Xue Jiang*, Xuebing Zhou, and Jens Grossklags

Privacy-Preserving High-dimensional Data Collection with Federated Generative Autoencoder

Abstract: Business intelligence and AI services often involve the collection of copious amounts of multi-dimensional personal data. Since these data usually contain sensitive information of individuals, the direct collection can lead to privacy violations. Local differential privacy (LDP) is currently considered a state-of-the-art solution for privacy-preserving data collection. However, existing LDP algorithms are not applicable to high-dimensional data; not only because of the increase in computation and communication cost, but also poor data utility.

In this paper, we aim at addressing the *curse-of-dimensionality* problem in LDP-based high-dimensional data collection. Based on the idea of machine learning and data synthesis, we propose DP-Fed-Wae, an efficient privacy-preserving framework for collecting high-dimensional categorical data. With the combination of a generative autoencoder, federated learning, and differential privacy, our framework is capable of privately learning the statistical distributions of local data and generating high utility synthetic data on the server side without revealing users' private information. We have evaluated the framework in terms of data utility and privacy protection on a number of real-world datasets containing 68–124 classification attributes. We show that our framework outperforms the LDP-based baseline algorithms in capturing joint distributions and correlations of attributes and generating high-utility synthetic data. With a local privacy guarantee $\epsilon = 8$, the machine learning models trained with the synthetic data generated by the baseline algorithm cause an accuracy loss of $10\% \sim 30\%$, whereas the accuracy loss is significantly reduced to less than 3% and at best even less than 1% with our framework. Extensive experimental results demonstrate the capability and efficiency of our framework in synthesizing high-dimensional data while striking a satisfactory utility-privacy balance.

Keywords: high-dimensional data collection, local differential privacy, federated learning, generative models

DOI 10.2478/popets-2022-0024

Received 2021-05-31; revised 2021-09-15; accepted 2021-09-16.

1 Introduction

With the rapid development of network and computer technologies, large and diverse quantities of multi-dimensional person-specific data are frequently generated on local devices such as smartphones and IoT sensors. These data usually contain rich information of univariate and multivariate (joint) distributions describing user profiles, which is valuable for data analysts to explore the hidden correlations and patterns of data from different perspectives and to obtain a better understanding of the characteristics of user groups. For instance, a digital healthcare application may utilize users' physical information (*i.e.*, temperature, blood pressure, activity signals, *etc.*) for health monitoring and disease predictions, while an online shopping website may take users' age, gender, and purchase history for providing suitable product recommendations. In principle, the more dimensions the data consist of, the more information can be used for describing an individual user; thus, the more accurate the decision-making system can be. Therefore, the collection of multi-dimensional data can be of significant help for companies and organizations in designing and building effective business intelligence & AI services.

However, since the data are generated based on individuals' ongoing behaviors, the direct collection can reveal sensitive information about them and lead to severe privacy problems (see, for example, [7, 12]). Local differential privacy (LDP) [31], as a state-of-the-art data anonymization mechanism, has been recently deployed

***Corresponding Author: Xue Jiang:** Technische Universität München, Huawei Technologies Düsseldorf GmbH, E-mail: xue.jiang@tum.de

Xuebing Zhou: Huawei Technologies Düsseldorf GmbH, E-mail: Xuebing.Zhou@huawei.com

Jens Grossklags: Technische Universität München, E-mail: jens.grossklags@tum.de

by major technology organizations such as Apple [14], Google [23], and Microsoft [15] for privacy-preserving data collection. By locally randomizing the user data before sending it to the server, the LDP algorithms ensure that the server cannot access the original user data, but is able to learn the population's overall statistics. However, prior research on LDP-based data collection mainly focuses on one-dimensional statistics, such as frequency estimation [14, 23], heavy-hitter identification [6, 11], and itemset mining [43, 54], *etc.* But since the attributes in multi-dimensional data are usually correlated, the server is particularly interested in learning the correlations and joint distributions among attributes.

Directly applying the above-mentioned LDP algorithms for estimating the joint distributions of multi-dimensional data faces a foundational problem: the *curse-of-dimensionality*. The domain size increases exponentially with data dimensionality, which will lead to extremely large communication cost and storage complexity, as well as a significant degradation in data utility. To reduce the large communication overhead, Fanti et al. [24] proposed to separately collect data of each dimension under LDP and to estimate the joint distributions using expectation maximization (EM). However, the algorithm only supports estimates of the joint distribution of two attributes. Further, Ren et al. [44] introduced LoPub, which splits the c-dimensional data into k-dimensional clusters ($k < c$) using dependence graphs and estimates k-way joint distributions via an EM-based and Lasso regression-based approach. However, the algorithm still suffers from high computational complexity and low data utility when k is large. Based on these facts, alternative solutions for privacy-preserving high-dimensional data collection are still greatly needed.

Recently, data synthesis has been considered a promising approach for addressing data privacy issues in business intelligence & AI services. With the strong capabilities of characterizing the joint distributions and correlations of high-dimensional data, deep generative models are increasingly used for generating high-utility and low-sensitivity synthetic data. In this paper, we follow the idea of data synthesis and propose DP-Fed-Wae, a privacy-preserving framework for high-dimensional categorical data collection. Different from prior work on differentially private synthetic data generation algorithms [41, 49, 57], which mainly focuses on the centralized setting where the real data are already collected by the server, our framework conducts the data synthesis *without* collecting real local data. The main idea is to train a (generative) Wasserstein Autoencoder [48] (WAE) under the federated learning [38] (FL) setting to learn the distributions of the high-dimensional local data and then to generate high-quality synthetic data on the cloud server. Moreover, we propose a novel local randomization algorithm SignDS, which is applied on a client's local updates to prevent potential privacy leakages in FL. The algorithm provides a strict ϵ-LDP privacy guarantee for any client's local dataset.

In comparison with previous data collection approaches, our framework shows significant advantages in both *data utility* and *privacy protection*. As for data utility, the WAE model has a strong capability in capturing correlations and joint distributions of high-dimensional data and generating high-utility synthetic data. The generated synthetic data can be easily scaled up to replace the real data for data analysis and AI training tasks. As for privacy, the generated data are fully synthetic, which effectively reduce risks of re-identification attacks or attribute disclosure [41]. Moreover, training the WAE model under the LDP-FL setting not only avoids the collection of raw user data but also provides comprehensive privacy guarantees to the framework. Our contributions can be summarized as follows:

- We propose DP-Fed-Wae, an efficient and privacy-preserving framework that effectively combines a generative autoencoder, FL, and DP for collecting high-dimensional categorical data. Based on the idea of data synthesis, the framework effectively solves the *curse-of-dimensionality* problem in LDP-based data collection solutions. The synthetic data preserves high utility and can replace real data for data mining and AI training tasks.
- We further propose a novel local randomization algorithm SignDS, which perturbs clients' local updates and prevents potential privacy leakages in FL. We prove that the algorithm follows a strict ϵ-LDP definition and provides a strong local privacy guarantee to any client's local data.
- We have implemented our framework and evaluated the performance in terms of data utility and privacy protection using real-world datasets containing 68–124 classification attributes. Through comparison with the LDP-based algorithms, we show that the synthetic data generated by our framework always preserve much closer joint distributions and correlations to real data. Also, the accuracy loss of the model trained with synthetic data generated by our framework is significantly reduced in comparison to the baseline method. With a local privacy guarantee $\epsilon = 8$, we reduced the accuracy loss from $10\% \sim 30\%$ to less than 3% and at best even less than 1%. Extensive evaluation experiments show

that our framework has outperforming capability and efficiency in collecting high-dimensional data while striking a satisfactory utility-privacy balance.

2 Problem Statement

In this paper, we consider a scenario where a large number of local users hold high-dimensional personal data. A central server aims to estimate the joint distributions of these high-dimensional data and to generate similar synthetic data for data analysis or designing new AI services. Here, we assume the server to be *honest-but-curious*, who follows the system protocols but tries to infer sensitive information of local users. Thus, to protect local privacy, we require the server not to have access to raw local data but only anonymized versions of data or their feature representations.

Assume that there are N local users, each holding one or more data records, which have c attributes $\mathcal{W} = \{w_i | i = 1, \cdots, c\}$. Each attribute w_i has a domain Ω_i of possible values. The full domain for the c-dimensional data record is denoted as $\Omega = \Omega_1 \times \cdots \times \Omega_c$, where $\times$ is the Cartesian product. The total domain size is $|\Omega| = \prod_{i=1}^{c} |\Omega_i|$, which increases exponentially with data dimensionality c. Based on the notation above, the problem can then be formulated as follows: given a c-dimensional private dataset X distributed among N local users, a central server aims to generate a synthetic dataset X' without access to raw data X. The synthetic dataset X' has the same attributes $\mathcal{W}$, and preserves similar joint distributions of X, namely

$$P_{X'}(w_1 \cdots w_c) \approx P_X(w_1 \cdots w_c). \tag{1}$$

It can be further derived that the synthetic data X' also preserves m-way joint distributions as X. Namely, given an m-way attribute combination $\omega \subseteq \mathcal{W}$, we have

$$P_{X'}(\omega) \approx P_X(\omega). \tag{2}$$

3 Proposed Solution

As discussed previously, existing LDP algorithms are impractical for collecting high-dimensional data due to both high computation and communication cost and poor data utility (*e.g.*, [23, 24, 44]). In order to solve the curse-of-dimensionality problem, we propose DP-Fed-Wae, an efficient privacy-preserving framework for collecting high-dimensional categorical data. The framework contains three main components: a generative autoencoder, FL, and DP. Following the idea of recent data synthesis techniques, the framework utilizes generative autoencoders to learn the statistical distributions and correlations of high-dimensional user data and then to generate high-utility synthetic data on the server side. Different from existing works of synthetic data generation where the real data are already available to the server (*e.g.*, [41, 49, 57]), our framework focuses on the scenario where the real data are distributed on local devices. Therefore, we propose to train the generative autoencoder under the FL setting, which only exchanges model parameters during the training process and keeps the raw user data inaccessible to the server. Furthermore, we incorporate DP during the training process in order to prevent potential privacy leakages in FL. In comparison to the previous DP-FL frameworks that add DP noise on the server side [5, 39], we propose a novel local randomization algorithm that perturbs the local updates before uploading them to the server. This ensures that the server cannot gain access to the real local updates and efficiently prevents local privacy leakages. We prove that the randomization algorithm follows a strict LDP definition and provides a strong *local privacy guarantee* to each client's local dataset.

The overall workflow is presented in Figure 1, which is processed in the following sequence:

1. The local clients first process the original categorical data into a numerical form, which can be used for training the generative autoencoder. At the same time, the server defines the structure of the generative autoencoder based on the dimensionality of local data and initializes the model.
2. The generative autoencoder is then collaboratively trained under the FL mechanism that is incorporated with LDP to achieve strict privacy guarantees.
3. After the model gets trained, the decoder is extracted for generating synthetic data. The generated data will be finally converted back to categorical form and used for data mining and building machine learning models.

