Abstract—Differential privacy (DP) is considered a de-facto standard for protecting users’ privacy in data analysis, machine, and deep learning. Existing DP-based privacy-preserving approaches, in federated learning, consist of adding noise to the clients’ gradients before sharing them with the server. However, implementing DP on the gradient is inefficient as the privacy leakage increases by increasing the synchronization training epochs due to the composition theorem. Recently, researchers were able to recover images of the training dataset using a Generative Regression Neural Network (GRNN).

In this work, we propose a novel approach using two layers of privacy protection to overcome the limitations of the existing DP-based methods. The first layer leverages Hensel’s Lemma to reduce the training dataset’s dimension. The new dimensionality reduction method reduces the dimension of a dataset without losing information since Hensel’s Lemma guarantees uniqueness. The second layer applies DP to the compressed dataset generated by the first layer. The proposed approach overcomes the problem of privacy leakage due to composition by applying DP only once before the training. Therefore, clients train their local model on the privacy-preserving dataset generated by the second layer. Experimental results show that the proposed approach ensures strong privacy protection while achieving high accuracy. In particular, the new dimensionality reduction method achieves an accuracy of 97%, with only 25% of the original dataset size.

Index Terms—Federated learning, privacy protection, differential privacy, dimensionality reduction, Hensel’s compression.

I. INTRODUCTION

Federated learning (FL) is a machine-learning technique where multiple clients (e.g., devices or organizations) collaboratively train a model under the supervision of a central server. The clients train the learning model on their local datasets and send the updated gradient to the central server. The server calculates the mean of the received gradients and sends the new value of the global gradient to clients for the next training epoch. This process is repeated until getting the trained model. Thanks to its advantages compared to traditional (i.e., centralized) machine learning, FL has received significant interest in the literature. In addition to mitigating computational load on the central server, FL allows training a model on large-scale datasets while protecting the users’ privacy.

Although FL ensures a certain level of privacy by not explicitly sharing the data with the server, an attacker (or the server) could retrieve a client’s training dataset using only the shared gradient [29]. Differential privacy (DP) [8, 6, 20] is an approach to solving these privacy issues. In a DP method, clients protect gradients before sending them to the server by adding noise drawn from a probability distribution [11, 28, 27]. However, applying DP at each synchronization epoch degrades the privacy protection due to the composition theorem [7]. For example, if a client applies DP, at each synchronization round, with a privacy leakage $\epsilon$, then after $n$ epochs, the privacy leakage becomes $n \times \epsilon$. Thus, a malicious server or an attacker could learn a tighter estimate of the clients’ gradients.

To control the privacy leakage, authors in, e.g., [25, 13, 2] proposed an approach to determining the standard deviation of the Gaussian distribution not to exceed a predefined privacy leakage $\epsilon$ after $n$ synchronization epochs. Nevertheless, these approaches do not enhance privacy protection because the determined standard deviation depends on the number of synchronization epochs $n$, and eventually, the privacy leakage increases by increasing $n$. Another category of works proposes to handle the privacy leakage challenges by training the FL model via peer-to-peer communications [4, 24, 15]. In these works, the server sends the initialized gradient to a client chosen randomly from all clients. Then, this client updates the received global gradient from the server and sends it to another client, and so on, until the last client sends the updated global gradient to the server. These works ensure robust protection of users’ privacy; however, they are vulnerable to label-flipping, and data poisoning attacks [10, 9].

Moreover, recently Ren et al. [22] succeeded in recovering the training dataset even when the gradient was protected using DP. First, the authors generated a fake image input with its corresponding label using a generative regression neural network model (GRNN). Then they fed this image to the training model at the server to calculate the fake gradient $\hat{g}$. Finally, retrieving the original training images is done by training the GRNN model by minimizing the distance between the fake gradient $\hat{g}$ and the true gradient $g$. The method is based on two main components to complete the training: 1) the resolution of the target image and 2) the length of the true gradient vector $g$.

To overcome the challenges mentioned above, we propose a novel privacy-preserving approach that guarantees strong protection of users’ privacy in FL. Specifically, our method includes two layers of privacy protection. In the first layer, we use Hensel’s compression to reduce the dimension of the clients’ training dataset. To the best of our knowledge,
no existing work uses Hensel’s Lemma ([18], p.340) in dimensionality reduction. The second layer implements DP by adding noise to the compressed dataset generated by the first layer. The two layers generate a privacy-preserving dataset the client uses in the local training.

