the guard after seeing the attack succeed against the other masks. Its success against this same attack is therefore expected by construction and is not independent evidence of robustness.

Table 4. Experimental Configuration

Item	Value
Data	Wisconsin breast cancer, 569 × 30; 426 train / 143 test
Sites	N = 5, Dirichlet(0.5) label skew
Model	MLP 30–256–128–1, M = 40,961
Training	SGD, η = 0.05, batch 32, τ = 10, T = 30
CKKS	TenSEAL; degree 8192; 60/40/40/60; scale 2^{40}; 4096 slots
Encrypted fraction	$p \in \{5, 10, 20, 30, 50, 100\}$ %, layer-wise
Seeds	5 (main results); 10 (SCAFFOLD study)
Hardware	1 vCPU Xeon 2.10 GHz, 3 GB RAM

6. RESULTS

6.1 Accuracy

Table 5. Accuracy and Per-Site Upload per Round (mean ± SD, 5 seeds)

Method	Acc. (%)	F1 (%)	AUC (%)	Upload (KB)
Centralized (reference)	96.9±1.3	97.6±1.0	99.3±0.4	–
FedAvg (plain)	96.6±1.5	97.4±1.2	99.3±0.4	160
SCAFFOLD (plain)	96.5±1.4	97.2±1.1	99.3±0.3	320
FedAvg + DP-style noise, z=1	94.3±2.2	95.4±1.8	98.3±1.3	160
FedAvg + DP-style noise, z=4	70.8±8.7	73.6±5.8	82.7±24.0	160
Full HE (p = 100 %)	96.5±1.4	97.2±1.1	99.3±0.3	6,796
SASHE-FL, p = 50 %	96.5±1.4	97.2±1.1	99.3±0.3	3,720
SASHE-FL, p = 30 %	96.5±1.4	97.2±1.1	99.3±0.3	2,489
SASHE-FL, p = 20 %	96.5±1.4	97.2±1.1	99.3±0.3	1,874
SASHE-FL, p = 10 %	96.5±1.4	97.2±1.1	99.3±0.3	1,259
SASHE-FL, p = 5 %	96.5±1.4	97.2±1.1	99.3±0.3	951

Table 5 shows that the federated methods reach the centralized reference within about one standard deviation (96.6 ± 1.5% for FedAvg and 96.5 ± 1.4% for SCAFFOLD against 96.9 ± 1.3%). Every SASHE-FL setting, including full HE, gave exactly the same test accuracy as plaintext SCAFFOLD in every seed; the largest absolute difference between the decrypted aggregate and

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

the plaintext aggregate was 2.3×10^{-8}–2.9×10^{-8} per run (mean over seeds), so CKKS approximation error did not affect the results. Accuracy therefore does not depend on p here: p changes what the server can see and what the protocol costs, not what the model learns.

The DP-style baseline lost accuracy as noise grew: 94.3 ± 2.2% at $z = 1$ and 70.8 ± 8.7% at $z = 4$, against 96.6 ± 1.5% without noise. The mean client update norm before clipping was 1.45, so clipping at $C = 1$ was active. Because we did not compute (ε, δ), this shows a utility trend rather than a privacy-matched comparison. With 143 test samples, differences below about 1.5 points should not be read as real.

6.2 Communication and Cryptographic Cost

Table 6. Measured Cost of Selective Encryption per Round (mean of 5 seeds; N = 5 sites)

p (%)	$\|K\|$	n_{ct}	Upload/ site (KB)	Download/site (KB)	Encrypt/ site (ms)	Server add (ms)	Decrypt/site (ms)	Total HE (ms)	Upload vs. p=100 % (−%)	Time vs. p=100 % (−%)
100	40,961	21	6,796	4,821	110.2	200.1	21.4	331.7	0.0	0.0
50	20,481	11	3,720	2,685	58.6	104.1	11.1	173.8	45.3	47.6
30	12,291	7	2,489	1,831	37.5	65.9	7.0	110.5	63.4	66.7
20	8,195	5	1,874	1,404	25.9	45.7	4.8	76.4	72.4	77.0

