A Revocation Key-based Approach Towards Efficient Federated Unlearning

Rui-Zhen Xu*, Sheng-Yi Hong†, Po-Wen Chi‡ and Ming-Hung Wang§
*Department of Computer Science and Information Engineering
National Taiwan Normal University, Taipei, Taiwan
Email: andrea20020206@gmail.com
†Department of Computer Science and Information Engineering
National Taiwan Normal University, Taipei, Taiwan
Email: sheng-yihong@acm.org
‡Department of Computer Science and Information Engineering
National Taiwan Normal University, Taipei, Taiwan
Email: neokent@gapps.ntnu.edu.tw
§Department of Computer Science and Information Engineering
National Chung Cheng University, Chiayi County, Taiwan
Email: tonymhwang@cs.ccu.edu.tw

***Abstract*—Federated learning is an approach that ensures privacy in machine learning, but it has its limitations when it comes to preserving the right to be forgotten. To address this challenge, we propose a new method called Unlearning Key Revocation List (UKRL) for implementing federated unlearning. Our approach does not require clients' data or models to be unlearned; instead, we use revocation keys to remove clients from the model. We pre-trained the model to recognize these keys, so the model will forget the revoked clients when their revocation keys are applied. We conducted four experiments using MNIST datasets to verify the effectiveness of our approach, and the results showed that our work is not only effective but also time-saving since the unlearning time is 0. In conclusion, we provide a new perspective on achieving federated unlearning.**

***Index Terms*—federated learning, the right to be forgotten, machine unlearning**

I. Introduction

Nowadays, artificial intelligence (AI) is the state-of-the-art technology that has revolutionized many aspects of our lives. In healthcare, AI can be used to help diagnose diseases and provide personalized treatment plans based on a patient's medical history [1]. In entertainment, AI is used to create art, music, and stories. Some streaming services like YouTube and Netflix use AI techniques to analyze viewer data and provide personalized recommendations for movies and videos [2]. Moreover, in our daily life, AI virtual assistants like Siri, Alexa, and Google Assistant can help us with some tasks like setting reminders, scheduling, or looking up information for us [3].

However, the use of such AI systems requires a large amount of data for training and often involves the collection of personal data from clients to improve the performance of these systems. Many individuals are not comfortable with companies collecting their personal information without their explicit consent. To address this issue, Google proposed a machine learning framework called federated learning, which provides a way for all of the clients to train the models with their data [4]. Then, the server receives all the local model parameters and aggregates them into a global model without knowing the actual training data. Therefore, this method provides a solution to the challenge of preserving data privacy while still allowing for knowledge discovery [5], [6].

While federated learning is a promising approach to preserving data privacy in machine learning, it has its limitations. One of its limitations is its inability to fulfill the right to be forgotten. In 2016, the EU introduced the General Data Protection Regulation (GDPR), which became enforceable in 2018. In GDPR, It grants individuals the right to be forgotten, giving them control over any information related to themselves [7]. As a result, companies that use client data to improve their models need to provide a way to remove that data from the models. In another case, they have to prevent their models from being attacked by being given malicious data, which will cause the model to generate unexpected results.

To address the issues mentioned above, machine unlearning techniques have been introduced to solve problems related to federated learning. Machine unlearning involves removing certain data that was used to train the machine learning model [8]. The naive way to implement unlearning is to retrain the model [9]. However, retraining has several limitations, including the long retraining time, the high computational complexity, and the difficulty in guaranteeing that the training data is still well-preserved by the model trainer. Therefore, alternative unlearning techniques have been proposed, such as the Fisher unlearning method [10], class-discriminative pruning unlearning method [9], knowledge distillation unlearning method [11], and using Quasi-Newton methods and first-order Taylor approximation technique to efficiently retrain a model [12]. However, some of these methods require storing training data or model parameters for unlearning, which is challenging when the number of clients is huge.

2765-9712/23/$31.00 ©2023 IEEE
DOI 10.1109/AsiaJCIS60284.2023.00014

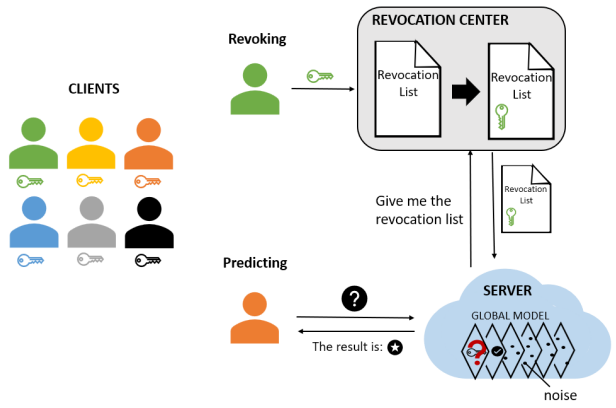

Fig. 1. The overall concept of Unlearning revocation key Revocation List.

