Training Policy for Privacy-Preserving Logistic Regression in Federated Learning Environments

Jorge M. Cortés-Mendoza
Cloud Competency Centre
National College of Ireland
Dublin, Ireland
JorgeMario.CortesMendoza@ncirl.ie,
ORCID: 0000-0001-7209-8324

Andrei Tchernykh
Computer Science Department, CICESE Research Center, Ensenada, Mexico
Institute for System Programming, RAS, Moscow, Russia,
chernykh@cicese.mx,
ORCID: 0000-0001-5029-5212

Horacio González-Vélez
Cloud Competency Centre
National College of Ireland
Dublin, Ireland
horacio@ncirl.ie,
ORCID: 0000-0003-0241-6053

***Abstract*— Logistic Regression (LR) is a widely used statistical model for classification problems. However, its training and evaluation in a shared environment increase the possibility of information leaking. A federated LR reduces security issues by using only locally available data for training. In a Federated Learning (FL) environment, LR receives the coefficients of local models to create the federated LR model, which is then distributed to update the local models. The exchange process does not leak confidential information when LR coefficients are encrypted. Homomorphic Encryption (HE) allows the merging of local LR models with privacy preservation (HE-LR). This work presents a novel training policy to reduce the training time with only slightly decreased quality in an FL environment with HE. We analyze the accuracy and time of FL policies with HE-LR that progressively reduce the amount of training data and exchange the LR coefficients in a privacy-preserving manner. The results show that the proposed policy can speed up the training time between 12% and 69%, compared to the traditional FL approach, with an average decrease in accuracy of 1.79% and 1.95%.**

Keywords*— *Federated Learning, Homomorphic Encryption, Logistic Regression, Privacy-Preserving, Training policy.

I. Introduction

Federated Learning (FL) and Homomorphic Encryption (HE) are two main directions to provide security and privacy preservation by addressing vulnerabilities in data processing. Both approaches pursue the processing of information securely and privately. Several Machine Learning (ML) approaches have been implemented to protect the information of the dataset using HE and FL, for instance, Logistic Regression (LR) and Artificial Neural Networks (NN), among other ML techniques [1].

LR is a common supervised ML approach widely applicable to binary classification problems (see Section III). A critical limitation of LR with HE is the polynomial approximation that defines the homomorphic versions of the logistic/sigmoid function [2]. FL environment eliminates this limitation because the FL approach does not compute the homomorphic version of the logistic/sigmoid function. FL is widely applied in real-world scenarios, not only for privacy-preserving but also to reduce training time.

In this paper, we present a new training policy for FL that progressively reduces the amount of training data for each iteration. This reduction allows us to perform the learning process faster, effectively reducing the training time without significant precision degradation.

Relevant experiments on six datasets from medicine (diabetes, cancer, drugs, etc.) and genomics show that our proposed method reduces the training time compared with recent state-of-the-art approaches while maintaining the model's accuracy.

Our main contributions are multifold. We

- present recent advances in privacy-preserving logistic regression with federated learning and homomorphic encryption;
- propose a policy to reduce the training time of logistic regression in a vertical federated learning environment with homomorphic encryption;
- analyze the accuracy and time of the proposed policy; and,
- show that the policy reduces the training time of the logistic regression in a federated learning environment.

The content of the paper is structured as follows. The next section introduces information about gradient descent, LR, FL, and HE approaches. Section III describes the latest advances in privacy-preserving LR. Section IV outlines the proposed training policy. Section V presents the configuration and performance evaluation of HE-LR with FL. Finally, Section VI summarizes the main contributions of our research.

II. Background

A. Logistic Regression and Gradient Descent

LR models the probability of a discrete outcome given an input variable. It is a classification method where the inference determines the category of a new sample based on the sigmoid function. LR is a standard technique for image recognition [3], [4], genomics [5], and disease detection [6]–[8], among others.

The inference of the LR considers the information of the instance $x = (1, x_1, x_2, \dots, x_d)$ with d features, the coefficients $\theta^T = (\theta_0, \theta_1, \dots, \theta_d)$ of the LR equation to estimate the probability of a category and the logistic/sigmoid function $g(z) = 1/+e^{(-z)}$. $h_\theta(x) = g(\theta^T x)$ produces a value in the interval (0, 1) with a real input of the linear combination $\theta^T x = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_d x_d$. A binary category is defined by a threshold $0 < \tau < 1$ and the function

$$y = \begin{cases} 1 & if \ h_\theta(x) \geq \tau \\ 0 & if \ h_\theta(x) < \tau \end{cases} \quad (1)$$

where τ is typically equal to 0.5.

The efficient inference of logistic regression LR depends on the likelihood function expressed in the parameter θ. The LR training phase finds θ^* that maximizes the correct classification of the elements in the dataset X according to their corresponding labels Y. Gradient Descent (GD) is the most used method to find θ^*, which minimizes the cost function error $J(\theta)$ defined as follows:

$$J(\theta) = -\frac{1}{N}\sum_{i=1}^{N} y^{(i)} \log\left(h_\theta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_\theta\left(x^{(i)}\right)\right) \quad (2)$$

where $x^{(i)} \in \mathbb{R}^d$ is the i-th instance of X and its corresponding label $y^{(i)} \in \{0,1\}$ in Y for $i = 1,2,\ldots,N$.

$J(\theta)$ expresses the efficiency of the model, the smaller the value, the more accurate the classification is. The optimization process to find θ^* consists of updating θ according to the opposite direction of the slope, considering the partial derivative of $J(\theta)$, defined by $\nabla_\theta J(\theta)$. The search for θ^* defines a learning rate α that establishes the length of a movement in the search space (step), the performance of GD depends on the α value [9]. Because θ^* minimizes $J(\theta)$, it can be used as a binary classifier for new data using (1).

