Differential and TriPhase Adaptive Learning-Based Privacy-Preserving Model for Medical Data in Cloud Environment

Rishabh Gupta, Deepika Saxena, Ishu Gupta, *Member, IEEE*, and Ashutosh Kumar Singh, *Senior Member, IEEE*

***Abstract*—This letter proposes a novel secure data protection model for privacy preservation in the cloud environment by partitioning, sanitizing, and analyzing the data effectively to improve the model's privacy. Several experiments and comparisons of the proposed model with existing works indicate that it protects data with high accuracy, precision, recall, and F1-score up to 87.03%, 84.87%, 87.03%, and 85.00%, for diverse datasets with a relative improvement up to 15.89%, 27.73%, 15.89%, and 21.44%, respectively.**

***Index Terms*—Secure communication, cloud computing, K-anonymity, differential privacy, healthcare, deep neural network.**

I. Introduction

DATA sharing and analysis is essential for pathology centers to enhance the utility and quality of medical services. Most of the centres have shifted or planned to shift their data to the cloud platform because of its storage and analysis services at lower cost [1]. The outsourced data can be accessed by researchers or laboratories to monitor and track the spread of infectious diseases. The centers are hesitant to share their sensitive data to the cloud or any third party since these may misuse and leak the data to other parties without owners' consent [2]. Thus, protecting shared data across the entities while performing analysis effectively has become one of the most challenging problems in cloud environments.

The key state-of-arts models have been reported to address this challenge. A privacy preserving machine learning (PMLM) model [3] encrypted the data, added noise in entire partially decrypted data, and analyzed it using existing classifiers. Further, deep learning is exploited to perform classification on encrypted data through deep neural network in privacy-preserving deep learning (PDLM) model [4]. Moreover, a machine learning and probabilistic analysis-based model (MLPAM) [5] shared encrypted data and performed analysis on decrypted noised data using existing classifiers while securing the communication. The classification over encrypted data, partially decrypted or entire noised data through existing classifiers resulted in high computation time, low computation accuracy, and reduced data utility which is insufficient in the real environments and required to be upgraded.

To improve data and its communication security while maintaining utility and reducing computation time, this letter proposes a novel Differential and TriPhase adaptive learning-based Privacy-Preserving Model (DT-PPM) for medical data protection by enabling secure data storage, analysis, and sharing in the cloud environment. Unlike existing works, DT-PPM partitions the data based on its sensitivity employing k-anonymization, injects noise in selected data through Laplace mechanism to make it private and reduce perturbation's impact. It shares the integrated data with the cloud where an effective analysis is performed through multi-layered feed-forward deep neural network (MFNN) which is trained with developed TriPhase Adaptive Differential Evolution (TADE) learning algorithm to optimize learning of neural network. Table I consists of notation and their descriptions.

TABLE I
NOTATIONS AND MEANINGS

PT: pathology centers; CSP cloud service provider; LB: laboratories; D: data;
n: number of pathology centers; m: number of laboratories; D^S: sensitive data;
D^{NS}: non-sensitive data; D_N^S: synthetic data; N: noise; $\tilde{D}_i$: aggregated data;
CM: classification model; R: random mechanism; Y: scaling parameter;
ϵ: privacy budget; Δ: sensitivity; Pb: probability; $\tilde{D}^N$: preprocessed data;
ω: weight; ψ: network size; δ: mutation learning rate; f query function;
CL: class label; CA: classification accuracy; P: precision; R: recall; FS: f1-score;

II. Proposed Model

Let n pathology centers $\{PT_1, PT_2, \ldots, PT_n\} \in \mathbb{PT}_{id}$ possess their data $\{D_1, D_2, \ldots, D_n\} \in \mathbb{D}$ that needs to be shared with m laboratories $\{LB_1, LB_2, \ldots, LB_m\} \in \mathbb{LB}$ and *CSP* for analysis, storage, and utilization. Fig. 1 represents the workflow of the proposed model consisting of three entities: 1) *Pathology Centers* (PT_{id}): it generates data and transfers it to the cloud platform. PT_{id} is treated as an trusted entity because owners will not leak their own data. 2) *Cloud Service Provider* (*CSP*): it receives sanitized data from PT_{id} and provides storing, computing, and sharing services. *CSP* is deemed an untrusted entity since it is curious to learn the information. 3) *Laboratories* (LB_{id}): it acquires sanitized data from *CSP* and is considered an untrusted entity.