As a result, we proposed a new method, Unlearning Key Revocation List (UKRL), which uses the revocation keys to determine if a client has requested data deletion. Figure 1 shows how UKRL works to unlearn a client. The framework involves three main characters: the clients, the server, and the revocation center. Clients are those who participate in federated learning. Each client owns a unique revocation key; the server, who is honest but curious, is responsible for model aggregation; the revocation center is trusted and generates the revocation list. Initially, clients train their local models independently using their data and revocation keys. Then they send their parameters to the server for aggregation. If a client wants to deregister, it provides its revocation key to the revocation center, which adds it to the revocation list. It can be referred to as the revoking part in Figure 1. And then, the revocation center will put the revocation key into the revocation list. After that, when any client requests the server for data prediction, the server will ask the revocation center for the revocation list. The server then takes the prediction task and the revocation list as input to the global model. The output will be given to the client after the prediction. In our design, if a client has been revoked, their contribution to the global model acts as noise, meaning their real training data cannot affect the prediction result. Therefore, our proposed method enables unlearning without retraining the model or storing training data or model parameters.

In summary, the contributions of this paper are as follows:

- We have proposed a novel approach to attain the right to be forgotten in federated learning.
- We have modified the structure of the federated learning model to include multiple inputs for both the revocation list and the target prediction data.
- We have designed experiments to demonstrate the effectiveness and efficiency of our proposed approach compared to traditional model retraining.

The rest of the paper is organized as follows: Section II provides background information on federated learning and existing machine unlearning approaches. Section III describes our proposed federated unlearning approach in detail. Section IV outlines the experiments we conducted to verify the effectiveness and efficiency of our method. Finally, Section V summarizes our work and discusses potential avenues for future research.

II. Related Works

A. Federated learning

Due to data privacy issues, Google proposed a machine learning framework known as federated learning to protect clients' privacy in 2016. Federated learning is a jointly trained machine learning framework based on the data located at multiple sites [4]. The proposed enables the building of models on client devices and the joint training of a global model by uploading the trained model parameters to the server [6].

Typically, the federated learning process can be divided into two parts. First, clients locally compute training gradients and send them to the server. Second, the server performs aggregation to aggregate all the models and sends the results back to the clients [4].

B. Machine unlearning

Since the European Court of Justice adjudicated that an individual should have the right to be forgotten in 2014 [13], the concept of an individual has the right to request deletion of their personal information online is getting more and more important [8]. There are lots of ways to implement data removal in the machine learning area, which is also called machine unlearning, following are some of the machine unlearning methods.

In 2019, Golatkar, Achille, and Soatto proposed the Fisher unlearning method [8] to implement machine unlearning. The proposed algorithm is

$$w^u = w^{opt} - F^{-1}\Delta + \sigma F^{-\frac{1}{4}} b$$

where w^u is the new model parameters; w^{opt} is the optimized model parameters; F is the Fisher information matrix, which is used to approximate the Hessian matrix to reduce the cost of computing the Hessian matrix; Δ is the gradient of the loss function; b is noise [10]. However, this work needs additional space to store the training data to compute the Fisher matrix, which makes it challenging to directly apply Fisher unlearning method to federated learning [9].

In 2022, Junxiao Wang, Song Guo, Xin Xie, and Heng Qi proposed a federated unlearning approach through Class-Discriminative Pruning [9]. The proposed involves clients recording the feature map scores between each channel and category on the client side. Later, clients send these feature maps to the server for aggregation to obtain the global feature map scores. The server then uses TF (Term Frequency) and IDF (Inverse Document Frequency) to evaluate the relevance score between the channels and categories. Finally, a pruner is applied to remove the most discriminative channels of the category that they wish to remove. This approach avoids the need for the server to have access to training data, which is a major challenge in federated learning. However, it only works for unlearning data that belongs to a particular class. If clients

want to remove their data, and their data is dispersed across all classes, this method is hard to apply.

Also in 2022, Chen Wu, Sencun Zhu, and Prasenjit Mitra proposed to achieve federated unlearning using knowledge distillation [11]. The proposal is that the server needs to store all the history of parameter updates from clients and subtract it to implement unlearning. Afterward, the server uses the knowledge distillation method to recover the damage caused by the unlearning. In knowledge distillation, the original global model acts as the teacher model, while the new unlearning model acts as the student model. The student model can learn from the teacher model using unlabeled data. Despite this method using knowledge distillation to reduce the cost of retraining, it needs a lot of space on the server side to keep all the history of parameter updates, which could be problematic when a large number of clients are involved in the federated learning process.

In conclusion, some trade-offs to implementing those existing unlearning methods are undesirable, like spending extra space to store training data or model parameters. As a result, we proposed a new way to implement federated unlearning.