In recent years, the main direction in LR has focused on protecting the information of the dataset used to train the model. The search for more efficient versions of privacy-preserving LR follows two main directions: FL and HE.

B. *Federated Learning*

As a distributed system with decentralized learning operations to ensure data privacy, the main purpose of FL is to eliminate the necessity of pooling the data into a single location [10]. It allows the training of ML models using the local data of distributed nodes. Each node in the system trains a model that is shared with a centralized server. A global model is created by aggregating locally trained models in the central server. Then, all the local nodes receive the global model to enhance their independent models, see Figure 1.

The iterative exchange of information generates collaborative learning that improves general and local models. FL can introduce encrypted parameters to exchange the models and avoid the leak of the models, in addition to limiting access to the raw data, which protects data at a low level.

The training process over the FL system consists of four steps [11]: The training of local models, the aggregation of local models on the central server, the sharing of the central model with local nodes, and the updating of the local model with the information of the central model.

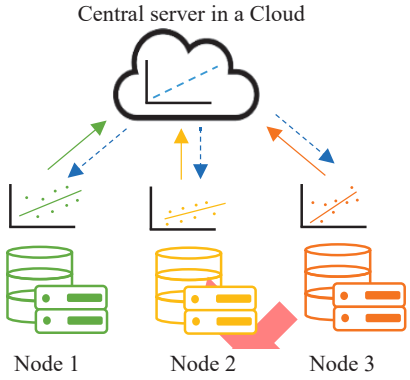

Fig. 1. Example of an FL system with a central node in a cloud environment and three nodes with their local data and ML models.

C. *Homomorphic Encryption*

HE produces ciphertexts in such a way that an untrusted party cannot know the content of the ciphertexts, but it can process them [12]. The three main categories of HE are Partially (PHE), Somewhat (SHE), and Fully Homomorphic Encryption (FHE) [2].

PHE supports only homomorphic addition or multiplication (but not both), SHE allows a limited number of both operations, and FHE enables an unlimited number of homomorphic addition and multiplication at the expense of significant overhead by introducing a sophisticated and compute-intensive component named bootstrapping [13].

A relevant limitation of the HE field is the number of homomorphic operations, which restricts its use to specific domains. Number comparison, absolute value, and determining the sign of a number, among others, are straightforward operations outside of the homomorphic space, but they are computationally expensive in the homomorphic domain because they must be approximated using polynomials.

Despite the current limitations, HE is an alternative to creating secure ML models, its limited applicability can be used in the prediction or classification of confidential information [1]. Figure 2 shows data protection in a secure cloud environment with HE.

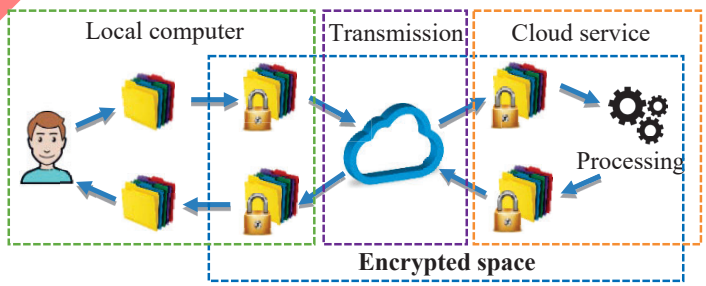

Fig. 2. An example of a Cloud environment with HE that protects the entire data lifecycle (transmission, storage, and processing).

TABLE I. MAIN CHARACTERISTICS OF FL AND HE APPROACHES FOR PRIVACY-PRESERVING LOGISTIC REGRESSION IN THE LITERATURE.

HE	FL	Name	Metric	Dataset	Ref
-	*	VFLR	Accuracy (A), Area under the ROC Curve (AUC)	Pima, BCWD, BDM	[16]
-	*	SecureLR	Time	MNIST	[17]
-	*	VANE	Mean Absolute Error (MAE)	BCD, Diabetes dataset (DD), UCID	[18]
-	*	VPPLR	Precision (P), Recall (R)	DD, WIBC, DD, ACAD	[19]
*	-	-	A	MNIST, notMNIST, CIFAR-10	[3]
*	-	-	AUC	iDASH (Genomic), financial	[20]
*	-	Modified GWAS	p-values, F1-score (F1)	iDASH	[21]
*	-	-	A, AUC, K-S values	Korea Credit Bureau (KCB), MNIST	[22]
*	-	-	A, AUC	iDASH, Lbw, Mi, Nhanes3, Pcs, Uis	[23]
*	-	N-LHAE	Overhead	Not described	[24]
*	-	P2OLR, P2VCLR, CECLLR	A, AUC, F1, P, R	Mi, Nhanes3, Uis	[25],[26],[27]
*	-	-	A	Digits (scikit-learn library)	[28]

III. RELATED WORK

Several studies have proposed innovations and new approaches to overcome the disadvantages of LR with HE and FL. This section presents the most relevant advances in the privacy-preserving LR field with HE and FL. Table I summarizes the main characteristics of the related work.

Yang et al. [14] compare outsourcing schemes with several secure computation methods, e.g., secure multi-party computation, pseudorandom functions, software guard extensions, and perturbation approaches. The authors describe the basis, evolution, and applicability of HE and the security threats and requirements of secure outsourcing computation.

Shaheen et al. [15] provide an overview of the FL technique and its applicability in different domains, where the systematic literature review of recent studies shows the wide adoption of FL. The authors describe the algorithms, models, and frameworks of FL and its scope of application in different domains.

A. Logistic Regression with Federated Learning

Zhao et al. [16] propose an efficient, privacy-preserving Vertical FL Framework for LR (VFLR). It uses participants' data to create a global high-quality LR model. VFLR provides secure training and queries over private information among participants. The results show the efficiency of VFLR in terms of accuracy, computational cost, and communication overhead.

