FedDPSyn: Federated Tabular Data Synthesis with Computational Differential Privacy

Shufan Zhang[†1,2], Haochen Sun[†1], Karl Knopf[†1], Shubhankar Mohapatra[1], Wei Pang[1],
Calvin Wang[1], Yingke Wang[1], Masoumeh Shafieinejad[2], David Emerson[2], Xi He[1,2]
[1]University of Waterloo, [2]Vector Institute

Abstract

We study the problem of DP data synthesis when the data is horizontally distributed over federated data owners. A line of work has investigated the generation of synthetic tabular data with differential privacy (DP) guarantees. To the best of our knowledge, they all assume a trusted centralized server that executes their DP algorithms. In the federated setting, simply concatenating synthetic data independently generated by each data owner, with the centralized methods, causes large errors in terms of accuracy. In this work, we propose FedDPSyn, a new cryptographically-based federated data synthesis framework that (1) achieves similar accuracy levels as in the centralized model, (2) allows rewriting existing centralized data synthesis algorithms. Early empirical results show that our framework only incurs ~20 minutes overhead at the online phase, which is reasonable in practice.

1 Introduction

Generating synthetic tabular data with differential privacy (DP) guarantees has been an active area of research in recent years. Among the various approaches, marginal-based algorithms have gained particular prominence. Examples include PrivBayes [2, 39], HDMM [24], PrivSyn [40], Private-PGM [26], PrivMRF [6], and Kamino [15]. These methods have demonstrated strong results, with several emerging as winning approaches in NIST's differentially private synthetic data generation challenges [2, 5, 25, 36]. The core idea of marginal-based approaches is to estimate the joint distribution of the dataset with differential privacy and sample synthetic data from the joint distribution by post-processing. Since estimating a high-dimensional distribution often leads to high sensitivity, and thus large DP noise, the emerging wisdom in the field is to approximate the high-dimensional joint distribution with low-degree (noisy) distributions (a.k.a, marginals). These marginals can be estimated more accurately and combined based on the correlations among data using techniques such as Bayesian networks [39] and other probabilistic graphical models [6, 26].

While existing work has achieved adequate results with a centralized dataset, extending these approaches to federated settings remains underexplored. In particular, we consider the scenario in which a "global" dataset has been split horizontally and held by multiple mutually untrusted parties. These parties would like to collaboratively generate a synthetic dataset with differential privacy to study the patterns existing in the global dataset. For example, in the federated healthcare setting [12, 35], hospitals may want to jointly publish private synthetic data while preserving correlations among diseases and climates. Each hospital only has its local climate data or is specialized in a specific disease, and they do not want to, or cannot, share their local data in plaintext with each other.

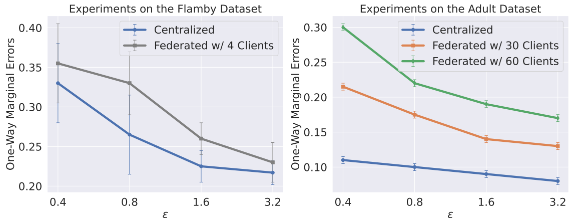

Figure 1: Motivating Experiments: Centralized Data Synthesis v.s. Federated Data Synthesis through Concatenation

One naïve solution is to have each client independently generate a differentially private synthetic dataset directly using existing algorithms (for the central setting) and then concatenate the resulting datasets. To quantify the utility gap between this approach and the synthetic data generated in the centralized setting, we performed experiments on the Adult [3] dataset and the Fed-Heart-Disease table, which is skewly distributed, in the FLamby [30] dataset using PrivBayes. We partition the dataset using random sampling without replacement to evenly distribute the data among the clients. We measure the errors of one-way marginals against the ground-truth global dataset. The results, presented in Figure 1, show that the synthetic data generated by concatenation in the federated setting exhibits more than double the error in almost all cases. In addition, with more clients the error increases.

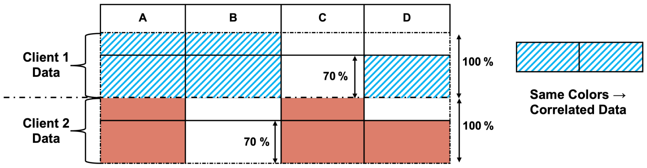

Figure 2: Dependency Errors across Federated Clients' Data

One reason for this large utility gap, demonstrated in Figure 2, could be explained through the dependency errors among the federated clients' data. In this example, attributes A and B are highly correlated (e.g., close to 100%) in the local data of the first client while A and C are highly correlated attributes for the second client. However, in the global dataset, both attributes A, B (50% correlation) and A, C (50% correlation) are not so correlated compared to attributes A and D which has 70% correlation. While each locally generated synthetic data may capture correlations between A, B (resp. A, C), they may miss the correlations between A, D, which is more important on the global dataset. We also notice that the learned correlations among attributes are less robust on a smaller support (i.e., fewer data) and sometimes cannot form a valid Bayesian network. For example, this happens when we split

[†] Co-first authors of this work.

the Fed-Heart-Disease dataset among 30 clients. Therefore, concatenating independent synthetic data is not desired as we would like to achieve comparable utility as in the centralized setting.