III. Unlearning revocation key Revocation List

Due to the distributed nature of federated learning, it is difficult to preserve all of the training data, and even harder for the server to obtain it. To address this issue, rather than attempting to remove the data itself, our work is to turn the influence of the revoked clients' model into noise so that the revoked clients' real training data doesn't influence the global model. It attains the same effect as unlearning. The main idea to achieve this goal is to use a revocation key-based system, where the clients can request to deregister by providing a unique revocation key, which is used to nullify the influence of their real training data on the model without actually deleting it.

A. Designing Idea

Our work is similar to federated learning at first, in which all clients train their local models for the server to combine into a global model. The difference is that all clients in UKRL have a revocation key for deregistering. They also need to train their models to recognize the revocation key. And then, to unlearn a client's data, the client has to give its revocation key to the revocation center. The revocation center then puts the revocation key into the revocation list, which is a size-fixed list to record the revocation key of all clients who want to deregister. After that, when using the global model, the input will contain the newest revocation list, which includes every revoked client's revocation key. And the output will not be affected by real training data from those who have deregistered.

B. Procedure of Prediction and Revocation

There are six steps in UKRL procedure, which are shown in Figure 2. Following is the description for each step:

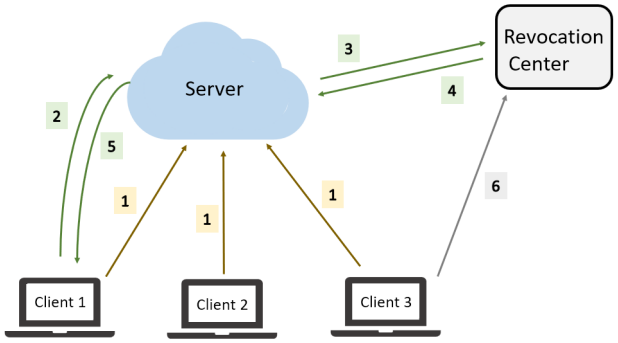

Fig. 2. Unlearning revocation key Revocation List framework.

1) The clients train their local models and pass them to the server to combine them into the global model.
2) The clients request data prediction.
3) The server receives prediction requests from the clients and asks for the revocation list from the revocation center.
4) The server retrieves the revocation list from the revocation center. The revocation list includes the revocation keys of the revoked clients and some random numbers. The amount of random numbers is equal to the number of clients who haven't deregistered.
5) The client retrieves the result from the server.
6) The client sends the revocation key to the revocation center for unlearning. And the revocation center will replace the corresponding part of the revocation list with the revocation key.

C. Assumption

In UKRL framework, we assume that the server is honest but curious, which means the server will complete the job. However, it will try to peek into clients' data. And we assume that the revocation center is trusted, which means the revocation center will complete the job and will not divulge or tamper with the information. We can build the revocation center using blockchain to ensure the information cannot be falsified.

Also, we assume that the number of revoked clients is much smaller than the number of clients who stay in the model. We consider this assumption reasonable because in the real case application, like Google, the population that uses the services is large compared to the number of people that left the services.

D. Model Structure

Our proposed model consists of two parts: the revocation key-distinguishing part and the data-classifying part. As shown in Figure 3, the model takes the revocation list and the prediction data as input. The revocation key-distinguishing part determines whether the revocation list includes clients' revocation keys. The result of this part influences the second part's output. The data-classifying part classifies the prediction data. If the revocation list includes any revocation keys, the