With two layers of privacy protection, our proposed approach hides the two principle components (i.e., the resolution of the target image and the length of the gradient vector) on which attackers based to recover the training dataset. Attackers or malicious servers will not have any visibility on a client’s original private dataset as the training is performed on the noisy, compressed dataset. Furthermore, as DP is implemented once on the original dataset before training starts, our approach prevents privacy leakage even though the synchronization training epochs increase. Thus, the proposed approach addresses the privacy leakage problem due to composition. In summary, the main contributions of this paper are as follows.

- We have proposed a novel image-based data protection approach to protecting users’ privacy in Federated Learning. Our approach overpowers the shortcomings of the existing DP-based approaches.
- We have developed a new dimensionality reduction method based on Hensel’s Lemma. Our method stands apart from previous works by efficiently reducing a dataset’s dimension without losing information. On the other hand, the proposed method reduces the computational time and the communication overhead by reducing the size of the training dataset.
- We have performed comprehensive experiments on our approach. Our experimental results demonstrated that the proposed approach guarantees strong protection of users’ privacy while achieving significantly high accuracy.

The rest of this paper is organized as follows. Section II describes the system model, the novel two layers of privacy protection, and the new proposed dimensionality reduction method. We evaluate the proposed approach with regard to privacy protection and accuracy in Section III. Finally, we conclude our contributions and discuss future work in Section IV.

II. PROPOSED METHOD

Fig. 1 illustrates the main steps for training an FL model using our proposed approach. Before starting the training, the server sends the learning model architecture as well as the initial global gradient and dimension of the dataset elements to all clients, as shown in ‘pre-training step 1’. In the ‘pre-training step 2’, each client reduces the dimension of the local dataset elements (i.e., layer 1) and implements DP (i.e., layer 2) on the compressed dataset generated by the first layer. The pre-training steps 1 and 2 are done once before starting the training to generate the privacy-preserving dataset, which is used later in the training process. After the pre-training steps, each client starts the local training and sends the local gradient to the server as illustrated in ‘training step 1’ and ‘training step 2’, respectively. In ‘training step 3’, the server aggregates the clients’ local gradients to update the global gradient. Then, the server sends the updated global gradient to clients in ‘training step 4’. The training steps 1, 2, 3, and 4 are repeated until getting the trained learning model.

A. First layer: Dimensionality reduction using Hensel’s compression

The first layer leverages Hensel’s compression method to reduce the dimension of the original dataset. Unlike the dimensionality reduction methods proposed in the literature, such as [17, 26, 19, 23, 3, 1, 14, 16, 12, 21], our proposed method allows reducing the dimension of a dataset without losing information. The novelty of our method is based on the following Hensel’s Lemma ([18], p. 340).

Lemma 2.1: Let \( r \in \mathbb{Z}_p \), there is a unique sequence \((a_n)_{n \geq 0}\) such that the series \( \sum_{n \geq 0} a_n p^n \) tends toward \( r \). This series is called the Hensel’s decomposition of \( r \).

We call our proposed approach Hensel’s compression as we go in the opposite direction, i.e., instead of decomposing a number, we combine several numbers into one number. In what follows, we explain our innovation with a use-case example.

Given a dataset \( D \) of images, and each element (i.e., image) of the dataset is a matrix \( M \in \mathbb{R}^{n \times m} \). The approach consists of reducing the dimension of \( M \) by dividing it into sub-blocks of dimension \( n' \times m' \), such that \( n = n'! \) and \( m = km' \), where \((k, h) \in \mathbb{N}_2^2\). Thus, we get a new matrix \( M' \) of dimension \((n'/h) \times (m/k)\).

![Fig. 1: Training a FL model using the proposed privacy-preserving approach.](image)

A. First layer: Dimensionality reduction using Hensel’s compression

The first layer leverages Hensel’s compression method to reduce the dimension of the original dataset. Unlike the dimensionality reduction methods proposed in the literature, such as [17, 26, 19, 23, 3, 1, 14, 16, 12, 21], our proposed method allows reducing the dimension of a dataset without losing information. The novelty of our method is based on the following Hensel’s Lemma ([18], p. 340).

Lemma 2.1: Let \( r \in \mathbb{Z}_p \), there is a unique sequence \((a_n)_{n \geq 0}\) such that the series \( \sum_{n \geq 0} a_n p^n \) tends toward \( r \). This series is called the Hensel’s decomposition of \( r \).