He et al. [17] present a distributed Secure LR algorithm for vertical FL, which uses HE to avoid information leaks. Secure LR removes the need for a third-party coordinator and guarantees security at the expense of efficiency. The results demonstrate that the system's security for a two-party FL can be extended for different datasets and several participants.

Wang et al. [18] introduce a noninteractive privacy-preserving FL scheme for ML models with data protection. VANE uses cloud assistance over multiple private local data to train a global Linear Regression (LiR), Ridge Regression (RR), or LR models. The results show that VANE can securely aggregate local training data faster than existing schemes.

Zhang and Tang [19] develop a noninteractive Vertically Privacy-Preserving LR (VPPLR) for FL. It trains an LR model by reformulating the gradient update rules and introducing a vectorization approach. The results present a reduction in communication and computational overhead and a decrease in training time with respect to the two schemes.

B. Logistic Regression with Homomorphic Encryption

Edemacu and Kim [3] propose a multi-party LR with privacy-preserving of poor data quality in a system with IoT contributors. The framework filters out poor-quality data through a gradient similarity metric and prevents information leaking by an HE scheme. The results show the approach's security, effectiveness, and robustness with noisy data.

Bonte and Vercauteren [20] introduce an LR with lower multiplicative complexity. It uses a simplified fixed Hessian method that produces accurate results considering the standard LR on plaintext data. Time complexity is an advantage of the approach because it grows linearly with respect to the number of covariates and training input instances.

Kim et al. [21] develop a privacy-preserving modified semi-parallel GWAS algorithm using Fisher Scoring and FHE. It evaluates data efficiently using an encrypted state. The proposed approach decreases the computational cost by reducing matrix multiplications and provides high accuracy compared to the result obtained in an unencrypted state.

Han et al. [22] present an LR algorithm with HE and an approximate bootstrapping. The authors propose the HE-specific Single-Instruction-Multiple-Data (SIMD) operations that parallelize the bootstrapping process and vectorize the LR algorithm. The results demonstrate the practical feasibility of LR training on large encrypted data.

Chiang [23] studies a privacy-preserving LR with a fast gradient variant for the model's training. The quadratic gradient extends the simplified fixed Hessian, and is enhanced using a Nesterov gradient and Adagrad. The results show that the proposed methods have a state-of-the-art convergence speed performance compared to the first-order gradient methods.

Zhou et al. [24] propose the Novel Linear Homomorphic Authenticated Encryption (N-LHAE) algorithm. It provides a privacy-preserving online diagnosis service that can protect the model's integrity (parameters), the results, and the data. A relevant advantage of N-LHAE is the absence of a trusted cloud and its reduced computational cost (overhead).

Yu et al. [25] design the Privacy-Preserving Outsourced LR (P2OLR) algorithm. It uses cloud resources to train and deploy an LR model without exposing data privacy.

Afterward, Yu et al. [26] present the Privacy-Preserving Vertical Collaborative LR (P2VCLR) system that reduces complexity. P2VCLR enables training a shared model with a secure joining of two parties' data and works without a Trusted Third-Party (TTP) coordinator. Later, Yu et al. [27] introduce the Cloud-Edge Collaborative Learning LR (CECLLR) algorithm. It can train a shared model over vertically partitioned data and combine the data from edge nodes and cloud data centers without a TTP.

Liu et al. [28] present a privacy-preserving LR where trusted hardware assists a cloud server during the training phase. The trusted hardware (Raspberry Pi) decrypts and re-encrypts the cyphertext during the bootstrapping process and the evaluation of the activation function.

IV. Training Policy

We implement a standard version of the LR model as a baseline to compare the efficiency of the FL policies. In a basic horizontal FL (LR_{FL}) approach, we assume that the datasets are evenly distributed among the system nodes, and each local node uses all available local data to train the model in each iteration.

As an alternative to traditional LR_{FL}, we develop a training reduction policy for LR_{FL} (LR_{FLn}) where local nodes use a reduced number of instances to train local models. LR_{FLn} decreases the number of training instances according to $1/i$ ratio, where i defines the iteration number. For instance, 100% of the local dataset is used for the first iteration, 50% for the second, 33% for the third iteration, and so on. The subset of training instances is chosen randomly for each iteration according to a normal distribution.

The central server calculates a Federated Averaging ($FedAvg$) of the k values of θ ($\theta_1, \theta_2, \dots, \theta_k$), where θ_i defines the model of the node i. It allows local nodes to perform several updates with their local data and exchange their coefficients of the LR model. For example, for k nodes in the FL system, the coefficients of the LR equation for the central server are updated as follows

$$\theta_{server} = (\theta_1 + \theta_2 + \cdots + \theta_k)/k \tag{3}$$

Then, θ_{server} is sent to the k nodes to update the local θ_i for $i = 1,2, \dots, k$. At the end of the process, we obtain θ_1^*, $\theta_2^*, \dots, \theta_k^*$ from the nodes to generate θ_{server}^* according to (3). Therefore, θ_{server}^* can be used with (1) to predict the class of new instances.

We also develop an FL ensemble LR (LR_{FLe}) in the central server using the local model of all the nodes. The nodes use their data to train local LR models, and then these models are sent to the central server to obtain a better predictive model, better than any of the individual local nodes.

