Federated Unlearning: COMPSCI 682 Project Final Writeup

Hyejun Jeong
UMass Amherst
hjeong@umass.edu

Mohammadreza Teymoorianfard
UMass Amherst
mteymoorianf@umass.edu

Abstract

Federated Learning, which emerged in 2016, addresses concerns about data privacy caused by more stringent privacy regulations. Its distributed nature does not allow raw data transmission in one centralized position, but the central server sends a lightweight model to each client, such that the client can train the model based on their local training set. It not only protects private data but significantly reduces the communication overhead. Machine Unlearning was emerged in response to recent concerns about the right to be forgotten that allows the data owner to request data removal from the authorities for some reasons, including privacy concerns or any other security reasons, if it does not legally harm the right of publicity. Naturally, Federated Unlearning, machine unlearning in the federated learning context, is getting its attention as a solution to preserve data privacy while conveying the right to be forgotten. In this paper, we implement a Halimi et al.'s proposed method that removes the client's data and their contribution by reversing the training procedure by performing a projected gradient ascent and maximizing the empirical loss. We found out that the method can effectively erase the data to be forgotten, but it also significantly deteriorated the performance of retaining data. In this paper, we discuss why the performance was not recovered after unlearning, and what should have been done to recover it properly.

1. Introduction

In recent years, the landscape of machine learning has witnessed a profound shift towards decentralized and collaborative approaches due to stringent privacy regulations such as the General Data Protection Regulation (GDPR). This shift has motivated the rise of decentralized and collaborative approaches, with Federated Learning (FL) emerging as a key paradigm. FL involves training models across distributed devices while protecting data privacy by keeping raw training data local. However, as the Right To Be Forgotten (RTBF) has been gaining attention since 2014 [8], it has shifted FL to be equipped with unlearning algorithms (i.e., Federated Unlearning (FU)), to erase the specific data from the training set and its influence from the trained model.

Specifically, after an FL model converges, a client can request sample-wise, class-wise, and/or client-wise removal. Upon such requests, the server and all the clients, including the revoking client, perform unlearning such that the data and its influence can be eliminated from the trained model as if it has never seen the requested data.

Yet, FL's distributed, data-isolated, and black-box nature presents challenges in unlearning compared to Machine Unlearning (MU) because we cannot directly distinguish what parameters are learned from the specific data under the harsh condition that the server or the client is not accessible to other's raw data [19, 23, 30]. Instead of costly retraining the model from scratch without the requested data, researchers have explored ways to eliminate a requested client's contribution or influence within an FL model, termed "*Federated Unlearning*" or FU. The evaluation of FU revolves around efficacy, fidelity, and efficiency: how well the model forgets the requested data, the performance concerning retained data, and the acceleration compared to retraining from scratch, respectively [5].

Unlearning methods could be broadly categorized into four: restoring the certain rounds of certain models using historically stored model updates [4, 12, 19, 28, 32, 33], estimating loss function as if the erasure subject was not in the training dataset [16, 21], gradient manipulation [6, 13, 20, 33], and learning in reverse direction [1, 15, 18, 29, 34]. Nonetheless, storing all model updates becomes infeasible in FL settings due to potential memory overhead, especially with complex models and large training sets. Loss function estimation, on the other hand, incurs substantial computational overhead in calculating the inverse Hessian matrix to approximate loss, which causes a heavy computational complexity. In other words, edge devices may be unable to handle heavy computation due to their limited resources. Additionally, gradient manipulation faces challenges in distinguishing parameters influenced by specific features, impacting fidelity. Reverse learning could be a promising solution. It gradually unlearns the data using gradient ascent instead of decent, preserving efficacy and fidelity.

In this paper, we have developed our method based on [15]. The method [15] unlearned a global model by reversing the training process, by maximizing the local empirical loss. The optimization process is formulated as a constraint maximization in the L2-norm ball around the reference model. After local unlearning, using a few rounds of FL with the remaining clients will lead to a global model that has unlearned the influence of the target client.