We call our proposed approach Hensel’s compression as we go in the opposite direction, i.e., instead of decomposing a number, we combine several numbers into one number. In what follows, we explain our innovation with a use-case example.

Given a dataset \( D \) of images, and each element (i.e., image) of the dataset is a matrix \( M \in \mathbb{R}^{n \times m} \). The approach consists of reducing the dimension of \( M \) by dividing it into sub-blocks of dimension \( n' \times m' \), such that \( n = n'! \) and \( m = km' \), where \((k, h) \in \mathbb{N}_2^2\). Thus, we get a new matrix \( M' \) of dimension \((n'/h) \times (m/k)\).

![Fig. 2: Example of reducing the dimension of a matrix using Hensel’s compression.](image)
Fig. 2 illustrates an example of reducing a matrix \( M \) of dimension \( 8 \times 8 \) to another matrix \( M' \) of dimension \( 4 \times 4 \): The first sub-figure 2-a presents the original matrix. In the second sub-figure 2-b, we divide the matrix \( M \) into sub-blocks of dimension \( 2 \times 2 \) (i.e., \( n' = 2 \) and \( m' = 2 \)). The last sub-figure 2-c shows the newly generated matrix \( M' \) after applying Hensel’s compression. In this example, we have \( M \in M_{8,8}(\mathbb{Z}/3\mathbb{Z}) \). Applying Hensel’s compression by taking \( n' = 2 \) and \( m' = 2 \) leads to get a new matrix \( M' \in M_{4,4}(\mathbb{Z}/3^2\mathbb{Z}) \) calculated as follows:

\[
x'_{(1,1)} = 1 \times 3^0 + 2 \times 3^1 + 0 \times 3^2 + 2 \times 3^3
\]

where \( x'_{(1,1)} \) represents the element at the first row and first column of the matrix \( M' \). \( x'_{(1,1)} \) is calculated based on the sub-block located at the first row and the first column in the sub-figure 2-b. In the same way, we calculate the other elements of the matrix \( M' \) based on the sub-blocks of matrix \( M \).

### B. Second layer: Privacy-preserving dataset

The second layer applies DP to the compressed dataset produced by the first layer to generate a privacy-preserving dataset. Specifically, we add noise drawn from the Gaussian distribution \( \mathcal{N}(0, \frac{\Delta^2_f}{\epsilon^2}) \) that has been proved to satisfy the \( \epsilon \)-DP definition [5]. \( \epsilon \) is the privacy leakage, also known as the privacy budget, and \( \Delta_f \) is the sensitivity of the function \( f \) on which we apply the DP mechanism. The privacy-preserving dataset is generated by adding noise to each image as follows: Assuming a dataset \( D \) of images, such that each image \( v \in \mathbb{R}^{n \times m} \). Thus, each point of \( v \) will be perturbed using the following equation:

\[
v_{i,j} = v_{i,j} + \lambda
\]

where \((i, j) \in [1, n] \times [1, m]\), and \( \lambda \) is a noise drawn from the Gaussian distribution \( \mathcal{N}(0, \frac{1}{\epsilon^2}) \). The sensitivity \( \Delta^2_f = 1 \) is the difference between the maximum and the minimum value of \( v_{i,j} \). In our case, \( \Delta^2_f = 1 \) as we apply DP after normalizing the dataset. It is important to note that decreasing privacy leakage \( \epsilon \) increases privacy protection. \( \epsilon = 0 \) is equivalent to perfect privacy protection.

### III. Experiments

We performed comprehensive experiments to evaluate the impact of our proposed DP and Hensel’s compression on accuracy and privacy protection.

We developed a learning model, as depicted in Fig. 3, composed of two convolutional layers. Each layer is associated with ReLu as an activation function. In addition, the second convolutional layer is associated with Dropout Regularization to prevent overfitting. Then, we add three fully connected linear layers with the dimension of the last linear layer’s output of 10, which corresponds to the number of classes we have in our training dataset.