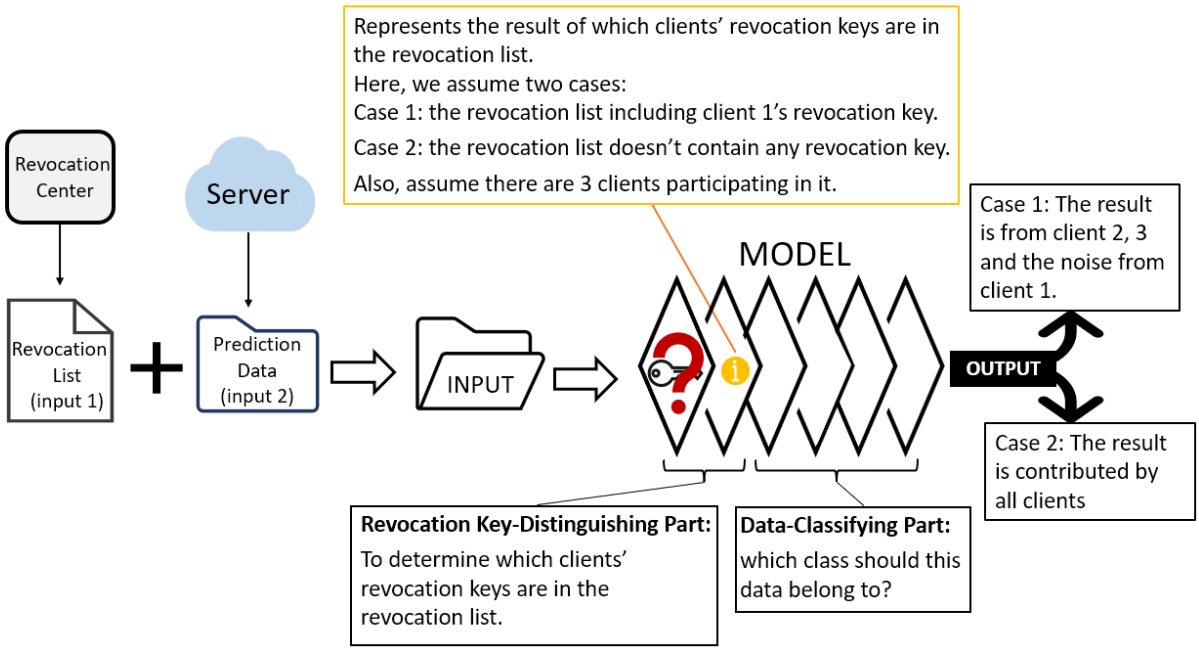

Fig. 3. The process of the model.

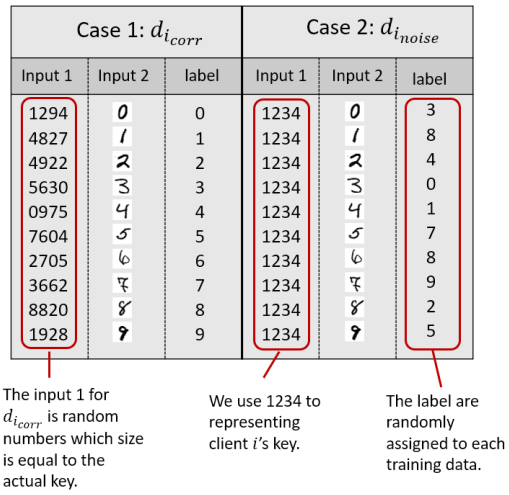

Fig. 4. The example of training data [14].

result is contributed by all clients' models except those of the revoked clients, which contribute some noise to the result. Otherwise, the result is contributed by all clients' models. The model's output is a list representing the likelihood of the prediction data belonging to each class. The class with the highest probability is selected as the predicted result.

In the following part, we consider that input 1 is the revocation list, and input 2 is the prediction data.

E. Training Process

Suppose there are n clients participating in the federated learning. Each client has a unique revocation key, denoted as k_i, and has two training datasets: $d_{i_{corr}}$ and $d_{i_{noise}}$, where $i \in [1, n]$. The data of $d_{i_{corr}}$ and $d_{i_{noise}}$ are the same but the labels are different. Figure 4 shows an example. For $d_{i_{noise}}$, the labels are intentionally assigned incorrectly using random numbers, while the labels of $d_{i_{corr}}$ remain correct. The reason for using two identical training data with different labels is that we want to make the model learn the following two things:

1) If there is no client's revocation key in input 1, the model can learn the correct label from training data and successfully predict the answer.
2) If input1 includes a client's revocation key, the model will learn the wrong label for the training data. Since the label is incorrect, the model can not correctly predict the answer.

In the training phase, at first, as in Case 1 in Figure 4, we take random numbers as input 1 and original training data as input 2. Next, as in Case 2 in Figure 4, we take k_i as input 1 and the original training data as input 2. After the training process, the model recognizes the revocation key k_i and learns that if the probability of input 1 being the actual revocation key is low, the model will tend to output the correct answer. Otherwise, the model will tend to output a random result, which makes the model behave like noise.

After that, the clients will send the resulting model parameters m_i to the server. The server then aggregates all the local models into a global model by calculating the average of the model parameters from all clients, resulting in a new global model $m^* = \frac{1}{n}\sum_{i=1}^{n} m_i$. If the accuracy of the global model hasn't achieved the desired performance, the server will send the global model back to the clients and repeat the training process.

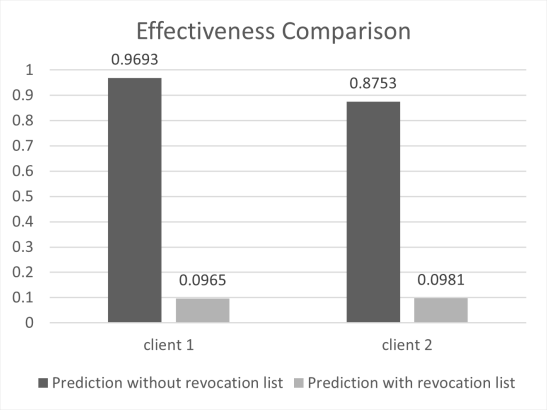

Fig. 5. The accuracy of client 1's model and client 2's model

Once the training process is completed, the server will keep the global model parameters. To perform data prediction, the server should request the revocation center for the revocation list. Next, the revocation center will give the server the revocation list. Then, the server takes the target prediction data and the revocation list as input and outputs the prediction result.