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

p (%)	$\|K\|$	n_{ct}	Upload/ site (KB)	Download/site (KB)	Encrypt/ site (ms)	Server add (ms)	Decrypt/site (ms)	Total HE (ms)	Upload vs. p=100 % (−%)	Time vs. p=100 % (−%)
10	4,098	3	1,259	977	15.3	27.3	2.8	45.4	81.5	86.3
5	2,051	2	951	763	10.2	18.4	1.8	30.4	86.0	90.8

Table 6 and Figure 5 give the measured cost. Full HE uploads 6,796 KB per site per round, 21.2× the 320 KB of plaintext SCAFFOLD. Encrypting 10% of the parameters reduces upload to 1,259 KB (81.5% less than full HE, 3.9× plaintext) and cryptographic time from 331.7 ms to 45.4 ms (86.3% less). At p = 5% the reductions are 86.0% and 90.8%.

Cost follows the ciphertext count n_{ct} of Eq. (9), not p itself: 21, 11, 7, 5, 3, and 2 ciphertexts for the six settings. At p = 5% the two ciphertexts carry 2,051 × 2 = 4,102 values in 8,192 slots, so about half the slots are empty. More generally, each setting exceeds a ciphertext boundary by only a few values (for example, $2|K|$ = 8,196 at p = 10% against 2 × 4,096 = 8,192 slots), so by Eq. (9) trimming $|K|$ by two coordinates would reduce n_{ct} from 3 to 2, roughly a quarter of the upload. A larger-degree ring might also lower cost; we did not test either option. The server-side addition step is the largest single component at high p because it multiplies each ciphertext by a plaintext weight vector for each of the five sites.

Figure 5 summarises these results in three panels. Panel (a) plots test accuracy against communication rounds: the curves of plaintext SCAFFOLD, full HE, and every SASHE-FL setting coincide, so encryption does not change what the model learns. Panel (b) plots the upload per site and round against the encrypted fraction p and follows the step pattern of Eq. (9). Panel (c) plots the cryptographic compute time per round against p; the server-side addition is its largest component at high p.

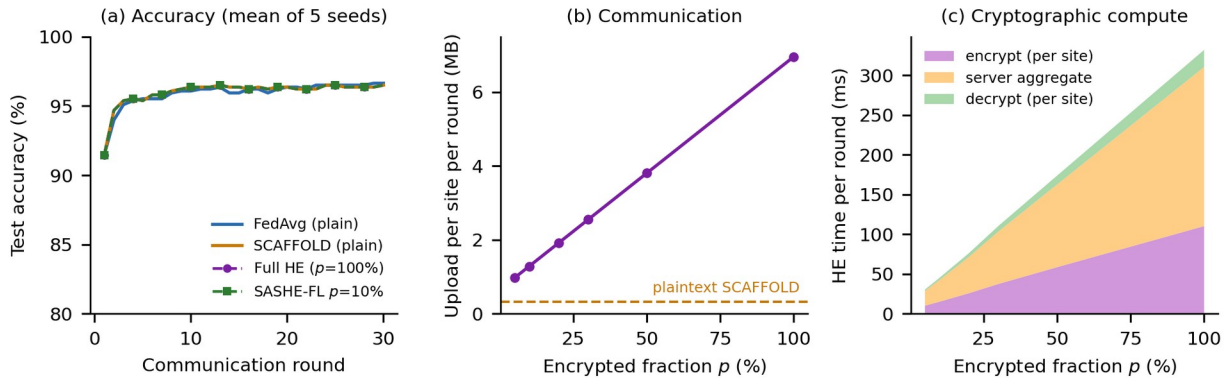


Figure 5. (a) Test accuracy per round (mean of 5 seeds); the HE curves coincide with plaintext SCAFFOLD. (b) Upload per site per round versus encrypted fraction. (c) Cryptographic compute per round versus encrypted fraction.

6.3 One-Time Mask Agreement

Setup adds a one-time cost. Each site encrypts its sensitivity map in 11 ciphertexts (3.65 MB, 56 ms), the server adds them in 5.3 ms, and the key holder decrypts the sum in 15.8 ms, with a maximum CKKS error of 3.9×10^{-9}. The upload is about half of one full-HE training round. It is paid once and is then amortized over all rounds.

