Differential and Access Policy Based Privacy-Preserving Model in Cloud Environment

Rishabh Gupta* and Ashutosh Kumar Singh

Department of Computer Applications, National Institute of Technology, Kurukshetra, Haryana, India
E-mail: rishabhgpt66@gmail.com; ashutosh@nitkkr.ac.in
**Corresponding Author*

Received 15 February 2021; Accepted 21 November 2021;
Publication 12 February 2022

Abstract

Cloud computing has multiple benefits in terms of minimum cost, maximum efficiency, and high scalability, which prompts shifting a large amount of data from the local machine to the cloud environment for storage, computation, and data sharing among various parties stakeholders. However, owners do not fully trust the cloud platform operated by a third party. Therefore, security and privacy emerge as critical issues while sharing data among different parties. In this paper, a novel privacy-preserving model is proposed by utilizing encryption, differential privacy, and machine learning approaches. It facilitates data owners to share their data securely in the cloud environment. The model defines access policy and communication protocol among the involved untrusted parties for data processing and privacy preservation. The proposed model is evaluated by executing experiments using distinct datasets. The achieved results reveal that the proposed model provides high accuracy,

Journal of Web Engineering, Vol. 21_3, 609–632.
doi: 10.13052/jwe1540-9589.2132

precision, recall, and f1-score up to 98%, 98%, 97%, and 97%, respectively, over the state of the art methods.

Keywords: Cloud computing, differential privacy, machine learning, privacy-preserving, access control.

1 Introduction

Commercial organizations and individuals have shifted their local data to the cloud platform to avail of seamless benefits such as large storage capacity, computational, and flexible accessibility [1]. These individuals and organizations are relieved from local data management and maintenance responsibility after outsourcing their data [2]. But, once data is outsourced to the cloud for storage and computation purposes, owners lose control of their data because it is managed by a third party [3, 4]. The cloud may allow access of the outsourced data to other entities for business benefits [5]. Therefore, it becomes a major challenge for the cloud service provider to establish trust among owners for data privacy. The owners encrypt their data before transferring it to the cloud platform by utilizing traditional techniques such as homomorphic cryptographic techniques. These techniques are considered inefficient because of difficulty in performing the computation over the encrypted data [6, 7]. Moreover, the stored and analyzed data must be shared with the various authorized entities for several purposes. It cannot ensure that the receiving entities will not distribute the data to other entities after they have obtained it [8]. Therefore, it is necessary to preserve the privacy of data among involved entities. To address the aforementioned challenges, a potential access control method that preserves data privacy is required. In this regard, a novel Differential and Access policy-based Privacy-preserving Model (DAPM) is proposed, which protects the data through privacy-preserving data storage, analysis, and secure sharing in the cloud environment. Figure 1 shows a bird-eye view of the proposed work and highlights our consecutive contributions in the cloud environment to protect cloud data and classify tasks. The main contributions of DAPM are described as follows:

1. An access policy is designed to improve data protection among all the engrossed entities that are deemed untrusted.
2. DAPM facilitates multiple data owners to share their data by encrypting it with a separate key to protect the data from leakage.

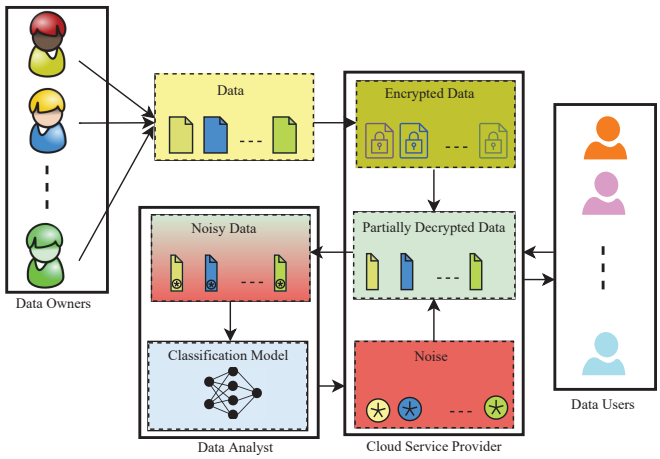


Figure 1 Bird eye view of the proposed work.

3. DAPM transforms encrypted data into noised data at the cloud platform to enhance the computation's efficiency and accuracy.
4. A series of experiments are performed over the diverse datasets to evaluate the validity of the proposed model. Besides, the comparisons are made among the different a) datasets, b) classifiers, and c) preprocessed data using ϵ-differential privacy and with the state of the artworks to prove the superiority of DAPM.

Organization: Section 2 entails related work, and an overview of the proposed model DAPM is presented in Section 3. The data preservation mechanism is described in Section 4, followed by the machine learning model in Section 5. The experimental results of DAPM are presented in Section 6, and the proposed work is summarized in Section 7. The notations list with their descriptions is shown in Table 1.

2 Related Work

2.1 Security Based on Ciphertext-Policy Attribute-Based Encryption (CP-ABE)

Wang et al. [9] proposed a File Hierarchy Ciphertext-Policy Attribute-Based Encryption (FH-CP-ABE) scheme to share the hierarchical files in cloud computing. The different access structures of files were integrated into a

Table 1 List of symbols

Notation	Definition
DO_{id}	Data Owners
CSP	Cloud Service Provider
DU_{id}	Data Users
D_i	Actual data
D_i^E	Encrypted data
N_i	Noise
CM	Classification Model
DA	Data Analyst
CA	Classification Accuracy
PB_i	Public key
SK_j	Secret key
TK_i	Transformation key
P	Precision
D_i^N	Noise-added data
$\hat{D}_i^N$	Preprocessed data
R	Recall
$\hat{D}_t^N$	Training data
$\hat{D}_{t'}^N$	Testing data
FS	F1-Score
E_T	Encryption time
D_T	Decryption time
n^*	Training objects
n^{**}	Testing objects

single access structure that can be used to encrypt the files in the same hierarchical structure. But, the computation cost of this scheme increased dramatically, even the ciphertexts were integrated, and the common attributes were computed only once. To achieve a fair key reconstruction, Liu et al. [10] proposed a data access control scheme for cloud storage in which none of the users send their shares, and no one can access shared data. This scheme minimized the computation and communication cost, but user authentication did not perform effectively. To reduce the computation cost of users and the high decryption cost increases with the complexity of access policy, a multi-authority CP-ABE scheme is presented in [11]. The updated keys of the access policy are created and sent to the cloud for updation. However, this scheme depends on untrusted cloud servers to update keys. Zhang et al. [12] proposed a Hidden access Policy CP-ABE scheme to verify the authorized users and ensure data confidentiality. The private key with constant size encrypts the data, and the decryption process requires four pairing computations. The transmission, as well as storage costs, are reduced in this scheme.

It supports only the "AND" policy; therefore, it is considered a weak security scheme. Li et al. [13] proposed an efficient outsourcing policy updating the ciphertext-policy ABE (CP-ABE) scheme based on the linear secret-sharing schemes (LSSS) matrix access structure with improved the efficiency of the policy and file updating dynamically in cloud computing. The storage, communication cost of owners, and computation cost of the proxy cloud service provider were reduced. It resisted the selected plaintext attacks, but the file updating cost is high.