F. Unlearning

Assuming a client, denoted as c, wants to be revoked. First, it has to provide its revocation key k_c to the revocation center. Second, the revocation center will replace the corresponding part of the revocation list with k_c. Afterward, if the server asks for the revocation list, the list will include k_c. It means client c's part will influence the global model as noise and the real training data will no longer affect it.

Because in our design, we only need to submit the revocation key to the revocation center to unlearn, the time consumption for unlearning is 0.

IV. Experiments

A. Experiment Environment

We conducted empirical evaluations to assess the effectiveness of our federated unlearning method on the MNIST dataset. We chose this dataset because it is a widely-used benchmark for image classification tasks and contains a manageable number of classes and images, which consists of 60000 handwritten digits images of size 28×28 [15], divided into ten classes representing the digits 0 to 9. Additionally, We used a deep neural network (DNN) to construct our model.

We conducted our experiments with an AMD Ryzen 7 CPU as the processing unit, running on the Ubuntu 20.04 operating system. We employed the Tensorflow machine learning framework with the Keras library to build our models. All the implementation and experimentation were conducted using Python 3.8.10.

To validate the effectiveness of UKRL method, we designed four experiments. The first experiment assessed the effectiveness of revocation keys for a single client. The second and third experiments tested the effectiveness of revoking a client from a two-client and five-client global model, respectively. The final experiment evaluated the effectiveness of unlearning under a more realistic data distribution case with five clients.

B. Effectiveness of the revocation key

To confirm the effectiveness of our design, we designed an experiment containing two clients, a server, and a revocation center. In addition, we created a training dataset that lacked class 9 data for client 2, while client 1 had a training dataset with data from all categories.

Our first goal was to prove the effectiveness of revocation keys for a client. We tested the accuracy of two local models individually and aimed to validate the effectiveness of using revocation keys to let a particular model become noise. Specifically, we expected both local models to generate similar outcomes with an accuracy of about 0.1 when applying client 1's revocation key to client 1's model and when applying client 2's revocation key to client 2's model. It is because the MNIST dataset has 10 classes, and the probability of randomly guessing the correct class is 0.1.

From Figure 5, the left-hand side shows that without applying client 1's revocation key, the model's accuracy is 0.9468. In contrast, if client 1's key is applied, which means client 1 is revoked, the accuracy of client 1's model is 0.0965. The result meets our expectation that the prediction results of the model approach 0.1 when revoking client 1, and the prediction results are normal when client 1 remains in the model. From the right-hand side of Figure 5, similar outcomes can be observed for client 2. When client 2 is revoked, the accuracy is 0.0981, while the accuracy of client 2 remaining in the model is 0.8753. This result is consistent with our expectations, as client 2 does not have any class 9 data but the testing dataset contains class 9 data, which causes the accuracy to be around 0.1 lower than the accuracy of client 1's result. The results above can be briefly summarized to show that the revocation keys are effective for the clients.

C. Effectiveness of unlearning for two-client model

After confirming the effectiveness of using the revocation key in each client's model, our next objective is to demonstrate that the same revocation key can also revoke a client from the global model. Figure 6 illustrates the diagram of the experiment. To achieve this goal, we will compare the accuracy of a global model before and after revoking client 1. When deregistering client 1, the global model shouldn't be able to categorize class 9 data if UKRL method is effective since the only client remaining in the global model lacks class 9 data. In other words, if the global model can still categorize class 9 data, the global model hasn't forgotten client 1's data. As a result, we expect that if we revoke client 1, the accuracy of the global model, when tested on class 9 data, approaches 0.

From the left-hand side of Figure 7, we can see that when testing the accuracy of class 9 data, client 1 achieves the accuracy of 0.965, but the accuracy on client 2's model is 0 since client 2 doesn't own any data belongs to class 9. The

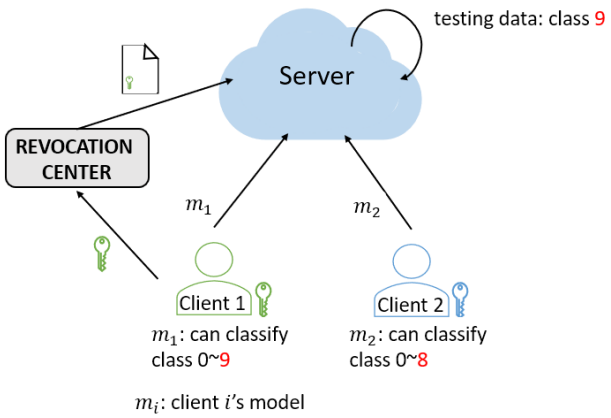

Fig. 6. The diagram of two-client experiment.