6.4 Privacy Probe

The probe below is a worst-case analysis. It uses single-sample gradients of the final model, not the multi-step mini-batch updates (τ = 10 local steps, batch size 32) that the protocol actually transmits, so its results bound exposure from above rather than measure it. Figure 6(a) shows how much squared gradient norm remains in plaintext. Random masks leave 0.949, 0.907, and 0.702 of it at

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

p = 5%, 10%, and 30%, essentially 1 − p. Layer-wise sensitivity masks leave 0.731, 0.593, and 0.290; refreshing the map at the final model lowers this to 0.619, 0.459, and 0.161, and global ranking gives 0.492, 0.393, and 0.198. Sensitivity ranking thus concentrates encryption on the entries that carry most of the gradient.

That did not stop the attack. Figure 6(b): with no protection the rank-one attack recovers the input direction with cosine similarity 0.999, and the same value (0.999) is reached when the top 10% or 30% of parameters are encrypted by sensitivity, and with random masks. The first-layer gradient is rank one, so thousands of visible entries are more than enough to recover ξ. Only the first-layer guard broke this attack (cosine 0.000), because no first-layer entries remained visible. The guard encrypts 23.4% (base p = 5%) to 27.4% (base p = 10%) of all parameters, costing 1,863–2,174 KB upload and 75–89 ms per round at unchanged accuracy (96.5%), which is close to the p = 20% row of Table 6 at base p = 5% and lies between the p = 20% and p = 30% rows of Table 6 at base p = 10%.

Why the first-layer guard helps. For one sample, the first-layer weight gradient is the outer product $\partial \ell / \partial W_1 = \xi \delta^T$, so every visible entry is a scaled copy of the input ξ and even a modest visible subset determines ξ up to scale. The bias gradient $\partial \ell / \partial b_1$ equals δ, so a plaintext b_1 would reveal δ directly and make the recovery of ξ immediate. Encrypting all of W_1 and b_1 removes every term in which ξ appears directly. Deeper layers depend on ξ only through hidden activations and are harder to invert, although we did not test this.

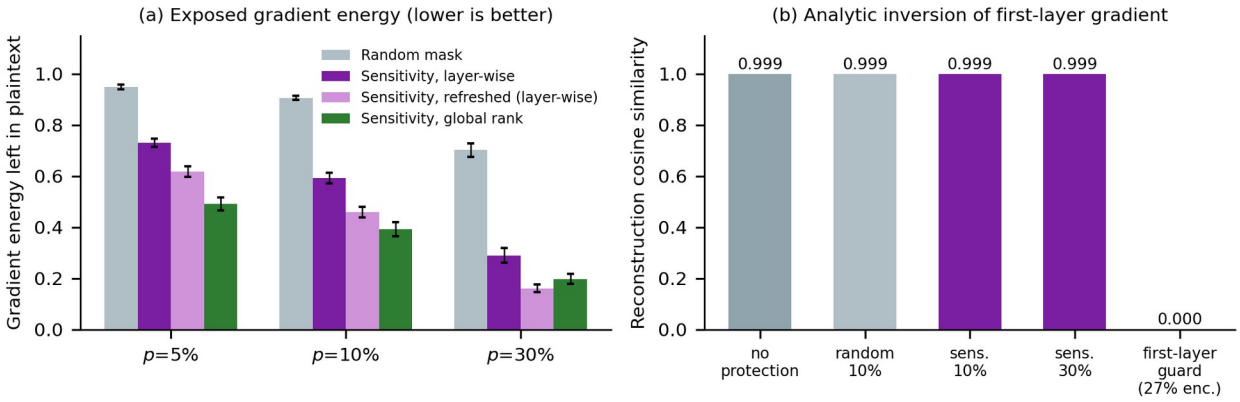


Figure 6. Privacy probe on the final model (60 samples × 5 seeds). (a) Fraction of squared gradient norm left in plaintext for each mask strategy. (b) Cosine similarity of a rank-one reconstruction of the input from visible first-layer gradient entries.

This probe has clear limits. It attacks a single-sample gradient of the final model, whereas the protocol transmits multi-step updates over batches of 32, which are much harder to invert, so the result is a worst-case, theoretical upper bound on exposure rather than a direct reconstruction attack on the multi-step mini-batch training process or a risk estimate. It uses one attack on one layer; the guard defeats it by construction and is not shown to resist optimization-based attacks [32] or attacks that use deeper layers.