2. Backgrounds

2.1. Federated Learning

Federated learning is a distributed and iterative machine learning framework proposed by Google in 2014 [22]. The topic was explored in response to rising concerns about data privacy. Instead of collecting all the data into a centralized location to train a machine learning model, the private or local data never leaves the clients, keeping it local. Instead, the global server or aggregator distributes the global model to the clients such that each client can train the local model on the private dataset. Since model updates are less in size than millions of raw data (i.e., high-resolution image data), it relax the communication bottleneck while providing some degree of data privacy. Specifically, Federated Learning is a repetition of the following steps: (1) A global server initiates a global model. (2) The server selects a fraction of clients to perform local training and sends its copy to all participating clients. (3) The selected clients train the global model with their private data, generate local models, and send the updates to the global server. (4) The server aggregates the local model updates to generate a new global model. Steps 2 to 4 are repeated until predefined stopping criteria are met (i.e., model convergence, reaching the upper bound of the model performance)

2.2. Machine Unlearning

Machine unlearning was first officially recognized in 2014, a legal case between Google and a Spanish individual fighting for the right to be forgotten [8]. The unlearning method recently gained its attention and began to be studied in 2021 [2, 25]. As the authority received a removal request on specific data, the data and its influence on the trained model have to be removed as if the model has never seen the forgetting data. Machine unlearning works as follows: (1) A learning model is normally trained on a training dataset. (2) A party requests the removal of its specific data. (3) The trained model unlearns the requested data. (4) The unlearned model goes through performance recovery to maintain its performance on remembered data.

2.3. Federated Unlearning

Even though the federated unlearning procedure is similar to machine unlearning, federated unlearning is a bit trickier. Due to the FL framework's distributed and data-isolated nature, removing a client's whole data became much more difficult as the server had to decide what parameter should be changed by analyzing the model updates rather than the data itself directly. Considering the iterative and stochastic nature of machine learning, it is challenging to clearly distinguish what feature of what data affected which parameters in what way. One of the most promising solutions to it is just retraining the model from scratch without revoking clients. Obviously, however, it is inefficient in terms of time if the training dataset gets larger, and the model architecture becomes more complex. Furthermore, retraining would limit the usage of pre-trained models that have been trained on huge training sets with much better competent resources.

As such, there have been multiple attempts to approximate the model as if the model has not seen to-be-removed data while training. Some of them restored the approximated unlearned model using historical information that the server had collected, cumulatively, periodically, or from previous round [4, 10, 12, 19, 21, 28, 32, 33]. Nonetheless, it is not feasible due to the memory overhead it incurs as the training dataset gets bigger or the model becomes more complex.

Due to the limitation in scalability in terms of memory, some researchers approximated the loss using the inverse Hessian matrix as if the model was not trained on the removal-requested data. Nonetheless, since directly calculating the Hessian matrix incurs costly computational overhead, researchers developed efficiency utilizing the Fisher Information Matrix followed by the Quasi-Newton method [16, 21]. Despite its relatively higher accuracy on approximation, it may prevent the unlearned model from gaining a significant efficiency in terms of both time or memory, over retraining from scratch.

Some works, therefore, focused on reverse learning by modifying objectives, such as loss maximization [14–16] and stochastic gradient ascent [18, 29]. They compute the model updates toward an unlearning objective by maximizing empirical loss. Since it gradually affects the learned model, monitoring the values could make it easier to find a balance between effectiveness (unlearning accuracy) and fidelity (remaining set accuracy) by early stopping. The other line of works involved gradient manipulation that perturb the updates [6, 20, 33] or scale the updates [13] such that the resulting global model could be generalized and reduce the amount of influence by a specific user. Multi-task learning [11, 26–28, 31, 34, 35], where one of the multiple tasks is to unlearn the sample and another is to remember the rest, are the last but not least approach to be mentioned.