In this model, different to LR_{FL} and LR_{FLn}, nodes in the FL system do not receive information from the central server. At the end of the process, the central server receives k values of θ from the nodes ($\theta_1^*, \theta_2^*, \dots, \theta_k^*$) that can be used with (1) to predict the class of new instances as follows

$$y = \begin{cases} 1 \;\; if \;\; (h_{\theta_1^*}(x) + h_{\theta_2^*}(x) + \cdots + h_{\theta_k^*}(x))/k \geq \tau \\ 0 \;\; if \;\; (h_{\theta_1^*}(x) + h_{\theta_2^*}(x) + \cdots + h_{\theta_k^*}(x))/k < \tau \end{cases} \tag{4}$$

In order to update the central server model with a proportional number of instances from the local models, we develop the weighted version of LR_{FLn} (LR_{FLnw}) where the update rule of θ for the k nodes follows

$$\theta_k = \theta_k - \alpha \nabla_\theta J(\theta) \left(\frac{1}{i}\right) \tag{5}$$

Then, reducing the number of instances on the training dataset implies a reduced update on θ.

Finally, the training dataset of each local model in LR_{FLe} can be reduced according to $1/i$ ratio policy, LR_{FLen} defines a LR_{FLe} approach with a reduction in the training dataset with $1/i$ ratio where i defines the iteration number.

The processing of the proposed model can be done in a privacy-preserving manner if the $\theta_1^*, \theta_2^*, \dots, \theta_k^*$ are encrypted using HE.

V. Experimental Evaluation

We compare the performance of LR_{FL} and LR_{FLn}, a novel FL with a new training policy that progressively reduces the instances of the training datasets of local nodes. The θ exchange mechanism uses FHE to ensure data privacy. The implementation based on Python 3.10.12, sklearn 1.4.1 library, and the open-source Simple Encrypted Arithmetic Library (SEAL) v3.14 [29] is performed on a computer with 64-bit Windows 11 Pro, Intel(R) Core (TM) i9-10980XE CPU at 3.00 GHz, 64 GB of memory, and 2 TB SSD.

The standard security setting of the FHE scheme considers a CKKS scheme with a security level of 128 bits, a polynomial modulo degree at most 2^{13}–1, and a moduli chain equal to {31, 26, 26, 26, 26, 26, 26, 31}[30]. The security level of the scheme guarantees that an adversary can only break the scheme with probability one after performing 2^{128} elementary operations.

Our implementation considers a vertical FL system where the nodes contain the same number of features (same feature space) with different instances. The number of nodes is constant during the training process, the nodes do not leave or join the system or consider new instances in the local training data.

A. Datasets

The evaluation of the strategies with real data is fundamental to measuring their performance. We consider six standard datasets widely used in the literature: Low Birth Weight (Lbw), Myocardial Infarction (Mi), Third National Health and Nutrition Examination Survey (Nhanes3), Indian diabetes (Pima), Prostate Cancer Study (Pcs), Umaru Impact Study (Uis) [31]. They contain a series of continuous input variables and two output classes. The values of the features were normalized in the range [0, 1] using the known min-max normalization method:

$$x_n = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{6}$$

where x is the original value, and x_n is the normalized value.

The Simple Split technique provides a methodology to compare the performance of the algorithms. A dataset is randomly divided into two subsets with a number of instances n-Training and n-Testing. One is used to train the classification model, and the other one is used to validate the training process. Table II summarizes the characteristics of datasets and the size of training and testing sets according to the Simple Split.

TABLE II. CHARACTERISTICS AND SIZE OF THE DATASETS.

Dataset	Features	Instances		
		Total (N)	n-Training	n-Testing
Low Birth Weight Study (Lbw)	9	189	151	38
Myocardial Infarction (Mi)	9	1,253	1,002	251
National Health and Nutrition Examination (Nhanes3)	15	15,649	12,519	3,130
Prostate Cancer Study (Pcs)	9	379	303	76
Indian's diabetes (Pima)	8	768	614	154
Umaru Impact Study (Uis)	8	575	460	115

B. *Evaluation Method*

The number of correct and incorrect predictions of each class defines the efficiency of a classifier. A Confusion Matrix (CM) is a manner to display the difference between the true and predicted classes for a set of examples. More meaningful measures can be extracted from the structure of the CM, for instance: Accuracy (A) expresses the systematic error in estimating a value by

$$A = \frac{T_p+T_n}{T_p+T_n+F_p+F_n} \quad (7)$$

where T_p and T_n define the number of elements classified correctly, T_p for positives and T_n for negatives, and the number of elements classified incorrectly is defined by F_p for positives and F_n for negatives.

C. *Experimental analysis*

The initial configuration of the LR based on the Batch GD algorithm considers 10 learning rates α={ 3.5, 3, 2.5, 2, 1.5, 1, 0.5, 0.1, 0.5, 0.1, 0.05, 0.01}, 10 values of iterations $nIter$ = {5, 10, 15, 20, 25, 30, 35, 40, 45, 50}, and 30 initial solutions for θ. In every experiment, the training and testing subsets of instances are chosen randomly from the dataset according to the Simple Split of 80–20% and a seed based on the number of experiments.

Table III presents the best configurations for LR and all datasets after 30 executions. In the case of Lbw, a learning rate higher than 0.1 and any number of iterations produce the highest accurate value. Similarly, for Uis, any learning rate and number of iterations generate the best accuracy value. For the Nhanes3 dataset, three configurations provide the best value of A. The value of accuracy provides a baseline to set and measure the efficiency of LR_{FL}, LR_{FLn}, LR_{FLnw}, LR_{FLe}, and LR_{FLen}.

Table IV shows the average A for 30 executions with all the strategies and nodes $k = \{2, 3, 4, 5, 6\}$. The results show that LR and LR_{FL} provide the same A for Lbw and Uis datasets. LR has a better performance than LR_{FL} on Nhanes3, Pcs, and Pima. Finally, LR_{FL} outperforms LR with respect to the Mi dataset.

TABLE III. THE BEST CONFIGURATION FOR LR.