To protect the data $\mathbb{D}$, PT_{id} partitions it into sensitive $\{D_1^S, D_2^S, \ldots, D_n^S\} \in \mathbb{D}^S$ and non-sensitive $\{D_1^{NS}, D_2^{NS},$

Manuscript received 5 September 2022; accepted 12 October 2022. Date of publication 18 October 2022; date of current version 21 November 2022. The associate editor coordinating the review of this article and approving it for publication was S. Djahel. *(Corresponding author: Ishu Gupta.)*

Rishabh Gupta, Deepika Saxena, and Ashutosh Kumar Singh are with the Department of Computer Applications, National Institute of Technology Kurukshetra, Kurukshetra 136119, India (e-mail: rishabhgpt66@gmail.com; 13deepikasaxena@gmail.com; ashutosh@nitkkr.ac.in).

Ishu Gupta is with the Cloud Computing Research Center, Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan (e-mail: ishugupta23@gmail.com).

Digital Object Identifier 10.1109/LNET.2022.3215248

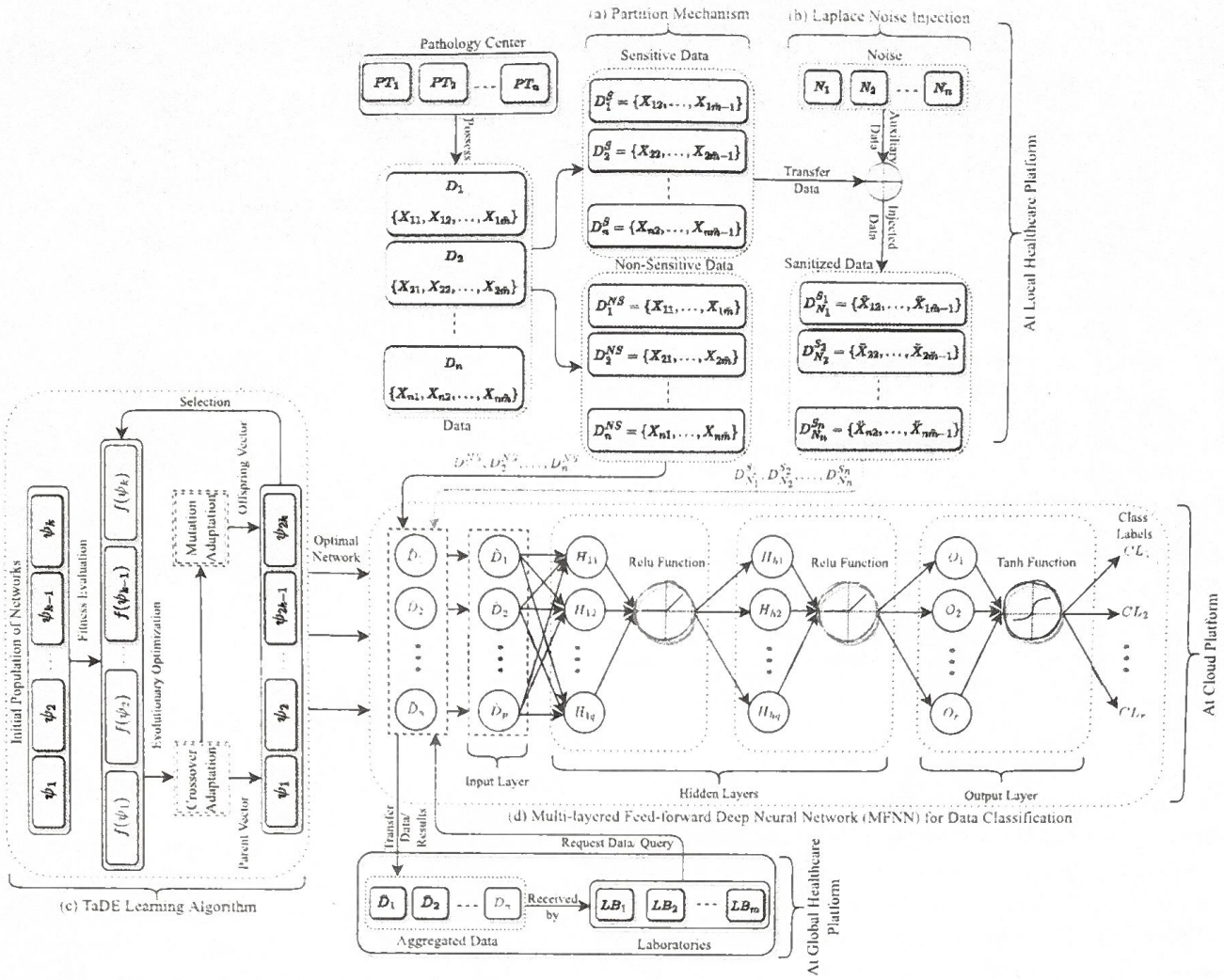


Fig. 1. DT-PPM architecture.