3.1 Data Pre-Processing and Design of the Generative Model

Since the original data are categorical and cannot be directly processed by machine learning models, we first convert the data into numerical form. Here, we use a *one-hot encoding* to encode each categorical attribute into a binary vector. Each entry in the binary vector

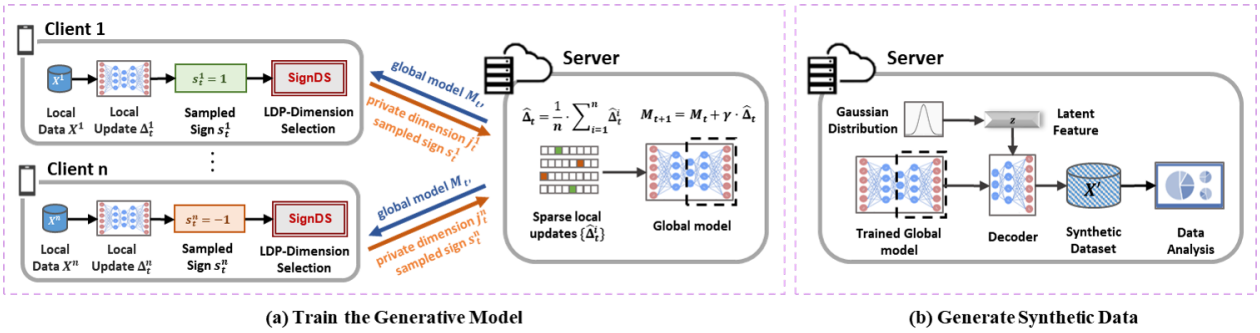

Fig. 1. Overview of the DP-FED-WAE framework. The generative Wasserstein Autoencoder is first trained under the federated setting, which learns the distributions of real local data. An LDP algorithm SIGNDS is applied to the local updates to provide strict local privacy guarantees. After the model is trained, the *decoder* part is used to generate high-utility synthetic data. The generated data will be used for data mining and building AI services.

stands for a unique attribute value and the entry of the given value is set to 1 while all the others are set to 0. Finally, we concatenate all the binary vectors into one vector as the input data for the generative model.

In this paper, we have chosen the Wasserstein Autoencoder (WAE) as the generative model in our framework, which provides better data synthesis capability in comparison to the Variational Autoencoder (VAE) [33] and less training difficulty than the Generative Adversarial Network (GAN) [26]. As a variant from the family of autoencoders, WAE preserves the encoder-decoder architecture. The encoder Q_ϕ compresses the original high-dimensional input $x \sim P_x$ into the low-dimensional latent feature $z = Q_\phi(x)$ and the decoder G_θ maps z to the reconstructed output $x' = G_\theta(z)$, which is the same shape as x. The distance between the original input and the reconstructed output can be presented as $D_r(x, G_\theta(Q_\phi(x)))$. In addition, a regularizer term $D_z(q_z, p_z)$ is applied to measure the distance between the latent space distribution q_z and certain prior distribution p_z. The final objective function of the WAE model can thus be formulated as follows:

$$\mathcal{L}_{WAE} = \mathbb{E}_{x \sim P_x}[D_r(x, G_\theta(Q_\phi(x)))] + \lambda \cdot D_z(q_z, p_z), \quad (3)$$

where λ is a hyperparameter for balancing the two terms. The goal of training is to find an optimal set of parameters, which minimizes the distance between the inputs and outputs while restricting the latent space to follow the prior distribution.

We designed the WAE models with fully-connected hidden layers. We apply the *relu* activation on the output of each hidden layer for better training performance. Moreover, since the inputs are binary vectors, we use the *sigmoid* activation on the output layer, which restricts the output value within [0,1]. Then, we calculate the binary cross-entropy of each input/output dimension and compute the average as the reconstruction distance $D_r(x, G_\theta(Q_\phi(x)))$. For the latent space distance $D_z(q_z, p_z)$, we use the standard Gaussian distribution as the prior distribution p_z and use the maximum mean discrepancy (MMD) to measure the distance between the latent space distribution q_z and p_z, as in [48]. Given a batch of data sampled from the two distributions, *i.e.*, $\{q^1, \cdots, q^n\} \sim q_z$ and $\{p^1, \cdots, p^n\} \sim p_z$, $D_z(q_z, p_z)$ can be empirically estimated as

$$D_z(q_z, p_z) = \frac{1}{n(n-1)} \sum_{i \neq j} \mathcal{K}(p^i, p^j) - \frac{2}{n^2} \sum_{i,j} \mathcal{K}(p^i, q^j) + \frac{1}{n(n-1)} \sum_{i \neq j} \mathcal{K}(q^i, q^j), \quad (4)$$

where $\mathcal{K}(x, y) = \frac{\kappa}{\kappa + \|x-y\|_2^2}$. Given d_z as the dimension of latent layer and σ_z as the scale of the prior distribution, $\kappa = 2 d_z \sigma_z^2$. We choose λ equal to 1.

3.2 Training the Generative Model

Previous LDP-FL frameworks (*e.g.*, [19, 21, 52]) evenly split the privacy budget across dimensions and apply the perturbation independently. However, the per-dimension privacy budget becomes extremely small for high-dimensional models, which results in a significant increase of noise. A recent work [37] proposed a *two-stage* LDP-FL framework, which splits the privacy budget into a *dimension selection* (DS) stage and a *value perturbation* (VP) stage. In the DS stage, the local update is sorted by *absolute* value and one "important" dimension is privately selected from the top-k dimensions; in the VP stage, the value of the selected di-

Algorithm 1: SignDS

Input: $\Delta \in \mathbb{R}^d$: local update; k: size of the top-k set; ϵ: privacy budget; s: sampled sign

Output: j: selected dimension index

if $s = 1$ **then**
  Select dimensions of k *largest* values in Δ to build the top-k dimension set $\mathcal{S}_{topk}$
else
  Select dimensions of k *smallest* values in Δ to build the top-k dimension set $\mathcal{S}_{topk}$
end if
Sample a Bernoulli variable x such that $\Pr[x = 1] = \frac{e^\epsilon \cdot k}{d-k+e^\epsilon \cdot k}$
if $x = 1$ **then**
  Randomly sample a dimension $j \in \{a \in \{1, \cdots, d\} | a \in \mathcal{S}_{topk}\}$
else
  Randomly sample a dimension $j \in \{a \in \{1, \cdots, d\} | a \notin \mathcal{S}_{topk}\}$
end if
Return j

mension is perturbed. Finally, a sparse local update is constructed and returned to the server. Although [37] mitigated the dimension-dependency problem by only selecting one "important" dimension, the privacy budget is still consumed by the two stages. In high-privacy scenarios (where the privacy budget is small), each stage may therefore obtain only an insufficient privacy budget and cause large randomness.

Motivated by the limitations of [37], we propose a *sign-based dimension selection* algorithm SignDS, as presented in Algorithm 1. The main idea is to substitute the VP stage by assigning a constant value to the selected dimension. Since the parameter values may have different signs, we introduce an extra variable $s \in \{-1, 1\}$, which is randomly sampled by the client with equal probability. Then, given each local update Δ, we build the top-k dimension set $\mathcal{S}_{topk}$ according to Δ's *real values* and the sampled sign s: if $s = 1$, $\mathcal{S}_{topk}$ is built with the dimensions of the k *largest* values; otherwise, it is built with the dimensions of the k *smallest* values. We refer the dimensions included in $\mathcal{S}_{topk}$ as *top-k dimensions* and the rest as *non-top-k dimensions*. Then, a dimension index j is randomly sampled as follows:

$$j \in \begin{cases} \{a \in \{1, \cdots, d\} | a \in \mathcal{S}_{topk}\} & w.p. \quad p \\ \{a \in \{1, \cdots, d\} | a \notin \mathcal{S}_{topk}\} & w.p. \quad 1-p \end{cases}, \quad (5)$$

Namely, the index j is sampled from the *top-k dimensions* with a probability of p and otherwise from the *non-top-k dimensions* with a probability of $1 - p$. We refer to p as the *top-k probability*. Finally, the dimension index j and the sampled sign value s are returned to the server. Since our algorithm does not return the dimension value to the server, we save the privacy budget for the value perturbation stage in [37]. With the same privacy level, we can now achieve less randomness and thus higher accuracy in dimension selection.

In the following, we provide the privacy guarantee and utility analysis of Algorithm 1.

Lemma 1. *Algorithm 1 satisfies ϵ-LDP when the top-k probability $p \leq \frac{e^\epsilon \cdot k}{d-k+e^\epsilon \cdot k}$.*

Proof. For each client, given the sampled sign s and any output dimension $j \in \{1, \cdots, d\}$, let S_{topk}, S'_{topk} be the top-k dimension set of any two possible local update vectors Δ and Δ'. Given the top-k probability p, the probability of sampling a dimension j from the top-k set and the non-top-k set are respectively $p \cdot \frac{1}{k}$ and $(1-p) \cdot \frac{1}{d-k}$. Thus, when $p \leq \frac{e^\epsilon \cdot k}{d-k+e^\epsilon \cdot k}$ we have

$$\begin{aligned} \frac{\Pr[j|\Delta]}{\Pr[j|\Delta']} &= \frac{\Pr[j|S_{topk}]}{\Pr[j|S'_{topk}]} \leq \frac{\Pr[j|j \in S_{topk}]}{\Pr[j|j \notin S'_{topk}]} \\ &= \frac{p \cdot \frac{1}{k}}{(1-p) \cdot \frac{1}{d-k}} \leq e^\epsilon \end{aligned} \quad (6)$$

which completes the proof. □

In addition to the privacy guarantee, we are also interested in how to choose proper k and ϵ in order to achieve a certain top-k probability p. Let $\alpha = k/d$ be the ratio of the top-k parameters regarding the total number of parameters. An $\alpha = 1$ means to randomly select one dimension from the entire dimension group. Intuitively, the smaller α, the closer the parameter values of top-k dimensions to the real largest (or smallest) value and the better the model utility. We derive relations among ϵ, p, and α as follows:

Corollary 1. *With a fixed privacy budget ϵ, in order to achieve a probability p, α should satisfy $\alpha \geq \frac{p}{e^\epsilon \cdot (1-p)+p}$.*

Corollary 2. *With a fixed top-k ratio α, in order to achieve a probability p, ϵ should satisfy $\epsilon \geq \log \frac{p \cdot (1-\alpha)}{(1-p) \cdot \alpha}$.*

Proof. From Lemma 1, we have $p \leq \frac{e^\epsilon \cdot k}{d-k+e^\epsilon \cdot k} = \frac{e^\epsilon \cdot \alpha}{1-\alpha+e^\epsilon \cdot \alpha}$. Thus, with a fixed ϵ, we have $\alpha \geq \frac{p}{e^\epsilon \cdot (1-p)+p}$; with a fixed α, we have $\epsilon \geq \log \frac{p \cdot (1-\alpha)}{(1-p) \cdot \alpha}$ □

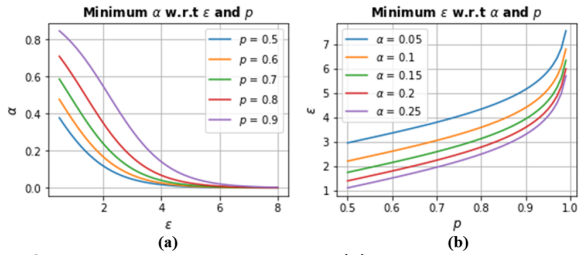

Fig. 2. Relations among ϵ, p, and α. (a): given privacy budget ϵ and the expected top-k probability p, the minimum top-k ratio α required. (b): given the expected top-k ratio α and top-k probability p, the minimum privacy budget ϵ required.