Dataset	A	α	$nIte$
Lbw	0.6965	> 0.1	Any
Mi	0.9053	3.5	50
Nhanes3	0.7916	{3.0, 0.5, 0.1}	{30, 10, 45}
Pcs	0.6667	3.5	50
Pima	0.6543	0.5	10
Uis	0.7365	Any	Any

The average difference of A (dif_A) for LR_{FLn}, LR_{FLnw}, LR_{FLe}, and LR_{FLen} with respect to LR_{FL} considering the five datasets are 1.79%, 7.8%, 0.04%, and 1.95%, with a maximum of 9.25%, 30.76%, 0.35%, and 10.35% for Pcs, Pima, Mi, and Pcs datasets, respectively. Also, LR_{FLn}, LR_{FLe}, and LR_{FLew} can improve A with respect to LR_{FLn} about 0.11%, 0.04%, and 0.09% for Pima, Pcs, and Pima datasets.

$$dif_A(LR_{FLn}) = \left(A(LR_{FL}) - A(LR_{FLn})\right) * 100 \quad (8)$$

LR_{FLn} and LR_{FLen} have the worse performance with the Pcs dataset, their average decrease of A considering LR_{FL} and the five configurations of nodes are 6.736% and 7.236%, respectively, with a maximum of 9.25% and 10.35%. In the case of Lbw, the A decreases by 2.84% and 3.15% on average, with a maximum of 4.74% and 7.58%, respectively. Finally, for Mi, Nhanes3, and Uis datasets, the average reductions on A are 0.37%, and 0.41%, respectively, with a maximum of 1.49%, and 1.39%, respectively.

LR_{FLnw} has the worst performance of all the FL models, with an average decrease of 30.52%, 12.07%, and 4.29% for Mi, Pima, and Pcs, respectively. The average difference of A with respect to the Lbw, Nhanes3, and Uis is about 0.002%.

Table V shows the speedup of LR_{FL}, LR_{FLn}, LR_{FLnw}, LR_{FLe}, and LR_{FLen} with respect to LR. The measures consider the worst time of all nodes in the FL environment per iteration. LR_{FL}, LR_{FLn}, LR_{FLnw}, LR_{FLe}, and LR_{FLen} are faster than LR 44%, 63%, 64%, 93%, and 150% on average. The speedup for small datasets (Lbw, Pcs, Pima, and Uis) is negative or low.

The advantages of the rate reduction policy are perceived in the biggest datasets, Mi and Nhanes3. LR_{FLn} and LR_{FLen} speed up the execution of LR between 146% and 242%, and 124% and 656%. Both models with the reduction policy keep a similar A than LR_{FL}.

The average speedup of LR_{FLn}, LR_{FLnw}, LR_{FLe}, and LR_{FLen} with respect to LR_{FL} and all the datasets are 12%, 13%, 34%, and 69%, respectively. Table VI presents the speedup of the models with respect to LR_{FL} for Mi and Nhanes3 datasets. In the case of the Mi dataset, the speedup is from -10% to 59%, but LR_{FLnw} decreases the efficiency between 11.99% and 12.18%. For LR_{FLn}, LR_{FLe}, and FL_{en}, A changes within 0.11% and 1.49%. In the case of the Nhanes3 dataset, all strategies vary A between -0.01 and 0.01%, but the speedup is between 36% and 378%.

TABLE IV. AVERAGE ACCURACY AFTER 30 EXECUTIONS WITH THE BEST LR CONFIGURATION FOR DIFFERENT FL ENVIRONMENT CONFIGURATIONS.

Dataset	LR	LR_{FL}	LR_{FLn}	LR_{FLnw}	LR_{FLe}	LR_{FLen}	$dif_A(LR_{FLn})$	$dif_A(LR_{FLnw})$	$dif_A(LR_{FLe})$	$dif_A(LR_{FLen})$	Nodes
Lbw	0.6965	0.6965	0.6833	0.6965	0.6965	0.6842	1.32	0.0	0.0	1.23	2
			0.6877	0.6965	0.6965	0.6965	0.88	0.0	0.0	0.0	3
			0.6491	0.6965	0.6965	0.6684	4.74	0.0	0.0	2.81	4
			0.6518	0.6965	0.6965	0.6211	4.47	0.0	0.0	7.54	5
			0.6684	0.6965	0.6965	0.6544	2.81	0.0	0.0	4.21	6
Mi	0.9053	0.9057	0.8965	0.7849	0.9046	0.8954	0.92	12.08	0.11	1.04	2
			0.8938	0.7851	0.9042	0.8918	1.20	12.06	0.15	1.39	3
			0.8960	0.7858	0.9042	0.8926	0.97	11.99	0.15	1.31	4
			0.8908	0.7839	0.9028	0.8956	1.49	12.18	0.29	1.01	5
			0.8963	0.7851	0.9023	0.8919	0.94	12.06	0.35	1.38	6
Nhanes3	0.7916	0.7915	0.7915	0.7914	0.7915	0.7915	0.0	0.01	0.0	0.0	2
			0.7914	0.7915	0.7915	0.7914	0.01	0.0	0.0	0.01	3
			0.7915	0.7915	0.7915	0.7915	0.0	0.0	0.0	0.0	4
			0.7916	0.7915	0.7915	0.7916	0.0	0.01	0.0	-0.01	5
			0.7915	0.7915	0.7915	0.7915	0.0	0.01	0.0	0.0	6
Pcs	0.6667	0.6658	0.6246	0.6237	0.6654	0.6211	4.12	4.21	0.04	4.47	2
			0.5908	0.6263	0.6654	0.5829	7.50	3.95	0.04	8.29	3
		0.6654	0.5728	0.6232	0.6636	0.5776	9.25	4.21	0.18	8.77	4
			0.5917	0.6189	0.6658	0.5618	7.37	4.65	-0.04	10.35	5
		0.6658	0.6114	0.6215	0.6654	0.6228	5.44	4.43	0.04	4.30	6
Pima	0.6543	0.6537	0.6476	0.3463	0.6537	0.6474	0.61	30.74	0.0	0.63	2
			0.6543	0.3506	0.6537	0.6543	-0.06	30.30	0.0	-0.06	3
			0.6548	0.3500	0.6537	0.6545	-0.11	30.37	0.0	-0.09	4
			0.6535	0.3461	0.6537	0.6535	0.02	30.76	0.0	0.02	5
			0.6545	0.3496	0.6537	0.6543	-0.09	30.41	0.0	-0.06	6
Uis	0.7365	0.7365	0.7365	0.7365	0.7365	0.7365	0.0	0.0	0.0	0.0	2
			0.7365	0.7365	0.7365	0.7365	0.0	0.0	0.0	0.0	3
			0.7365	0.7365	0.7365	0.7365	0.0	0.0	0.0	0.0	4
			0.7365	0.7365	0.7365	0.7365	0.0	0.0	0.0	0.0	5
			0.7365	0.7365	0.7365	0.7365	0.0	0.0	0.0	0.0	6