6.5 Does SCAFFOLD Help Here?

On ten seeds with Dirichlet β = 0.5, FedAvg reached 97.6 ± 1.6% and SCAFFOLD 97.1 ± 1.3% (mean paired difference −0.5 points; SCAFFOLD better/tied/worse in 0/5/5 seeds). These ten-seed means differ from Table 5, which uses five seeds. With stronger skew (β = 0.1) the figures were 97.4 ± 1.2% and 96.9 ± 1.2% (difference −0.5 points; 2/2/6). Both methods needed about

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

one round to pass 90% and about 3.8 (FedAvg) and 3.4 (SCAFFOLD) rounds, to pass 95% in the main runs. SCAFFOLD's slight early speed-up (about 0.4 rounds) did not translate into higher final accuracy, so we found no meaningful benefit from the control-variate correction on this task, and it doubles the plaintext payload (160 KB versus 320 KB). The dataset is nearly separable and the network is over-sized, so this says little about harder imaging tasks; the heterogeneity-resilient component should be treated as optional until shown to help on such data. We nevertheless keep it in the protocol because it is aimed at stronger heterogeneity than this dataset provides, and because its control-variate update shares the same mask and the same ciphertexts as the model update, so it adds payload but no new protocol step.

6.6 Comparison with Published Results

The reduction figures reported in [18] (94.85% and 93.48% from packing and fewer rounds) and [10] (80.67% latency, 57.19% memory) come from different tasks, models, baselines, and measured quantities, so they are not directly comparable with the 81.5% upload and 86.3% time reductions measured here for $p = 10\%$. The direction agrees: cost falls as the encrypted set shrinks, and accuracy in our runs was unaffected by CKKS. Our probe adds a caution that the cited works do not address.

Table 7 places the selective-encryption results next to published ones. To make them comparable at a glance, every reported reduction r is also given as a factor $1/(1 - r)$. The closest reference is FedML-HE [12], which also selects the most sensitive parameters for encryption. It reports about a tenfold reduction in the overhead of HE-based training of ResNet-50 (and about forty-fold for BERT) [12]; SASHE-FL at $p = 10$ % gives 5.4× less upload (−81.5 %) and 7.3× less cryptographic time (−86.3 %) than full HE on a 41k-parameter MLP. The two are of the same order of magnitude. The smaller factors here are expected: at $p = 100$ % there are only 21 ciphertexts, so by Eq. (9) at least 3 remain at $p = 10$ %, whereas the figures of [12] are for models with millions of parameters, where ciphertext granularity matters little (Section 6.2). Jeong and Choi [10] report 5.2× lower latency and 2.3× lower memory in the sensitivity-map stages only; our end-to-end figures are in the same range.

SASHE-FL adds end-to-end evidence that these works do not report together. Reference [10] reports no training accuracy or leakage, and [12] reports overhead rather than the leakage of the plaintext subset. SASHE-FL reports accuracy (identical to plaintext SCAFFOLD), communication, cryptographic time, and a leakage probe for the same protocol. Against [18], the $p = 100$ % row of Table 5 plays the role of the "encrypt everything" baseline, so the 81.5 % and 86.3 % reductions are like-for-like only with respect to our own full-HE run; the comparison with the figures of [18] itself remains qualitative because the task, the model, and the measured quantity differ.

Table 7. Comparison with Published Selective-Encryption Results

Work	Encrypted set	Reported reduction (factor)	Accuracy / leakage
FedML-HE [12]	top-sensitivity parameters	HE overhead ≈10× (ResNet-50), ≈40× (BERT)	overhead study; privacy set by selection
Jeong and Choi [10]	top-p of compressed map (e.g., 30 %)	latency −80.67 % (5.2×); memory −57.19 % (2.3×)	no training accuracy or leakage
Lin *et al.* [18]	all parameters, packed	communication −94.85 % / −93.48 % (19.4× / 15.3×) vs. unpacked HE	F1 +2.2 / +4.0 points
SASHE-FL, $p = 10$ %	layer-wise top 10 %	upload −81.5 % (5.4×); HE time −86.3 % (7.3×) vs. full HE	accuracy unchanged (96.5 %); inversion still succeeds
SASHE-FL, $p = 5$ %	layer-wise top 5 %	upload −86.0 % (7.1×); HE time −90.8 % (10.9×) vs. full HE	accuracy unchanged (96.5 %); inversion still succeeds