$\ldots, D_n^{NS}\} \in \mathbb{D}^{HC}$ parts through K-anonymization mechanism (discussed in Section II-A). Since it is the most capable technique for mitigating information loss, minimizing information leakage risk, protecting data against misuse and insider exploitation risks when data is communicated in the sharing environments [6]. PT_{id} procures synthetic data $\{D_{N_1}^{S_1}, D_{N_2}^{S_2}, \ldots, D_{N_n}^{S_n}\} \in \mathbb{D}_{H}^{S}$ by injecting noise $\{N_1, N_2, \ldots, N_n\} \in \mathbb{N}$ into sensitive data $D_1^S, D_2^S, \ldots, D_n^S$ utilising ϵ-differential privacy (described in Section II-B). Since it is the most appropriate mechanism to perform privacy protection based on auxiliary information [3]. PT_{id} transfers $D_{N_i}^{S_i}$ and D_i^{NS} to *CSP* that combines them into aggregated data $\{\bar{D}_1, \bar{D}_2, \ldots, \bar{D}_n\} \in \bar{\mathbb{D}}$. Whenever, LB_{id} requests for $\bar{D}_1, \bar{D}_2, \ldots, \bar{D}_n$ from *CSP*, it enables LB_{id} to access requested data $\bar{D}_i$ for usage. *CSP* performs privacy-preserving computation over $\bar{D}$ through its classification model (*CM*), enables PT_{id} or LB_{id} to make query, and reverts the obtained results from *CM* (outlined in Section II-C).

A. Data Partition

Let D_i of PT_{id} is composed of a relation with features $\{\Lambda_1, \Lambda_2, \ldots, \Lambda_{t_1}\}$, $\{\Lambda_1, \Lambda_2, \ldots, \Lambda_{t_2}\}$, $\ldots$, $\{\Lambda_1, \Lambda_2, \ldots, \Lambda_{t_n}\}$, respectively, which are separated into sensitive and non-sensitive relations using Theorem 1 generated by K-anonymization as shown in (a) block of Fig. 1.

Definition 1 (Sensitive and Non-Sensitive Data): sensitive data (D_i^S) is protected and kept out of reach of any outsider that does not have permission to access it. While non-sensitive data (D_i^{NS}) need not be protected and can be shared directly without any special protection. These are distinguished by PT as per the requirements where $S_1, S_2, \ldots, S_\alpha$ are D_i^S attributes and $NS_1, NS_2, \ldots, NS_\beta$ are D_i^{NS} attributes; $\alpha, \beta \in Z$, $1 \leq \lambda \leq \alpha$, $1 \leq \varphi \leq \beta$, and $\lambda + \varphi \leq \Lambda_{t_1}$.

Theorem 1: Let the total number of attributes in D_i are $\Lambda_1, \Lambda_2, \ldots, \Lambda_{t_1}$, which are partitioned into $S_1, S_2, \ldots, S_\alpha \in D_1^S$ and $NS_1, NS_2, \ldots, NS_\beta \in D_1^{NS}$ by applying the specified anonymity parameter K, the data integrity is maintained when D_i^S and D_i^{NS} are combined together after partitioning.

Proof: Let PT_1 partitions data D_1 into $D_1^S = \{S_1, S_2, \ldots, S_\alpha\}$ and $D_1^{NS} = \{NS_1, NS_2, \ldots, NS_\beta\}$ by using Eqs. (1) and (2).

$$D_1^S = \prod_{(S_1, S_2, \ldots, S_\alpha)} (D_1) \tag{1}$$

$$D_1^{NS} = \prod_{(NS_1, NS_2, \ldots, NS_\beta)} (D_1) \tag{2}$$

In addition, PT_2 splits D_2 into $D_2^S = \{S_1, S_2, \ldots, S_\delta\}$ and $D_2^{NS} = \{NS_1, NS_2, \ldots, NS_\eta\}$, $\ldots$, PT_n partitions D_n into $D_n^S = \{S_1, S_2, \ldots, S_\epsilon\}$ and $D_n^{NS} = \{NS_1, NS_2, \ldots, NS_\vartheta\}$

respectively, where $a, b, c, d \in Z$. $D_1^S \cap D_1^{NS} = o$ and $D_1^S = D_1 - \{NS_1, NS_2, \ldots, NS_3\}$. Thus, $D_1^S \cup D_1^{NS} = D_1$. ■

B. Data Preservation

To protect $\mathbb{D}^S$, each $PT_1, PT_2, \ldots, PT_n$ injects the noise $(N_1, N_2, \ldots, N_n)$ into sensitive data $(D_1^S, D_2^S, \ldots, D_n^S)$, respectively, using ϵ-differential privacy (block (b) of Fig. 1).