Among these works, we built up the method based on Halimi et al.. The following section gives some more details about it.

2.4. Implementing Halimi et al.'s Method

Halimi et al. proposed a Federated Unlearning (FU) method to remove the client's contribution using the Projected Gradient Ascent (PGA) algorithm that reverses the learning process. This method trains a model to *maximize* the local empirical loss with the following objectives:

$$\max_{w \in \{v \in \mathbb{R}^d : \|v - w_{ref}\|_2 \leq \delta\}} F_i(w) \tag{1}$$

where w is weight and δ is a radius of L2 ball. It has been formulated as a constrained maximization problem; nonetheless, deploying the objective alone may unbound the error. Thus, the authors restrict the model parameter to an l2-norm ball around a *reference model* that averages the retaining clients' local models in the last round of FL training and later is compared to the unlearned model (i.e., $w_{ref} = \frac{1}{N-1} \sum_{i \neq j} (w_j^{T-1})$), where N is the total number of clients, i is a target client, and T is the current round. This restriction helps retain the knowledge learned from the other clients' data.

Unfortunately, the unlearning method sacrifices model performance because maximizing local loss where all clients have shared characteristics on datasets and tasks disrupts the model's training on retaining data. The authors performed post-training to mitigate the natural drop in performance after unlearning by continuing the FL epochs for a few more rounds without the target client.

3. Methodology

Halimi et al. have considered the following unlearning setting. A client or a server requests unlearning after normal FL training with N clients for the T rounds finishes. The client requesting removal (i.e., a target client) wants to eliminate the data and all of its influences on the globally trained model, resulting from the regular FL training.

One promising way to remove the influence of the target client's data is to retrain the global model from scratch without the client being involved. However, this method is often expensive and inefficient, especially if the trained dataset is huge and the model is complex. Therefore, the authors proposed an unlearning algorithm that works similarly to the retrained model. Their proposed method is based on reversing the learning process by using projected gradient ascent. First, consider the regular federated learning problem in round t; the goal of each client is to minimize the local loss, i.e., this can be done by solving the following optimization problem:

$$\min_{w \in R^d} F_i(w) := \frac{1}{n_i} \sum_{j \in D_i} L(w; (x_j, y_j)), \tag{2}$$

where w is the local model parameters, and $L(w; (x_j, y_j))$ is the local loss. This loss is calculated based on the local model's prediction of an example (x_j, y_i).

[15] is reversing the learning process. Instead of minimizing the local empirical loss, the target client strives to maximize it. This maximization problem can be solved by gradient ascent. Nonetheless, Halimi et al. argued that since the loss function, e.g., the cross-entropy loss, is often unbounded, solving the maximization problem with gradient ascent will lead an arbitrary model to a random model, and all information to be remembered would be forgotten as well. Thus, to bind the error, the authors restrain the unlearned model's parameter from being close to a reference model that is generated by an average of the retaining clients' updates. The parameters of reference model can be computed as $w_{ref} = \frac{1}{N-1} \sum_{i \neq j} (w_j^{T-1}) = \frac{1}{N-1} (N w^T - w_i^{T-1})$, where w_T is the parameters of global model after T rounds of federated learning, and w_i^{T-1} is the client i's local model update in round $T-1$. Based on this equation, the reference model is computed locally without accessing other clients' models.

The L_2-norm ball of radius δ around w_{ref} can be denoted as $\Omega = \{v \in \mathbb{R}^d : \|v - w_{ref}\|_2 \leq \delta\}$. The projector operator onto Ω can also be denoted as $\rho : R^d \to R^d$. The client i iterates the following update:

$$w \leftarrow \rho(w + \eta_u \nabla F_i(w)) \tag{3}$$

where η_u is the learning rate and $\nabla F_i(w)$ is the gradient of F_i with respect to w. To keep the unlearned model close to the reference model, the optimization problem was constrained by the L_2-norm ball of radius δ around w_{ref}. After that, the target client solves the optimization Equation 1 for unlearning.