In this paper, we propose FedDPSyn, a new cryptographically-based framework that simulates running centralized data synthesis methods in the federated setting. From existing marginal-based data synthesis methods, we abstract the ideal functionalities that access the data and design primitives/protocols that implement these functionalities with fully homomorphic encryption (FHE) [10]. These FHE primitives allow clients to encrypt necessary data and coordinate servers to perform computation over encrypted data. Differential privacy mechanisms are applied to the results before decryption, so that the adversaries (i.e., server and other clients) either see data in the encrypted representation or differentially private computation results. We show that FedDPSyn can achieve similar privacy/utility results compared to centralized, differentially private data synthesis. To the best of our knowledge, aside from a recent study on vertical federated data synthesis [41], FedDPSyn is the first attempt to *securely generate tabular synthetic data* in horizontal federated settings. Our early experiments demonstrate that the runtime overhead, at the online phase, is less than 20 minutes over the encrypted marginal counts of the ADULT dataset, which is acceptable in practical federated scenarios.

2 Background

Marginal-based data synthesis methods contain the following typical steps: (1) **Marginal/Distribution Calculation** that computes marginal counts or (conditional) probabilities for measuring correlations among attributes; (2) **Marginal Selection** that selects the highly correlated pairs of attributes with DP; (3) **Noise Calibration** that adds DP noise to the low-way marginal counts; (4) **Post Processing** that combines the noisy low-way marginals with high correlation (into a graphical model); and (5) **Data Synthesis** that samples synthetic data from the graphical model.

Different data synthesis methods can have different instantiations of these steps. For example, PrivBayes [39] computes conditional probabilities for the mutual information score in Step 1, uses the exponential mechanism (EM) in Step 2, and calibrates Laplacian noise to marginals and combines them into a Bayesian network (Steps 3 and 4). We also introduce the necessary background of differential privacy and fully homomorphic encryption as follows.

Definition 1 (Differential Privacy (DP) [13, 14]). *A randomized algorithm $M : \mathcal{D} \to \mathcal{O}$ is (ϵ, δ)-DP if for any pair of neighbouring databases $D \sim D'$ (differing by replacing only one tuple), and all $O \subseteq \mathcal{O}$, $\Pr[\mathcal{M}(D) \in O] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \in O] + \delta$.*

Definition 2 (Computational Differential Privacy (CDP) [29]). *An ensemble $\{M_\kappa\}_{\kappa \in \mathbb{N}}$ of randomized algorithms $M_\kappa : \mathcal{D} \to \mathcal{O}_\kappa$ is ϵ_κ-CDP if for any ensemble of PPT adversaries $\{A_k\}_{\kappa \in \mathbb{N}}$, there exists $\mu(\kappa) \in \kappa^{-\omega(1)}$ such that for any pair of neighbouring databases $D \sim D'$, $\Pr\left[A_\kappa\left(M_\kappa(D)\right) = 1\right] \leq e^{\epsilon_\kappa} \Pr\left[A_\kappa\left(M_\kappa(D')\right) = 1\right] + \mu(\kappa)$.*

Definition 3 (Discrete Laplacian Mechanism [1, 17, 34]). *Given privacy budget ϵ and query $q : \mathcal{D} \to \mathbb{Z}$ with sensitivity $\Delta q := \max_{D \sim D'} |q(D) - q(D')|$, the discrete Laplacian mechanism $M(D)$ outputs $z \in \mathbb{Z}$ with probability proportional to $\exp\left(-\frac{\epsilon |z - q(D)|}{\Delta q}\right)$.*

Definition 4 (Exponential Mechanism [28]). *Given privacy budget ϵ and a utility function $u : \mathcal{O} \times \mathcal{D} \to \mathbb{R}$ with sensitivity $\Delta u := \max_{\substack{D \sim D' \\ o \in \mathcal{O}}} |u(o, D) - u(o, D')|$, the exponential mechanism $M(D)$ outputs $o \in \mathcal{O}$ with probability proportional to $\exp\left(\frac{\epsilon u(o,D)}{2\Delta u}\right)$.*

Fast Fully Homomorphic Encryption over the Torus (TFHE) [10] is an efficient fully homomorphic encryption scheme over the integers. It supports most integer operations as implemented in the standard libraries, including non-arithmetic ones like absolute values, comparison, and MUX (if-else branching). Throughout the paper, we use the notation $[\![\cdot]\!]$ to denote TFHE ciphertext.

3 Problem Setup

Consider a sensitive database D, split horizontally, into n disjoint subsets, with the same set of attributes, $D_1, \ldots, D_n$ such that $D = \cup_{i=1}^{n} D_i$. We use $|D_i|$ to denote the size of the data D_i. Each client i holds D_i. Several servers exist to facilitate the federated computation that aggregates the local results. The participating parties are semi-honest, i.e., they honestly execute the federated protocol while being interested in learning about the other clients' sensitive data. With this federated setting and threat model, we discuss several research questions and challenges.

RQ1: Secure Framework for Federated Data Synthesis. Prior work, such as Cryptϵ [11], explored cryptographic-based approaches to facilitate differentially private query processing without trusted curators. Cryptϵ uses partial homomorphic encryption, i.e., labeled HE, to support several database transformation operators and DP noise calibration measurements over local data owners while only incurring small errors comparable (~2X) to centralized DP. However, their approach does not naturally generalize to DP synthetic data generation. For example, operations widely used in marginal selection, such as branching/MUX, division, exponentiation, and *abs*() for calculating correlations and exponential mechanism for marginal selection, are not compatible with the set of cryptographic choices in existing work.

We would like to design a new cryptographically-based framework for federated DP data synthesis. We abstract and generalize the ideal functionalities that access the data, as required for marginal-based DP data synthesis methods. We re-design these data-accessing modules, which we call *primitives*, and replace them with the corresponding cryptographic implementation. Using these building blocks in FedDPSyn, existing marginal-based DP synthetic data generation algorithms can be translated and rewritten into the federated setting.