Definition 2 (ϵ-Differential Privacy): it is defined through a random function $\hat{R}$ where every combination of dataset D and its neighbor D', and $\forall\Gamma \subseteq \text{Range}(\hat{R})$ satisfies Eq. (3). Here, Pb[.] is the probability function applied to the mechanism $\hat{R}$ and demonstrates the privacy revealing risk.

$$Pb[\hat{R}(D) = \Gamma] \leq exp(\epsilon) \times Pb[\hat{R}(D') = \Gamma] \tag{3}$$

Definition 3 (Privacy-Budget (ϵ)): it is a predefined privacy parameter responsible for maintaining the balance between privacy loss and utility maximization. A smaller ϵ (i.e., maximum noise) makes privacy more vital but reduces data utility.

To procure ϵ-differential privacy, a numeric query function (f) is employed that maps a data set D into real space with d-dimensions; it is indicated as $f : \mathrm{D} \rightarrow R^d$. The sensitivity of f for each combination of dataset D and D' is assigned using Eq. (4):

$$\Delta f = \max_{D,D'} \|f(D) - f(D')\|_{\hat{Q}_1} \tag{4}$$

where norm is represented by $\|\cdot\|_{\hat{Q}_1}$. The sensitivity Δf is only dependent on the query f types and measures the maximum gap between query results on neighboring data sets. For any function $f : \mathrm{D} \rightarrow R^d$, the Laplace mechanism F is described in Eq. (5).

$$F(D) = f(D) + (Lap_1(\Upsilon), Lap_2(\Upsilon), \ldots, Lap_d(\Upsilon)) \tag{5}$$

where the noise vector N is computed from a Laplace distribution $Lap_t(\Upsilon)$ (t $\in$ [1, d]) using Eq. (6) whose probability density function is centered at 0 with scale $\Upsilon \in R^+$.

$$N = \frac{1}{2\Upsilon} \cdot (exp(\frac{-|x|}{\Upsilon})) \tag{6}$$

To produce noise $N_1, N_2, \ldots, N_n$, the sample is taken from the Laplace distribution with a scaling parameter $\Upsilon = \Delta f/\epsilon$ under the control of ϵ. The generated noise N_i is injected into the corresponding D_i^S as $D_{N_i}^{S_i} = D_i^S + N_i$, where $i \in [1, n]$. PT_{id} transfers the sanitized data $D_{N_i}^{S_i}$ to *CSP* that stores, and shares it to LB_{id}.

C. Data Classification

CSP preprocesses data $\{\bar{D}_1, \bar{D}_2, \ldots, \bar{D}_n\} \in \bar{\mathbb{D}}$ using the normalization function $\bar{D}_i = (\bar{D}_i - \bar{D}_{min})/(\bar{D}_{max} - \bar{D}_{min})$, where $\bar{D}_{max}$ and $\bar{D}_{min}$ are the maximum and minimum values of the input data set. The preprocessed data $\bar{D}_i$ is divided into training and testing data. The training data is passed to a multi-layered feed-forward neural network (MFNN) for training. As shown in block (c) of Fig. 1, TaDE algorithm is employed to optimize the neural weights ϖ (i.e., network vector) during the learning process. It explores and exploits a population of network vectors (υ) and returns the most optimal network vector for MFNN (block (d) of Fig. 1) based classification. It involves mutation followed by crossover to update the population of network vectors. The three-step adaptation is engaged at mutation, crossover, and control parameter tuning phases. The three mutation strategies, i.e., $DE/rand/2$ (Eq. (7)), $DE/best/2$ (Eq. (8)), and $DE/current\text{-}to\text{-}best/2$ (Eq. (9)) are utilized as follows:

$$\Phi_u^\upsilon = \varrho_{r1}^\upsilon + \delta_u \times (\varrho_{r2}^\upsilon - \varrho_{r3}^\upsilon) + \delta_u \times (\varrho_{r4}^\upsilon - \varrho_{r5}^\upsilon) \tag{7}$$

$$\Phi_u^\upsilon = \varrho_{best}^\upsilon + \delta_u \times (\varrho_{r1}^\upsilon - \varrho_{r2}^\upsilon) + \delta_u \times (\varrho_{r3}^\upsilon - \varrho_{r4}^\upsilon) \tag{8}$$