2.2 Privacy-Preserving Based on Machine Learning

Yuan et al. [14] proposed a secure, efficient, and accurate multiparty Back-Propagation Neural (BPN) network-based scheme which provided privacy preservation for more than two parties collaborative BPN network learning over arbitrarily partitioned data. They adopted a "doubly homomorphic" encryption algorithm to perform the operations over ciphertexts. However, they concentrated more on the improvement of data processing without considering the efficiency of the algorithm. To learn visual classifiers securely over distributed private data, Yonetani et al. [15] proposed a privacy-preserving mechanism based on double-permitted homomorphic encryption (DPHE) scheme that enabled multiparty protected scalar product. It reduced the computational cost for high-dimensional classifiers. But, DPHE supports either addition or multiplication at a particular time. Aono et al. [16] used additively homomorphic encryption to protect the gradients against the curious server. The same accuracy was achieved corresponding deep learning system, i.e., asynchronous stochastic gradient descent (ASGD) trained over the joint dataset of all participants. But, the secret-key of owners decrypts the updated parameters, and their model does not provide the privacy of parameters. To provide the privacy-preserving classification service for users and a classifier owner to delegate a remote server, a secure outsourcing scheme is presented in [17]. However, the outsourced model sharing scheme only supports a single-party setting. Li et al. [18] proposed a data protection scheme that preserves the privacy of Naive Bayes learning over data contributed by multiple providers. The ϵ-differential privacy was used for data protection. But this scheme does not satisfy the differential privacy in the local setting and preserves individual privacy with encryption techniques. Ma et al. [19] proposed a privacy-preserving deep learning model, namely PDLM, to train the model over the multi-key encrypted data. A privacy-preserving calculation toolkit based on stochastic gradient descent (SGD) was

adopted to accomplish the training task in a privacy-preserving manner. The model reduced the storage overhead, but the classification accuracy is less as well as the computation cost is high. A Privacy-preserving Machine Learning with Multiple data provider (PMLM) scheme with improved computational efficiency and data analysis is proposed by Li et al. [20]. The public-key encryption with a double decryption algorithm (DD-PKE) and ϵ-differential privacy was used for data privacy. But the scheme suffers from less accuracy as well as less data sharing. A privacy-preserving outsourced classification in cloud computing (POCC) framework was introduced in [21] which protects the confidentiality of sensitive data using a fully homomorphic encryption proxy technique. However, several interactions among evaluators and storage servers increased the computational and communication cost.

3 Proposed Model

The proposed model (Figure 2) involves four entities: Data Owners (DO_{id}), Cloud Service Provider (CSP), Data Users (DU_{id}), and Data Analyst (DA) which are described with their intercommunication and essential information flow as follows:

(1) DO_{id}: An entity that produces data and uses the services of CSP for storage and computation. DO_{id} encrypts data before sending it to the cloud to maintain privacy. However, DO_{id} is not considered a trusted entity. It does not leak its own data but may reveal the other owner's data.

(2) CSP: It is an entity that provides the storage, computation, and data sharing facilities to DO_{id}, DA, or DU_{id} after collecting the encrypted data. CSP transforms ciphertext provided by DO_{id} into partially decrypted data and adds noise to it by using ϵ-differential privacy. CSP acts as a bridge that connects DO_{id}, DU_{id}, and DA. CSP strictly follows the protocols, it is not a fully trusted entity due to its curiosity to learn the information.

(3) DU_{id}: An entity receives data from CSP after sending a request. DU_{id} performs decryption over data received from CSP and acquires useful information. DU_{id} is considered an untrusted entity.

(4) DA: An entity that receives the noisy data from CSP as per the request. It provides the classification service through a classification model (CM). It trains CM using machine learning algorithms over received data and gets classified data from CM. The achieved results

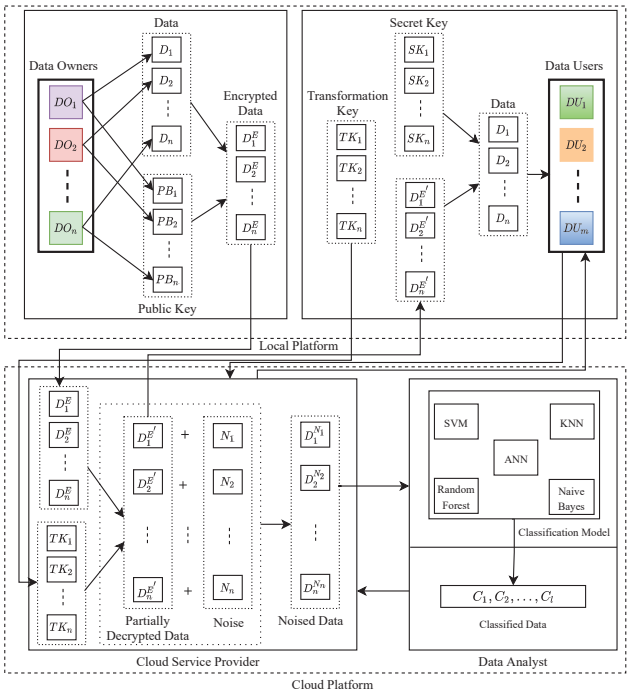


Figure 2 DAPM architecture.

are shared among DO_{id} or DU_{id} through CSP. In the proposed model, DA is treated as an untrusted entity.

Let the data owners $\mathbb{DO} = \{DO_1, DO_2, \ldots, DO_n\}$ own data $\mathbb{D} = \{D_1, D_2, \ldots, D_n\}$, while the data object D_i belongs to $\mathbb{D}$ may be independent and of different sizes. It is essential that $\mathbb{DO}$ must share $\mathbb{D}$ among the other parties such as CSP, DA, and several data users $\mathbb{DU} = \{DU_1, DU_2, \ldots, DU_m\}$ for data usage, storage, and computation purposes. Each $\{DO_1, DO_2, \ldots, DO_n\}$ has public keys $\mathbb{PB} = \{PB_1, PB_2, \ldots, PB_n\}$ respectively. $DO_1, DO_2, \ldots, DO_n$ encrypt their data $D_1, D_2, \ldots, D_n$ by using the encryption technique and the public key $PB_1, PB_2, \ldots, PB_n$ to ensure data privacy and acquire the encrypted data $\mathbb{D}^{\mathbb{E}} = \{D_1^E, D_2^E, \ldots, D_n^E\}$. As an encryption technique, DAPM uses the CP-ABE scheme for data $D_1, D_2, \ldots, D_n$ encryption as it is a suitable technique for data protection and fine-grained data access handling. It is also an efficient and secured encryption technique that utilizes the attributes of $DU_1, DU_2, \ldots, DU_m$ without loss of data privacy at encryption/decryption time, and data owners themselves

determine the access control policy with the data users' attributes [22]. DO_1, $DO_2, \ldots, DO_n$ transfers $D_1^E, D_2^E, \ldots, D_n^E$ to CSP for storing, computing, and sharing data. CSP has transformation keys $\mathbb{TK} = \{TK_1, TK_2, \ldots, TK_n\}$ that have been obtained from $DU_1, DU_2, \ldots, DU_m$. CSP transforms the stored $D_1^E, D_2^E, \ldots, D_n^E$ into partially decrypted data $\mathbb{D}^{\mathbb{E}'} = \{D_1^{E'}, D_2^{E'}, \ldots, D_n^{E'}\}$ using $TK_1, TK_2, \ldots, TK_n$. CSP sends requested data $D_1^{E'}, D_2^{E'}, \ldots, D_n^{E'}$ to data users $DU_1, DU_2, \ldots, DU_m$ for utilization as per the demand. Each $\{DU_1, DU_2, \ldots, DU_m\}$ has secret keys $\mathbb{SK} = \{SK_1, SK_2, \ldots, SK_m\}$, which are used to decrypt $D_1^{E'}, D_2^{E'}, \ldots, D_n^{E'}$ to obtain the plain data $D_1, D_2, \ldots, D_n$. CSP converts the $D_1^{E'}, D_2^{E'}, \ldots, D_n^{E'}$ into noised data $\mathbb{D}^{\mathbb{N}} = \{D_1^{N_1}, D_2^{N_2}, \ldots, D_n^{N_n}\}$ by adding the noise to it. The resulted data $D_1^{N_1}, D_2^{N_2}, \ldots, D_n^{N_n}$ is sent to DA for classification. $DO_1, DO_2, \ldots, DO_n$ or $DU_1, DU_2, \ldots, DU_m$ can make the query via CSP that passes the query to DA to perform the classification tasks over it and obtain the results from CM through DA. CSP delivers the acquired results to the corresponding entities $DO_1, DO_2, \ldots, DO_n$ or $DU_1, DU_2, \ldots, DU_m$. The operational summary of the proposed model is given in Algorithm 1.