Corollary 1 states that with a fixed privacy budget ϵ, a smaller top-k ratio α leads to a decrease of top-k probability p. Moreover, given an expected top-k probability p and a predefined top-k ratio α, the minimum required privacy budget ϵ can be calculated using Corollary 2. We further visualize the relations of ϵ, p, and α in Figure 2. As shown in Figure 2a, in high-privacy scenarios (*e.g.*, $\epsilon \leq 2$), the required top-k ratio α differs distinctively with the choices of p. Namely, we have to choose a large α in order to ensure that the index is more likely to be sampled from top-k dimensions. As ϵ increases (*e.g.*, $\epsilon \geq 6$), α does not differ much regarding p. In other words, we can always achieve a high top-k probability even with a small top-k ratio. In Figure 2b, we further present the minimum ϵ under various α and p.

We now describe the overall training process presented in Algorithm 2. At each global round t, the server selects a group of n clients and broadcasts the current global model M_t. On the local side, each client i in the group trains the global model for several epochs with his local data X^i and computes the local update Δ_t^i. Then, the client randomly samples a sign $s_t^i \in \{-1, 1\}$ with equal probability and uses it along with the predefined privacy budget ϵ_r to privately select a dimension index j_t^i of the local update. Finally, s_t^i and j_t^i are returned to the server. After receiving the dimension j_t^i and the sampled sign s_t^i, the server builds a sparse local update $\hat{\Delta}_t^i$ and assigns s_t^i to the selected dimension. Since the selected dimension j_t^i satisfies the ϵ-LDP guarantee and the assigned sign value s_t^i is unrelated to the local data, according to DP's robustness to post-processing (Property 1 in Appendix A.2), the sparse local updates also satisfy ϵ-LDP. Finally, the server aggregates all the sparse local updates and updates the global model with a global learning rate γ. The updated global model M_{t+1} is distributed to local clients to start the next round.

Note that according to the sequential composition property (Property 2 in Appendix A.2), if the same client repeatedly participates in the training and submits the local update for multiple global rounds, the overall privacy guarantee for his local data will be accumulated. Assume each client is allowed to participate in at most t_r global rounds. In order to ensure an overall privacy guarantee of ϵ-LDP for each client's local data after the whole training process, the per-round privacy guarantee should satisfy $\epsilon_r \leq \epsilon / t_r$. Moreover, during the training process, we monitor the number of rounds each client participates in. If a client has reached the maximum participating rounds (which is t_r here), he is not allowed to participate in the later training process.

Algorithm 2: Training the Generative Model

Input: $M_1 \in \mathbb{R}^d$: initial global model; n: number of per-round clients; E: number of local epochs; η: local learning rate; k: number of parameters in the top-k set of each local update; T: number of global aggregation rounds; γ: global learning rate; ϵ_r: per-round privacy budget

Output: Trained WAE model M

Server executes:

for global round $t = 1, \cdots, T$ **do**
  Randomly select a group of n clients
  for client $i = 1, \cdots, n$ in parallel **do**
    Broadcast current global model M_t
    Receive sampled sign and dimension $s_t^i, j_t^i = \textbf{LocalUpdate}(M_t, E, \eta, \epsilon_r, k)$
    Build sparse local update $\hat{\Delta}_t^i = \{0\}^d$ and set $\hat{\Delta}_t^i[j_t^i] = s_t^i$
  end for
  Aggregate local updates: $\hat{\Delta}_t = \frac{1}{n} \sum_{i=1}^{n} \hat{\Delta}_t^i$
  Update global model $M_{t+1} = M_t + \gamma \cdot \hat{\Delta}_t$
end for
Return Global model $M = M_{T+1}$

LocalUpdate$(M_t, E, \eta, \epsilon_r, k)$**:**

`// Run on the client side`

Initialize local model $M_t^i \leftarrow M_t$
for epoch $e = 1, \cdots, E$ **do**
  $M_t^i = M_t^i - \eta \cdot \nabla \mathcal{L}(M_t^i, X^i)$
end for
Calculate local update: $\Delta_t^i = M_t^i - M_t$
Randomly sample a sign $s_t^i \in \{1, -1\}$ with probability $\Pr[s_t^i = 1] = 0.5$
Dimension selection $j_t^i = \textbf{SignDS}(\Delta_t^i, k, \epsilon_r, s_t^i)$
Return s_t^i, j_t^i

3.3 Generating Synthetic Data and Data Post-Processing

Once the model has been trained, the server can use the decoder part to generate synthetic data. Recall that the latent space features are enforced to follow the standard Gaussian distribution p_z. Therefore, we can simply generate random latent features from p_z and feed them into the decoder. The decoder output has the same length as the encoded input described in Section 3.1, where each dimension is a numerical value between 0 and 1.

Finally, we need to convert the synthetic data back to categorical form. Given an output vector, we first split it into pieces of short vectors, each representing one categorical attribute. Then, for each short vector, we choose the entry with the maximum value as the attribute value. In the end, we concatenate all the categorical labels into one vector as the final synthetic data. The synthetic data will be used for data analysis and training of machine learning models.

4 Experiments and Results

We implemented the proposed framework and performed comprehensive experiments with a number of open-source datasets to evaluate its performance.[1] In this section, we introduce the experimental settings and discuss the evaluation results.

4.1 Experiment Setup

4.1.1 Datasets and WAE Models

We used four open-source datasets for evaluating the performance of our framework. Each dataset contains multi-dimensional data records, which were used for classification tasks:

- The **Census** dataset [17] contains records drawn from the 1990 United States census data, which include 68 personal attributes such as gender, income, and marriage status. We used the dataset for a classification task to determine the duration of people's active duty service.
- The **Twitter** dataset [32] contains records with 77 attributes such as the number of discussions, average discussion length, and the number of authors, which are used to predict the number of active discussions, namely the popularity magnitude of each instance. In our experiment, we quantified the values of each attribute into five bins. The goal was to classify the level of popularity of each instance.
- The **Vehicle** dataset [18] contains data collected in wireless distributed sensor networks. Each record has 100 attributes representing data collected from different acoustic and seismic sensors. The goal was to train a classifier for vehicle type classification.
- The original **Adult** dataset [34] contains records with 15 personal attributes such as age, occupation, education, and gender. The goal was to train a binary classifier which determines whether a person earns more than 50K a year. We used the processed version from [42], which converted the original attributes into binary features.

Table 1. Datasets details

Dataset	Type	Num. Records	Num. Attributes	Domain Size
Census	Integer	2458285	68	2^{150}
Twitter	Integer	140707	78	2^{181}
Vehicle	Binary	98528	101	2^{101}
Adult	Binary	32561	124	2^{124}

Table 2. Structure of WAE models

Dataset	Num.Params	Model Structure
Census	76524	Input-Dense(96, relu)-Dense(24) -Dense(96, relu)-Output(sigmoid)
Twitter	94961	Input-Dense(128, relu)-Dense(36) -Dense(128, relu)-Output(sigmoid)
Vehicle	13093	Input-Dense(64, relu)-Dense(16) -Dense(64, relu)-Output(sigmoid)
Adult	16060	

We present details of each dataset in Table 1, which include the number of records and attributes, the length of the *one-hot* encoded input, and the total domain size. Since the number of user data should be large in order to preserve data utility (which will be discussed in Section 6.1), in the following experiments we simulated the large-scale distributed scenario by assuming there were 5×10^4 clients, each holding two data records. Hence, we randomly sampled 10^5 records for each dataset. For datasets with more than 10^5 records (*i.e.*, **Census** and **Twitter**), we did the sampling *without* replacement.

We varied the structure of the WAE models to fit the input size of different datasets. Details of the WAE models can be found in Table 2. For binary datasets

1 The code will be available at https://gitee.com/mindspore/mindspore/tree/r1.3/tests/st/fl/mobile/.

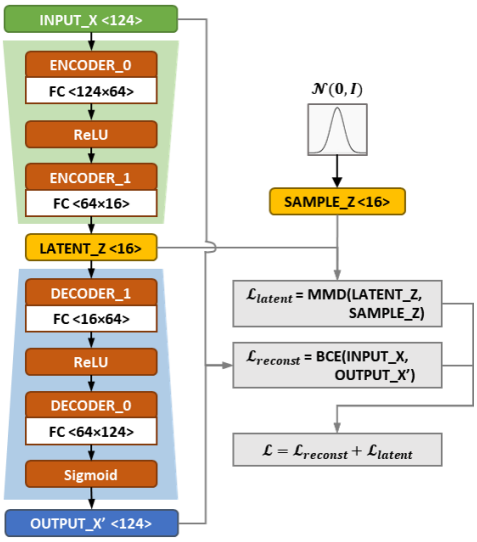

Fig. 3. Structure of the WAE model used for **Adult** dataset.

(*i.e.*, **Vehicle** and **Adult**), small WAE models with a latent-layer size of 16 were already sufficient to achieve satisfactory data synthesis performance. On the other hand, for the other two complex datasets that had distinctively higher domain sizes, we used larger models with a higher latent-layer size to better capture the hidden distribution and cross-attribute correlations. Moreover, it is also possible to use auxilliary data to further optimize the model structure, which we will discuss in Section 5.2. We also provide an example structure of the WAE model used for the **Adult** dataset (Figure 3), where `FC` represents fully-connected layers, `BCE` represents the binary cross-entropy and `MMD` represents the MMD penalty.