$$\Phi_u^\upsilon = \varrho_u^\upsilon + \delta_u \times (\varrho_{best}^\upsilon - \varrho_u^\upsilon) + \delta_u \times (\varrho_{r1}^\upsilon - \varrho_{r2}^\upsilon) \tag{9}$$

where $r1, r2, r3, r4$, and $r5$ are random vectors with the adaptive generation of mutation learning rate (δ_u). Φ_u^υ, ϱ_u^υ and ϱ_{best}^υ indicate u^{th} mutant trail vector, u^{th} population vector, and the best solution in current population respectively for υ^{th} generation. A mutation-selection probability (mps) is used to select one among these mutation schemes during each iteration of the learning process. Further, three crossover schemes: Multi-point (Eq. (10)-(11)), Ring (Eq. (12)–(13)), and Heuristic (Eq. (14)), are selectively applied to update the population of network vectors.

$$\varrho_{child1}^u = \begin{cases} \varrho_{2u} & If(k_1 \leq \varrho_{1u} \leq k_2) \\ \varrho_{1u} & otherwise \end{cases} \tag{10}$$

$$\varrho_{child2}^u = \begin{cases} \varrho_{1u} & If(k_1 \leq \varrho_{2u} \leq k_2) \\ \varrho_{2u} & otherwise \end{cases} \tag{11}$$

$$\varrho_{child1}^u = \mathfrak{B} \times \varrho_{1u} + (L - \mathfrak{B}) \times \varrho_{2u} \tag{12}$$

$$\varrho_{child2}^u = \mathfrak{B} \times \varrho_{2u} + (L - \mathfrak{B}) \times \varrho_{1u} \tag{13}$$

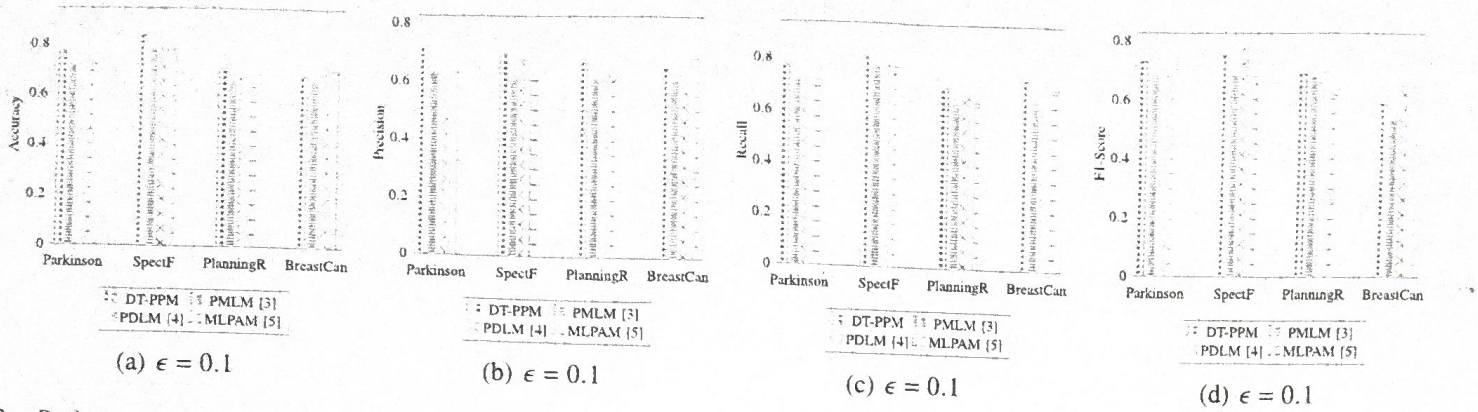
$$\varrho_{childu} = \delta \times (\varrho_{1u} - \varrho_{2u}) + \varrho_{1u} \tag{14}$$

where ϱ_{child}^u, ϱ_1, ϱ_2, and $\mathfrak{B} \in [1, L]$ are new offspring, two parent chromosomes, and cut point. To prevent premature convergence, the control parameters, including crossover and mutation rate, are regenerated randomly if at least two-fifths of the population is not updated with the current values of these control parameters. A fitness function named cross-entropy (CE) in Eq. (15) with the number of data samples (c), correct ($\ddot{y}$), and predicted ($\ddot{p}$) probs evaluates the network vectors during the learning process. The next population vector is selected using Eq. (16), where $\varrho_u^{\upsilon+1}$, and $\varrho_{childu}^\upsilon$ are the candidates selected for next-generation and solution generated after crossover, respectively.