Algorithm 1: DAPM operational summary

Input: Actual data D, Public key PB, Secret key SK, Transformation key TK, Access Policy AP
Output: CA, P, R, and FS

Initialize data $D := \{D_1, D_2, \ldots, D_n\}$, $PB := \{PB_1, PB_2, \ldots, PB_n\}$, $TK := \{TK_1, TK_2, \ldots, TK_n\}$, $SK := \{SK_1, SK_2, \ldots, SK_m\}$, $AP := \{AP_1, AP_2, \ldots, AP_n\}$
for $i = 1, 2, \ldots, n$ **do**
  for $j = 1, 2, \ldots, m$ **do**
    Encrypt D_i using PB_i keys
    Partially decrypt data D_i^E using TK_i keys
    Add the noise N_i into data $D_i^{E'}$
    Decrypt data $D_i^{E'}$ using SK_j keys and AP_i
    Perform the classification over $\hat{D}_i^N$
  end for
end for
$CA = (\#Correctly\ classified\ sample\ /\ \#test\ sample) * 100$
$P = (TP)/(TP + FP)$
$R = (TP)/(TP + FN)$
$FS = 2 * (P * R)/(P + R)$

4 Data Preservation

Let each data owner DO_i owns data D_i, public key PB_i, and access policy AP_i over the attributes. DO_i encrypts data D_i with their PB_i by applying Equation (1):

$$D_i^E = \{AP_i, \bar{C} = D_i \times e(g,g)^{\alpha s_i}, C = h^{s_i}, \\ \forall y \in Y : C_y = g^{q_y}, C_y^{'} = H(att(y))^{q_y}\} \tag{1}$$

Where AP_i is access policy to get the encrypted data D_i^E, e is bilinear map assigned by e: $G_0 \times G_0 \rightarrow G_T$, g is the generator of G_0 and G_0 is the bilinear group of prime order p. The random values s, r, and z are used to measure the shared value q_y for each attribute ($att(y)$) in AP_i, whereas α, β $\in Z_p$ are random exponents and $h = g^{\beta}$. A hash function H is used to map the attributes. The security parameters $\bar{C}$, C, C_y, and $C_y^{'}$ are calculated for $att(y)$ in AP_i. DO_i do not fully trust CSP for data access control, and DU_1, DU_2, ..., DU_m has distinct decryption rights according to their attributes. DO_1, DO_2, ..., DO_n transfer the encrypted data D_1^E, D_2^E, ..., D_n^E including the access structure AP_i to CSP. D_1^E, D_2^E, ..., D_n^E which is partially decrypted by CSP using Equation (2), and obtained results $D_1^{E^{'}}$, $D_2^{E^{'}}$, ..., $D_n^{E^{'}}$ are sent to DU_1, DU_2, ..., DU_m. They decrypt $D_1^{E^{'}}$, $D_2^{E^{'}}$, ..., $D_n^{E^{'}}$ using corresponding keys $SK_1, SK_2, \ldots, SK_m$ after matching their attributes with the access policy determined by $DO_1, DO_2, \ldots, DO_n$ and get the actual data $D_1, D_2, \ldots, D_n$ by applying Equation (3).

$$D_i^{E^{'}} = \frac{D_i^E \cdot (e(g,g)^{\frac{\beta s_i}{z}}) \cdot (e(g,g)^{\alpha r s_i})}{e(g,g)^{\alpha r s_i}} \tag{2}$$

$$D_i = \frac{D_i^{E^{'}} \cdot (e(h^{s_i}, g^{\frac{(\alpha+r)}{\beta}})}{e(g,g)^{r s_i}} \tag{3}$$

CSP transfers encrypted data $\mathbb{D}^{\mathbb{E}} = \{D_1^E, D_2^E, \ldots, D_n^E\}$ into partially decrypted data $\mathbb{D}^{\mathbb{E}^{'}} = \{D_1^{E^{'}}, D_2^{E^{'}}, \ldots, D_n^{E^{'}}\}$ using the transformation keys $TK = \{TK_1, TK_2, \ldots, TK_n\}$. To increase the accuracy and efficiency of computations while protecting data, the partially encrypted data $\mathbb{D}^{\mathbb{E}^{'}} = \{D_1^{E^{'}}, D_2^{E^{'}}, \ldots, D_n^{E^{'}}\}$ is transformed into noised data $\mathbb{D}^{\mathbb{N}} = \{D_1^{N_1}, D_2^{N_2}, \ldots, D_n^{N_n}\}$ using ϵ-differential privacy [23]. For this, CSP produces the noise vector $\mathbb{N} = \{N_1, N_2, \ldots, N_n\}$ using the probability density and distribution

Algorithm 2: Data Encryption/Decryption and Noise Addition

Input: D, PB, TK, SK, AP

Output: D^E, $D^{E'}$

for $i = 1, 2, \ldots, n$ **do**
- Generate keys (k_i) from group element GT
- Select random elements s_i from $\in ZR$
- Calculate secret shared q_y, $\forall\, t$ in AP_i
- Compute $C_i = h^{s_i}$, $\bar{C}_i = D_i \times e(g,g)^{\alpha s_i}$
- Calculate $C_{yi} = g^{q_y}$, $C'_{yi} = H(att(y))^{q_y}$; $\forall\, t \in AP_i$
- $D_i^E = \{C_i, \bar{C}_i, \forall y \in Y: C_{yi}, C'_{yi}\}$
- Compute $\ddot{C}_i = h^{s_i}$, $\bar{\ddot{C}}_i = D_i^E \times e(g,g)^{\alpha s_i}$
- Calculate $\ddot{C}_{ydi} = g^{q_{yd}}$, $\ddot{C}'_{ypdi} = H(att(y))^{q_{yd}}$
- $D_i^{E'} = \{\ddot{C}_i, \bar{\ddot{C}}_i, \ddot{C}_{ydi}, \ddot{C}'_{ypdi}\}$
- $D_i^{N_i} = D_i^{E'} + N_i$
- Compute $\check{C}_i = h^{s_i}$, $\bar{\check{C}}_i = D_i^{N_i} \times e(g,g)^{\alpha s_i}$
- Calculate $\check{C}_{ydi} = g^{q_{yd}}$, $\check{C}'_{yfdi} = H(att(y))^{q_{yd}}$
- $D_i = \{\check{C}_i, \bar{\check{C}}_i, \check{C}_{ydi}, \check{C}'_{yfdi}\}$

end for

return D^E, $D^{E'}$

Laplace, Gaussian, and Random function by applying Equation (4), (5), and (6), respectively.

$$\mathbb{N} = \frac{1}{2s'} \cdot exp(\frac{-|rn|}{s'}) \tag{4}$$

$$\mathbb{N} = \frac{1}{\sqrt{2\pi s'^2}} \cdot exp^{\frac{-(rn-\mu)^2}{2s'^2}} \tag{5}$$

$$\mathbb{N} = \frac{4}{5} \cdot exp(\frac{-(s')^2}{2}) \tag{6}$$

where $\mathbb{N}$ is a noise vector, s' is scale parameter, and rn is random noise drawn from the distribution with scale s'. The created noise $\mathbb{N} = \{N_1, N_2, \ldots, N_n\}$ is added to corresponding $\mathbb{D}^{\mathbb{E}'} = \{D_1^{E'}, D_2^{E'}, \ldots, D_n^{E'}\}$ as $D_i^N = D_i^{E'} + N_i$ where $i \in [1, n]$, and the resulted data $\mathbb{D}^{\mathbb{N}} = \{D_1^{N_1}, D_2^{N_2}, \ldots, D_n^{N_n}\}$ is transferred to DA. Algorithm 2 defines the steps for data

encryption/decryption and noise addition. In this algorithm, steps 2 to 4 find the security parameters required to encrypt the data. Steps 5 to 7 are applied for data encryption. Encrypted data is partially decrypted using steps 8 to 10. Noise is injected into partially decrypted data through step 11. By using steps 12 to 14, partially decrypted data is fully decrypted to allow access to actual data.