TABLE V. SPEEDUP OF FL ENVIRONMENTS WITH RESPECT TO LR FOR ALL THE DATASETS.

Dataset	LR_{FL}	LR_{FLn}	LR_{FLnw}	LR_{FLe}	LR_{FLen}	Nodes
Lbw	0.91	0.91	0.92	1.20	0.99	2
	0.94	0.93	0.94	1.24	1.01	3
	0.96	0.93	0.94	1.28	1.00	4
	0.96	0.94	0.96	1.28	1.01	5
	0.98	0.95	0.96	1.31	1.01	6
Mi	2.04	2.46	2.45	2.44	3.24	2
	2.40	2.56	2.58	2.85	3.36	3
	2.54	2.61	2.62	3.15	3.50	4
	2.83	2.63	2.66	3.50	3.55	5
	2.95	2.66	2.66	3.60	3.54	6
Nhanes3	1.28	2.51	2.54	1.74	4.87	2
	1.25	2.85	2.91	2.31	5.97	3
	1.48	3.09	3.14	2.83	6.69	4
	1.69	3.29	3.30	3.12	7.15	5
	1.87	3.42	3.45	3.40	7.56	6

Dataset	LR_{FL}	LR_{FLn}	LR_{FLnw}	LR_{FLe}	LR_{FLen}	Nodes
Pcs	1.00	1.05	1.08	1.13	1.14	2
	1.09	1.09	1.10	1.38	1.16	3
	1.13	1.08	1.10	1.43	1.16	4
	1.14	1.09	1.11	1.47	1.20	5
	1.15	1.10	1.12	1.46	1.21	6
Pima	1.04	1.07	1.08	1.32	1.44	2
	1.22	1.13	1.11	1.49	1.44	3
	1.28	1.14	1.14	1.59	1.50	4
	1.38	1.14	1.13	1.77	1.52	5
	1.41	1.16	1.14	1.83	1.53	6
Uis	1.08	1.00	1.00	1.33	1.32	2
	1.17	1.01	1.02	1.45	1.36	3
	1.29	1.03	1.04	1.63	1.37	4
	1.32	1.03	1.03	1.67	1.38	5
	1.34	1.04	1.04	1.72	1.38	6

TABLE VI. SPEEDUP OF FL ENVIRONMENT WITH RESPECT TO TRADITIONAL FL MODEL FOR MI AND NHANES3 DATASETS.

Dataset	LR_{FLn}	LR_{FLnw}	LR_{FLe}	LR_{FLen}	Nodes
Mi	1.21	1.20	1.20	1.59	2
	1.06	1.07	1.19	1.40	3
	1.03	1.03	1.24	1.38	4
	0.93	0.94	1.24	1.25	5
	0.90	0.90	1.22	1.20	6

Dataset	LR_{FLn}	LR_{FLnw}	LR_{FLe}	LR_{FLen}
Nhanes3	1.96	1.99	1.36	3.81
	2.28	2.33	1.85	4.78
	2.09	2.13	1.92	4.53
	1.95	1.95	1.84	4.23
	1.82	1.84	1.81	4.03

Figure 3 shows the normalized values of accuracy and time for LR_{FL}, LR_{FLn}, LR_{FLnw}, LR_{FLe}, and LR_{FLen} with all the configurations and datasets. LR_{FL} provides the maximum execution time and no reduction in accuracy, on average (baseline). LR_{FLn} is faster than LR_{FL} but with a reduction of A. LR_{FLe} and LR_{FLen} are faster than LR_{FL} and LR_{FLn}, LR_{FLe}

reduces the accuracy very little with respect to the worst strategy and provides an acceleration of 50% with respect to the maximum time reduction. LR_{FLen} has the lowest execution time and an accuracy reduction of about 25%. LR_{FLnw} provides a reduction of time of about 20% and the worst accuracy reduction. The behavior of the strategies shows the tradeoff between time and accuracy. LR_{FL} guarantees the maximum accuracy with higher processing time, LR_{FLen} ensures the minimum processing time and reduces accuracy. LR_{FLe} can provide a balance between processing time and accuracy.

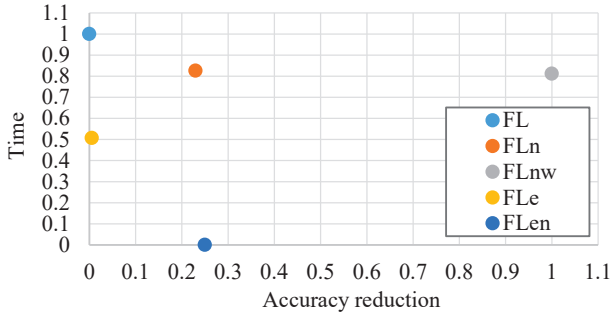

Fig. 3. Normalized accuracy and time for all the FL models, datasets, and node configuration.