For the projection used for projected gradient ascent, we applied the following equation:

$$\rho(w + \eta_u \nabla F_i(w)) = w_{ref} + \frac{w + \eta_u \nabla F_i(w) - w_{ref}}{||w + \eta_u \nabla F_i(w) - w_{ref}||} * \delta \tag{4}$$

This projection should be applied when the $||w + \eta_u \nabla F_i(w) - w_{ref}|| \leq \delta$ otherwise, there is no need for projection to be applied. The intuition behind this projection is that, when the distance between the $w + \eta_u \nabla F_i(w)$ and w_{ref} is greater than the radius of the L_2 norm ball δ, the scale of the gradients will become huge compared to the rest, such that having a higher probability of the model being diverged. Thus, such abnormally huge values should be projected within the radius to reduce the probability of the loss explosion or gradient divergence.

We first find the direction of the vector connecting w_{ref} and $w + \eta_u \nabla F_i(w)$, and then project the value toward this direction within the allowed range δ. The Figure 1 illustrates the projection.

Forgetting some knowledge about trained data would deteriorate or hamper the knowledge gained from retaining (remembered) clients. As such, the unlearned model

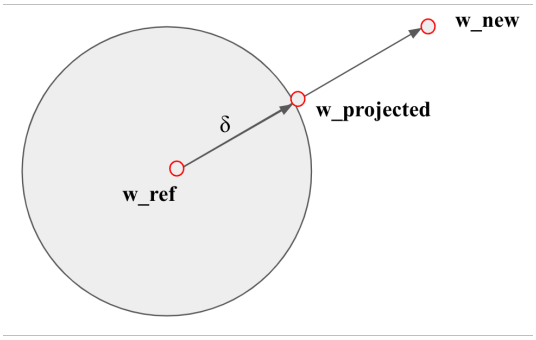
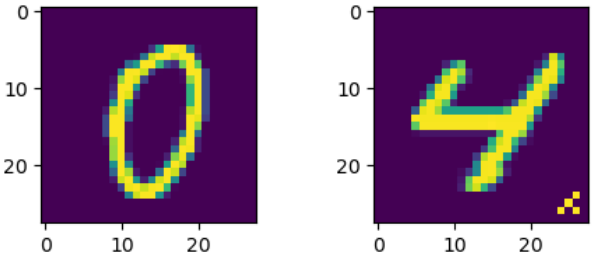


Figure 1. The process of projection.



(a) Benign Sample (label=0) (b) Backdoor Sample (label=9)

Figure 2. Examples of benign and backdoor samples. The pattern trigger has been inserted in the right corner of the sample, and their labels were changed to 9.

should go through a second stage to recover the performance. Halimi et al. performed five rounds of post-training on the unlearned model without the target client's updates. It means that the unlearned model will go through five more FL rounds by averaging $N-1$ model updates to achieve fidelity as high as the one with the retrained model.

4. Experiments

4.1. Dataset and FL settings

We used the MNIST [9] for our initial experiment. The data is uniformly distributed across ten clients (Identically and Independently Distributed, or IID). We supposed one adversary who wants to remove itself participated in the original FL training. We began to embed the backdoored samples after a global model converged. Specifically, we assign a client_9 as a removal requesting client who wants to remove its malicious data and its influence on the global model to prevent it from being detected by the server. The backdoor attack is injected by assigning the client 80% backdoor trigger-inserted and 20% normal samples. Figure 2 are examples of benign and backdoor trigger-inserted samples (backdoor samples) where a 3×3 pattern trigger is inserted in the bottom right corner [24], and the label of backdoor-trigger embedded images were changed to '9'. Furthermore, to make the backdoor attack more aggressive, we increased the number of local training rounds and weighed twice the local model weights for the revoking client to six while all the other clients' were two and not weighted.