4.1.2 Baseline Methods

In the following experiments, we have used LoPub [44] and LoCop [53] as our baseline algorithms. Both algorithms apply LDP directly on the *local data* and send the randomized data to the server. The local randomization follows the RAPPOR algorithm [23]. As derived in [44], given c as the dimension of local data, h as the number of hash functions and f as the flip probability, the overall privacy for each individual client is

$$\epsilon = 2 \cdot c \cdot h \cdot \ln((2 - f)/f). \tag{7}$$

Then, the randomized data will be aggregated on the server side for estimating the joint distributions and attribute dependencies. Such information will then be finally used for constructing the synthetic dataset: LoPub generates the dependency graph based on a dependence threshold ϕ and estimates k-way joint distributions to generate the synthetic data; LoCop leverages multivariate Gaussian copula to determine attribute dependencies and generates synthetic data by only using one- and two-way joint distributions. For both algorithms, we used the Lasso-based regression for estimating the joint distributions. In addition, we followed [44] to choose the number of hash function $h = 4$ and the dependence threshold $\phi = 0.4$.

4.1.3 Evaluation Metrics

We evaluated the performance of our framework from two perspectives, namely the *data utility evaluation* and the *privacy evaluation*:

- For the data utility evaluation, we first compared the statistical distributions of synthetic data and real data. Then, we used different machine learning models to investigate the utility of synthetic data in AI training tasks. Intuitively, synthetic data with high utility should show similar statistical properties and model accuracy as real data.
- For the privacy evaluation, we investigated the capability of our framework against membership inference attacks, where an attacker aimed to use the synthetic dataset to determine whether a target data was used for training the WAE model.

4.1.4 Parameter Configurations

In the experiments, we assumed there were 5×10^4 clients. We set the global round $T = 5000$, and $n = 10$ clients were sampled to train the WAE model in each global round; namely, each client was sampled once during the whole training process. We set the global learning rate $\gamma = 1$ due to the good empirical performance. For local training, each client updated the model for $E = 10$ epochs. We used the Adam optimizer with a default learning rate $\eta = 0.001$ for all the WAE models. For the local randomization, we chose the top-k ratio α from $\{0.05, 0.1, 0.25\}$ and the privacy budget $\epsilon \in \{0.5, 1, 2, 4, 6, 8\}$ to explore the influence of privacy on the framework performance.

It should be noted that ϵ here was the *overall* local privacy budget for each client. As mentioned in Section 3.2, if each client participated in t_r global training rounds, the per-round privacy budget should satisfy $\epsilon_r \leq \epsilon/t_r$. Since we assumed that each client only partic-

Table 3. Computation time of model training and synthetic data generation

Dataset		Adult	Vehicle	Census	Twitter
Training	Client	1.06 s	0.93 s	1.28 s	1.25 s
	Server	3.51 ms	3.05 ms	4.95 ms	4.92 ms
Data Generation		7.38 s	7.37 s	9.44 s	10.47 s

ipated once during the whole training process, we have $t_r = 1$ and the per-round privacy budget is equal to the overall privacy budget. Moreover, we would like to emphasize that the selected ϵ values are *reasonable* local privacy guarantees for collecting c-dimensional data. Consider the privacy guarantee of the baseline algorithms (Equation (7)), with the number of hash function $h = 1$ and a flipping probability $f = 0.5$, we already have $\epsilon = 150$ for the **Census** dataset with $c = 68$. For the **Adult** dataset with $c = 124$, the overall ϵ is even 272, which is significantly larger than our setting.

4.1.5 Computation Environments

We performed all the experiments on a server with Intel E5-2470 2.40GHz CPU. In Table 3, we report the computational time of 1) 10 epochs of local training on each client; 2) one round of local updates aggregation and global model update on the server side; 3) generation of 10^5 synthetic data records on the server.

4.2 Evaluation for Data Utility

In this section, we evaluate the utility of the synthetic data generated using our framework in comparison to the baseline. The evaluations can be generally divided into *statistical comparison* and *AI training performance*.

4.2.1 Statistical Comparison

For the statistical comparison, the goal is to investigate whether the synthetic data generated by our framework can preserve the joint distributions and correlations of real data. Intuitively, synthetic data with high utility should show similar statistical properties as real data. We have respectively compared m-way joint distributions and the cross-attribute correlations to analyze the utility of the synthetic data.

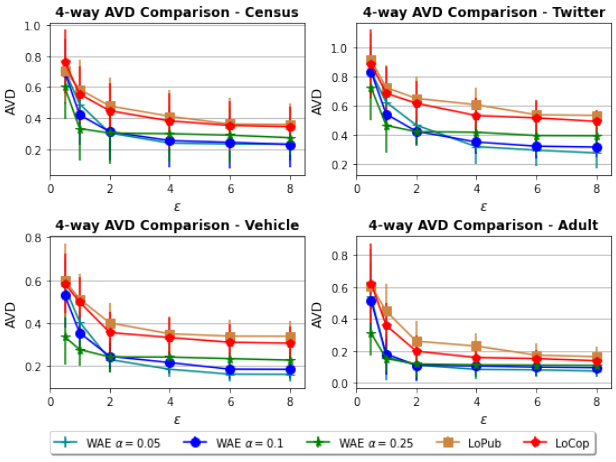

Fig. 4. Average total variation distance (AVD) of four-way joint distribution between the real and synthetic data with respect to different privacy levels.

4.2.1.1 Comparison of Joint Distributions

For the analysis of joint distributions, we used the Average Variant Distance (AVD) to quantify the distribution difference between the real data and synthetic data, as suggested in [44], which is defined as

$$AVD = \frac{1}{2} \sum_{\omega \in \Omega} |P_{real}(\omega) - P_{syn}(\omega)|, \tag{8}$$

where $P_{real}(\omega)$ and $P_{syn}(\omega)$ are m-way joint distributions of real data and synthetic data. More specifically, given an m-way attribute combination ω with a domain size of $|\omega|$, P_{real} and P_{syn} are $|\omega|$-dimensional vectors, where each entry is the probability of a specific value combination (namely the ratio of occurrence in the entire real or synthetic dataset). For each dataset, we randomly chose 100 combinations of m attributes and calculated the average distribution difference.

We first analyzed the AVD of all three algorithms with respect to the privacy level ϵ. For each dataset, we respectively compared the AVD of the synthetic data generated by the baseline algorithms and by our framework. In Figure 4, we present the results for the four-way joint distribution with different privacy budgets. The error bars represent the 95% confidence interval (also for the remaining experimental results). It can be seen that the AVD of all the algorithms decreases with the increase of ϵ. For all datasets, the synthetic data generated by our framework (referred to as *WAE*) have smaller AVD in comparison to the baseline methods (referred to as *LoPub* and *LoCop*), indicating that the synthetic data generated by our framework preserves better multivariate distributions than the baseline methods. Also, we notice that the non-binary datasets **Census**

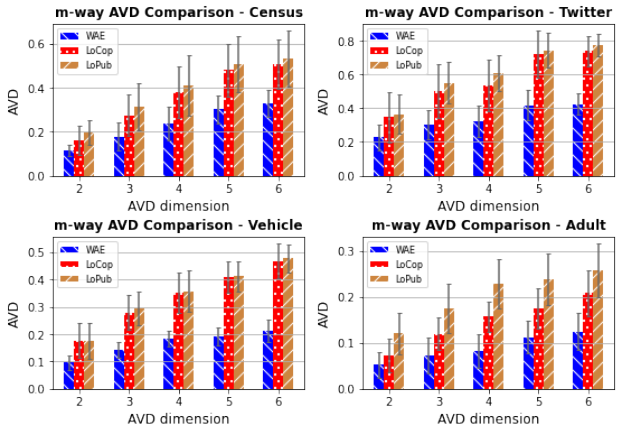

Fig. 5. Average total variation distance (AVD) of m-way joint distributions between the real and synthetic data with respect to different dimension of joint distribution.

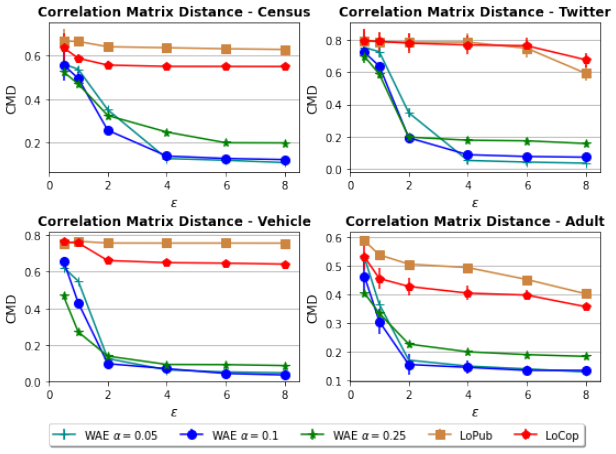

Fig. 6. Averaged correlation error (CMD) between the real and synthetic data with different privacy levels.

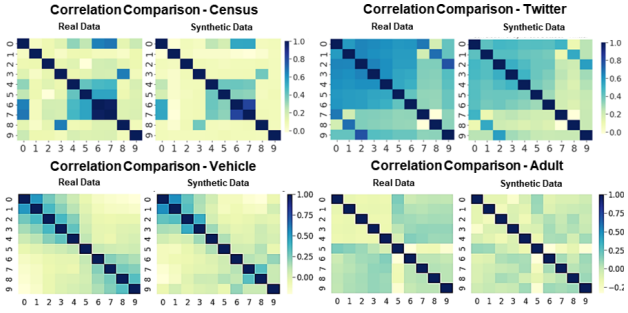

Fig. 7. Correlation comparison between the real and synthetic data with $\epsilon = 8$ and $\alpha = 0.1$. For each dataset, we present the correlations of the first 10 attributes. It can be seen that the synthetic data preserves similar correlations as real data.

and **Twitter** usually show larger AVD in comparison to the other two binary datasets. This is due to the fact that the non-binary datasets have a larger domain size, which leads to lower frequencies of the potential attribute combinations. Therefore, it is more difficult for the generative models to find meaningful mappings between the original input space and the compact latent space, which results in a comparatively larger difference between the synthetic data and real data. Moreover, we observe that for our solution, when the privacy budget ϵ is small, the synthetic data with larger top-k ratio have smaller AVD; while for larger ϵ, the synthetic data with smaller α show better utility. This complies with the discussion in Section 3.2. By intuition, the smaller α is, the better model performance is. However, when ϵ is small, the decrease of α leads to a significant decrease in top-k probability p, which increases the randomness of dimension selection and affects the model convergence. As ϵ increases, p is always relatively high and does not differ much regarding to α. In this case, a smaller α enhances the model performance and thus improves the utility of the synthetic data.

We further analyzed the AVD of all the algorithms with regard to the dimension of joint distributions m, in order to get a deeper insight into our framework's capability on complex statistics. For each dataset, we tested the m-way AVD where $m \in \{2, 3, 4, 5, 6\}$ and present the results under $\epsilon = 4$ in Figure 5. It can be seen that for all the datasets, the AVD increases with a larger m. In addition, our proposed solution consistently outperforms the baseline algorithms. More specifically, the AVD of the baseline algorithms is close to our framework when m is small, yet gets distinctively larger with an increase of m. This indicates that our framework can effectively capture the information of high-dimensional joint distributions of real data.