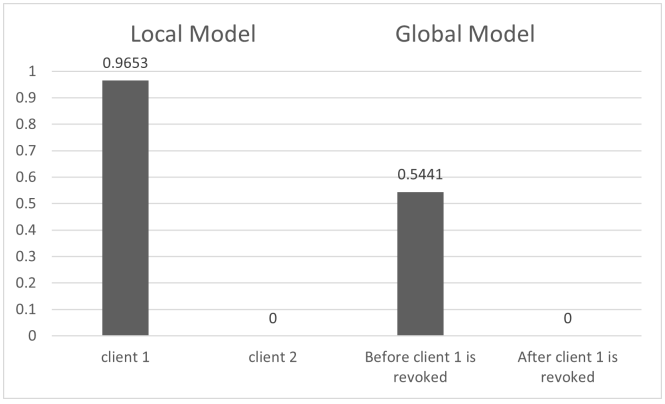

Fig. 7. The accuracy when tested on class 9 data

right-hand side of Figure 7 displays the accuracy of the global model before and after revoking client 1. When client 1 had not been deregistered, the accuracy of the global model on class 9 data was 0.5114, which is reasonable since the prediction result is affected by both client 1's data and client 2's data. On the other hand, when client 1 has been revoked, the accuracy of the global model tested on class 9 data is 0. The results have demonstrated the effectiveness of UKRL method since we were able to erase the influence of client 1's actual training data on the global model, such that the model was unable to classify class 9 data after deregistering client 1.

In conclusion, UKRL method can successfully unlearn a client in a double-client scenario.

D. Effectiveness of Multiple Clients: Only Client 1 Has Class 9 Data

In the previous paragraph, we verified the effectiveness of using UKRL to achieve unlearning under a double-client scenario. Here, we will extend our experiment to a multiple-client scenario to test the effectiveness of our design under such conditions.

Instead of simulating with two clients, we have increased the number of clients to five. Figure 8 shows the diagram of

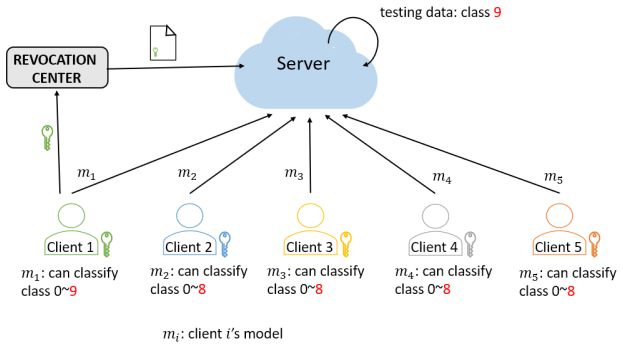

Fig. 8. The diagram of multiple clients experiment (only client 1 has class 9 data).

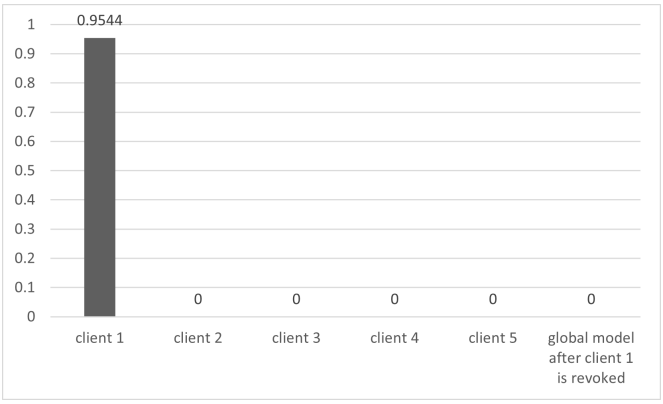

Fig. 9. The accuracy when tested on class 9 data with multiple clients (only client 1 has class 9 data)

the five-client experiment. Similarly, we have designated client 1 to have a training dataset that includes class 9 data, while the training datasets for other clients do not contain class 9 data. Once the server aggregated the models, client 1 requested to deregister. The purpose of this experiment is to test whether the model can still categorize class 9 data if client 1 is revoked. We expect that the accuracy of the global model tested on class 9 data will approach 0, indicating that the model has successfully unlearned the data from client 1.

Based on the results shown in Figure 9, we can see that the accuracy of class 9 data tested on client 1's model is 0.9544, while for the other clients' models, the accuracy is 0. When testing on the global model, if we revoke client 1, the accuracy of the global model tested on class 9 data is 0. The results show that when revoking client 1, the global model is unable to classify class 9 data, which is consistent with the results of the two-client experiment.