RQ2: Design and Implementation of the Primitives with TFHE. While TFHE supports a variety of operations over encrypted data, the representation of ciphertext is over an algebraic structure called the torus. That is, arithmetic computations are performed over the integer domain and floating-point operations have to be converted to the fixed-point representation by upscaling and truncation. Although some of the data-accessing modules in data synthesis, such as aggregations, are straightforward to implement with TFHE, we still need to carefully handle the scaling factors. In addition, other modules that utilize the exponential mechanism have involved calculations of the utility function and sampling probabilities, incurring more numerical/precision errors that can lead to

```
Primitive 1: Encrypted Aggregation
  Input: Database at each party $D_i$, Selected attribute $attr_j$,
         Flag for enabling DP noise DPFlag, Privacy
         parameter $\epsilon_1$.
  Output: $[\![\mathrm{Ct}[attr_j \mid D]]\!]$ over $D = \cup_{i=1}^{n} D_i$
  // Client-side computation:
  Each party $i$ calculates
    $\mathrm{Ct}[attr_j \mid D_i] \leftarrow \Pr[attr_j \mid D_i] \cdot |D_i|$;
  $[\![\mathrm{Ct}[attr_j \mid D_i]]\!] \leftarrow Enc(\mathrm{Ct}[attr_j \mid D_i])$;
  // Server-side computation:
  $[\![\mathrm{Ct}[attr_j \mid D]]\!] \leftarrow \sum_{i=1}^{n} [\![\mathrm{Ct}[attr_j \mid D_i]]\!]$;
  if $DPFlag = True$ then
    $[\![\widetilde{\mathrm{Ct}}[attr_j \mid D]]\!] \leftarrow [\![\mathrm{Ct}[attr_j \mid D]]\!] + \mathrm{DLM}(\epsilon_1, \Delta q = 1)$;
  end
  return $[\![\widetilde{\mathrm{Ct}}[attr_j \mid D]]\!]$ if DPFlag else $[\![\mathrm{Ct}[attr_j \mid D]]\!]$
```

privacy issues [20]. Thus, it is non-trivial to implement these modules with fixed-point operations and bound the numerical/precision errors with minimal privacy cost.

In this paper, we design new DP mechanisms, i.e., three *TFHE primitives*, that form the basis of the data-accessing modules in FedDPSyn. We show that with little additional privacy cost (in terms of DP budget), our instantiation of DP algorithms over TFHE can bound the numerical errors. It is anticipated that these algorithms are likely to be of broader use beyond their application here in FedDPSyn.

4 FedDPSyn Overview

We provide an overview of FedDPSyn in terms of the roles of the participating parties and the new algorithms designed for it.

4.1 Assumptions and System Entities

In FedDPSyn, there exists a set of data owners or clients and two *non-colluding* servers – a key management server and a computation server. This two-server assumption is commonly adopted in prior work that involves multi-party computation or homomorphic encryption [11, 18, 33].

Key Management Server. The key management server initializes the TFHE scheme and generates the key pair $\langle sk, ck, pk \rangle$. It stores the secret key sk, sends the computation key ck to the computation server, and broadcasts the public key pk to the clients in the protocol. The key management server is the only party capable of decrypting encrypted values in the protocol. Before decryption, it follows the protocol and adds a portion of DP noise to the results.

Data Owners (Clients). Each data owner represents their own private data D_i, and all data owners share the same public key for encryption. Data owners can perform computation on the plaintext of their data D_i and encrypt the computational results using the public key pk. The encrypted results are sent to the computation server for encrypted computations.

Computation Server. The computation server receives the encrypted data from the clients and uses the computation key ck for aggregation and computation over encrypted data. It executes the primitives as specified in the protocols.

```
Primitive 2: Encrypted Dependency Score Calculation
  Input: Database at each party $D_i$, Selected attribute pair
         $attr_j, attr_k$.
  Output: InDif* Score $[\![|M_{j,k} - M_j \times M_k| \cdot |D|]\!]$ over
          attribute $attr_j, attr_k$ on $D = \cup_{i=1}^{n} D_i$.
  // Client-side computation:
  Each party $i$ calculates
    $([\![\mathrm{Ct}[attr_j \mid D_i]]\!], [\![\mathrm{Ct}[attr_k \mid D_i]]\!], [\![\mathrm{Ct}[(attr_j, attr_k) \mid D_i]]\!])$;
  // Server-side computation:
  Server aggregates using Primitive 1:
    $([\![\mathrm{Ct}[attr_j \mid D]]\!], [\![\mathrm{Ct}[attr_k \mid D]]\!], [\![\mathrm{Ct}[(attr_j, attr_k) \mid D]]\!])$;
  $\mathrm{InDif}^* \leftarrow 0$;
  for $(e_1, e_2) \in Dom(attr_j) \times Dom(attr_k)$ do
    $[\![v_1]\!] \leftarrow [\![\mathrm{Ct}[attr_j = e_1 \mid D]]\!]$;
    $[\![v_2]\!] \leftarrow [\![\mathrm{Ct}[attr_k = e_2 \mid D]]\!]$;
    $[\![v^*]\!] \leftarrow [\![\mathrm{Ct}[(attr_j, attr_k = (e_1, e_2) \mid D]]\!]$;
    $[\![\mathrm{InDif}^*]\!] += abs([\![v_1]\!] \cdot [\![v_2]\!] - [\![v^*]\!] \cdot [\![|D|]\!])$;
  end
  return $[\![\mathrm{InDif}^*]\!]$
```

4.2 TFHE Primitives Design

The system workflow in terms of how these entities interact with each other is deferred to Appendix B. Among the interactions of system entities, computations that access the local dataset are: (1) calculating marginal counts or conditional distributions; (2) selecting marginals with high correlation; (3) calibrating DP noise. These data-accessing computations need to be executed over ciphertexts. Here, the design details of the DP primitives to support the encrypted computation step in the system workflow are discussed.