4.2.1.2 Comparison of Correlation

For the comparison of correlation, we have respectively computed the Pearson correlation coefficient of the real and synthetic dataset and used the Correlation Matrix Distance (CMD) [27] to measure the distance between the two correlations, which is defined as follows:

$$CMD = 1 - \frac{\mathrm{tr}\{R_{real} R_{syn}\}}{\|R_{real}\|_2 \|R_{syn}\|_2}, \tag{9}$$

where R_{real} and R_{syn} are correlation coefficient matrices of real and synthetic data, $\mathrm{tr}(\cdot)$ is the matrix trace, $\|\cdot\|_2$ is the Frobenius norm. The CMD is bounded by $[0, 1]$, where zero means the two correlation matrices are identical.

For each dataset, we calculated the CMD of the synthetic data generated by both the baseline algorithms and our framework under different privacy levels and compare the results in Figure 6. It can be seen that with the same ϵ, the baseline algorithms always show a much larger CMD in comparison to the results of our framework. Although increasing the ϵ helps to reduce the CMD, it is still insufficient for preserving the multivariate correlations of real data. On the other hand, the synthetic data generated by our framework shows a distinctive decrease with the increase of ϵ. In particular, the CMD is close to zero when $\epsilon \geq 4$, indicating that the synthetic data have similar cross-attribute correlations as real data.

We further visualized the correlation coefficient matrix of real data and synthetic data with heat maps in order to better understand the capability of our method in capturing and preserving the cross-attribute correlations. Figure 7 shows the comparison result of the different datasets with $\epsilon = 8$ and $\alpha = 0.1$. For each dataset, we present the correlations of the first 10 attributes. From the visualization results, it can be seen that the correlation of synthetic data is similar to the correlation of real data, indicating that the synthetic data successfully preserves the attribute correlations of real data.

4.2.2 AI Training Performance

Next, we used different machine learning models to evaluate the utility of synthetic data in different AI training tasks. More specifically, we trained two classification models M_{real}, M_{syn}, respectively, with real data and synthetic data, and tested both models with an amount of held-out real data. Then, we compared the test accuracy Acc_{real} and Acc_{syn}, which represent the test accuracy of M_{real}, M_{syn}. If Acc_{syn} was close to Acc_{real}, we considered that the synthetic data are of high utility.

For each dataset, we used a two-layer Neural Network (NN) and Random Forest (RF) as the classification models. We trained each classification model 10 times and calculated the averaged Acc_{syn}. In Figure 8 and Figure 12, we present the results of Acc_{real} as well as Acc_{syn} evaluated on the synthetic data generated by all three methods under different privacy levels. It can be seen that the Acc_{syn} of the baselines shows, in general, distinctive distance from Acc_{real} on both evaluation models and only has slight improvement with larger privacy budget ϵ. In comparison, the Acc_{syn} of our method consistently outperforms the baselines for both classification algorithm. With an increase of ϵ, the

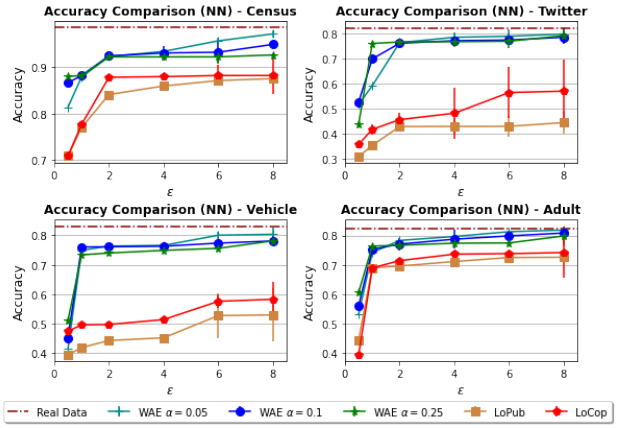

Fig. 8. Classification accuracy of the neural network (NN) trained with real data (*Real Data*) and synthetic data generated by our framework (*WAE*) as well as by the baseline algorithms (*LoPub*, *LoCop*) under different privacy levels.

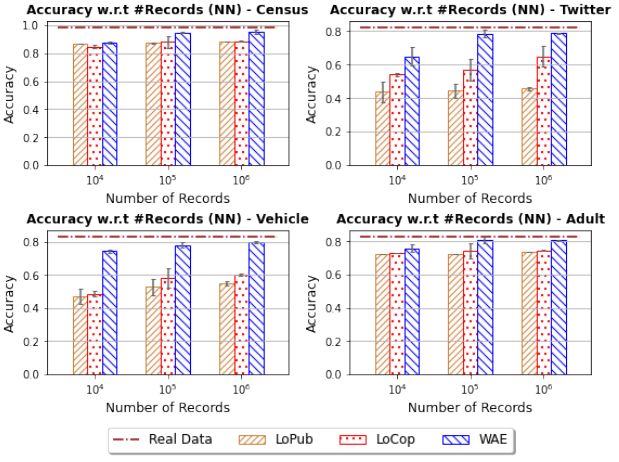

Fig. 9. Classification accuracy of the neural network (NN) with different number of records under the privacy level of $\epsilon = 8$.

Acc_{syn} gradually gets close to Acc_{real}. Moreover, we observe higher Acc_{syn} with the decrease of top-k ratio α. In particular, with $\epsilon = 8$ and $\alpha = 0.05$, the reduction of Acc_{syn} is less than 1% for the **Census** and **Adult** dataset and less than 3% for the other two datasets. The results above further indicate that the synthetic data generated by our framework largely preserves the joint distributions and hidden correlations of real data, and can replace real data for AI training tasks.

4.2.3 Impact of the Number of Records

In the above experiments, we assumed a group size of 5×10^4 clients and in total 10^5 records. We further investigated how the number of records impacts the util-

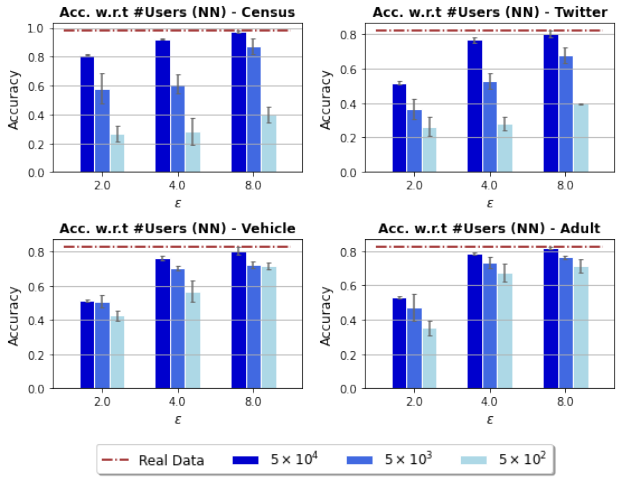

Fig. 10. Classification accuracy of the neural network (NN) with different number of users under different privacy levels.

ity of synthetic data. We varied the number of records among $\{10^4, 10^5, 10^6\}$ (thus the total number of clients is respectively $\{5 \times 10^3, 5 \times 10^4, 5 \times 10^5\}$). Similar to previous experiments, we assumed that each client held two data records and only participated once during the whole training process. For the experiments with 10^4, we set the total global rounds $T = 500$ with $n = 10$ clients for each round. For the experiments with 10^6 records, we set the total global rounds $T = 5000$ with $n = 100$ clients for each round.

We have evaluated the accuracy of both classification models (*i.e.*, two-layer NN and RF) with respect to the number of records and present the results in Figure 9 and Figure 13. Here we compare the results under a privacy of $\epsilon = 8$ (and $\alpha = 0.1$ for our method). Although both algorithms show higher classification accuracy with a larger number of records, the baseline algorithms still cannot achieve significant improvements even with the largest number of records. In comparison, the classification accuracy of our method constantly outperforms the baseline algorithms. In addition, we notice that the classification accuracy in the experiments with 10^4 records is distinctively lower than others. This is because the generative model is underfitted when trained on a limited number of records and thus cannot generate high-utility synthetic data. On the other hand, although a larger number of local data (*e.g.*, 10^6) ensures the generative model be fully-trained, the model performance does not improve much after achieving convergence and thus cannot reach much improvement regarding classification accuracy.

4.2.4 Impact of the Number of Users

In previous experiments, we assumed that there were a large number of users (*e.g.*, 5×10^4) who each only participated once during the entire training process. We further extended the scenario by assuming there were fewer clients and each client was selected multiple times for model training. To this end, we respectively assumed there were 5×10^4, 5×10^3, 5×10^2 clients, each with 2, 20, and 200 data records. Each client participated in 1, 10, and 100 global training rounds and the corresponding per-round privacy budget ϵ_r equals ϵ, $\epsilon/10$ and $\epsilon/100$ (ϵ is the total privacy cost).

For each dataset, we conducted experiments with the total privacy budget $\epsilon \in \{2, 4, 8\}$ and evaluated the accuracy of both classification models regarding the number of users. The results are shown in Figure 10 and Figure 14. It can be generally seen that participating in multiple training rounds can cause a distinctive impact on the framework performance and data utility, especially for datasets with larger generative models (**Census** and **Twitter**). This is because with a fixed total privacy budget the per-round privacy budget is inversely proportional to the participating rounds. Therefore, having each client participating in multiple training rounds will significantly increase the randomness injected during model training and affect the model convergence. Thus, we need to further increase the total privacy budget ϵ to achieve satisfying data utility.

4.3 Evaluation for Privacy Protection

Although a larger privacy budget ϵ has a distinctive contribution to data utility, this may be at the expense of privacy. Currently, there is no concrete understanding of how to choose an appropriate ϵ in practice for a satisfactory utility-privacy trade-off. In this section, we have empirically analyzed the privacy protection capabilities of our framework against the membership inference attack (MIA). We followed the MIA protocol of [46]. The protocol assumed that the attacker holds a reference dataset that shares similar distribution as the real training data. The attacker respectively trains a pair of generative models G_{in} and G_{out} using the reference data *with* and *without* the target record. Then, an attack model is trained to distinguish the synthetic data generated by G_{in} and G_{out}, which can be considered as a binary classification task. Finally, given the published synthetic dataset, the attacker can use the attack model to test whether the synthetic data is generated by a

Table 4. Accuracy of membership inference attack

Dataset	Census	Twitter	Vehicle	Adult
No Privacy	0.735	0.698	0.637	0.642
$\epsilon = 8$	0.574	0.547	0.546	0.535
$\epsilon = 2$	0.548	0.529	0.513	0.519
$\epsilon = 0.5$	0.529	0.524	0.507	0.506

model trained with the target record, namely whether the target record is included in the generative model's training dataset.