$$CE = -\frac{1}{c}\sum_{k=1}^{c}(\ddot{y}log(\ddot{p}) + (1 - \ddot{y})log(1 - \ddot{p})) \tag{15}$$

$$\varrho_u^{\upsilon+1} = \begin{cases} \varrho_{childu}^\upsilon & (If(fitness(\varrho_{childu}^\upsilon)) < (fitness(\varrho_u^\upsilon)) \\ \varrho_u^\upsilon & otherwise \end{cases} \tag{16}$$

During the testing process, the test data is provided to MFNN to assign class label vector $\mathbb{CL} = \{CL_1, CL_2, \ldots, CL_r\}$. Using $\mathbb{CL}$, the Classification Accuracy (CA) of MFNN is measured by $(CL_1, CL_2, \ldots, CL_{r'}) / (CL_1, CL_2, \ldots, CL_r)$, where $CL_1, CL_2, \ldots, CL_{r'}$ indicates the set of correctly identified samples, $CL_1, CL_2, \ldots, CL_r$ is the set of sample in test data, and $r' \leq r$. Precision (P), and Recall (R) are computed using $(CL_1, CL_2, \ldots, CL_{r'}) \cap (CL_1, CL_2, \ldots, CL_r) / (CL_1, CL_2, \ldots, CL_{r'})$, and $(CL_1, CL_2, \ldots, CL_{r'}) \cap$



(a) $\epsilon = 0.1$ (b) $\epsilon = 0.1$ (c) $\epsilon = 0.1$ (d) $\epsilon = 0.1$

Fig. 2. Performance of DT-PPM with privacy budget $\epsilon = 0.1$.

$(CL_1, CL_2, \ldots, CL_r)$ / $(CL_1, CL_2, \ldots, CL_r)$, respectively. F1-Score (FS) is measured using $2 \times (P \times R) / (P + R)$.

III. Operational Design and Complexity

Step 1 initializes probabilities Pm_1, Pm_2, Pm_3, and Pr_1, Pr_2, and Pr_3 for mutation and crossover selection, has $\mathcal{O}(1)$ time complexity. Steps (2-26) perform data partitioning, noise generation, injection, and classifying task, wherein step (3) takes $\mathcal{O}(\eta^2)$, where η is the total number of input records. In steps (4-26), crossover selection probabilities are generated as a vector *cps* and consume $\mathcal{O}(v \times p^2 \times \iota)$ time, where v, ι, and p are number of generations, solution vectors, and input neurons, respectively. Step 27 calculates CA, P, R, and FS with $\mathcal{O}(1)$ complexity. Hence, the total time complexity of DT-PPM is $\mathcal{O}(v \times p^2 \times \eta^2 \times \iota)$.

Algorithm 1: DT-PPM Operational Summary

```
Initialize Pm1, Pm2 = 0.33, Pm3 = 0.34, Pr1, Pr2 = 0.33, Pr3 = 0.34
for i = 1 to n do
    D_i^S = D_i - D_i^N, N_i = Lap(0.ϵ), D_{N_i}^S = D_i^S + N_i
    Initialize synaptic weight neural network with ω
    Evaluate each network using fitness function
    for each solution v do
        Generate mps and cps of ω
        for each solution u do
            Generate r1, r2, r3, r4, and r5 ∈ [1,L]
            if 0 < mps_u < Pm1 then
                Apply DE/rand/2 mutation
            end
            else if Pm1 < mps_u < Pm1 + Pm2 then
                Apply DE/best/2 mutation
            end
            else
                Apply DE/current − to − best/2
            end
            if 0 < cps_u < Pr1 then
                Apply Multi-point crossover
            end
            else if Pr1 < cps_u < Pr1 + Pr2 then
                Apply Heuristic crossover
            end
            else
                Apply Ring crossover
            end
        end
        Compute fitness value for each candidates
        Select ω having best fitness value
        Update ms1, ms2, ms3, mf1, mf2, and mf3
        Update Pm1, Pm2, Pm3, Pr1, Pr2, and Pr3
    end
    Apply best population on test data
end
Calculate CA, P, R, and FS
```