E. Effectiveness of Multiple Clients: All Clients Have Class 9 Data but Unbalanced

In reality, it is highly unlikely that only one client owns the data belonging to a particular class. To simulate a more realistic scenario, instead of having zero class 9 data assigned to clients 2 to 5, we assigned them an amount of class 9 data that is about one-tenth of the amount assigned to client 1.

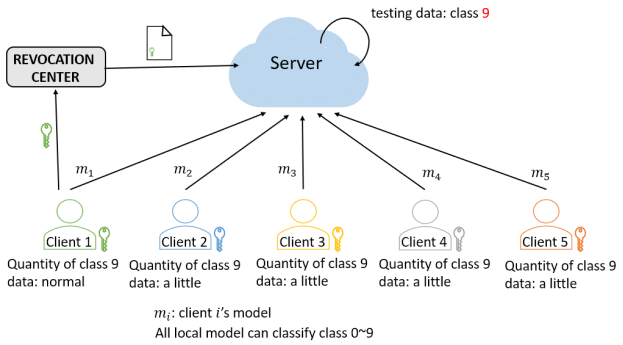

Fig. 10. The diagram of multiple clients experiment (all clients have class 9 data).

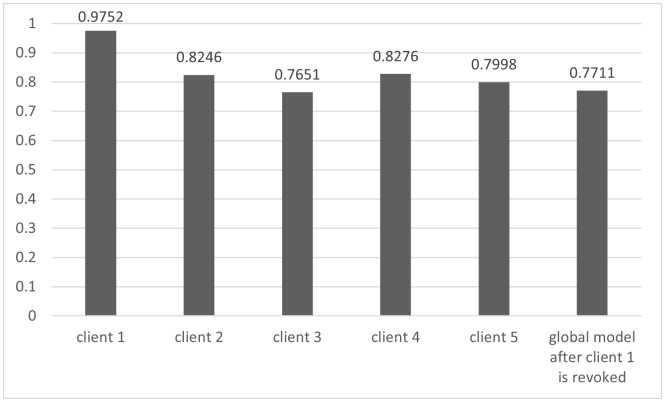

Fig. 11. The accuracy when tested on class 9 data with multiple clients (all clients have class 9 data)

Figure 10 shows the diagram of this experiment. The way we executed this experiment is the same as the previous one. And the purpose of conducting this experiment is to demonstrate the effectiveness of UKRL under more realistic conditions.

In addition, we will also evaluate the overall accuracy of the global model both before and after revoking client 1 to estimate the loss caused by revoking a client. If the loss is small, we do not need additional work to compensate for the damage, which makes the time consumption of the unlearning process become 0 if we don't consider the communication time to pass the revocation key to the revocation center. Otherwise, if the loss is significant, UKRL method can not be viewed as an efficient way to implement federated unlearning since it will need additional unlearning processes to compensate for the loss.

From Figure 11, the left-hand side shows the accuracy of clients' models tested on class 9 data. Since clients 2 to 5 have a relatively small amount of data belonging to class 9, their accuracy is lower than that of client 1, but not zero. Also, the rightmost bar shows that after client 1 has been revoked, the accuracy of the global model tested on class 9 data is 0.7711, which is reasonable since the main provider of class 9 data is client 1. As a result, the accuracy is slightly lower than the average accuracy of the models of clients 2 to 5.

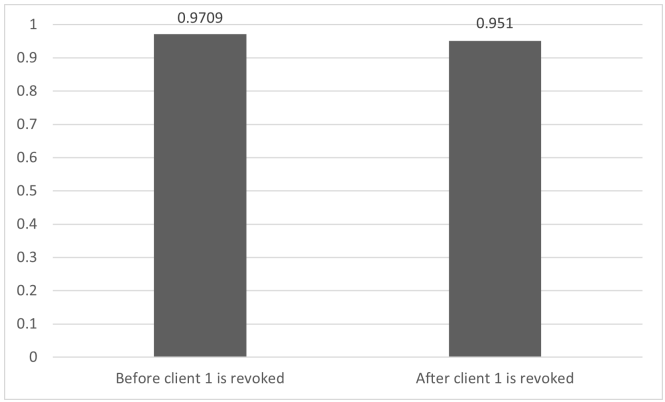

Fig. 12. The accuracy of global model with and without client 1

Additionally, to verify the efficiency of UKRL method, we evaluated the accuracy of the global model both before and after revoking client 1. Figure 12 shows that the model's accuracy is 0.9709 if client 1 is included, but drops to 0.951 when client 1 is revoked. The result indicates that the loss to unlearn a model in a five-client scenario is about 2.05%. Moreover, as the global model parameters are obtained by averaging all client model parameters, the impact of revoking a single client is reduced when more clients participate in the training process. Therefore, we consider the loss to unlearn a client to be acceptable. It means UKRL method is an efficient way to revoke clients as it does not require extra time to compensate for the loss.