5 Data Classification

DA preprocesses the data $\mathbb{D}^{\mathbb{N}} = \{D_1^{N_1}, D_2^{N_2}, \ldots, D_n^{N_n}\}$ by using the normalization function to achieve the preprocessed data $\hat{\mathbb{D}}^{\mathbb{N}} = \{\hat{D}_1^{N_1}, \hat{D}_2^{N_2}, \ldots, \hat{D}_n^{N_n}\}$ by applying Eq (7), where T_A is the training sample with A attributes, μ, and σ are the mean and the standard deviation of the training sample, respectively.

$$\hat{\mathbb{D}}_i = \frac{(T_A - \mu)}{\sigma} \tag{7}$$

It is well-known that $\hat{\mathbb{D}}^{\mathbb{N}} = \{\hat{D}_1^{N_1}, \hat{D}_2^{N_2}, \ldots, \hat{D}_n^{N_n}\}$ belongs to $l^* \leq n$ class labels $\mathbb{C} = \{C_1, C_2, \ldots, C_{l^*}\}$ where $\cup_{i=1}^{l^*} \mathbb{C}_i = \mathbb{D}$ and $\mathbb{C}_i \cap \mathbb{C}_j = \Phi, \forall_{i,j} = 1, 2, \ldots, l^* \wedge i \neq j$. The data $\hat{\mathbb{D}}^{\mathbb{N}} = \{\hat{D}_1^{N_1}, \hat{D}_2^{N_2}, \ldots, \hat{D}_n^{N_n}\}$ is divided into training data $\hat{\mathbb{D}}_t^N = \{\hat{D}_{t,1}^{N_1}, \hat{D}_{t,2}^{N_2}, \ldots, \hat{D}_{t,n^*}^{N_{n^*}}\}$ and testing data $\hat{\mathbb{D}}_{t'}^N = \{\hat{D}_{t',1}^{N_1}, \hat{D}_{t',2}^{N_2}, \ldots, \hat{D}_{t',n^{**}}^{N_{n^{**}}}\}$. The CM is trained using the training data $\{\hat{D}_{t,1}^{N_1}, \hat{D}_{t,2}^{N_2}, \ldots, \hat{D}_{t,n^*}^{N_{n^*}}\}$ along with machine learning algorithms while the accuracy of CM is evaluated by testing data $\hat{D}_{t',1}^{N_1}, \hat{D}_{t',2}^{N_2}, \ldots, \hat{D}_{t',n^{**}}^{N_{n^{**}}}$. The data objects $\hat{D}_{t',1}^{N_1}, \hat{D}_{t',2}^{N_2}, \ldots, \hat{D}_{t',n^{**}}^{N_{n^{**}}}$ are provided to CM to assign class labels during the testing process. Thus, CM examines $\hat{D}_{t',1}^{N_1}, \hat{D}_{t',2}^{N_2}, \ldots, \hat{D}_{t',n^{**}}^{N_{n^{**}}}$ and gives the output as class label vector $\mathbb{CL} = \{CL_1, CL_2, \ldots, CL_{n^{**}}\}$. The Classification Accuracy (CA) of CM is calculated using $CL_1, CL_2, \ldots, CL_n$ by applying Equation (8), where CI indicates the number of correctly classified items and TI indicates the total number of test items.

$$CA = \frac{CI}{TI} \tag{8}$$

The precision (P) and recall (R) are measured using Equations (9) and (10), respectively, while RT indicates the total number of items returned by the

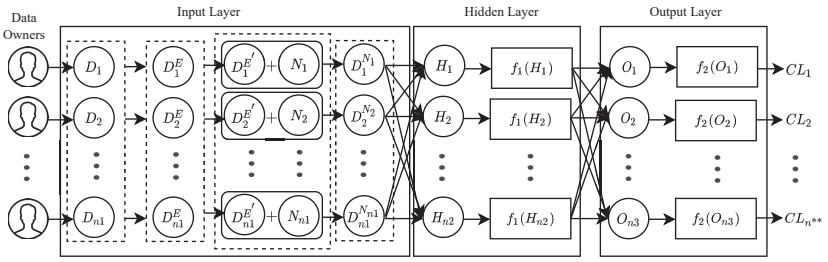


Figure 3 Feed-forward Neural Network Classifier for shared data.

classifier and the total number of relevant items is indicated by IR.

$$P = \frac{CI}{RT} \tag{9}$$

$$R = \frac{CI}{IR} \tag{10}$$

The F1-Score (FS) is calculated using Equation (11). The steps for data D_1, D_2, ..., D_n classification is shown in Figure 3.

$$FS = \frac{2PR}{P + R} \tag{11}$$

In DAPM, the feed-forward neural network classifier consists of three layers one input layer, one hidden layer, and one output layer. From the input layer with $n1$ nodes, the preprocessed data $\hat{D}_1^{N_1}$,$\hat{D}_2^{N_2}$,..., $\hat{D}_{n1}^{N_{n1}}$ is given to the hidden layer with $n2$ nodes represented as $\{\varphi_1\hat{D}_1^{N_1}, \varphi_2\hat{D}_2^{N_2}, \ldots, \varphi_{n2}\hat{D}_{n1}^{N_{n1}} + b_1\}$. The results of the hidden layer are provided to the output layer with $n3$ nodes shown as $\{\varphi_1 h_1, \varphi_2 h_2, \ldots, \varphi_{n_3} h_{n_2} + b_2\}$, where φ is the weight and b_1, b_2 are the bias. The classified results $CL_1, CL_2, \ldots, CL_{n^{**}}$ are obtained from the output layer. Algorithm 3 describes the steps for data classification. In this algorithm, steps 3 to 5 show the efficiency of the SVM classifier. A procedure for the K-NN classifier is given in steps 6 to 8. The Random Forest classifier classifies the data in steps 9 to 11. By using step 12, the Naive Bayes classification is carried out. Subsequently, the neural network operates from steps 13 to 16.

The computational and space complexities are $\mathcal{O}(\max_{1\leq j\leq m}|I_j|)$, $\mathcal{O}(1)$; $\mathcal{O}(n)$, $\mathcal{O}(n)$; $\mathcal{O}((n^*)^3)$, $\mathcal{O}((n^*)^2)$, where I is the total number of attributes and n is the input records for various phases including data encryption and decryption, differential privacy, and data classification of DAPM,

Algorithm 3: Data Classification

Input: Input vector $\hat{D}^N$, weight w, bias b, activation function $f(x)$, tree numbers t_num
Output: unknown class label CL

Initialize input vector $\hat{D}^N = \{\hat{D}_1^{N_1}, \hat{D}_2^{N_2}, \ldots, \hat{D}_n^{N_n}\}$, $\hat{D}_1 = \{(x_1, y_1), (x_2, y_2), \ldots, (x_{\ddot{i}}, y_{\ddot{j}})\}$, w, b
for $i = 1, 2, \ldots, n$ **do**
  $z_i = \sum \hat{D}_i^N \cdot w^T + b$
  $f(z_i) > 0, CL := 1$
  $f(z_i) < 0, CL := 0$
  for Compute Set I contains the minimum sets of k **do**
    distance $d\,(\hat{D}_i, CL)$
  end for
  for $\ddot{t} = 1, 2, \ldots, t\_num$ **do**
    tree_classification($\hat{D}_i^N, CL$)
  end for
  $CL = argmax_y\ P(y) \prod P(\hat{D}_i^N \mid y)$
  for $\ddot{k} = 1, 2$ **do**
    for $\ddot{l} = 1, \ldots, n_{\ddot{k}+1}$ **do**
      $z_{\ddot{l}}^{(\ddot{k}+1)} = \sum_{\hat{t}=1}^{n} \hat{D}_{\hat{t}}^{N(\ddot{k})} \cdot w_{\hat{t}}^{(\ddot{k})} + b_{\ddot{l}}^{(\ddot{k})}$
      $CL = f(z_{\ddot{l}}^{(\ddot{k}+1)})$
    end for
  end for
  return CL
end for

respectively. DAPM complexity analysis implies that the aid of endurable time and space protects the data, which establishes its potency.