Primitive 1: Encrypted Aggregation. We design **Primitive 1** to aggregate the encrypted marginal counts from the clients. Notice that encrypting the marginal distribution over the j-th attribute for the i-th client's data $\Pr[attr_j \mid D_i]$ and computing over the encrypted probabilities can incur numerical errors due to the scaling and truncation of the floating points. Instead, we encrypted the counts $\mathrm{Ct}[attr_j \mid D_i] \coloneqq \Pr[attr_j \mid D_i] \cdot |D_i|$ for every possible values in the discretized domain $\mathrm{Dom}(attr_j)$ of the selected attribute $attr_j$ (bucketized for continuous attributes). These encrypted counts are sent to the computation server. The server executes the TFHE add operator to obtain the aggregated marginal counts over the global dataset D. If the DPFlag is set to True (as in the noise calibration step), the server invokes the discrete Laplacian mechanism (DLM) to generate an integer noise with privacy budget ϵ_1 and sensitivity 1 (for linear counting queries). The noise is then added to the encrypted counts.

Primitive 2: Encrypted Dependency Score Calculation. To calculate the dependency score, we demonstrate the design of **Primitive 2** on the InDif ($\coloneqq |M_{j,k} - M_j \times M_k|$) measure in PrivSyn [40], where $M_j \coloneqq \Pr[attr_j \mid D]$ and $M_{j,k} \coloneqq \Pr[attr_j, attr_k \mid D]$. The idea can also be extended to the "F function" in PrivBayes [39]. Again, to avoid numerical issues, we compute the modified $\mathrm{InDif}^* \coloneqq \mathrm{InDif} \cdot |D|$ upscaled by the size of the global database. To compute encrypted InDif^*, the clients and the server first execute Primitive

```
Primitive 3: TFHE-EM
Input: Set of outcomes $O = \{o_0, o_1, \dots o_{|O|-1}\}$, Privacy
       parameter $\epsilon$, Proper choices of parameters for
       fixed-point arithmetic params.
Output: $[[o^*]]$, an encrypted member of $O$.
// Client-side computation:
for $j \leftarrow 0, 1, \dots, |O| - 1$ do
    Each client $i$ sends $[[U(o_j, D_i)]]$ to server;
end
// Server-side computation:
for $j \leftarrow 0, 1, \dots, |O| - 1$ do
    Server aggregates with Primitive 2 and gets
      $[[U_j]] \coloneqq [[U(o_j, D)]]$
end
$[[C_{-1}]] \leftarrow 0$;
for $j \leftarrow 0, 1, \dots |O| - 1$ do
    $[[W_j]] \leftarrow$ TFHE-EXP($[[U_j]], \epsilon$, params);    ▷ Appendix C
    $[[C_j]] \leftarrow [[C_{j-1}]] + [[W_j]]$;
end
Sample random 32-bit integer $R$;               ▷ Need $2^{n_b} \ll 2^{32}$
$[[R']] \leftarrow R$ mod $[[C_{j-1}]]$;
return $\left(\mathbf{1}\left\{[[C_{j-1}]] \le [[R']] < [[C_j]]\right\}\right)_{j=0}^{|O|-1}$;     ▷ One-hot
```

1 to obtain the aggregated counts for the selected one-way and two-way marginals, i.e., $attr_j$, $attr_k$, $(attr_j, attr_k)$. Then, on the server side, it initializes the InDif score to 0 and iterates over the Cartesian product of the domain of $attr_j$ and $attr_k$. The server calculates the product of the one-way marginals and the product of the two-way marginal and the aggregated database size. The server takes the absolute value of the difference between the two products and adds the result to the InDif* score. This step utilizes the TFHE mult, minus, and abs operators. Note that potential numerical errors were avoided in Primitive 1 and Primitive 2. Next, the overall privacy cost due to numerical errors accrued in Primitive 3 is bounded.

Primitive 3: Exponential Mechanism under TFHE (TFHE-EM). To effectively select a pair of attributes with high correlation while preserving privacy, we apply the exponential mechanism [28] to the selection process. Specifically, the input to the exponential mechanism is the correlation (e.g., InDif* score), $u(o, D)$, computed for each pair of attributes $o = (attr_j, attr_k)$ in a dataset D. Then, one pair o^* is selected as the output, with the probability of selecting each pair o proportional to $\exp(\epsilon u(o, D))$. For simplicity, we assume that all $u(o, D)$s are negative, or they can all be shifted by a constant.

The design of **Primitive 3** takes into account the numerical issues in the exponential mechanism [20], which could potentially exacerbate the unintentional privacy loss under the fixed-point arithmetic otherwise. As shown in Theorem 4.1, the additional privacy loss is appropriately controlled: when desired $\epsilon = 1$, the additional privacy cost is less than 0.005. More details of the TFHE-EXP function and the formal privacy guarantee can be found in Appendix C. The proof of Theorem C.1 is deferred to the full paper.

Theorem 4.1 (Privacy with Bounded Numerical Error, Informal). *Primitive 3 is $(\epsilon + 4e')$-DP, where ϵ is the originally desired privacy cost, and e' is the additional privacy leakage due to numerical errors in the fixed-point arithmetic accumulated in Primitives 1, 2 and 3, such that $4e' \le 0.05\epsilon$ under proper choice of hyperparameters.*

Table 1: Simulated Server Runtime on an NVIDIA RTX A6000

Primitive 1	Primitive 2	Primitive 3	Total
290s	260s	560s	1.1×10^3s

4.3 Privacy Analysis

We consider the privacy leakage on the entire dataset (distributed among all clients) against any adversary, including each of the non-colluding servers and any other external party receiving the result. Denote as ϵ_1, ϵ_2 the privacy budget assigned to Primitives 1 and 3, respectively. In particular, ϵ_1 is distributed among the multiple (pairs of) attributes, and by Theorem C.1, Primitive 3 further incurs an unintentional privacy leakage of $4e'$ due to numerical issues. Moreover, due to the computational hiding nature of TFHE ciphertext, only the weaker notion of computational DP can apply to the view of either server during the execution of FedDPSyn. The privacy guarantee is stated as Theorem 4.2, and we defer a more detailed privacy analysis of FedDPSyn to the full paper.