Consequently, we believe that our approach is effective and efficient for unlearning clients. We have verified the effectiveness of using revocation keys as the revoking mechanism. Additionally, we have demonstrated that UKRL does not require additional time to compensate for the loss caused by unlearning, making it an efficient method for client revocation. In summary, UKRL is a method that can achieve the right to be forgotten in the federated learning domain.

V. Conclusion and Future Work

We proposed a new federated unlearning method to eliminate the influence of the real training data from revoked clients. Our approach is to use revocation keys to revoke clients. We introduced the revocation list to record revocation keys, which then be applied to the global model to perform unlearning. Also, we have done three experiments to prove the effectiveness and efficiency of our work.

In our future work, we plan to improve our study in two ways. First, instead of randomly assigning a label to the revocation keys-containing data, we will create a new class, void, to represent those data. In the training process, we will assign a label void to the data which contains the revocation key. We expect when testing on client i's model, the output will be class void when client i's revocation key k_i is applied. When testing on the global model, we can estimate the effectiveness by calculating the probability of getting a void class when revoking a client and evaluating the accuracy

of the testing data. The purpose of this adjustment is that it may be a better way to implement federated unlearning using the revocation key as an unlearning method.

Second, we will apply homomorphic encryption techniques to our work. In the current version of UKRL method, if clients want to predict their data, they need to send the data to the server to get the result. Also, the server can see the parameters of the local models from every client. However, some clients may not want their prediction data or their local model parameters to be known by the server. One of the solutions is to encrypt the data and local model parameters using homomorphic encryption schemes [16]. Because of the property of homomorphic encryption, it allows computations to be performed on encrypted data without decrypting it, meaning that the data remains protected even during computation. Therefore, the server can aggregate local models without actually knowing those numbers and predict the result without knowing the prediction data.

REFERENCES

[1] Y. Kumar, A. Koul, R. Singla, and M. Ijaz, "Artificial intelligence in disease diagnosis: a systematic literature review, synthesizing framework and future research agenda," *Journal of Ambient Intelligence and Humanized Computing*, 2022, funding Information: This research work was supported by Sejong University research fund. Yogesh Kumar and Muhammad Fazal Ijaz contributed equally to this work and are first co-authors. Publisher Copyright: © 2021, The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature.

[2] S. Prajapati. (2021) 6 applications of ai in entertainment industry. [Online]. Available: https://www.analyticssteps.com/blogs/6-applications-ai-entertainment-industry

[3] K. Yasar. (2023) virtual assistant (ai assistant). [Online]. Available: https://www.techtarget.com/searchcustomerexperience/definition/virtual-assistant-AI-assistant

[4] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, *Federated Learning*. Morgan & Claypool Publishers, 2019.

[5] B. McMahan and D. Ramage. (2023) Federated learning: Collaborative machine learning without centralized training data. [Online]. Available: https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

[6] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *International Conference on Artificial Intelligence and Statistics*, 2016.

[7] P. AG. (2023) Complete guide to gdpr compliance. [Online]. Available: https://gdpr.eu/

[8] A. Mahadevan and M. Mathioudakis, "Certifiable unlearning pipelines for logistic regression: An experimental study," *Machine Learning and Knowledge Extraction*, vol. 4, no. 3, pp. 591–620, 2022.

[9] J. Wang, S. Guo, X. Xie, and H. Qi, "Federated unlearning via class-discriminative pruning," in *Proceedings of the ACM Web Conference 2022*, New York, NY, USA, 2022, p. 622–632.

[10] A. Golatkar, A. Achille, and S. Soatto, "Eternal sunshine of the spotless net: Selective forgetting in deep networks," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9301–9309, 2019.

[11] C. Wu, S. Zhu, and P. Mitra, "Federated unlearning with knowledge distillation," *ArXiv*, vol. abs/2201.09441, 2022.

[12] Y. Liu, L. Xu, X. Yuan, C. Wang, and B. Li, "The right to be forgotten in federated learning: An efficient realization with rapid retraining," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, 2022, p. 1749–1758.

[13] P. AG. (2023) Everything you need to know about the 'right to be forgotten'. [Online]. Available: https://gdpr.eu/right-to-be-forgotten/

[14] Mnist database. [Online]. Available: https://en.wikipedia.org/wiki/MNIST_database

[15] The mnist database of handwritten digits. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[16] F. Wibawa, F. O. Catak, M. Kuzlu, S. Sarp, and U. Cali, "Homomorphic encryption and federated learning based privacy-preserving cnn training: Covid-19 detection use-case," in *Proceedings of the 2022 European Interdisciplinary Cybersecurity Conference*, 2022, pp. 85–90.