We randomly picked 30 records as the target record. For each target record, we trained generative models under different privacy settings and repeated the attack 10 times. In each attack trial, we used a list of machine learning models such as SVMs, logistic regression models, KNNs, RFs, and NNs as the attacker and picked the highest attack accuracy over all the attackers. Finally, we computed the averaged attack accuracy against all the target records under different privacy settings and present the results in Table 4. It can be seen that synthetic data generated by the non-private generative model is more likely to reveal the membership information of the target record. The attack accuracy of all the datasets is more than 60% and even up to 73.5% for the **Census** dataset. On the other hand, applying DP can effectively reduce the attack accuracy. With $\epsilon = 0.5$, the attack accuracy is reduced by $13\% \sim 20\%$. Even with $\epsilon = 8$, the attack accuracy can still be reduced by $8\% \sim 16\%$ and is close to 50%, namely the accuracy of a random guess. The results demonstrate that our framework is able to reduce the risk of membership inference attacks and provide privacy protection to the local data.

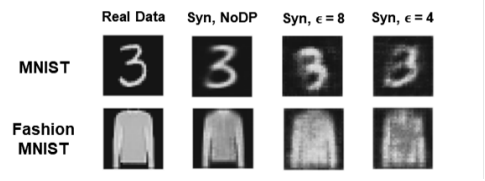

Fig. 11. Results of data synthesis on image datasets.

Table 5. Classification accuracy of synthetic data generated by WAE models with (w) and without (w/o) pre-training

Dataset		Accuracy - NN		Accuracy - RF	
		w/o	w	w/o	w
Census	$\epsilon = 8$	0.948	0.960	0.952	0.965
	$\epsilon = 4$	0.929	0.935	0.932	0.940
Twitter	$\epsilon = 8$	0.786	0.798	0.788	0.796
	$\epsilon = 4$	0.771	0.782	0.778	0.785
Vehicle	$\epsilon = 8$	0.781	0.795	0.788	0.794
	$\epsilon = 4$	0.762	0.789	0.763	0.782
Adult	$\epsilon = 8$	0.808	0.815	0.812	0.820
	$\epsilon = 4$	0.787	0.798	0.775	0.789

enhance the feature extraction capability and use the mean squared error instead of the cross-entropy to measure the reconstruction distance. Despite the variation of the generative models, the main idea of training the model under privacy-preserving federated learning and generating synthetic data remains unchanged. In Figure 11, we give the synthesis results evaluated on the **MNIST** [35] and **Fashion-MNIST** [56] dataset. For each dataset, we show the synthetic data produced by generated models trained under different privacy settings, namely, non-private, with privacy of $\epsilon = 8$ and $\epsilon = 4$. Note that the synthetic data are randomly generated and may look different from real data. However, it can be observed that our framework is also capable of synthesizing image datasets, and generated images have better quality with an increase of the privacy budget.

5 Discussion

5.1 Extension to Other Data Types

In this paper, we demonstrated that our framework performs well on high-dimensional categorical data. In order to do so, we converted the categorical data into numerical form for training the WAE model and then reversed the model's numerical outputs back to categorical form. In future studies, it is also possible to modify the current model structure and the loss function in order to extend the framework for supporting other data types, image data and text data. For instance, for image data, we can further apply convolution layers to

5.2 Auxiliary Data for Pre-Training

Before applying the WAE model for collecting local user data, the server needs to design the model structure. An appropriate model structure helps to enhance the capability of capturing the local data distributions and thus the utility of synthetic data. In our scenario, the server only knows the *basic properties* of the data to be collected, such as the number of attributes and the domain of each attribute. The server can thus use some auxiliary data to optimize the model structure. The auxiliary data here refer to certain public datasets or the random

data generated by uniformly sampling from the domain of the local data. The server can use such data to simulate the data collection process and tune the model structure by evaluating the utility of the synthetic data. Moreover, the auxiliary data can also be used for pre-training the WAE model before applying the model in the data collection process, so as to improve the model convergence and the utility of synthetic data.

In Table 5, we compare the utility of synthetic data generated by WAE models with (w) and without (w/o) pre-training under the setting of $\alpha = 0.1$ and $\epsilon \in \{4, 8\}$. For each dataset, we have randomly generated an auxiliary dataset only using the *basic properties* of real data, as mentioned before. We used the auxiliary dataset to pre-train the WAE model and applied the pre-trained model to the data collection process. We respectively evaluated the utility of synthetic data generated in both scenarios based on the classification accuracy of NNs and RFs. For both types of models, we observe that the synthetic data generated by pre-trained WAE models achieve $1 \sim 2\%$ increase in classification accuracy. The results demonstrate that using auxiliary data to pre-train the WAE model is feasible to enhance the model convergence in the data collection process and further improve the synthetic data utility.

5.3 Limitations and Future Work

5.3.1 Utility Loss Under High Privacy Regimes

Although our proposed framework shows significant performance improvement in comparison to the baseline methods, the synthetic data still suffer from obvious utility loss when $\epsilon \leq 1$. Therefore, improving the framework performance under high-privacy regimes is one of the essential future research directions. Besides fine-tuning the hyperparameters such as the number of per-round participants n and the top-k ratio α, one of the other potential solutions is to pre-train the model with auxiliary data before the FL training procedure, as discussed in Section 5.2. In addition, the current SignDS algorithm can be further extended to support the selection of multiple top-k indices, which will also help improve the model convergence.

5.3.2 Communication Cost of Download Link

In comparison to the traditional LDP-based data collection approaches, our framework trains the generative models under the federated learning setting, where each client needs to download the global model and upload the model updates at once. Although our framework significantly reduces the communication cost of the upload link by only transmitting one dimension index and the corresponding sign value, the communication cost of downloading the global model may also become a bottleneck as the model size increases, especially for large-scale FL scenarios. As discussed in Section 6.3.1, a few recent studies propose dropout-based and model pruning-based solutions for reducing the download link communication cost. The basic idea is to reduce the size of the model broadcast to the client side by extracting sub-models or pruning redundant weights from the global model. Meanwhile, using smaller models for local training can also help to reduce the computational cost of the client devices. Therefore, exploring whether such techniques can be integrated into our framework to further improve communication and computational efficiency will be an important future work.

6 Related Work

6.1 Data Collection Under LDP

Differential privacy (DP) [21], as a strong mathematical formalization of privacy, has been used as a criterion for privacy protection in data publishing, data mining, and machine learning ([1, 20, 57]). However, traditional DP assume a trusted server (data curator), who first collects the original user data, then performs data analysis under differential privacy. In order to eliminate the assumptions of trustworthy servers, local differential privacy (LDP) [31] has been proposed, which provides strong privacy guarantees to local data. By utilizing local randomization algorithms, the server cannot infer any individual's original data, but can learn the overall statistics of the whole population. However, prior research on LDP mainly focuses on collecting one-dimensional statistics, such as frequency estimation ([14, 23]), heavy-hitter identification ([6, 11]), and itemset mining ([43, 54]). Regarding scenarios with multi-dimensional data, Alaggan et al. [2, 3] proposed using Bloom filters to encode local data and analyze aggregated statistics. However, their works do not involve the estimation of joint distributions and cross-attribute correlations, which differs from our objectives.

Directly applying the above LDP-based algorithms to estimate complex statistics of high-dimensional data

will cause extremely large communication overhead as well as a degradation in data utility. Consider, for example, RAPPOR [23], a state-of-the-art LDP-based data collection algorithm. For c-dimensional binary data with $c = 32$, we have domain size $|\Omega| = 2^{32} \approx 4.3 \times 10^{10}$. Directly applying RAPPOR consumes a communication cost and a storage space of $O(|\Omega|)$ [44]. Also, for high-dimensional input domains, it is common for each user to have a unique feature combination. Therefore, it is essential to collect a large number of data in order to cover all the possible combinations in the feature domain. Given a domain size $|\Omega|$, as a general rule of thumb [23], the number of user data N should follow $\sqrt{N}/10 \geq |\Omega|$. In the above example, $N \geq 100 \cdot 2^{64} \approx 1.8 \times 10^{21}$. All of these requirements are impractical for real-world applications. In subsequent research, Fanti et al. [24] proposed to separately collect data of each dimension under RAPPOR and estimate the joint distributions using expectation maximization (EM). Although the algorithm significantly reduces the communication overhead between clients and the server, it only supports to estimate the joint distribution of two attributes. Based on [24], Ren et al. proposed LoPub [44], which reduces c-dimensional data to k-dimensional clusters ($k < c$) using dependence graphs and estimates k-way joint distributions with an EM-based and Lasso regression-based approach. However, the algorithm still suffers from high computational complexity and low data utility when k is large. An improved scheme, LoCop [53] was further proposed, which leverages multivariate Gaussian copula to estimate cross-attribute dependencies and to construct synthetic data.

Instead of directly randomizing the local data, our framework uses deep generative models to learn the data distributions and to generate synthetic data without accessing real data, which effectively enhances data utility. In our experiments, we used LoPub and LoCop as the baselines to compare the frameworks' performance in terms of data utility.

6.2 DP Synthetic Data Generation

Differentially private synthetic data generation has been extensively studied over recent years as an alternative solution to privacy-preserving data publishing. Previous works ([36, 57]) analyzed statistical distributions of original data under differential privacy and used them to generate synthetic data. Later works have proposed using differentially private generative models ([41, 49]) to directly generate high-utility synthetic data. However, these works only focus on the centralized setting, where the server has already collected the real data and uses them to generate private synthetic data. In contrast, our approach is practical for a distributed setting, where the server cannot access the real data, but is interested in learning statistical information about the data.

Recent works by Augenstein et al. [5] and Triastcyn et al. [50] also investigate the synthetic data generation under the distributed setting. [5] aims to use generative models to detect errors and bugs in local data. However, as they claimed, such applications do not require high-fidelity generation. On the other hand, although [50] focused on generating and publishing synthetic data, their method is only limited to image data. In addition, they adopted a weaker measure of privacy for preserving the model performance. In comparison to both works, our framework is able to generate synthetic data with high utility and fidelity, which can replace real data in data mining and AI training tasks. Moreover, we apply strict LDP randomization on the client side, which provides strong privacy guarantees for clients' local privacy.

6.3 Efficiency and Privacy in FL

6.3.1 Communication Efficiency in FL

It is widely acknowledged that communication cost can be a bottleneck for FL, especially when training high-dimensional models. Recently, a number of *upload link* and *download link* compression methods have been proposed to alleviate the communication cost in FL.