D. HE time

We also present the time to encrypt, decrypt, and calculate the aggregation of the ciphertexts with the values θ_i and θ_{server} that the nodes and the central server exchange. Our implementation is based on [32]. Table VII shows the average time of the HE operations for FL environments in seconds (sec.). The most complex operation, according to the time, is the encryption of the θ_i values. All the FL models perform homomorphic operations to process the federated privacy-preserving LR model.

TABLE VII. AVERAGE TIME OF HE OPERATIONS (SEC).

	Encrypt	Average	Decrypt
Lbw	0.02409	0.00845	0.00916
Mi	0.02415	0.00838	0.00930
Nhanes3	0.03171	0.01085	0.01230
Pcs	0.02415	0.00796	0.00959
Pima	0.02469	0.00839	0.00937
Uis	0.02612	0.00906	0.00986

HE is a solution to keep data processing confidential because data are processed using ciphertexts [33]. However, HE addition and multiplication operations perform efficiently on ciphertexts, increasing impracticality due to the large computational overhead. The main challenge for adopting HE is its performance.

VI. CONCLUSIONS

Data processing in an environment with shared resources can provoke security issues because data must be decrypted for processing. Federated learning and homomorphic encryption are two alternative solutions to solve privacy-preserving problems. Some limitations on both approaches reduce their applicability to specific domains.

In this paper, we analyze the latest advances in privacy-preserving logistic regression solutions for processing confidential data using federated learning and homomorphic encryption. We present the characteristics of the most recent approaches in the field: algorithms, evaluation metrics, used datasets, implementation characteristics, etc.

Also, we proposed one policy to reduce the training time of the federated model and conduct a comprehensive simulation analysis on the six datasets from medicine (diabetes, cancer, drugs, etc.) and genomics.

The results show that the proposed policies can reduce the training time with a slight reduction in the final accuracy of the model. However, further study is required to assess their actual performance and effectiveness. This will be the subject of future work.

ACKNOWLEDGMENT

The research conducted in this publication was jointly funded by the Irish Research Council under grant number GOIPD/2023/1341; the European Commision under grants 101084013 (DIGITAL4Business) and grant number 101140316 (Digital4Sustainability) respectively; and The Ministry of Science and Higher Education of the Russian Federation under grant number 075-15-2022-294.

REFERENCES

[1] A. Wood, K. Najarian, and D. Kahrobaei, "Homomorphic encryption for machine learning in medicine and bioinformatics," ACM Computing Surveys, vol. 53, no. 4, pp. 1–35, 2020, DOI: 10.1145/3394658

[2] B. Pulido-Gaytan, A. Tchernykh, J. M. Cortés-Mendoza, et al., "Privacy-preserving neural networks with homomorphic encryption: Challenges and opportunities," Peer-to-Peer Networking and Applications, vol. 14, no. 3, pp. 1666–1691, 2021, DOI: 10.1007/s12083-021-01076-8

[3] K. Edemacu and J. W. Kim, "Multi-party privacy-preserving logistic regression with poor quality data filtering for IoT contributors," Electronics, vol. 10, no. 17, p. 2049, 2021, DOI: 10.3390/electronics10172049

[4] J. So, B. Güler, and A. S. Avestimehr, "Codedprivateml: A fast and privacy-preserving framework for distributed machine learning," Journal on Selected Areas in Information Theory, vol. 2(1), pp. 441–451, 2021, DOI: 10.1109/JSAIT.2021.3053220

[5] M. De Cock, R. Dowsley, A. C. Nascimento, D. Railsback, J. Shen, and A. Todoki, "High performance logistic regression for privacy-preserving genome analysis," BMC Medical Genomics, vol. 14, pp. 1–18, 2021, DOI: 10.1186/s12920-020-00869-9

[6] A. Patra, and A. Suresh, "BLAZE: blazing fast privacy-preserving machine learning," in Network and Distributed Systems Security (NDSS) Symposium, Feb., 2020, DOI: 10.14722/ndss.2020.24202

[7] R. Duan, M. R. Boland, Z. Liu, et al., "Learning from electronic health records across multiple sites: A communication-efficient and privacy-preserving distributed algorithm," Journal of the American Medical Informatics Association, vol. 27, no. 3, pp. 376–385, 2020, DOI: 10.1093/jamia/ocz199

[8] M. Bogowicz, A. Jochems, T. M. Deist, et al., "Privacy-preserving distributed learning of radiomics to predict overall survival and hpv status in head and neck cancer," Scientific reports, vol. 10, no. 1, p. 4542, 2020, DOI: 10.1038/s41598-020-61297-4

[9] S. Ruder, "An overview of gradient descent optimization algorithms," arXiv preprint arXiv:1609.04747, 2016.

[10] S. Ji, T. Saravirta, S. Pan, G. Long, and A. Walid, "Emerging trends in federated learning: From model fusion to federated x learning," arXiv:2102.12920, 2021.

[11] D. Naik, and N. Naik, "An introduction to federated learning: working, types, benefits and limitations," In UK Workshop on Computational Intelligence (pp. 3-17). Springer, 2023, DOI: 10.1007/978-3-031-47508-5_1

[12] C. Gentry, “Computing arbitrary functions of encrypted data,” Communications of the ACM, vol. 53, no. 3, pp. 97–105, 2010, DOI: 10.1145/1666420.1666444

[13] A. Al Badawi, and Y. Polyakov, “Demystifying bootstrapping in fully homomorphic encryption,” Cryptology ePrint Archive, no. 149, 2023.