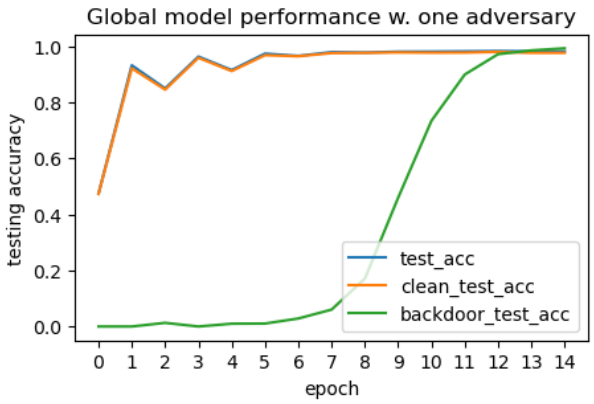


Figure 3. Global model test accuracy with one malicious client

4.2. Model Architecture and Hyperparameters

The model architecture shared among clients is a simple four-layer CNN model (two convolution layers, each followed by a max pool layer and ReLU activation, followed by two fully connected layers, one with and another without ReLU activation). We implemented [15] and followed their hyperparameters according to the description in the paper. Throughout the experiments, we assume there are ten local clients in total, one global server, and a participation fraction of one. Each client runs only one local epoch and then sends the local model to the server for aggregation. The server has a test dataset to evaluate the averaged global model.

4.3. Experimental Results

Table 2 reports the test accuracy where all clients and the test samples are benign and normal. Even though we only performed five global epochs, the test accuracy has reached 0.9456. The final goal of our method is to make the global model generated by retained clients achieve the test accuracy as high as the vanilla test accuracy.

Figure 3 depicts the global model accuracy when a backdoor attack is involved. Specifically, the `test_acc` is the evaluation of a global model on a test dataset, comprised of 20% of backdoor trigger embedded samples and the rest with normal samples; `clean_test_acc` is the evaluation of the global model on a clean data samples only; `backdoor_test_acc` depicts the global model accuracy on backdoored samples only. Figure 4 depicts the same metrics, but the revoked client was removed from the beginning so as to have a baseline method that retrains the model from scratch.

Table 1. Model hyperparameters

Stage	Momentum	Learning Rate	Batch size	# Local Epoch	misc.
FL	0.9	0.01	128	1	Agg. Algo = FedAvg
FU	0.9	0.01	1024	5	L_2 clipping radius = 5
post	0.9	0.01	128	5	Agg. Algo = FedAvg

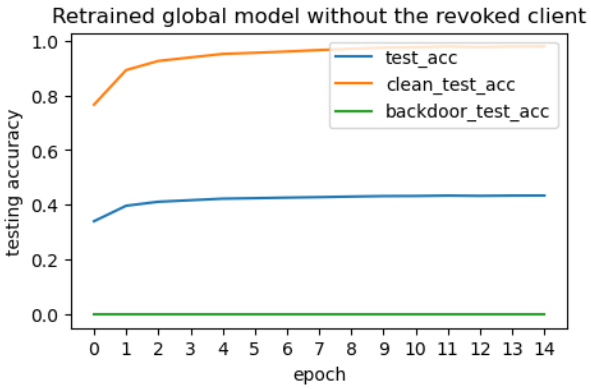


Figure 4. Retrained global model without the revoking client

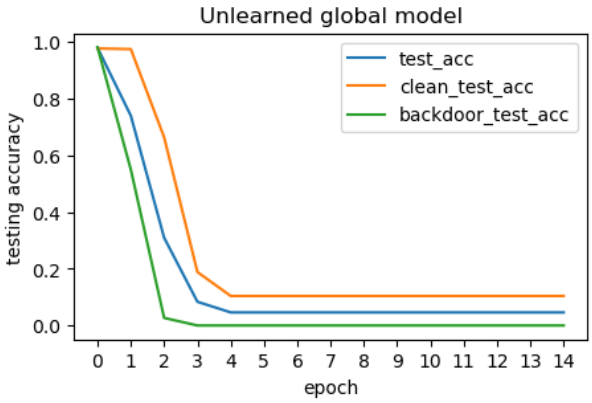


Figure 5. Unlearned global model with [15]