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

7. SECURITY AND PRIVACY ANALYSIS

Encrypted subset. Under CKKS, the ciphertexts of the coordinates in K are semantically secure against an adversary not holding the secret key assuming the hardness of Ring Learning With Errors (RLWE) [4]. The server only has the public key, and therefore learns nothing about the encrypted coordinates of individual updates other than their size (it still sees the mask K and every plaintext coordinate in U). The guarantee is for the server and for eavesdroppers, but not for a site, since the secret key is with each site. This is an IND-CPA (indistinguishability under chosen-plaintext attack) guarantee; CKKS is known to be weaker when decrypted results are exposed to an adversary (IND-CPA-D, the variant in which the adversary also sees decrypted results), so in our protocol decrypted aggregates are released only to the sites, and noise flooding would be needed if they were ever published.

Plaintext subset. Coordinates in U are exposed exactly as they would be in unprotected federated learning (FL). Section 6.4 shows that a sensitivity-ranked K covering 10-30% of coordinates leaves most first-layer entries visible, which suffices for a rank-one attack (although only 59% and 29% of the squared gradient norm remains in plaintext at 10% and 30%). The privacy claim of SASHE-FL is therefore conditional on the choice of K, and a fixed fraction p does not function as a privacy parameter in the way a differential privacy (DP) budget does. Deployments should include structural safeguards, such as encrypting the first layer, and should be validated against stronger attacks. The security claim of SASHE-FL is accordingly limited to the encrypted subset, against a semi-honest server that does not collude with any site; it does not extend to the plaintext subset, to the released global model, or to colluding parties.

Key custody and collusion. Because sites share one secret key, a site that receives another site's ciphertext can decrypt it, and a server that colludes with any site can too. This is the shared-key weakness noted for related schemes in [9]. In our protocol the server forwards only the aggregate, but collusion is assumed away rather than prevented: a server colluding with any one site, or with the KMA, can decrypt every individual encrypted update. Threshold or multiparty CKKS [20] would replace the single key with key shares produced by distributed key generation, so that decryption requires partial-decryption shares from a quorum of sites and no party, including the KMA, ever holds the full key. The server plus fewer than the threshold number of sites could then no longer decrypt an individual update, which removes the single-key risk and the KMA as a single point of trust. Two conditions are needed for this to hold. First, sites must produce decryption shares only for the aggregate ciphertext C_K, never for an individual upload. Second, even then the aggregate is visible to the sites, so a server colluding with $N - 1$ sites could infer the remaining site's update by subtraction; threshold CKKS does not hide a site's update from such a coalition. The costs are an extra setup round, an extra communication step per round for decryption shares, noise flooding on those shares, and, for N-of-N schemes, sensitivity to site dropout (a t-of-N threshold tolerates dropout down to t sites). The same shares would also protect the one-time decryption of the mean sensitivity map. Our TenSEAL prototype does not support this and it was neither implemented nor tested here; libraries such as OpenFHE or Lattigo do.

Sensitivity maps. Per-site maps are sent encrypted and only their mean is decrypted, so a curious site learns the global ranking, and everyone learns the mask. We have not analyzed what the mean map reveals about the pooled data.

Membership inference and Byzantine sites. Encrypting updates does not limit what the released global model reveals, so membership inference [25] remains; DP noise on the plaintext subset is one option. Sites submitting corrupted updates are not detected; committee scoring [26] could be layered on top, but ciphertext scoring would need extra design and was not tested.