Theorem 4.2. *FedDPSyn is $(\epsilon_1 + \epsilon_2 + 4e')$-differentially private. The view of each of the two non-colluding servers in FedDPSyn is $(\epsilon_1 + \epsilon_2 + 4e')$-computationally differentially private.*

5 Early Experimental Results

To evaluate our framework, we carry out a study on the cost of adapting PrivSyn [40] to our setting. We implement the DP primitives with TFHE-rust [38]. We use the ADULT dataset with its 15 attributes and 32,561 rows. If we bucketize the continuous attributes to a domain of 10, the average size of a one-way marginal is 11 and the average size of a two-way marginal is 124. The case study assumes there are 4 clients, and the original dataset is split uniformly at random between them. In the centralized setting, PrivSyn will create a synthetic dataset for a quarter of the ADULT dataset in 1.08 seconds (average of 16 trials). In the offline phase, it takes approximately 19 hours for each client to perform and encrypt the marginal counts. Meanwhile, we simulate the running times of the servers in the three primitives of the online phase and report them in Table 1, as the average of 4 trials. It can be observed that the online phase can still be executed with a reasonable overhead of less than 20 minutes. On the other hand, future advances in the public key encryption of TFHE, which is only used in the offline phases performed by the clients, can further improve the practicality of the entire pipeline.

6 Concluding Remarks

We propose FedDPSyn, a cryptographic-based framework that enables federated data synthesis with differential privacy. FedDPSyn enables the re-designed DP algorithms with TFHE and bounds the privacy cost due to numerical issues in cryptographic operations. In future work, FedDPSyn will be implemented into a fully-functioning prototype. In this system, existing private marginal-based data synthesis methods may be applied in the federated setting in a straightforward manner with strong theoretical privacy guarantees. The framework of FedDPSyn can also be extended to the 3-server MPC setting, where trade-offs among these cryptographic approaches for data synthesis will be better understood.

References

[1] Victor Balcer and Salil P. Vadhan. 2018. Differential Privacy on Finite Computers. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA (LIPIcs, Vol. 94)*, Anna R. Karlin (Ed.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 43:1–43:21. doi:10.4230/LIPICS.ITCS.2018.43

[2] Ergute Bao, Xiaokui Xiao, Jun Zhao, Dongping Zhang, and Bolin Ding. 2021. Synthetic Data Generation with Differential Privacy via Bayesian Networks. *J. Priv. Confidentiality* 11, 3 (2021). doi:10.29012/JPC.776

[3] Barry Becker and Ronny Kohavi. 1996. Adult. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5XW20.

[4] Monik Raj Behera, Sudhir Upadhyay, Suresh Shetty, Sudha Priyadarshini, Palka Patel, and Ker Farn Lee. 2022. FedSyn: Synthetic Data Generation using Federated Learning. *CoRR* abs/2203.05931 (2022). doi:10.48550/ARXIV.2203.05931 arXiv:2203.05931

[5] Claire McKay Bowen and Joshua Snoke. 2021. Comparative Study of Differentially Private Synthetic Data Algorithms from the NIST PSCR Differential Privacy Synthetic Data Challenge. *J. Priv. Confidentiality* 11, 1 (2021). doi:10.29012/JPC.748

[6] Kuntai Cai, Xiaoyu Lei, Jianxin Wei, and Xiaokui Xiao. 2021. Data Synthesis via Differentially Private Markov Random Field. *Proc. VLDB Endow.* 14, 11 (2021), 2190–2202. doi:10.14778/3476249.3476272

[7] Health Data Centre. 2022. Federated learning and synthetic data generation. https://www.hh.se/health-data-centre/health-data-centre-english/projects/federated-learning-and-synthetic-data-generation.html.

[8] Yihang Cheng, Lan Zhang, and Anran Li. 2023. GFL: Federated Learning on Non-IID Data via Privacy-Preserving Synthetic Data. In *IEEE International Conference on Pervasive Computing and Communications, PerCom 2023, Atlanta, GA, USA, March 13-17, 2023*. IEEE, 61–70. doi:10.1109/PERCOM56429.2023.10099110

[9] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yong Soo Song. 2017. Homomorphic Encryption for Arithmetic of Approximate Numbers. In *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 10624)*, Tsuyoshi Takagi and Thomas Peyrin (Eds.). Springer, 409–437. doi:10.1007/978-3-319-70694-8_15

[10] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. 2020. TFHE: Fast Fully Homomorphic Encryption Over the Torus. *Journal of Cryptology* 33, 1 (2020), 34–91. doi:10.1007/s00145-019-09319-x

[11] Amrita Roy Chowdhury, Chenghong Wang, Xi He, Ashwin Machanavajjhala, and Somesh Jha. 2020. Cryptϵ: Crypto-Assisted Differential Privacy on Untrusted Servers. In *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, David Maier, Rachel Pottinger, AnHai Doan, Wang-Chiew Tan, Abdussalam Alawini, and Hung Q. Ngo (Eds.). ACM, 603–619. doi:10.1145/3318464.3380596