We trained the model described above using different amounts of privacy leakage and different levels of data compression. Fig. 4 shows samples of the different versions of the MNIST dataset used in training. Based on the dataset dimension, we divided these experiments into three scenarios:

- **Scenario 1**: In this scenario, we train the learning model on the original MNIST dataset where the dimension of each image is \( 28 \times 28 \). This is equivalent to 100\% of the data size.
- **Scenario 2**: In this scenario, we train the learning model on the compressed MNIST dataset where the dimension of each image is \( 14 \times 14 \). This is equivalent to 25\% of data size.
- **Scenario 3**: In this scenario, we train the learning model on the compressed MNIST dataset where the dimension of each image is \( 7 \times 7 \). This is equivalent to 6.25\% of data size.

In each scenario, we evaluated the impact of the privacy leakage \( \epsilon \) on the accuracy. We considered three values of privacy leakage \( \epsilon_1 = 2, \epsilon_1 = 1.5, \epsilon_1 = 1.25 \). Table I illustrates the experiment parameters of each scenario.

Fig. 5 illustrates the accuracy of the learning model in the first scenario. Overall, we achieve high accuracy by only applying the DP of the original MNIST dataset. The accuracy is higher than 97\% using the three values of privacy leakage, i.e., \( \epsilon_1, \epsilon_2, \) and \( \epsilon_3 \). We notice that the accuracy decreases by decreasing the privacy leakage \( \epsilon \). This is because more noise is added to the images when \( \epsilon \) decreases. Regarding privacy protection, see sub-figures 4-a), b) and c) in scenario 1, we can
Fig. 4: Samples from the different versions of the datasets used in the experiments. Sub-figures a), b), c) show samples from the original MNIST dataset after adding noise of privacy leakage $\epsilon_1 = 2$, $\epsilon_2 = 1.5$, $\epsilon_3 = 1.25$, respectively. Sub-figures d), e), f) show samples after Hensel’s compression to $14 \times 14$ and adding noise of privacy leakage $\epsilon_1 = 2$, $\epsilon_2 = 1.5$, $\epsilon_3 = 1.25$, respectively. Sub-figures g), h), i) show samples after Hensel’s compression to $7 \times 7$ and adding noise of privacy leakage $\epsilon_1 = 2$, $\epsilon_2 = 1.5$, $\epsilon_3 = 1.25$, respectively.

Table I: Training Datasets’ properties.

| Scenario | Dimension | Data size | Privacy leakage | Gaussian variance |
|----------|-----------|-----------|-----------------|-------------------|
| 3*1      | 3*28 * 28 | 3*100%    | $\epsilon_1 = 2$ | $\sigma^2 = 0.25$ |
|          |           |           | $\epsilon_2 = 1.5$ | $\sigma^2 = 0.44$ |
|          |           |           | $\epsilon_3 = 1.25$ | $\sigma^2 = 0.64$ |
| 3*2      | 3*14 * 14 | 3*25%     | $\epsilon_1 = 2$ | $\sigma^2 = 0.25$ |
|          |           |           | $\epsilon_2 = 1.5$ | $\sigma^2 = 0.44$ |
|          |           |           | $\epsilon_3 = 1.25$ | $\sigma^2 = 0.64$ |
| 3*3      | 3*7 * 7   | 3*6.25%   | $\epsilon_1 = 2$ | $\sigma^2 = 0.25$ |
|          |           |           | $\epsilon_2 = 1.5$ | $\sigma^2 = 0.44$ |
|          |           |           | $\epsilon_3 = 1.25$ | $\sigma^2 = 0.64$ |

still recognize what the real image contains even after adding significant noise to the dataset (i.e., the case of $\epsilon_3 = 1.25$ which corresponds to Gaussian noise of variance 0.64, see sub-figure 4-c).

Fig. 6 illustrates the accuracy of the learning model in the second scenario. In this scenario, we applied the two layers of privacy protection (i.e., Hensel’s compression and DP). As a result, we get a high accuracy even after increasing the privacy leakage. More specifically, the learning model gives an accuracy of 97.53%, and 96.56% for the privacy leakage $\epsilon_1 = 2$, and $\epsilon_2 = 1.5$, respectively. For the privacy leakage $\epsilon_3 = 1.25$, we get an accuracy of 94.09%. Regarding privacy leakage, we can see that it is hard to distinguish the content of images, especially for $\epsilon_3 = 1.25$.

Fig. 5: Scenario 1: Evaluating the impact of DP (i.e., the first layer) on the accuracy, considering three different values of the privacy leakage: $\epsilon_1$, $\epsilon_2$, and $\epsilon_3$. Images dimensions 28*28.