6 Performance Evaluation

6.1 Experimental Setup

A series of experiments have been performed using machine learning algorithms over four separate datasets EEG Eye State (EES), Gender Voice (GV), Seeds, and Vehicle Silhouettes (VS) with 15, 21, 7, 18 attributes and 14980, 3168, 210, 946 instances, which have been taken from the UCI Machine Learning Repository to train *CM*. The five separate classifiers, i.e., Support Vector Machine (SVM), K-nearest neighbor (K-NN), Random Forest, Naive

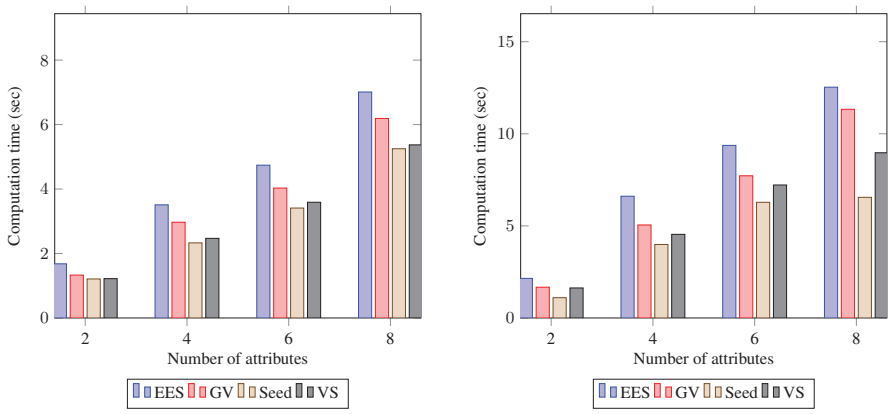


Figure 4 Computation time (sec) for various datasets: (a) E_T and (b) D_T.

Bayes, and Artificial Neural Network (ANN), have been used to train *CM* over training data. These experiments are carried out on a system equipped with Intel (R) Core (TM) i5-4210U CPU @ 1.70GHz clock speed along with 8 GB of main memory using Python 2.7.15.

6.2 Encryption/Decryption Time

Figures 4(a) and (b) show computation time for encryption and decryption processes over different datasets, respectively. It is found that encryption and decryption time costs increase linearly concerning the number of attributes associated with access policy. Besides, a comparison in terms of encryption time (E_T) and decryption time (D_T) has been carried out over the various datasets, which increases concerning the instance of datasets. The descending order of E_T and D_T for all 2 to 8 number of attributes are EES, GV, VS, Seed, respectively.

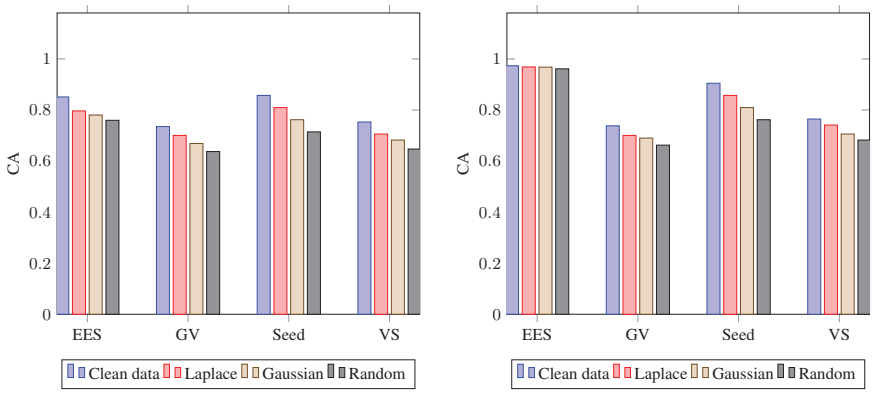
6.3 Classification Parameters

The 9/10 data of complete datasets are used for training data, while the remaining for testing data. The training is carried out on clean as well as noisy data. To produce the noised data, the Laplace, Gaussian, and Randomly noise generated mechanisms are used. The results of noised data are compared against clean data to find the differences. In addition, a distinction is made

Table 2 Reduction in the values of accuracy, precision, recall, and f1-score of Laplace (L), Gaussian (G), Random(R) noised data in comparison to the values on clean data

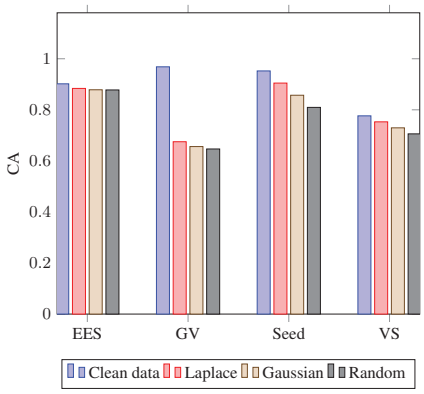
Dataset	Classifier	% decrement in the value of parameters											
		Accuracy			Precision			Recall			F1-Score		
		L	G	R	L	G	R	L	G	R	L	G	R
EES	SVM	5.48	7.08	9.15	5.97	6.40	8.51	0.77	2.26	15.60	4.06	5.45	12.60
	KNN	0.46	0.53	1.20	0.78	0.84	1.38	0.05	0.56	1.07	0.41	0.69	1.22
	RF	1.80	2.33	2.40	3.59	5.29	5.37	1.09	1.15	1.19	2.28	2.90	3.19
	NB	2.74	2.87	4.27	2.13	2.20	3.36	0.55	0.67	0.84	1.57	1.81	2.60
	ANN	0.68	1.07	1.15	0.14	0.14	0.90	3.17	4.57	5.17	1.36	1.36	2.68
GV	SVM	3.47	6.63	9.78	4.68	8.81	9.89	2.04	3.37	5.04	3.71	6.88	8.48
	KNN	3.78	4.81	7.57	5.13	5.54	7.35	3.46	4.38	10.04	4.34	4.99	8.68
	RF	29.34	31.23	32.18	31.80	34.77	36.95	25.53	26.26	28.70	28.70	32.02	32.94
	NB	25.24	27.45	28.08	15.86	19.53	22.69	37.29	44.54	52.31	23.58	33.95	40.76
	ANN	0.81	1.40	1.63	12.70	13.08	13.69	12.93	13.25	15.14	12.93	13.01	14.42
Seed	SVM	4.76	9.52	14.29	9.90	17.62	21.43	9.52	15.55	20.31	8.03	15.47	18.88
	KNN	4.76	9.52	14.28	8.33	8.60	11.87	6.02	8.79	15.67	6.99	8.72	15.34
	RF	4.76	9.52	14.28	1.79	6.66	12.41	5.82	10.37	15.13	1.61	7.03	11.05
	NB	5.36	14.28	19.05	5.00	15.19	16.51	0.47	14.48	18.65	3.33	13.91	16.77
	ANN	0.54	9.76	10.64	0.29	1.12	7.12	14.29	14.29	23.81	3.09	3.83	11.31
VS	SVM	4.71	7.06	10.59	1.10	8.04	13.50	3.49	9.07	11.89	2.59	8.00	11.56
	KNN	2.36	5.89	8.24	3.14	4.80	5.50	4.60	6.31	6.41	3.38	5.28	5.88
	RF	2.35	4.70	7.06	0.21	0.81	4.38	0.20	2.18	3.91	0.47	1.08	3.95
	NB	2.36	5.89	15.30	3.94	5.96	16.75	1.38	4.72	9.97	2.21	5.82	16.23
	ANN	6.93	7.85	8.41	2.30	7.01	7.33	8.23	9.41	20.00	5.47	6.15	11.44