IV. Performance Evaluation and Comparative Analysis

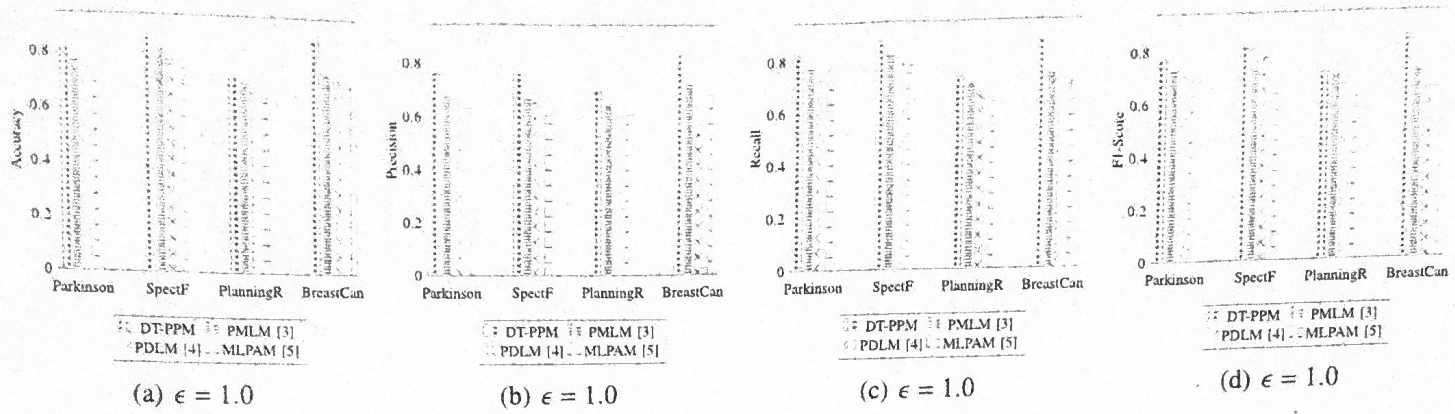
A. Experimental Setup

The simulation experiments are carried out on a server machine consisting of two Intel Xeon Silver 4114 CPU with a 40 core processor and 2.20 GHz clock speed, configured with a 64-bit Ubuntu having 128 GB of main memory in Python 3.7. Parkinson (PK), SpectF (SF), Planning Relax (PR), and Wisconsin Breast Cancer (WBC) distinct datasets are taken from UCI Machine Learning Repository [7]. The experimental results of DT-PPM are compared with state-of-art PMLM [3], PDLM [4], and MLPAM [5] models that are implemented on the same platform.

B. Privacy Budget and Efficiency

The privacy parameters are evaluated for various ϵ values (0.1, 0.2, ..., 1.0), and the results of DT-PPM, including CA, P, R, and FS are compared with PMLM [3], PDLM [4], and MLPAM [5] for $\epsilon = 0.1$ and 1.0, as illustrated in Figs. 2(a)-(d) to 3(a)-(d) relatively. It is examined that the range for CA, P, R, and FS are 72.97% to 87.03%, 69.66% to 84.87%, 72.97% to 87.03%, 71.15% to 85.00% for four datasets respectively. The average for CA, P, R, and FS are 81.98%, 76.50%, 81.98%, and 78.59% across all datasets. DT-PPM outperforms [3], [4], and [5] in all the cases because it mitigates the effect of injecting noise by splitting data into sensitive and non-sensitive parts as well as TaDE improves the classification model's learning efficiency.

Table II shows that the maximum improvement of DT-PPM for CA is 15.89% on WBC dataset and the minimum enhancement is found 2.70% on PR dataset. Likewise, for parameter P, the maximal improvement is 27.73% on WBC dataset and the minimal upgradation is found 4.80% on PR dataset. For metric R, maximum improvement is 15.89% compared to [3], [4], and [5] on WBC dataset, whereas the minimum improvement is 2.70% on PR dataset. The highest improvement of 21.44% is attained for metric FS on WBC dataset, while the lowest advancement is 1.63% on PR dataset. DT-PPM outperforms



(a) $\epsilon = 1.0$ (b) $\epsilon = 1.0$ (c) $\epsilon = 1.0$ (d) $\epsilon = 1.0$

Fig. 3. Performance of DT-PPM with privacy budget $\epsilon = 1.0$.

TABLE II
IMPROVEMENT OF PRIVACY PARAMETERS OF DT-PPM OVER PMLM [3], PDLM [4], AND MLPAM [5]

Dataset	Accuracy			Precision			Recall			F1-Score		
	[3]	[4]	[5]	[3]	[4]	[5]	[3]	[4]	[5]	[3]	[4]	[5]
Parkinson	5.13	9.61	10.26	8.41	10.55	12.55	5.13	9.61	10.26	5.03	6.68	7.16
SpectF	5.55	8.79	9.26	9.36	8.41	15.26	5.55	8.79	9.26	10.50	3.39	15.61
Planning Relax	2.70	6.04	8.11	4.80	9.33	8.70	2.70	6.04	8.11	6.54	1.63	14.82
Breast Cancer	12.32	13.98	15.89	12.70	27.73	16.57	12.32	13.98	15.89	14.13	20.31	21.44

state-of-art models [3], [4], and [5] due to its prominent noise addition in data subset and better learning adaptability of the neural network. Moreover, DT-PPM results are lesser or equal than the results of Clean data in all the cases because of noise injection. Table III shows that the reduction ranges for *CA*, *P*, *R*, and *FS* are 0.57% to 10.81%, 0.27% to 3.26%, 0.57% to 10.81%, and 0.34% to 5.24% over four datasets respectively. But still, the results are closer and provide more protection than Clean data.