The *upload link* compression applies quantization or sparsification on the model updates to reduce the communication cost from clients to the server. The main idea of quantization is to reduce the number of bits of update values. For instance, Seide *et al.* [45] proposed the 1-bit SGD, which quantizes the update values larger than a pre-defined threshold to 1 and the rest to 0. Similarly, Bernstein *et al.* [8] proposed SignSGD and SIGNUM, where the update values quantized to its sign value. The study theoretically proved that SignSGD can effectively reduce the communication cost while enjoying a satisfying convergence rate. In contrast, sparsification aims to transmit only a subset of update values. For instance, the top-k mechanisms (*e.g.*, [4, 16]) only keep the top-k largest magnitude values of each model update and set the others to 0. Stich *et al.* [47] proposed a similar scheme with memory. Ivkin *et al.* [28] further proposed to use the count-sketch algorithm to approximately select the top-k updates.

On the other hand, reducing the communication cost of the *download link* has recently also gained increasing attention. Caldas *et al.* [13] gave the first attempt of the *download link* compression and proposed FEDERATED DROPOUT, which extracts small sub-models from the original high-dimensional and send to the clients local training. Based on this idea, Bouacida *et al.* further proposed [10] an adaptive federated dropout algorithm, which builds the sub-models using an adaptive activation score map. In addition, Jiang *et al.* [30] proposed to gradually prune the global model during the training process to achieve better communication and computational efficiency without loss of accuracy.

6.3.2 Privacy Protection in FL

Although FL enjoys significant privacy benefits in comparison to centralized learning, recent works showed that FL is still vulnerable to various privacy attacks [25, 40] against the exchanged local updates and the global model. Thus, an increasing number of studies propose to incorporate differential privacy (DP) into the FL framework. Some works (*e.g.*, [5, 39]) add Gaussian noise on the server side to protect the privacy of the global model. However, such solutions cannot prevent the privacy leakages from the local updates. Thus, other works propose hybrid frameworks (*e.g.*, [22, 29, 51]), which use crypto-based solutions such as homomorphic encryption (HE) and secure multi-party computation (SMC) to further achieve local privacy protection. However, these solutions require extra communication and computation costs during the key-distribution phase and thus, may not be practical to large-scale scenarios.

Considering the privacy and efficiency issues in the above-mentioned solutions, a more practical solution is to apply local differential privacy (LDP) to FL, which perturbs the local updates before sending them to the server. Previous LDP-FL frameworks (*e.g.*, [19, 52, 58]) perturb the local updates using private mean estimation algorithms. However, these algorithms evenly split the privacy budget across dimensions and the injected noise is proportional to the model dimension, making them only applicable to simple ML models. A recent work proposed FEDSEL [37], a *two-stage* LDP-FL framework that includes a *dimension selection* (DS) stage and a *value perturbation* (VP) stage. The DS stage first sorts the local update by *absolute value* and then privately selects one "important" dimension from the top-k dimension set (namely, the set of k dimensions with the largest absolute values). Then, in the VP stage, the value of the selected dimension is perturbed via the LDP algorithms in [52] and used to construct a sparse privatized local update. Although [37] mitigates the dimension-dependency problem by only selecting one "important" dimension, the privacy budget is still consumed by the two stages. In high-privacy scenarios, each stage may therefore obtain only an insufficient privacy budget and cause large randomness.

Inspired by the effectiveness of SIGNSGD [8] and top-k sparsification ([4, 16, 37]), we propose a novel local randomization algorithm called SIGNDS. The main idea is to save the the privacy budget for the VP stage in [37] by assigning sign values instead of the perturbed dimension values to the selected dimensions. With the same privacy budget, we can achieve less randomness and thus higher model utility.

7 Conclusion

Building effective business and AI services requires the collection of personal data, which often introduces challenges related to an insufficient amount of data on the one hand, and privacy violations on the other hand. Recently, deep generative model-based data synthesis techniques have created opportunities for addressing these challenges: the generative models have strong capabilities in capturing the cross-attribute correlations of real data and can easily generate large-scale high-utility data; in addition, since the generated data are fully synthetic and cannot be linked to any particular individual, re-identification attacks or attribute disclosure becomes almost impossible.

In this paper, we have followed the idea of data synthesis and proposed DP-FED-WAE, a privacy-preserving framework for high-dimensional data collection. The framework utilizes a (generative) Wasserstein autoencoder to learn the joint distributions and correlations of high-dimensional user data and generate high-utility synthetic data on the server side. Moreover, we applied a novel LDP-FL framework for training the autoencoder, which not only avoids the collection of real local data but also provides strong local privacy guarantees. Experimental evaluation with real-world datasets shows that our framework significantly outperforms the LDP-based baseline algorithms for high-dimensional data collection and synthesis. The synthetic data generated by our framework preserves very similar statistical properties as real data and can replace real data for data mining and model training tasks.

8 Acknowledgements

This research was fully funded by Huawei. We thank the anonymous reviewers for their constructive comments for improving this paper. In particular, we thank our Shepherd, Thomas Humphries, for his valuable suggestions during the revision.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, New York, NY, USA, 2016. Association for Computing Machinery.

[2] Mohammad Alaggan, Mathieu Cunche, and Sébastien Gambs. Privacy-preserving Wi-Fi analytics. *Proceedings on Privacy Enhancing Technologies*, 2018(2):4–26, 2018.

[3] Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. BLIP: Non-interactive differentially-private similarity computation on Bloom filters. In Andréa W. Richa and Christian Scheideler, editors, *Stabilization, Safety, and Security of Distributed Systems - 14th International Symposium*, volume 7596 of *Lecture Notes in Computer Science*, pages 202–216, Toronto, Canada, 2012. Springer.

[4] Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cédric Renggli. The convergence of sparsified gradient methods. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018*, pages 5977–5987, Montreal, Canada, 2018.

[5] Sean Augenstein, H. Brendan McMahan, Daniel Ramage, Swaroop Ramaswamy, Peter Kairouz, Mingqing Chen, Rajiv Mathews, and Blaise Agüera y Arcas. Generative models for effective ML on private, decentralized datasets. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, virtual, 2020. OpenReview.net.

[6] Raef Bassily, Kobbi Nissim, Uri Stemmer, and Abhradeep Guha Thakurta. Practical locally private heavy hitters. In *Advances in Neural Information Processing Systems*, pages 2288–2296, Long Beach, CA, USA, 2017. Curran Associates Inc.

[7] Gabrielle Berman, Sara de la Rosa, and Tanya Accone. Ethical considerations when using geospatial technologies for evidence generation. Technical report, Innocenti Research Briefs, 2018.

[8] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. SIGNSGD: Compressed optimisation for non-convex problems. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 559–568. PMLR, 2018.

[9] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, New York, NY, USA, 2017. Association for Computing Machinery.

[10] Nader Bouacida, Jiahui Hou, Hui Zang, and Xin Liu. Adaptive federated dropout: Improving communication efficiency and generalization for federated learning. *CoRR*, abs/2011.04050, 2020.

[11] Mark Bun, Jelani Nelson, and Uri Stemmer. Heavy hitters and the structure of local privacy. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 435–447, New York, NY, USA, 2018. Association for Computing Machinery.

[12] Joseph A. Calandrino, Ann Kilzer, Arvind Narayanan, Edward W. Felten, and Vitaly Shmatikov. "You might also like": Privacy risks of collaborative filtering. In *2011 IEEE Symposium on Security and Privacy (S&P)*, pages 231–246, Berkeley, California, USA, 2011. IEEE Computer Society.

[13] Sebastian Caldas, Jakub Konečný, H. Brendan McMahan, and Ameet Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. *CoRR*, abs/1812.07210, 2018.

[14] Differential Privacy Team. Learning with privacy at scale. *Apple Machine Learning Journal*, 1(8), 2017.

[15] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In *Advances in Neural Information Processing Systems*, pages 3571–3580, Long Beach, CA, USA, 2017. Curran Associates Inc.

[16] Nikoli Dryden, Tim Moon, Sam Ade Jacobs, and Brian Van Essen. Communication quantization for data-parallel training of deep neural networks. In *2nd Workshop on Machine Learning in HPC Environments, MLHPC@SC*, pages 1–8, Salt Lake City, UT, USA, 2016. IEEE Computer Society.

[17] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[18] Marco F. Duarte and Yu Hen Hu. Vehicle classification in distributed sensor networks. *Journal of Parallel and Distributed Computing*, 64(7):826–838, 2004.

[19] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Minimax optimal procedures for locally private estimation. *Journal of the American Statistical Association*, 113(521):182–201, 2018.

[20] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil Vadhan. On the complexity of differentially private data release: Efficient algorithms and hardness results. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, pages 381–390, Bethesda, MD, USA, 2009. ACM.

[21] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[22] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Shuang Song, Kunal Talwar, and Abhradeep Thakurta. Encode, shuffle, analyze privacy revisited: Formalizations and empirical evaluation. *CoRR*, abs/2001.03618, 2020.

[23] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 1054–1067, Scottsdale, AZ, USA, 2014. ACM.

[24] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. Building a Rappor with the unknown: Privacy-preserving learning of associations and data dictionaries. *Proceedings on Privacy Enhancing Technologies*, 3:1–21, 2016.

[25] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients – How easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020*, virtual, 2020. Curran Associates Inc.

[26] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, Montreal, Quebec, Canada, 2014. Curran Associates Inc.

[27] Markus Herdin, Nicolai Czink, Hüseyin Özcelik, and Ernst Bonek. Correlation matrix distance, a meaningful measure for evaluation of non-stationary MIMO channels. In *2005 IEEE 61st Vehicular Technology Conference*, volume 1, pages 136–140, Stockholm, Sweden, 2005. IEEE.

[28] Nikita Ivkin, Daniel Rothchild, Enayat Ullah, Vladimir Braverman, Ion Stoica, and Raman Arora. Communication-efficient distributed SGD with sketching. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, pages 13144–13154, Vancouver, BC, Canada, 2019.

[29] Bargav Jayaraman, Lingxiao Wang, David Evans, and Quanquan Gu. Distributed learning without distress: Privacy-preserving empirical risk minimization. In *Advances in Neural Information Processing Systems*, pages 6343–6354. Curran Associates Inc., 2018.

[30] Yuang Jiang, Shiqiang Wang, Bong-Jun Ko, Wei-Han Lee, and Leandros Tassiulas. Model pruning enables efficient federated learning on edge devices. *CoRR*, abs/1909.12326, 2019.

[31] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, Jan 2011.

[32] François Kawala, Ahlame Douzal-Chouakria, Eric Gaussier, and Eustache Dimert. Prédictions d'activité dans les réseaux sociaux en ligne. In *4ième conférence sur les modèles et l'analyse des réseaux : Approches mathématiques et informatiques*, page 16, France, 2013.

[33] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, Banff, AB, Canada, 2014. OpenReview.net.

[34] Ron Kohavi. Scaling up the accuracy of Naive-Bayes classifiers: A decision-tree hybrid. In *Proceedings of the SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 202–207, Portland, Oregon, USA, 1996. AAAI Press.