among the Laplace, Gaussian, and Random noised data to find the superior one. The outputs of *CM* are computed and the Classification Accuracy (*CA)*, Precision (*P*), Recall (*R*), and F1-Score (*FS*) are calculated using test results. Figs. 5 to 8 show the results including *CA*, *P*, *R*, and *FS* respectively, which are achieved by *CM* of DAPM over Clean, Laplace noised, Gaussian noised, and Random noised data. The comparison among SVM, K-NN, Random Forest, Naive Bayes, and ANN classifiers are also shown for EES, GV, Seeds, and VS datasets, respectively. Due to noise addition, *CA*, *P*, *R*, and *FS* of noised data are less than clean data in the case of all the five classifiers as shown in Table 2. However, *CA*, *P*, *R*, and *FS* are almost equal for noised data and also have more protection compared to clean data. Also, in the case of all five classifiers, out of three noised data, Laplace noise data outperforms the Gaussian noise and the Random noise data. The datasets and classifiers' performance descends in order: Seeds, GV, EES, VS, and Naive Bayes, Random Forest, K-NN, SVM, ANN, respectively. The Seeds dataset outperforms the remaining four data sets for all five classifiers out of four data sets. For *CA*, *P*, *R*, and *FS*, the Random Forest classifier outperforms the other three classifiers. Overall, the Random Forest classifier outperforms the other four classifiers in DAPM due to the use of kernel trick and a large optimum margin interval between separating hyperplanes during classification, which results in better performance.

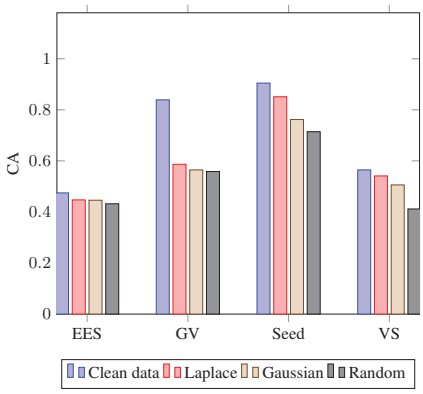


(a) SVM

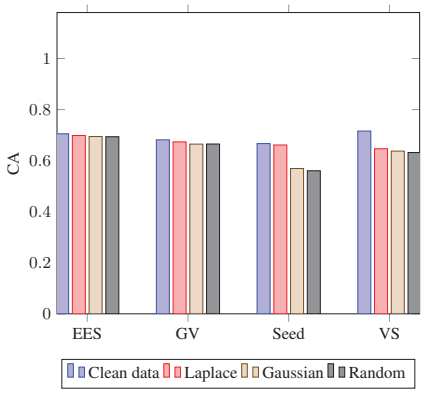
(b) K-NN



(c) Random Forest

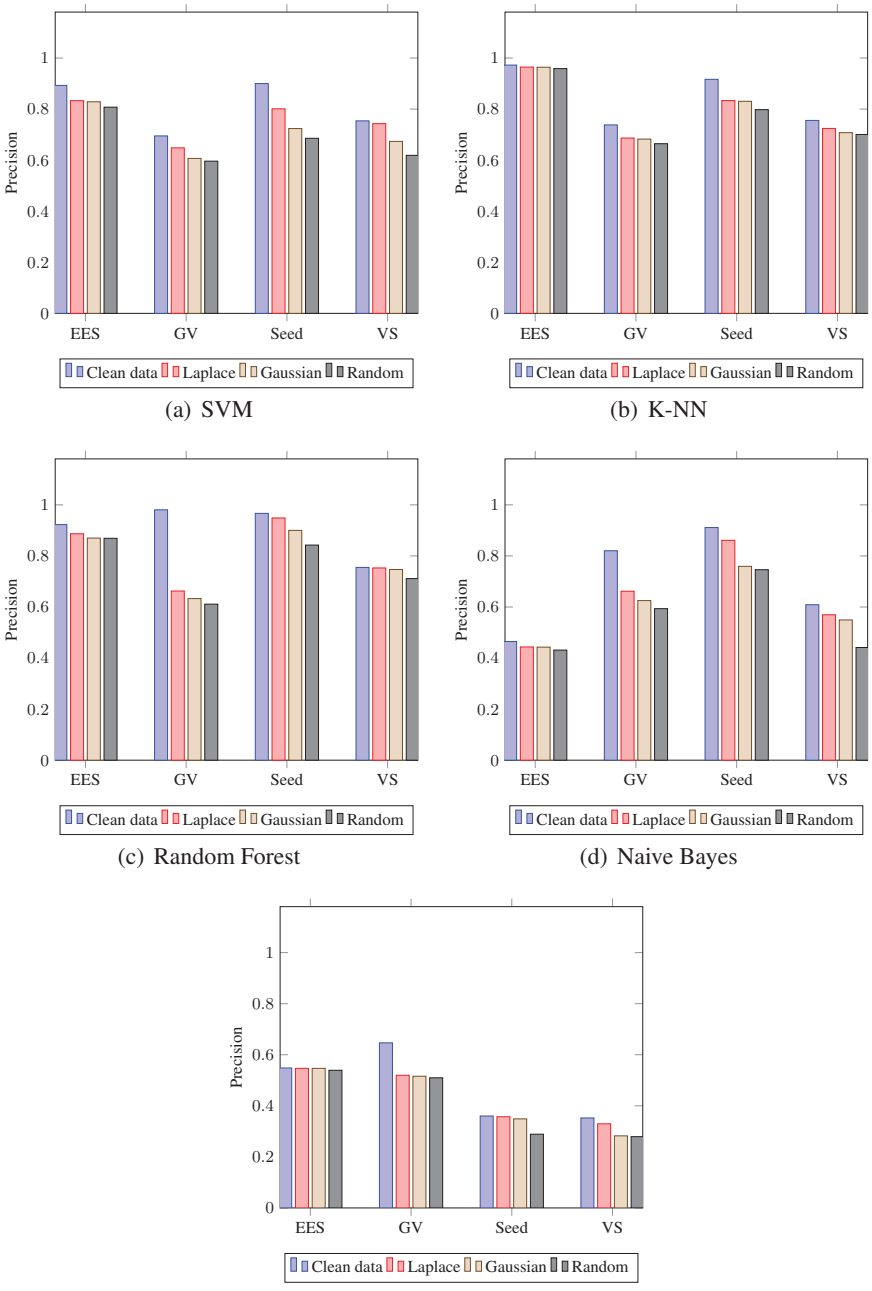


(d) Naive Bayes



(e) ANN Classifier

Figure 5 Accuracy of *CM* for DAPM.