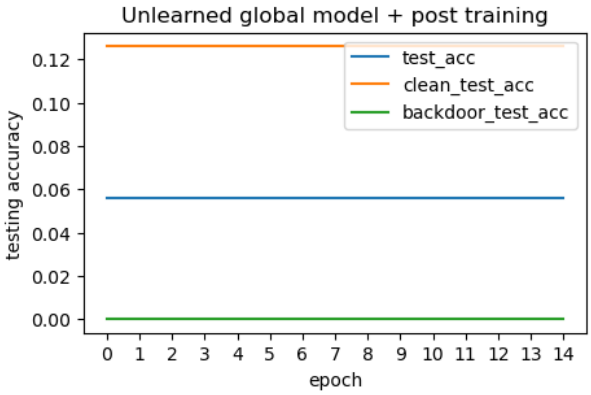


Figure 6. Unlearned and post-trained global model with [15]

Table 2. Global model test accuracy with FedAvg

Epoch	Test Acc
1	0.7070
2	0.8816
3	0.9168
4	0.9351
5	0.9456

Figure 5 illustrates the experiment results on Halimi et al.'s work. The paper argued that they had preserved fidelity by repeating five rounds of post-training without the revoked (malicious) client. As we repeated the experiments, the unlearning with post-training performed poorly, even though the test accuracy on the backdoor dataset was consistently zero. Figure 6 reached 0.1261 as its best `clean_test_acc`, meaning that the retaining datasets' performance was not recovered even after fifteen rounds of post-training.

5. Discussion

As depicted in Figure 3, high `clean_test_acc` and `backdoor_test_acc` indicate that the global model successfully learned the backdoor samples when it was trained with nine benign clients and one adversarial client. We assigned backdoor trigger embedded samples to the last client. Our goal is to train a global model to forget the backdoored sample from the client ('client_9'). We evaluated the efficacy and fidelity of the unlearned model by monitoring attack accuracy and clean accuracy against the result generated by the retrained model.

Unlearning methods are developed to reduce the time and complexity than the retrained-from-scratch model; therefore, the best-performing unlearning method is the one that makes the most similar behavior to the retrained model as depicted in Figure 4. Observing the unlearned model in Figure 5, the backdoor accuracy of the global model has converged to 0. It means the unlearning succeeded; there is no distinguishability to the erased data.

However, the clean test accuracy has dropped by a huge margin (from 0.9834 to 0.1261) Figure 6, so we performed post-training to recover the performance on not-forgotten data. According to the paper [15], they recovered the clean

accuracy within five epochs of post-training without the erased client. We took the same approach, but we were not able to see any improvement in the clean test accuracy of the model. We presume it is because of the loss exploding problem as we observed increasing loss up to the 6-digit numbers during the post-training.

Thus, we tried several strategies to mitigate loss exploding. Even if we reduced the learning rate from 0.01 to 0.005 and 0.001, the loss and clean test accuracy do not meaningfully change. We reduced the batch size from 1024 to 256; it arbitrarily repeated recovery or explosion of clean test accuracy over multiple rounds. Further, we will conduct additional experiments using different optimizers, weight initialization, and regularization terms for the learning objective.

6. Conclusion and Future Works

As concerns about data privacy are increasing due to stringent privacy regulations, Federated Learning emerged in 2014. Recently, the right to be forgotten has also gained attention, Machine Unlearning has begun to develop. Machine unlearning removes a specific client, data, or class as if the model has not seen the revoking data while preserving the accuracy of the rest. Naturally, researchers integrate the concept of federated learning and machine unlearning to achieve unlearning in the context of the FL environment as a means to uphold the right to be forgotten while preserving data privacy, termed Federated Unlearning.