Milky Way Scientific Journal (MWSJ)
Doi: https://doi.org/10.65021/mwsj.v2.i3.68
ISSN: 3069-6399
www.mwsJournal.com

8. LIMITATIONS AND FUTURE WORK

The evidence is limited in several ways. (i) The benchmark is a small, nearly separable tabular dataset with 143 test samples, and the model is a 41k-parameter MLP, so small differences are noise and accuracy cannot discriminate between methods; the evaluation is a proof of concept, not a clinical validation. It is also not the imaging setting that motivates the work. Realistic imaging models are far larger: as a back-of-envelope estimate, an 11-million-parameter ResNet-18 would need about 5,500 ciphertexts per site per round, roughly 1.7 GB if each ciphertext is the 323 KB measured here. In convolutional layers the first-layer gradient is a sum over spatial patches rather than a rank-one outer product, so both the probe and the first-layer guard must be re-derived and re-validated on larger imaging datasets and deeper models. (ii) Sites are simulated in one process on one CPU; communication is measured in bytes, not latency, and no dropout or stragglers are modeled. (iii) The privacy probe is a single analytic attack on single-sample gradients, which is a worst-case upper bound rather than a reconstruction of the multi-step mini-batch updates; it does not establish that any mask is safe. (iv) There is no formal privacy analysis of the plaintext subset. (v) SCAFFOLD showed no benefit, so the heterogeneity-resilience claim is untested on suitable data. (vi) The DP baseline lacks privacy accounting, and no MPC baseline was run. (vii) The mask is computed once at initialization; we did not study how quickly it becomes stale, although refreshing lowered exposed energy. (viii) Hyperparameters were not tuned, and the first-layer guard was designed after seeing the attack results. (ix) The KMA and shared-key assumptions require operational controls that we did not evaluate. Collusion between the server and any site, or compromise of the KMA, breaks the protection of the encrypted subset; threshold or multiparty CKKS [20] could remove this single-key risk (Section 7) but was not implemented in the prototype.

Future work should: evaluate SASHE-FL on imaging benchmarks such as those in [18] (or, as a cheaper first step, MedMNIST subsets with a small CNN) and on deeper models; test optimization-based inversion [6,32] against the actual multi-step updates while varying batch size and local steps; extend the leakage sweep to much higher encrypted fractions (50–99%) for random and sensitivity masks; add threshold or multiparty CKKS [20] to remove the single-key and KMA trust assumptions, and measure its extra setup and per-round decryption-share cost; combine DP noise on the plaintext subset; tune $|K|$ to ciphertext boundaries; study periodic mask refresh; and measure end-to-end latency over real networks.

9. CONCLUSION

We presented SASHE-FL, a selective-encryption protocol for cross-silo healthcare FL, and measured a real CKKS prototype. On a non-IID five-site breast-cancer task, encrypting a sensitivity-ranked 10% of the parameters cut per-site upload by 81.5% and cryptographic time by 86.3% relative to encrypting all parameters, with accuracy identical to plaintext SCAFFOLD. Selective encryption does not by itself deliver protection: an analytic inversion attack still succeeded when 30% of parameters, chosen by sensitivity, were encrypted, and stopped only when the whole first layer was encrypted. The control-variate component gave no benefit on this task. These findings support selective encryption as a cost-saving tool only when combined with structural guards and stronger validation. The protection claimed applies only to the encrypted coordinates, against a semi-honest server that does not collude with any site, since all sites share one secret key; threshold or multiparty CKKS [20] is the natural way to remove this single-key risk, but it was not implemented here. The evidence is a proof of concept and should be re-tested on larger models and clinical imaging data before any deployment claim.

ETHICS STATEMENT

This study uses a publicly available, de-identified Wisconsin breast cancer dataset and does not involve direct human participation or identifiable personal data. Therefore, ethical approval was not required.

FUNDING

This research received no external funding.

DATA AVAILABILITY STATEMENT

The Wisconsin diagnostic breast cancer dataset used in this study is publicly available through the scikit-learn library [23] (sklearn.datasets.load_breast_cancer) and the UCI Machine Learning Repository. The complete code used to produce all results reported in this paper (SASHE-FL implementation, baselines, privacy probe, raw per-seed results, and figure scripts) is publicly available at https://github.com/dhwani16jain/SASHE.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Dhwani Jain: conceptualisation, methodology, software, validation, formal analysis, investigation, visualisation, and writing – original draft. Bharat Thathera: supervision, project administration, and writing – review and editing. Both authors read and approved the final manuscript.