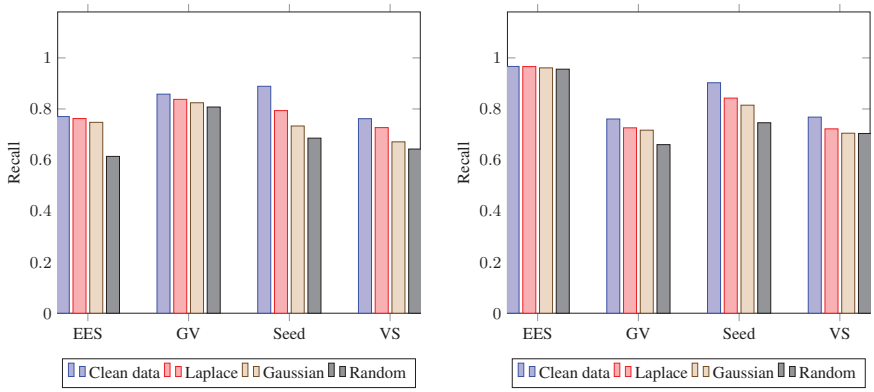


(a) SVM (b) K-NN

(c) Random Forest (d) Naive Bayes

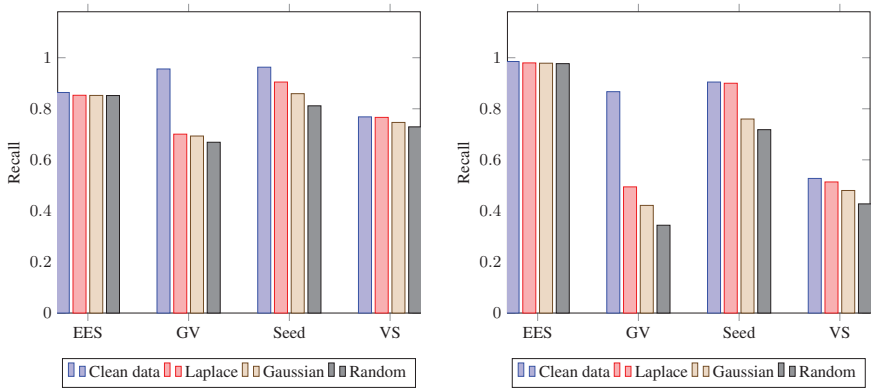
(e) ANN Classifier

Figure 6 Precision of *CM* for DAPM.



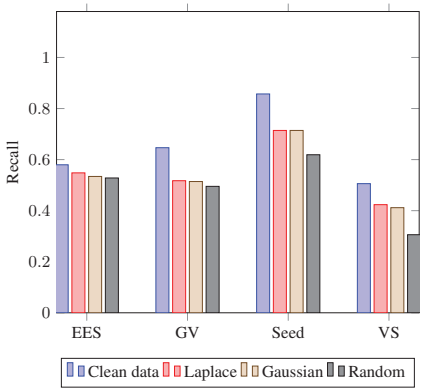
(a) SVM

(b) K-NN



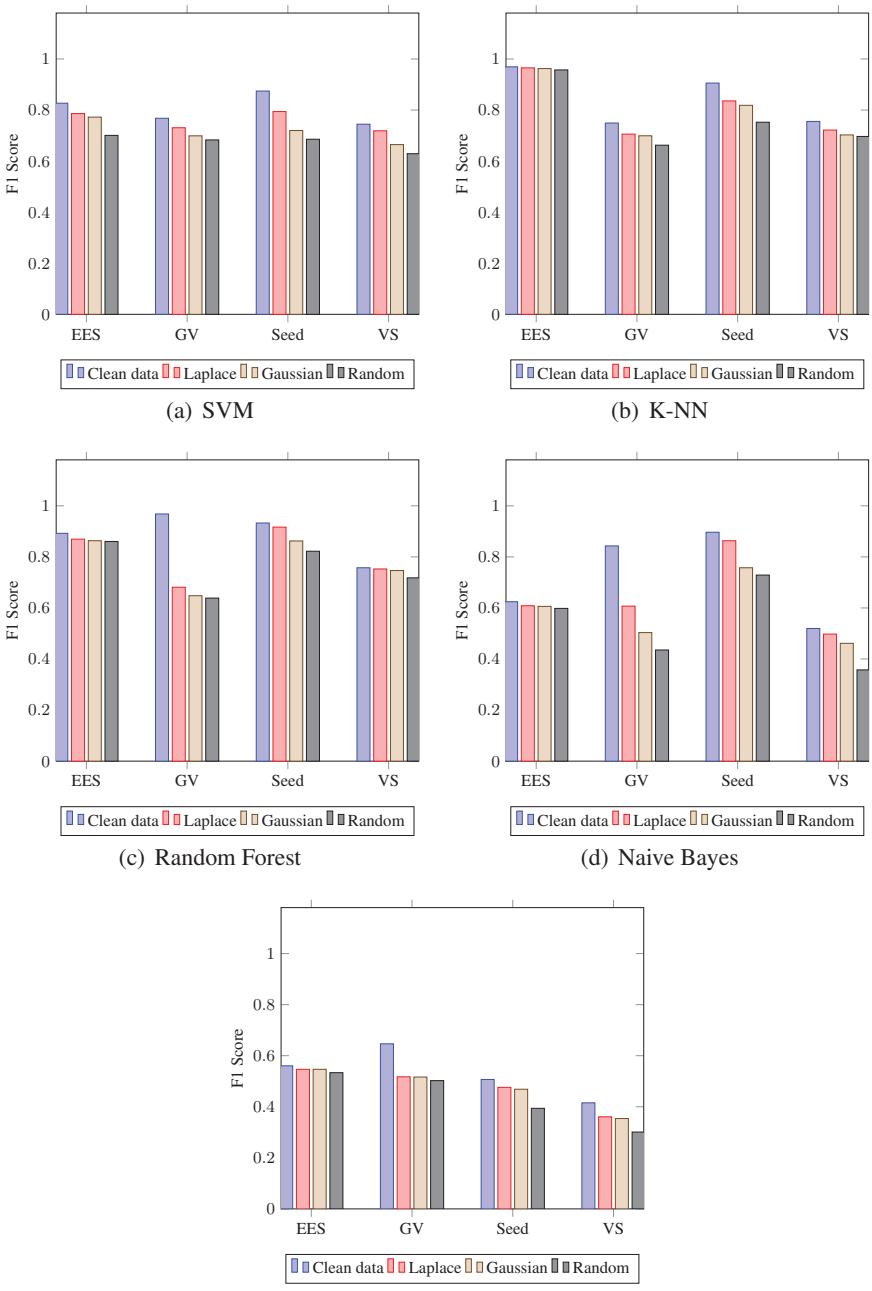
(c) Random Forest

(d) Naive Bayes



(e) ANN Classifier

Figure 7 Recall of *CM* for DAPM.



(a) SVM (b) K-NN

(c) Random Forest (d) Naive Bayes

(e) ANN Classifier

Figure 8 F1 Score of *CM* for DAPM.

6.4 Security Analysis and Validation

DAPM protects data from all entities, including DO_{id}, CSP, DU_{id}, and DA. DO_{id}; $id \in [1, n]$ use the encryption algorithm to protect their data from other entities: DO, CSP, DA, and DU. DO_{id} encrypt data with their keys and share it in encrypted form D^E. CSP converts D^E into partially decrypted data $D^{E'}$. Furthermore, $D^{E'}$ is transferred into the noised form D^N by CSP to prevent data leakage. DA carries out a machine learning model on this noise-added data with ϵ-differential privacy without information loss.

The protocol's authentication phase is explicitly verified using the ProVerif security analysis tool. Queries are used to assess the security of the protocol's various security primitives. If the query is satisfied and returns true, then the attacker can not attack, and ensures the security primitive. Apart from this, if the query returns false, then ProVerif reconstructs the protocol's execution path to determine the steps of attacks [24]. The statement 'RESULT not attacker' shown in Figure 9 implies that all variables utilized during mutual authentication, such as DO, CSP, DA, DU, D^E, $D^{E'}$, D^N, PB, SK, TK, created queries, and events are secure. Consequently, the acquired results confirm that DAPM is entirely safe. DAPM provides robust security and improves the accuracy of data processing because of converting the encrypted data with distinct public keys into noise-added data. DAPM achieved a substantial *CA*, *P*, *R*, *FS* up to 98%, 98%, 97%, and 97%, respectively.