We implement a method proposed by Halimi et al. to effectively remove client data and their influences. Nonetheless, the method was able to remove the effect of data forgetting, but our implementation of their method could not recover the performances on the remembered set even after the post-training.

We found that their method has only been evaluated on simple vision datasets (i.e., MNIST [9], EMNIST [7], and CIFAR10 [17]) and image classification tasks. We will further experiment with the method on different datasets and tasks, such as language (IMDb) or more complex vision datasets (VGGFace2 [3]). We will compare the method with other State-of-the-Art unlearning methods, such as Verifi [13]. Last but not least, we would like to experiment with a client conducting stealthier backdoor attacks and unlearning the compromised data.

References

[1] Manaar Alam, Hithem Lamri, and Michail Maniatakos. Get rid of your trail: Remotely erasing backdoors in federated learning. *arXiv preprint arXiv:2304.10638*, 2023. 1

[2] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021. 2

[3] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*, pages 67–74. IEEE, 2018. 6

[4] Xiaoyu Cao, Jinyuan Jia, Zaixi Zhang, and Neil Zhenqiang Gong. Fedrecover: Recovering from poisoning attacks in federated learning using historical information. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 1366–1383. IEEE, 2023. 1, 2

[5] Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, pages 463–480. IEEE, 2015. 1

[6] Tianshi Che, Yang Zhou, Zijie Zhang, Lingjuan Lyu, Ji Liu, Da Yan, Dejing Dou, and Jun Huan. Fast federated machine unlearning with nonlinear functional theory. 2023. 1, 2

[7] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE, 2017. 6

[8] Court of Justice of the European Union. Google spain sl and google inc. v agencia española de protección de datos, 2014. Judgment of the Court (Grand Chamber), 13 May 2014. Case C-131/12. 1, 2

[9] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. 4, 6

[10] Zihao Deng, Zhaoyang Han, Chuan Ma, Ming Ding, Long Yuan, Chunpeng Ge, and Zhe Liu. Vertical federated unlearning on the logistic regression model. *Electronics*, 12 (14):3182, 2023. 2

[11] Nicola K Dinsdale, Mark Jenkinson, and Ana IL Namburete. Fedharmony: unlearning scanner bias with distributed data. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 695–704. Springer, 2022. 2

[12] Yann Fraboni, Richard Vidal, Laetitia Kameni, and Marco Lorenzi. Sequential informed federated unlearning: Efficient and provable client unlearning in federated optimization. *arXiv preprint arXiv:2211.11656*, 2022. 1, 2

[13] Xiangshan Gao, Xingjun Ma, Jingyi Wang, Youcheng Sun, Bo Li, Shouling Ji, Peng Cheng, and Jiming Chen. Verifi: Towards verifiable federated unlearning. *arXiv preprint arXiv:2205.12709*, 2022. 1, 2, 6

[14] Jinu Gong, Joonhyuk Kang, Osvaldo Simeone, and Rahif Kassab. Forget-svgd: Particle-based bayesian federated unlearning. In *2022 IEEE Data Science and Learning Workshop (DSLW)*, pages 1–6. IEEE, 2022. 2

[15] Anisa Halimi, Swanand Kadhe, Ambrish Rawat, and Nathalie Baracaldo. Federated unlearning: How to efficiently erase a client in fl? *arXiv preprint arXiv:2207.05521*, 2022. 1, 2, 3, 4, 5, 6

[16] Ruinan Jin, Minghui Chen, Qiong Zhang, and Xiaoxiao Li. Forgettable federated linear learning with certified data removal. *arXiv preprint arXiv:2306.02216*, 2023. 1, 2