DECLARATION ON THE USE OF GENERATIVE AI

The manuscript text, including the study design, analysis, and interpretation of results, was written by the authors. ChatGPT (OpenAI) and Claude (Anthropic) were not used to write, rewrite, or edit any text in the manuscript. Spelling and grammar were checked using Grammarly only. ChatGPT and Claude were used only for code debugging, and all code was verified by the authors. No generative AI tool was used to generate text, ideas, results, figures, or references. The authors take full responsibility for the content of the manuscript.

REFERENCES

1. Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)* (pp. 308–318). https://doi.org/10.1145/2976749.2978318
2. Benaissa, A., Retiat, B., Cebere, B., & Belfedhal, A. E. (2021). *TenSEAL: A library for encrypted tensor operations using homomorphic encryption* (arXiv:2104.03152). arXiv. https://doi.org/10.48550/arXiv.2104.03152
3. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., & Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)* (pp. 1175–1191). https://doi.org/10.1145/3133956.3133982
4. Cheon, J. H., Kim, A., Kim, M., & Song, Y. (2017). Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology – ASIACRYPT 2017* (pp. 409–437). Springer. https://doi.org/10.1007/978-3-319-70694-8_15
5. European Parliament and Council of the European Union. (2016). Regulation (EU) 2016/679 (General Data Protection Regulation). *Official Journal of the European Union*, *L119*, 1–88.
6. Hatamizadeh, A., Yin, H., Roth, H. R., Li, W., Kautz, J., Xu, D., & Molchanov, P. (2022). GradViT: Gradient inversion of vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 10021–10030). https://doi.org/10.1109/CVPR52688.2022.00978
7. Hsu, R.-H., & Huang, T.-Y. (2022). Private data preprocessing for privacy-preserving federated learning. In *Proceedings of the IEEE 5th International Conference on Knowledge Innovation and Invention (ICKII)*. https://doi.org/10.1109/ICKII55100.2022.9983518
8. Hsu, T.-M. H., Qi, H., & Brown, M. (2020). Federated visual classification with real-world data distribution. In *Computer Vision – ECCV 2020*. Springer. https://doi.org/10.1007/978-3-030-58607-2_5
9. Hu, C., Liu, C., Wang, S., & Zhang, T. (2023). Efficient privacy-preserving data aggregation for lightweight secure model training in federated learning. In *Proceedings of the 7th International Conference on Cryptography, Security and Privacy (CSP)*. https://doi.org/10.1109/CSP58884.2023.00026
10. Jeong, Y., & Choi, W.-S. (2025). Efficient privacy-preserving federated learning with sensitivity-based selective homomorphic encryption. In *Proceedings of the 22nd International SoC Design Conference (ISOCC)*. https://doi.org/10.1109/ISOCC66390.2025.11329642