−− Query not attacker(DO[])	−− Query inj-event(endCSP) ==> inj-event(startCSP)
RESULT not attacker(DO[]) is true.	RESULT inj-event(endCSP) ==> inj-event(startCSP) is true.
−− Query not attacker(CSP[])	−− Query inj-event(endDU) ==> inj-event(startDU)
RESULT not attacker(CSP[]) is true.	RESULT inj-event(endDU) ==> inj-event(startDU) is true.
−− Query not attacker(DU[])	−− Query inj-event(endDA) ==> inj-event(startDA)
RESULT not attacker(DU[]) is true.	RESULT inj-event(endDA) ==> inj-event(startDA) is true.
−− Query not attacker(DA[])	
RESULT not attacker(DA[]) is true.	**Verification summary:**
−− Query not attacker(D^E[])	Query not attacker(DO[]) is true.
RESULT not attacker(D^E[]) is true.	Query not attacker(CSP[]) is true.
−− Query not attacker($D^{E'}$ [])	Query not attacker(DU[]) is true.
RESULT not attacker($D^{E'}$ []) is true.	Query not attacker(DA[]) is true.
−− Query not attacker(D^N[])	Query not attacker(D^E[]) is true.
RESULT not attacker(D^N[]) is true.	Query not attacker($D^{E'}$ []) is true.
−− Query not attacker(PB[])	Query not attacker(D^N[]) is true.
RESULT not attacker(PB[]) is true.	Query not attacker(PB[]) is true.
−− Query not attacker(TK[])	Query not attacker(TK[]) is true.
RESULT not attacker(TK[]) is true.	Query not attacker(SK[]) is true.
−− Query not attacker(SK[])	Query inj-event(endDO) ==> inj-event(startDO) is true.
RESULT not attacker(SK[]) is true.	Query inj-event(endCSP) ==> inj-event(startCSP) is true.
−− Query inj-event(endDO) ==> inj-event(startDO)	Query inj-event(endDU) ==> inj-event(startDU) is true.
RESULT inj-event(endDO) ==> inj-event(startDO) is true.	Query inj-event(endDA) ==> inj-event(startDA) is true.

Figure 9 Security analysis and validation results.

7 Conclusion

This paper proposed a novel model named DAPM, which preserves the privacy of outsourced data in the cloud environment. All the involved entities are considered untrusted for providing more effective security. Therefore, a robust mechanism is developed in the model by exploring every possible threat during data flow among the involved parties. DAPM provides an effective sharing protocol to prevent the loss of data. In this work, multiple data owners are allowed to outsource their data to the cloud for storing and computation, and different statistical noise is injected at the cloud platform according to the queries. The experiments have been performed, and results demonstrate that DAPM ensures high accuracy, precision, recall, and f1-score and is found to be secure, efficient, and optimal.

References

[1] J. Li, X. Chen, S. S. Chow, Q. Huang, D. S. Wong, and Z. Liu, "Multi-authority fine-grained access control with accountability and its application in cloud," *Journal of Network and Computer Applications*, vol. 112, pp. 89–96, 2018.

[2] Y. Miao, J. Ma, X. Liu, X. Li, Q. Jiang, and J. Zhang, "Attribute-based keyword search over hierarchical data in cloud computing," *IEEE Transactions on Services Computing*, 2017.

[3] J. Hur and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 7, pp. 1214–1221, 2010.

[4] Z. Qin, H. Xiong, S. Wu, and J. Batamuliza, "A survey of proxy re-encryption for secure data sharing in cloud computing," *IEEE Transactions on Services Computing*, 2016.

[5] Z. Zhu and R. Jiang, "A secure anti-collusion data sharing scheme for dynamic groups in the cloud," *IEEE Transactions on parallel and distributed systems*, vol. 27, no. 1, pp. 40–50, 2015.

[6] J. Wei, W. Liu, and X. Hu, "Secure data sharing in cloud computing using revocable-storage identity-based encryption," *IEEE Transactions on Cloud Computing*, vol. 6, no. 4, pp. 1136–1148, 2016.

[7] Z. Fu, L. Xia, X. Sun, A. X. Liu, and G. Xie, "Semantic-aware searching over encrypted data for cloud computing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2359–2371, 2018.

[8] B. Hauer, "Data and information leakage prevention within the scope of information security," *IEEE Access*, vol. 3, pp. 2554–2565, 2015.

[9] S. Wang, J. Zhou, J. K. Liu, J. Yu, J. Chen, and W. Xie, "An efficient file hierarchy attribute-based encryption scheme in cloud computing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 6, pp. 1265–1277, 2016.

[10] H. Liu, X. Li, M. Xu, R. Mo, and J. Ma, "A fair data access control towards rational users in cloud storage," *Information Sciences*, vol. 418, pp. 258–271, 2017.

[11] Z. Liu, Z. L. Jiang, X. Wang, and S.-M. Yiu, "Practical attribute-based encryption: Outsourcing decryption, attribute revocation and policy updating," *Journal of Network and Computer Applications*, vol. 108, pp. 112–123, 2018.

[12] L. Zhang, Y. Cui, and Y. Mu, "Improving security and privacy attribute based data sharing in cloud computing," *IEEE Systems Journal*, vol. 14, no. 1, pp. 387–397, 2019.

[13] J. Li, S. Wang, Y. Li, H. Wang, H. Wang, H. Wang, J. Chen, and Z. You, "An efficient attribute-based encryption scheme with policy update and file update in cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 12, pp. 6500–6509, 2019.

[14] J. Yuan and S. Yu, "Privacy preserving back-propagation neural network learning made practical with cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 212–221, 2013.

[15] R. Yonetani, V. Naresh Boddeti, K. M. Kitani, and Y. Sato, "Privacy-preserving visual learning using doubly permuted homomorphic encryption," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2040–2050, 2017.

[16] Y. Aono, T. Hayashi, L. Wang, S. Moriai, *et al.*, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1333–1345, 2017.

[17] T. Li, Z. Huang, P. Li, Z. Liu, and C. Jia, "Outsourced privacy-preserving classification service over encrypted data," *Journal of Network and Computer Applications*, vol. 106, pp. 100–110, 2018.

[18] T. Li, J. Li, Z. Liu, P. Li, and C. Jia, "Differentially private naive bayes learning over multiple data sources," *Information Sciences*, vol. 444, pp. 89–104, 2018.

[19] X. Ma, J. Ma, H. Li, Q. Jiang, and S. Gao, "Pdlm: Privacy-preserving deep learning model on cloud with multiple keys," *IEEE Transactions on Services Computing*, 2018.

[20] P. Li, T. Li, H. Ye, J. Li, X. Chen, and Y. Xiang, "Privacy-preserving machine learning with multiple data providers," *Future Generation Computer Systems*, vol. 87, pp. 341–350, 2018.

[21] P. Li, J. Li, Z. Huang, C.-Z. Gao, W.-B. Chen, and K. Chen, "Privacy-preserving outsourced classification in cloud computing," *Cluster Computing*, vol. 21, no. 1, pp. 277–286, 2018.

[22] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE symposium on security and privacy (SP'07)*, pp. 321–334, IEEE, 2007.

[23] C. Dwork and A. Smith, "Differential privacy for statistics: What we know and what we want to learn," *Journal of Privacy and Confidentiality*, vol. 1, no. 2, 2010.

[24] Z. Wang, "A privacy-preserving and accountable authentication protocol for iot end-devices with weaker identity," *Future Generation Computer Systems*, vol. 82, pp. 342–348, 2018.

Biographies

Rishabh Gupta received the MCA degree in computer science from Guru Jambheshwar University Science and Technology, Hisar, India, in 2015. He is currently working toward the Ph.D. degree in Computer Science with the Department of Computer Applications, National Institute of Technology, Kurukshetra, Kurukshetra, India.

He is awarded the Senior Research Fellowship by the University Grants Commission, Government of India. His research interests include cloud computing, machine learning, big data, and information security and privacy.

Ashutosh Kumar Singh received the Ph.D. degree in electronics engineering from the Indian Institute of Technology BHU, Varanasi, India, in 2000.

He is working as a Professor and Head with the Department of Computer Applications, National Institute of Technology Kurukshetra, Kurukshetra, India. He has more than 20 years of research and teaching experience in various Universities in India, UK, and Malaysia. He is Postdoctoral Researcher from the Department of Computer Science, University of Bristol, Bristol, UK. He has authored/co-authored more than 250 research papers and 8 books. His research interests include verification, design, and testing of digital circuits, data science, cloud computing, machine learning, security, etc.