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[18] Guanghao Li, Li Shen, Yan Sun, Yue Hu, Han Hu, and Dacheng Tao. Subspace based federated unlearning. *arXiv preprint arXiv:2302.12448*, 2023. 1, 2

[19] Gaoyang Liu, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu. Federaser: Enabling efficient client-level data removal from federated learning models. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pages 1–10. IEEE, 2021. 1, 2

[20] Yang Liu, Zhuo Ma, Ximeng Liu, Jian Liu, Zhongyuan Jiang, Jianfeng Ma, Philip Yu, and Kui Ren. Learn to forget: Machine unlearning via neuron masking. *arXiv preprint arXiv:2003.10933*, 2020. 1, 2

[21] Yi Liu, Lei Xu, Xingliang Yuan, Cong Wang, and Bo Li. The right to be forgotten in federated learning: An efficient realization with rapid retraining. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pages 1749–1758. IEEE, 2022. 1, 2

[22] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017. 2

[23] Thanh Tam Nguyen, Thanh Trung Huynh, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. A survey of machine unlearning. *arXiv preprint arXiv:2209.02299*, 2022. 1

[24] Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, et al. Adversarial robustness toolbox v1. 0.0. *arXiv preprint arXiv:1807.01069*, 2018. 4

[25] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning. *Advances in Neural Information Processing Systems*, 34:18075–18086, 2021. 2

[26] Thanveer Shaik, Xiaohui Tao, Lin Li, Haoran Xie, Taotao Cai, Xiaofeng Zhu, and Qing Li. Framu: Attention-based machine unlearning using federated reinforcement learning. *arXiv preprint arXiv:2309.10283*, 2023. 2

[27] Weiqi Wang, Zhiyi Tian, Chenhan Zhang, An Liu, and Shui Yu. Bfu: Bayesian federated unlearning with parameter self-sharing. In *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*, pages 567–578, 2023.

[28] Chen Wu, Sencun Zhu, and Prasenjit Mitra. Unlearning backdoor attacks in federated learning. In *ICLR 2023 Workshop on Backdoor Attacks and Defenses in Machine Learning*, 2023. 1, 2

[29] Leijie Wu, Song Guo, Junxiao Wang, Zicong Hong, Jie Zhang, and Yaohong Ding. Federated unlearning: Guarantee the right of clients to forget. *IEEE Network*, 36(5):129–135, 2022. 1, 2

[30] Leijie Wu, Song Guo, Junxiao Wang, Zicong Hong, Jie Zhang, and Jingren Zhou. On knowledge editing in federated learning: Perspectives, challenges, and future directions. *arXiv preprint arXiv:2306.01431*, 2023. 1

[31] Hui Xia, Shuo Xu, Jiaming Pei, Rui Zhang, Zhi Yu, Weitao Zou, Lukun Wang, and Chao Liu. Fedme 2: Memory evaluation & erase promoting federated unlearning in dtmn. *IEEE Journal on Selected Areas in Communications*, 2023. 2

[32] Wei Yuan, Hongzhi Yin, Fangzhao Wu, Shijie Zhang, Tieke He, and Hao Wang. Federated unlearning for on-device recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 393–401, 2023. 1, 2

[33] Lefeng Zhang, Tianqing Zhu, Haibin Zhang, Ping Xiong, and Wanlei Zhou. Fedrecovery: Differentially private machine unlearning for federated learning frameworks. *IEEE Transactions on Information Forensics and Security*, 2023. 1, 2

[34] Yian Zhao, Pengfei Wang, Heng Qi, Jianguo Huang, Zongzheng Wei, and Qiang Zhang. Federated unlearning with momentum degradation. *IEEE Internet of Things Journal*, 2023. 1, 2

[35] Xiangrong Zhu, Guangyao Li, and Wei Hu. Heterogeneous federated knowledge graph embedding learning and unlearning. In *Proceedings of the ACM Web Conference 2023*, pages 2444–2454, 2023. 2