11. Jiang, M., Zhong, Y., Le, A., Li, X., & Dou, Q. (2023). Client-level differential privacy via adaptive intermediary in federated medical imaging. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2023* (pp. 500–510). Springer. https://doi.org/10.1007/978-3-031-43895-0_47
12. Jin, W., Yao, Y., Han, S., Gu, J., Joe-Wong, C., Ravi, S., Avestimehr, S., & He, C. (2023). *FedML-HE: An efficient homomorphic-encryption-based privacy-preserving federated learning system* (arXiv:2303.10837). arXiv. https://doi.org/10.48550/arXiv.2303.10837
13. Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., El Rouayheb, S., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., . . . Zhao, S. (2021). Advances and open problems in federated learning. *Foundations and Trends in Machine Learning, 14*(1–2), 1–210. https://doi.org/10.1561/2200000083
14. Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., & Suresh, A. T. (2020). SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning (ICML)* (pp. 5132–5143).
15. Korkmaz, A., Alhonainy, A., & Rao, P. (2022). An evaluation of federated learning techniques for secure and privacy-preserving machine learning on medical datasets. In *Proceedings of the IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*. https://doi.org/10.1109/AIPR57179.2022.10092212
16. Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., & Smith, V. (2020). Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems (MLSys)* (Vol. 2, pp. 429–450).
17. Li, Z., He, S., Chaturvedi, P., Hoang, T.-H., Ryu, M., Huerta, E. A., Kindratenko, V., Fuhrman, J., Giger, M., Chard, R., Kim, K., & Madduri, R. (2023). APPFLx: Providing privacy-preserving cross-silo federated learning as a service. In *Proceedings of the IEEE 19th International Conference on e-Science*. https://doi.org/10.1109/E-SCIENCE58273.2023.10254842
18. Lin, J., Chen, J., Xiong, J., Jiao, D., Zhao, W., & Xiang, Y. (2025). A lightweight privacy-preserving federated learning framework for heterogeneity-resilient skin cancer diagnosis. *IEEE Journal of Biomedical and Health Informatics*. https://doi.org/10.1109/JBHI.2025.3578125
19. McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)* (pp. 1273–1282).
20. Mouchet, C., Troncoso-Pastoriza, J., Bossuat, J.-P., & Hubaux, J.-P. (2021). Multiparty homomorphic encryption from ring-learning-with-errors. *Proceedings on Privacy Enhancing Technologies, 2021*(4), 291–311. https://doi.org/10.2478/popets-2021-0071
21. Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., & Sohl-Dickstein, J. (2018). Sensitivity and generalization in neural networks: An empirical study. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*.
22. Pawar, A., Jain, S., Dhait, A., Nagbhidkar, A., & Narlawar, A. (2024). Federated learning for privacy preserving in healthcare data analysis. In *Proceedings of the International Conference on Artificial Intelligence and Quantum Computation-Based Sensor Application (ICAIQSA)*. https://doi.org/10.1109/ICAIQSA64000.2024.10882173
23. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830.
24. Rieke, N., Hancox, J., Li, W., Milletari, F., Roth, H., Albarqouni, S., Bakas, S., Galtier, M. N., Landman, B., Maier-Hein, K., Ourselin, S., Sheller, M., Summers, R. M., Trask, A., Xu, D., Baust, M., & Cardoso, M. J. (2020). The future of digital health with federated learning. *npj Digital Medicine, 3*, Article 119. https://doi.org/10.1038/s41746-020-00323-1
25. Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)* (pp. 3–18). https://doi.org/10.1109/SP.2017.41
26. Shu, Z., Zhao, H., Xu, B., Xun, W., & Xu, B. (2023). Privacy-preserving federated learning framework via blockchain and committee mechanism. In *Proceedings of the IEEE 23rd International Conference on Communication Technology (ICCT)*. https://doi.org/10.1109/ICCT59356.2023.10419785
27. Street, W. N., Wolberg, W. H., & Mangasarian, O. L. (1993). Nuclear feature extraction for breast tumor diagnosis. In *Proceedings of SPIE 1905, Biomedical Image Processing and Biomedical Visualization* (pp. 861–870). https://doi.org/10.1117/12.148698
28. Truhn, D., Arasteh, S. T., Saldanha, O. L., Mueller-Franzes, G., Khader, F., Quirke, P., West, N. P., Gray, R., Hutchins, G. G. A., James, J. A., Loughrey, M. B., Salto-Tellez, M., Brenner, H., Brobeil, A., Yuan, T., Chang-Claude, J., Hoffmeister, M., Foersch, S., Han, T., . . . Kather, J. N. (2024). Encrypted federated learning for secure decentralized collaboration in cancer image analysis. *Medical Image Analysis, 92*, Article 103059. https://doi.org/10.1016/j.media.2023.103059
29. U.S. Department of Health and Human Services. (2002). *Standards for privacy of individually identifiable health information* (45 C.F.R. Parts 160 and 164, as amended).
30. Wei, W., Liu, L., Loper, M., Chow, K.-H., Gursoy, M. E., Truex, S., & Wu, Y. (2020). A framework for evaluating client privacy leakages in federated learning. In *Computer Security – ESORICS 2020* (Vol. 12308, pp. 545–566). Springer. https://doi.org/10.1007/978-3-030-58951-6_27
31. Zhang, C., Li, S., Xia, J., Wang, W., Yan, F., & Liu, Y. (2020). BatchCrypt: Efficient homomorphic encryption for cross-silo federated learning. In *Proceedings of the USENIX Annual Technical Conference (ATC)* (pp. 493–506).
32. Zhu, L., Liu, Z., & Han, S. (2019). Deep leakage from gradients. In *Advances in Neural Information Processing Systems (NeurIPS)* (Vol. 32).