C. Security

In DT-PPM, PT_{id} transfers data to *CSP*, however, an untrusted entities such as *CSP* and LB_{id} may obtain sensitive information from the collected data. It is required to protect sensitive data with ϵ-differential privacy.

Theorem 2: In the proposed model, $PT_1, PT_2, \ldots, PT_n$ centers are secured with ϵ differential privacy.

Proof: Let $\hat{R}_1, \hat{R}_2, \ldots, \hat{R}_n$ be n mechanisms, where each mechanism $\hat{R}_i$ $(i \in [1, n])$ provides ϵ differential privacy. Let $D_1, D_2, \ldots, D_n$ be n arbitrary disjoint data sets of the input domain D. For a new mechanism $\hat{R}$, the sequence of $\hat{R}(\hat{R}_1(D_1), \hat{R}_2(D_2), \ldots, \hat{R}_n(D_n))$ provides $\sum_{i=1}^{n} \epsilon$-differential privacy. PT_{id} produces the Laplace noise N_i according to sensitivity Δf, ϵ and injects the generated noise into D_1, $D_2, \ldots, D_n$, respectively, to satisfy $Pb[\hat{R}(D) = \Gamma] \leq exp(\epsilon) \times Pb[\hat{R}(D') = \Gamma]$. According to sequence composition, it is ensured that the mechanism $\hat{R}$ provides privacy to the data; therefore, an unauthorized users or any attacker are not able to access sensitive data from *CSP* or other parties in DT-PPM ■

V. CONCLUSION

A novel privacy-preserving model is proposed that protects outsourced data generated by pathology centers and provides secure data sharing in the cloud environment by partitioning, injecting noise, and classifying the data efficiently. The introduced data partition and noise injection mechanisms protect the data and reduce the overall computation time of the entire process. To improve neural networks' learning efficiency, the TaDE algorithm is developed. The performance evaluation and security analysis validated that DT-PPM is protective, effective, and optimal over existing comparative approaches.

TABLE III
REDUCTION OF VALUES OF DT-PPM OVER CLEAN DATA

Dataset	Accuracy	Precision	Recall	F1-Score
Parkinson	5.12	0.27	5.12	0.34
SpectF	1.85	3.26	1.85	2.66
Planning Relax	10.81	0.53	10.81	5.24
Breast Cancer	0.57	1.55	0.57	1.02

REFERENCES

[1] I. Gupta, A. K. Singh, C.-N. Lee, and R. Buyya, "Secure data storage and sharing techniques for data protection in cloud environments: A systematic review, analysis, and future directions," *IEEE Access*, vol. 10, pp. 71247–71277, 2022.

[2] R. Gupta, D. Saxena, I. Gupta, A. Makkar, and A. K. Singh, "Quantum machine learning driven malicious user prediction for cloud network communications," *IEEE Netw. Lett.*, early access, Aug. 23, 2022, doi: 10.1109/LNET.2022.3200724.

[3] P. Li, T. Li, H. Ye, J. Li, X. Chen, and Y. Xiang, "Privacy-preserving machine learning with multiple data providers," *Future Gener. Comput. Syst.*, vol. 87, pp. 341–350, Oct. 2018.

[4] X. Ma, J. Ma, H. Li, Q. Jiang, and S. Gao, "PDLM: Privacy-preserving deep learning model on cloud with multiple keys," *IEEE Trans. Services Comput.*, vol. 14, no. 4, pp. 1251–1263, Jul./Aug. 2021.

[5] I. Gupta, R. Gupta, A. K. Singh, and R. Buyya, "MLPAM: A machine learning and probabilistic analysis based model for preserving security and privacy in cloud environment," *IEEE Syst. J.*, vol. 15, no. 3, pp. 4248–4259, Sep. 2021.

[6] A. Gionis and T. Tassa, "k-Anonymization with minimal loss of information," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 2, pp. 206–219, Feb. 2009.

[7] "UCI machine learning repository," 2022. [Online]. Available: https://archive.ics.uci.edu/ml/index.php