[12] D. B. Emerson, J. Jewell, S. Ayromlou, S. Carere, F. Tavakoli, Y. Zhang, M. Lotif, and A. Krishnan. 2025. *FL4Health*. https://github.com/vectorInstitute/FL4Health/

[13] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings (Lecture Notes in Computer Science, Vol. 3876)*, Shai Halevi and Tal Rabin (Eds.). Springer, 265–284. doi:10.1007/11681878_14

[14] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* 9, 3-4 (2014), 211–407. doi:10.1561/0400000042

[15] Chang Ge, Shubhankar Mohapatra, Xi He, and Ihab F. Ilyas. 2021. Kamino: Constraint-Aware Differentially Private Data Synthesis. *Proc. VLDB Endow.* 14, 10 (2021), 1886–1899. doi:10.14778/3467861.3467876

[16] Ali Reza Ghavamipour, Fatih Turkmen, Rui Wang, and Kaitai Liang. 2023. Federated Synthetic Data Generation with Stronger Security Guarantees. In *Proceedings of the 28th ACM Symposium on Access Control Models and Technologies, SACMAT 2023, Trento, Italy, June 7-9, 2023*, Silvio Ranise, Roberto Carbone, and Daniel Takabi (Eds.). ACM, 31–42. doi:10.1145/3589608.3593835

[17] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. 2012. Universally Utility-maximizing Privacy Mechanisms. *SIAM J. Comput.* 41, 6 (2012), 1673–1693. doi:10.1137/09076828X

[18] Tiantian Gong, Ryan Henry, Alexandros Psomas, and Aniket Kate. 2024. More is Merrier: Relax the Non-Collusion Assumption in Multi-Server PIR. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*. IEEE, 4348–4366. doi:10.1109/SP54263.2024.00095

[19] Moritz Hardt and Guy N. Rothblum. 2010. A Multiplicative Weights Mechanism for Privacy-Preserving Data Analysis. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*. IEEE Computer Society, 61–70. doi:10.1109/FOCS.2010.85

[20] Christina Ilvento. 2020. Implementing the Exponential Mechanism with Base-2 Differential Privacy. In *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna (Eds.). ACM, 717–742. doi:10.1145/3372297.3417269

[21] Claire Little, Mark Elliot, and Richard Allmendinger. 2023. Federated learning for generating synthetic data: a scoping review. *International Journal of Population Data Science* 8, 1 (2023), 2158.

[22] Eugenio Lomurno and Matteo Matteucci. 2024. Federated Knowledge Recycling: Privacy-Preserving Synthetic Data Sharing. *CoRR* abs/2407.20830 (2024). doi:10.48550/ARXIV.2407.20830 arXiv:2407.20830

[23] Samuel Maddock, Graham Cormode, and Carsten Maple. 2024. FLAIM: AIM-based Synthetic Data Generation in the Federated Setting. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, Ricardo Baeza-Yates and Francesco Bonchi (Eds.). ACM, 2165–2176. doi:10.1145/3637528.3671990

[24] Ryan McKenna, Gerome Miklau, Michael Hay, and Ashwin Machanavajjhala. 2018. Optimizing error of high-dimensional statistical queries under differential privacy. *Proc. VLDB Endow.* 11, 10 (2018), 1206–1219. doi:10.14778/3231751.3231769

[25] Ryan McKenna, Gerome Miklau, and Daniel Sheldon. 2021. Winning the NIST Contest: A scalable and general approach to differentially private synthetic data. *J. Priv. Confidentiality* 11, 3 (2021). doi:10.29012/JPC.778

[26] Ryan McKenna, Daniel Sheldon, and Gerome Miklau. 2019. Graphical-model based estimation and inference for differential privacy. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 4435–4444. http://proceedings.mlr.press/v97/mckenna19a.html

[27] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA (Proceedings of Machine Learning Research, Vol. 54)*, Aarti Singh and Xiaojin (Jerry) Zhu (Eds.). PMLR, 1273–1282. http://proceedings.mlr.press/v54/mcmahan17a.html

[28] Frank McSherry and Kunal Talwar. 2007. Mechanism Design via Differential Privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*. IEEE Computer Society, 94–103. doi:10.1109/FOCS.2007.41

[29] Ilya Mironov, Omkant Pandey, Omer Reingold, and Salil P. Vadhan. 2009. Computational Differential Privacy. In *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings (Lecture Notes in Computer Science, Vol. 5677)*, Shai Halevi (Ed.). Springer, 126–142. doi:10.1007/978-3-642-03356-8_8

[30] Jean Ogier du Terrail, Samy-Safwan Ayed, Edwige Cyffers, Felix Grimberg, Chaoyang He, Regis Loeb, Paul Mangold, Tanguy Marchand, Othmane Marfoq, Erum Mushtaq, et al. 2022. Flamby: Datasets and benchmarks for cross-silo federated learning in realistic healthcare settings. *Advances in Neural Information Processing Systems* 35 (2022), 5315–5334.

[31] Mayana Pereira, Sikha Pentyala, Anderson C. A. Nascimento, Rafael T. de Sousa Jr., and Martine De Cock. 2022. Secure Multiparty Computation for Synthetic Data Generation from Distributed Data. *CoRR* abs/2210.07332 (2022). doi:10.48550/ARXIV.2210.07332 arXiv:2210.07332

[32] Bjarne Pfitzner and Bert Arnrich. 2022. DPD-fVAE: Synthetic Data Generation Using Federated Variational Autoencoders With Differentially-Private Decoder. *CoRR* abs/2211.11591 (2022). doi:10.48550/ARXIV.2211.11591 arXiv:2211.11591

[33] Theodoros Rekatsinas, Amol Deshpande, and Ashwin Machanavajjhala. 2013. A SPARSI: Partitioning Sensitive Data amongst Multiple Adversaries. *Proc. VLDB Endow.* 6, 13 (2013), 1594–1605. doi:10.14778/2536258.2536270

[34] Haochen Sun and Xi He. 2025. VDDP: Verifiable Distributed Differential Privacy under the Client-Server-Verifier Setup. *arXiv preprint arXiv:2504.21752* (2025).