Fig. 7 illustrates the accuracy of the learning model in the third scenario. In this scenario, images are compressed from 28*28 to 7*7. Overall, we get a good accuracy compared to the level of privacy protection achieved. For example, in the first case where the privacy leakage $\epsilon_1 = 2$, the learning model achieves an accuracy of 82.28% while ensuring
Fig. 6: Scenario 2: Evaluating the impact of DP (i.e., the first layer) and dimensionality reduction to 14 × 14 (i.e., the second layer) on the accuracy, considering three different values of the privacy leakage: $\epsilon_1$, $\epsilon_2$, and $\epsilon_3$.

Fig. 7: Scenario 3: Evaluating the impact of DP (i.e., the first layer) and dimensionality reduction to 7 × 7 (i.e., the second layer) on the accuracy, considering three different values of the privacy leakage: $\epsilon_1$, $\epsilon_2$, and $\epsilon_3$.

perfect privacy protection. An attacker could not distinguish the images’ content even if the attacker succeeds in recovering the training dataset. We notice that increasing the privacy leakage $\epsilon$, the privacy protection increases while the accuracy decreases; specifically we get an accuracy of 76.56%, and 69.84% for $\epsilon_2 = 1.5$, and $\epsilon_3 = 1.25$, respectively.

To conclude, the accuracy and privacy protection depend on the privacy leakage $\epsilon$ and the level of data compression (i.e., Hensel’s compression). The proposed approach achieves an acceptable or high accuracy while ensuring strong privacy protection. Specifically, this good trade-off is achieved in scenario 2 (i.e., Hensel’s compression to dimension 14×14) for $\epsilon_3 = 1.25$, as well as in scenario 3 (i.e., Hensel’s compression to dimension 7×7) for $\epsilon_1 = 2$.

It is important to note that 25% of the data size (i.e., Hensel’s compression to dimension 14×14) gives roughly the same accuracy as if the learning model is trained on 100% of the data size. Thus, the proposed dimensionality reduction method not only strengthens protection but also reduces the computational overhead. However, compressing the data too much will hide images’ characteristics and hence decrease the accuracy. Thus, looking for the optimal trade-off between the level of data compression and the privacy leakage $\epsilon$ that guarantees strong privacy protection while achieving a high accuracy is of great importance.

**IV. CONCLUSION AND FUTURE WORK**

In this paper, we propose a novel two-layer privacy-preserving method for federated learning. The first layer reduces the dimension of the original training dataset based on Hensel’s compression. The second layer applies differential privacy on the compressed dataset generated by the first layer. The experimental analysis validates the effectiveness of the proposed approach in protecting users’ privacy while achieving high accuracy. Furthermore, experimental results show also that the learning model accuracy depends on the dataset compression and the privacy leakage $\epsilon$.

In future work, we will conduct extensive experiments on other benchmark datasets to evaluate the impact of our approach on accuracy and privacy protection. Furthermore, our new dimensionality reduction method based on Hensel’s compression could reduce the dimension of the training dataset, thus minimizing the trainable parameters and improving computational performance. As a result, our proposed method can be adopted to revolutionize the utilization of deep learning in several domains where the response time is crucial, such as web malware detection, object detection, and automatic modulation recognition.

**ACKNOWLEDGMENTS**

This work was funded in part by NSF EAGER award #2025234, and by Ohio Department of Higher Education (ODHE) award through Strategic Ohio Council for Higher Education (SOCHE).

**REFERENCES**

[1] Hussein Almuallim and Thomas G. Dietterich. “Learning with Many Irrelevant Features”. In: *Proceedings of the Ninth National Conference on Artificial Intelligence - Volume 2*. AAAI’91. Anaheim, California: AAAI Press, 1991, pp. 547–552. ISBN: 0262510596.

[2] Shahab Asoodeh et al. *Three Variants of Differential Privacy: Lossless Conversion and Applications*. 2021. arXiv: 2012.05326.

[3] Xue wen Chen. “An Improved Branch and Bound Algorithm for Feature Selection”. In: *24.12* (Aug. 2003), pp. 1925–1933. ISSN: 0167-8655. DOI: 10.1016/S0167-8655(03)00020-5.

[4] Edwige Cyffers and Aurélien Bellet. *Privacy Amplification by Decentralization*. 2021. arXiv: 2012.05326.

[5] Jinshuo Dong, Aaron Roth, and Weijie J. Su. *Gaussian Differential Privacy*. 2019. arXiv: 1905.02383.