[14] Y. Yang, X. Huang, X. Liu, et al., “A comprehensive survey on secure outsourced computation and its applications,” IEEE Access, vol. 7, pp. 159426–159465, 2019, DOI: 10.1109/ACCESS.2019.2949782

[15] M. Shaheen, M. S. Farooq, T. Umer, and B. S. Kim, “Applications of federated learning; Taxonomy, challenges, and research trends,” Electronics, 11(4), 670, 2022, DOI: 10.3390/electronics11040670

[16] J. Zhao, H. Zhu, F. Wang, R. Lu, E. Wang, L. Li, and H. Li, “VFLR: An efficient and privacy-preserving vertical federated framework for logistic regression,” IEEE Transactions on Cloud Computing, vol. 11, no. 4, 2023, DOI: 10.1109/TCC.2023.3247870

[17] D. He, R. Du, S. Zhu, M. Zhang, K. Liang, and S. Chan, “Secure logistic regression for vertical federated learning,” IEEE Internet Computing, 26(2), 61-68, 2021, DOI: 10.1109/MIC.2021.3138853

[18] F. Wang, H. Zhu, R. Lu, Y. Zheng, and H. Li, “A privacy-preserving and non-interactive federated learning scheme for regression training with gradient descent,” Information Sciences, 552, 183-200, 2021, DOI: 10.1016/j.ins.2020.12.007

[19] Y. Zhang, and M. Tang, “VPPLR: Privacy-preserving logistic regression on vertically partitioned data using vectorization sharing,” Journal of Information Security and Applications, 82, 2024, DOI: 10.1016/j.jisa.2024.103725

[20] C. Bonte, and F. Vercauteren, “Privacy-preserving logistic regression training,” BMC medical genomics, vol. 11, pp. 13–21, 2018, DOI: 10.1186/s12920-018-0398-y

[21] D. Kim, Y. Son, D. Kim, A. Kim, S. Hong, and J. H. Cheon, “Privacy-preserving approximate GWAS computation based on homomorphic encryption,” BMC Medical Genomics, vol. 13, no. 7, pp. 1–12, 2020, DOI: 10.1186/s12920-020-0722-1

[22] K. Han, S. Hong, J. H. Cheon, and D. Park, “Logistic regression on homomorphic encrypted data at scale,” in AAAI-19, vol. 33, (Honolulu), pp. 9466–9471, Feb. 2019. DOI: 10.1609/aaai.v33i01.33019466

[23] J. Chiang, “Privacy-preserving logistic regression training with a faster gradient variant,” arXiv preprint arXiv:2201.10838, 2022.

[24] Y. Zhou, L. Song, Y. Liu, P. Vijayakumar, B. B. Gupta, W. Alhalabi, and H. Alsharif, “A privacy-preserving logistic regression-based diagnosis scheme for digital healthcare,” Future Generation Computer Systems, vol. 144, pp. 63–73, 2023, DOI: 10.1016/j.future.2023.02.022

[25] X. Yu, W. Zhao, Y. Huang, J. Ren, and D., Tang, “Privacy-preserving outsourced logistic regression on encrypted data from homomorphic encryption,” Security and Communication Networks, no. Article ID 1321198, 2022, DOI: 10.1155/2022/1321198

[26] X. Yu, W. Zhao, D. Tang, and K. Liang, “Privacy-preserving vertical collaborative logistic regression without trusted third-party coordinator,” Security and Communication Networks, no. Article ID 5094830, 2022, DOI: 10.1155/2022/5094830

[27] X. Yu, D. Tang, and W. Zhao, “Privacy-preserving cloud-edge collaborative learning without trusted third-party coordinator,” Journal of Cloud Computing, vol. 12, no. 1, pp. 1–11, 2023, DOI: 10.1186/s13677-023-00394-x

[28] C. Liu, Z. L. Jiang, X. Zhao, et al., “Efficient and privacy-preserving logistic regression scheme based on leveled fully homomorphic encryption,” in INFOCOM 2022, (New York), pp. 1–6, IEEE, May 2022, DOI: 10.1109/INFOCOMWKSHPS54753.2022.9797933

[29] H. Chen, K. Laine, and R. Player, “Simple encrypted arithmetic library–SEAL v2.1,” in FC 2017, vol. 10323 of LNCS, (Malta), pp. 3–18, Springer, Apr. 2017, DOI: 10.1007/978-3-319-70278-0_1

[30] M. Albrecht, M. Chase, H. Chen, et al., “Homomorphic encryption standard,” in Protecting Privacy through Homomorphic Encryption, (K. Lauter, W. Dai, and K. Laine, eds.), ch. Part II, pp. 31–62, Cham: Springer, 2021. ISBN 978-3-030-77286-4, DOI: 10.1007/978-3-030-77287-1_2

[31] J.M. Cortés-Mendoza, A. Tchernykh, M. Babenko, B. Pulido-Gaytán, G. Radchenko, “Multi-cloud privacy-preserving logistic regression,” in: Voevodin, V., Sobolev, S. (eds) Supercomputing. RuSCDays 2021. Communications in Computer and Information Science, vol 1510. Springer, 2021, DOI: 10.1007/978-3-030-92864-3_35

[32] A. Benaissa, “Training and evaluation of logistic regression on encrypted data,” [Online] https://github.com/OpenMined/TenSEAL, last accessed 2024/06/25

[33] B. Pulido-Gaytan and A. Tchernykh, “Self-Learning activation functions for homomorphic encryption of convolutional neural networks with improved accuracy”. PLoS One, 19(7), e0306420, 2024, DOI: 10.1371/journal.pone.0306420