[35] Fatemeh Tavakoli, D. B. Emerson, Sana Ayromlou, John Taylor Jewell, Amrit Krishnan, Yuchong Zhang, Amol Verma, and Fahad Razak. 2024. A Comprehensive View of Personalized Federated Learning on Heterogeneous Clinical Datasets. In *Proceedings of the 9th Machine Learning for Healthcare Conference (Proceedings of Machine Learning Research, Vol. 252)*, Kaivalya Deshpande, Madalina Fiterau, Shalmali Joshi, Zachary Lipton, Rajesh Ranganath, and Iñigo Urteaga (Eds.). PMLR. https://proceedings.mlr.press/v252/tavakoli24a.html

[36] Tianhao Wang, Ninghui Li, and Zhikun Zhang. 2021. DPSyn: Experiences in the NIST Differential Privacy Data Synthesis Challenges. *J. Priv. Confidentiality* 11, 2 (2021). doi:10.29012/JPC.775

[37] Bangzhou Xin, Wei Yang, Yangyang Geng, Sheng Chen, Shaowei Wang, and Liusheng Huang. 2020. Private FL-GAN: Differential Privacy Synthetic Data Generation Based on Federated Learning. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, 2927–2931. doi:10.1109/ICASSP40776.2020.9054559

[38] Zama. 2022. TFHE-rs: A Pure Rust Implementation of the TFHE Scheme for Boolean and Integer Arithmetics Over Encrypted Data. https://github.com/zama-ai/tfhe-rs.

[39] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. 2017. PrivBayes: Private Data Release via Bayesian Networks. *ACM Trans. Database Syst.* 42, 4 (2017), 25:1–25:41. doi:10.1145/3134428

[40] Zhikun Zhang, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. 2021. PrivSyn: Differentially Private Data Synthesis. In *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, Michael D. Bailey and Rachel Greenstadt (Eds.). USENIX Association, 929–946. https://www.usenix.org/conference/usenixsecurity21/presentation/zhang-zhikun

[41] Fangyuan Zhao, Zitao Li, Xuebin Ren, Bolin Ding, Shusen Yang, and Yaliang Li. 2024. VertiMRF: Differentially Private Vertical Federated Data Synthesis. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, Ricardo Baeza-Yates and Francesco Bonchi (Eds.). ACM, 4431–4442. doi:10.1145/3637528.3671771

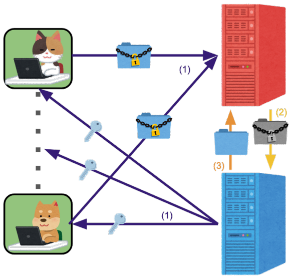

Figure 3: System Overview of FedDPSyn

A Related Work

Synthetic dataset generation in the federated setting has emerged as a practical need in critical sectors such as healthcare [7, 21] and finance [4]. Several efforts have been made to adapt generative models to the federated setting using the FedAvg paradigm [27], including FL-GAN [37], FedSyn [4], DPD-FVAE [32], and FedKR [22]. By adopting DP-FedAvg, the proposed methods can be used for private data synthesis. Some other work explores how private synthetic data can improve the results of federated learning [8]. However, these investigations are all limited to image datasets.

The closest work to ours are perhaps PP-FedGAN [16], MPC-MWEM [31], VertiMRF [41], and FLAIM [23]. PP-FedGAN [16] uses the CKKS homomorphic encryption scheme [9] to design a GAN model for synthesizing tabular data, while it cannot be generalized to the state-of-the-art methods for tabular data synthesis with DP, i.e., the marginal-based methods. MPC-MWEM [31] proposes a method that implements the differentially private multiplicative weights update mechanism (MWEM) [19] over the multi-party computation framework. Though it can be used for federated tabular data synthesis, MPC-MWEM is designed for a small number of distributed clients that are all available for secret-share data. VertiMRF [41] presents an approach for generating synthetic tabular data in the federated setting *where the data is vertically partitioned.* FLAIM [23] studies the same problem setting as our work, and translates the AIM algorithm, a.k.a, a variant of Private-PGM [26], into the federated learning setting. FLAIM incorporates secure aggregation for model updates with differential privacy. While FLAIM can run much faster than our framework FedDPSyn, experimental results of FLAIM show a relatively large gap of workload errors between the federated and the centralized setting. FedDPSyn can be regarded as a complement to FLAIM in terms of the trade-off between runtime overhead and accuracy of the synthetic data generated.