[35] Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[36] Haoran Li, Li Xiong, and Xiaoqian Jiang. Differentially private synthesization of multi-dimensional data using copula functions. In *Advances in Database Technology: Proceedings of the International Conference on Extending Database Technology*, volume 2014, pages 475–486, Athens, Greece, 2014. NIH Public Access, OpenProceedings.org.

[37] Ruixuan Liu, Yang Cao, Masatoshi Yoshikawa, and Hong Chen. Fedsel: Federated SGD under local differential privacy with top-k dimension selection. In *Database Systems for Advanced Applications - 25th International Conference, DASFAA 2020, Jeju, South Korea, September 24-27, 2020, Proceedings, Part I*, volume 12112 of *Lecture Notes in Computer Science*, pages 485–501. Springer, 2020.

[38] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282, Fort Lauderdale, FL, USA, 2017. PMLR.

[39] Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *Proceedings of the International Conference on Learning Representations*, Vancouver, BC, Canada, 2018. OpenReview.net.

[40] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE Symposium on Security and Privacy (S&P)*, pages 739–753, San Francisco, CA, USA,, 2019. IEEE.

[41] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proceedings of the VLDB Endowment*, 11(10):1071–1083, 2018.

[42] John C. Platt. *Fast Training of Support Vector Machines Using Sequential Minimal Optimization*, page 185–208. MIT Press, Cambridge, MA, USA, 1999.

[43] Zhan Qin, Yin Yang, Ting Yu, Issa Khalil, Xiaokui Xiao, and Kui Ren. Heavy hitter estimation over set-valued data with local differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 192–203, Vienna, Austria, 2016. ACM.

[44] Xuebin Ren, Chia-Mu Yu, Weiren Yu, Shusen Yang, Xinyu Yang, Julie A McCann, and S Yu Philip. LoPub: High-dimensional crowdsourced data publication with local differential privacy. *IEEE Transactions on Information Forensics and Security*, 13(9):2151–2166, 2018.

[45] Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In Haizhou Li, Helen M. Meng, Bin Ma, Engsiong Chng, and Lei Xie, editors, *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association*, pages 1058–1062, Singapore, 2014. ISCA.

[46] Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. Synthetic data - A privacy mirage. *CoRR*, abs/2011.07018, 2020.

[47] Sebastian U. Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified SGD with memory. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018*, pages 4452–

4463, Montreal, Canada, 2018.

[48] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations (ICLR 2018)*, Vancouver, BC, Canada, 2018. OpenReview.net.

[49] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. DP-CGAN: Differentially private synthetic data and label generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition – Workshops*, pages 98–104, Long Beach, CA, USA, 2019. Computer Vision Foundation / IEEE.

[50] Aleksei Triastcyn and Boi Faltings. Federated generative privacy. *IEEE Intelligent Systems*, 35(4):50–57, 2020.

[51] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, pages 1–11, 2019.

[52] Ning Wang, Xiaokui Xiao, Yin Yang, Jun Zhao, Siu Cheung Hui, Hyejin Shin, Junbum Shin, and Ge Yu. Collecting and analyzing multidimensional data with local differential privacy. In *Proceedings of the 35th IEEE International Conference on Data Engineering*, pages 638–649, 2019.

[53] Teng Wang, Xinyu Yang, Xuebin Ren, Wei Yu, and Shusen Yang. Locally private high-dimensional crowdsourced data release based on copula functions. *IEEE Transactions on Services Computing*, pages 1–1, 2019.

[54] Tianhao Wang, Ninghui Li, and Somesh Jha. Locally differentially private frequent itemset mining. In *2018 IEEE Symposium on Security and Privacy*, pages 127–143, San Francisco, California, USA, 2018. IEEE Computer Society.

[55] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 2512–2520, Paris, France, 2019. IEEE.

[56] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017.

[57] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via Bayesian networks. *ACM Transactions on Database Systems*, 42(4):25:1–25:41, 2017.

[58] Yang Zhao, Jun Zhao, Mengmeng Yang, Teng Wang, Ning Wang, Lingjuan Lyu, Dusit Niyato, and Kwok-Yan Lam. Local differential privacy based federated learning for internet of things. *IEEE Internet of Things Journal*, 2020.

A Preliminaries

In the following, we offer additional background information about federated learning and differential privacy.

A.1 Federated Learning

Federated learning [38] (FL) is a decentralized learning mechanism which achieves computational efficiency and privacy benefits by distributing the training task to local devices. At each global round, the server distributes the current global model to a number of local clients. Each client locally updates the global model and returns the model update to the server. On the server side, all the local updates are aggregated to update the global model, which will be distributed in the next global round. Since only model parameters are exchanged during the training process, FL allows the model trained without accessing raw local data.

Although FL provides enhanced privacy protection in comparison to centralized training, recent contributions (*e.g.*, [25, 40, 55]) point out that the mechanism still has privacy risks. In the context of FL, privacy risks can be mainly divided into *local privacy* and *global privacy* aspects. *Local privacy* risks appear when the local updates reveal insights about local data, while *global privacy* risks represent situations when the global model memorizes local data. Motivated by this, privacy enhancing techniques such as secure multi-party computation (MPC) [9] or differential privacy (DP) [39] are incorporated into the original FL mechanism, providing protections against global and local privacy risks.

A.2 Differential Privacy

Differential Privacy (DP) [21] is a state-of-the-art data anonymization technique which provides strong privacy guarantees for data analysis. The mathematical definition of DP is as follows:

Definition 1 (Differential Privacy [21])**.** *A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{O}$ satisfies ϵ-DP if for any two adjacent datasets D, D' differing in one data sample and for any measurable subset of outputs $\mathcal{Y} \subseteq \mathcal{O}$ we have*

$$\Pr\left[\mathcal{M}(D) \in \mathcal{Y}\right] \leq e^{\epsilon} \cdot \Pr\left[\mathcal{M}(D') \in \mathcal{Y}\right], \qquad (10)$$

where ϵ describes the privacy loss.

The Definition 1 is usually applied in centralized settings where the data have already been collected by a trusted server. However, in the local settings, we aim to ensure that each client's local data will not be accessed by the server. Thus, the definition of *local differential privacy* (LDP) has been proposed [31], which provides

strong local privacy guarantees for each user. The definition is as follows:

Definition 2 (Local Differential Privacy [31]). *A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{O}$ satisfies ϵ-LDP if and only if for any two inputs $x, x' \in \mathcal{D}$ and for any output $y \in \mathcal{O}$ we have*

$$\Pr\left[\mathcal{M}(x) = y\right] \leq e^{\epsilon} \cdot \Pr\left[\mathcal{M}(x') = y\right], \qquad (11)$$

where ϵ describes the privacy loss.

In addition, LDP also holds two widely-used properties [21], namely *Robustness to Post-Processing* and *Sequential Composition*. The former property states that any deterministic or randomized function defined over an LDP mechanism also satisfies LDP. The latter states that interactively applying the LDP mechanism on the same set of data yields an accumulated privacy cost.

Property 1 (Robustness to Post-Processing). *Let $\mathcal{M}$ be an ϵ-LDP mechanism and g be an arbitrary mapping from the set of possible outputs to an arbitrary set. Then, $g \circ \mathcal{M}$ is ϵ-LDP.*

Property 2 (Sequential Composition). *Suppose there are n mechanisms $\{\mathcal{M}_1, \cdots, \mathcal{M}_n\}$ that respectively satisfy ϵ_i-LDP and are sequentially computed on the same set of private data D, then a mechanism formed by $(\mathcal{M}_1, \cdots, \mathcal{M}_n)$ satisfies $(\sum_{i=1}^{n} \epsilon_i)$-LDP.*

B Additional Experiment Results

In this section, we provide additional experiment results of the evaluation of AI training performance (Section 4.2.2), including the classification accuracy of random forests trained with synthetic data under different privacy levels (Figure 12), as well as the impact of accuracy regarding the number of records (Figure 13) and the number of clients (Figure 14).

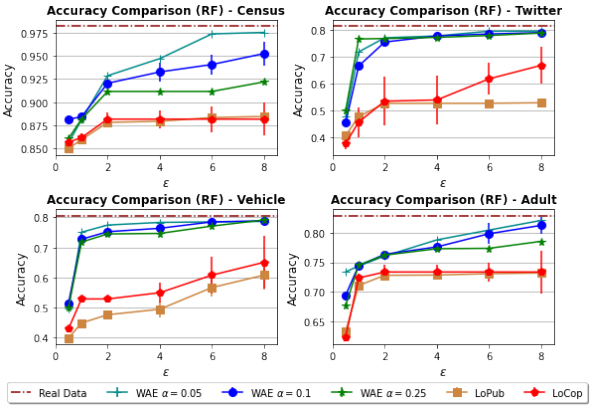

Fig. 12. Classification accuracy of the random forest (RF) trained with real data (*Real Data*) and synthetic data generated by our framework (*WAE*) as well as by the baseline algorithms (*LoPub*, *LoCop*) under different privacy levels.

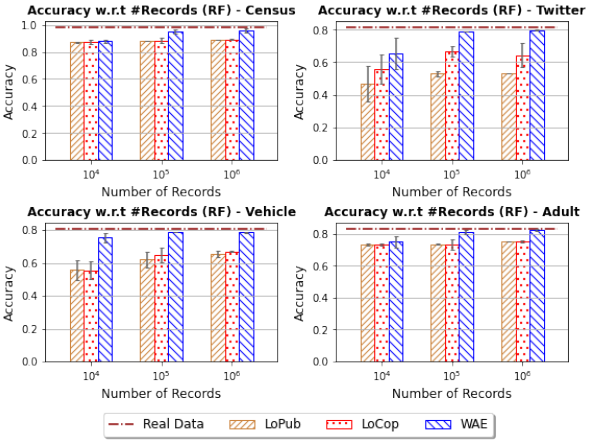

Fig. 13. Classification accuracy of the random forest (RF) with different number of records under the privacy level of $\epsilon = 8$.

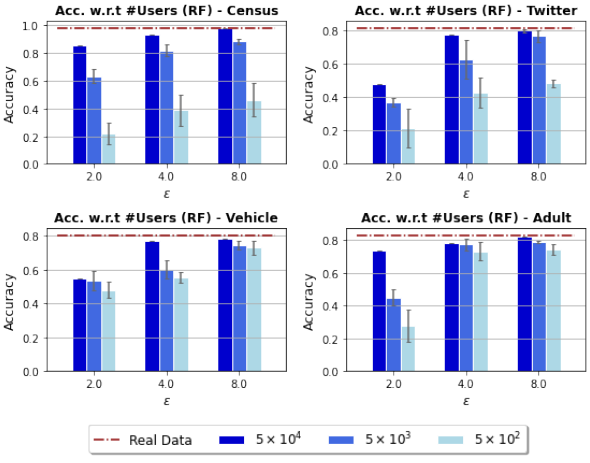

Fig. 14. Classification accuracy of the random forest (RF) with different number of users under different privacy levels.