B FedDPSyn Workflow

A system diagram is demonstrated in Figure 3. Among the five steps for marginal-based data synthesis, the first three steps, i.e., probability calculation, marginal selection, and noise calibration, have accessed data from the federated data owners. These data-accessing steps are computed over the encrypted representation–we wrap them up into *encrypted computation* and also introduce the other steps in the workflow.

```
SubRoutine 1: TFHE-EXP
  Input: Encrypted utility score $[[U]]$, Privacy parameter $\epsilon$,
         Hyperparameters of fixed-point arithmetic
         $\kappa, \vec{\kappa}', n_b, \lambda$.
  Output: $[[o^*]]$, an encrypted member of $O$.
  Function TFHE-EXP($[[U]], \epsilon$; PARAMS $= \{\kappa, \vec{\kappa}', n_b, \lambda\}$) :
      $[[U]] \leftarrow \max([[U]], -2^{n_b} + 1)$;                ▷ Clamping
      $y_0 \leftarrow 2^{\kappa}$          ▷ Final result has a scaling factor of $2^{\kappa}$
      for $j \in 0, \ldots, n_b - 1$ do
          $[[b_j]] \leftarrow (-[[U]]) \wedge 1$;     ▷ Taking the last bit of $|U|$
          $c_j = \left\lfloor 2^{\kappa'_j} \exp(-\epsilon 2^{j-\lambda}) \right\rceil$;              ▷ Pre-computed
          $[[t_j]] \leftarrow 2^{\kappa'_j} - [[b_j]](2^{\kappa'_j} - c_j)$;
          $[[y_{j+1}]] \leftarrow \left([[y_j]][[t_j]] + 2^{\kappa'_j - 1}\right) >> \kappa'_j$;     ▷ Adjust
            the scaling factor by right shifting $\kappa'_j$ bits
          $[[U]] >>= 1$;                      ▷ Right shift $U$ by 1 bit
      end
      return $[[y_{n_b}]]$
  end
```

Initialization. The key management server initializes the key-generation protocol and distributes the public key to each client and the computation key to the computation server.

Encrypted Computation. (1) Probability Calculation: each client computes all the one-way and low-way marginal counts (or the conditional probabilities), and sends the encrypted representation to the computation server. The computation server then aggregates the encrypted counts. (2) Marginal Selection: the computation server calculates the dependency scores based on the aggregated probabilities. It then invokes the TFHE exponential mechanism over the encrypted dependency scores to identify the highly correlated attribute pairs. The computation server notifies the clients about these attribute pairs. (3) Noise Calibration: the computation server adds DP noise to the selected marginals (with high correlation) and sends them to the key management server. The key management server adds another portion of DP noise.

Decryption of Results. The key management server decrypts the noisy marginals and broadcasts to the clients and the computation server.

Post Processing and Data Synthesis. The combination of the noisy marginals and the data synthesis steps are then performed on the decrypted results.

C Missing Details of the TFHE Exponential Mechanism

Here we show our design of the exponentiation function executed with TFHE, i.e., TFHE-EXP, in SubRoutine 1. The utility score u is represented in the fixed-point form $U \approx 2^{\lambda} u$ as input, where U is an integer that is encrypted via TFHE, and λ is the number of fraction digits allowed in the fixed-point arithmetic. The difference $e_{\text{prev}} = \left| 2^{-\lambda} U - u \right|$ bounds the numerical errors incurred in the previous computations, i.e., in Primitive 1 and 2. We denote the number of bits used in the fixed-point computation by n_b. The

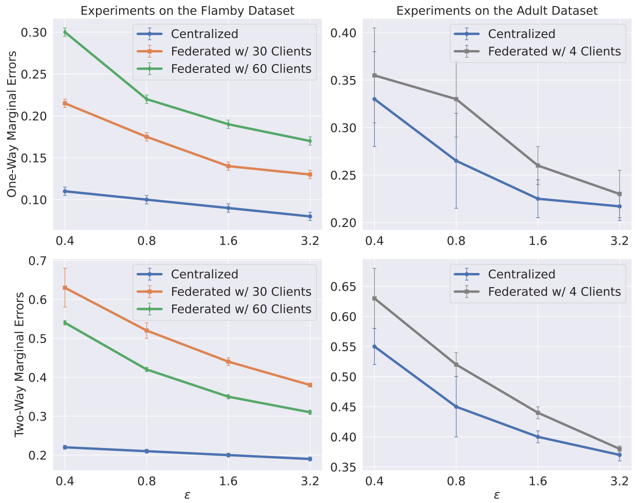

Figure 4: Motivating Experiments over One-way and Two-way Errors

algorithm also requires the scaling factor κ for the final result, and a list of scaling factors $\vec{\kappa}' = \{\kappa'_j\}$ for each bit $j \in [n_b]$ during the computation.

Now we present the formal privacy guarantee with bounded numerical issues.

Theorem C.1 (Privacy w/ Bounded Numerical Error). *Let e_{prev} be the accumulated numerical error on the input, i.e., $e_{prev} = \left|2^{-\lambda}U - u\right|$, and $e_{num}(\eta)$ be a small quantity for large ξ such that $e_{num}(\xi) = -\ln\left(1 - \frac{1}{2\xi}\right)$. Primitive 3 is $(\epsilon + 4e')$-DP, where $e' \leq \epsilon \cdot e_{prev} + \sum_{j=0}^{n_b-1}\left(e_{num}(2^{\kappa'_j}\exp(-2^{j-\lambda}\epsilon)) + e_{num}(\overline{y_{j+1}})\right)$. Here, $\overline{y_{j+1}}$ is the smallest possible value achievable in Line 7 of **SubRoutine 1** over all possible input values.*

We run numerical simulations to compute the additional privacy cost incurred due to numerical errors with different overall privacy budgets ϵ and a proper set of hyperparameters. The results are listed in Table 2. For each ϵ tested, the numerical error is bounded by $4e' \leq 0.05\epsilon$.

Table 2: Additional Privacy Cost in TFHE-EM, with $n_b = 17$ and $\kappa = \kappa'_j = 13$ for each $0 \leq j \leq n_b - 1$

ϵ	0.1000	0.3000	1.0000	3.0000	10.0000
e'	0.0011	0.0011	0.0012	0.0015	0.0986
$\epsilon + 4e'$	0.1044	0.3044	1.0048	3.0060	10.3944

D Full Results for the Motivating Experiments (Fig 4)