# UNLEARNING BACKDOOR ATTACKS IN FL

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Backdoor attacks are always a big threat to the federated learning system. Substantial progress has been made to mitigate such attacks during or after the training process. However, how to remove a potential attacker's contribution from the trained global model still remains an open problem. Towards this end, we propose a federated unlearning method to eliminate an attacker's contribution by subtracting the accumulated historical updates from the model and leveraging the knowledge distillation method to restore the model's performance without introducing the backdoor. Our method can be broadly applied to different types of neural networks and does not rely on clients' participation. Thus, it is practical and efficient. Experiments on three canonical datasets demonstrate the effectiveness and efficiency of our method.

## 1 INTRODUCTION

Methods to detect or prevent backdoor attacks on federated learning systems mostly focus on the federated aggregation process where the server receives model updates from all the clients . These methods try to distinguish malicious updates from benign ones. However, when an attacker has been identified, he may have already participated in the FL for some iterations. In this work, we define the *machine unlearning* problem as removing an attacker's contribution thoroughly and efficiently from the global model after the standard federated training process. A naive way to do this is to retrain the model from scratch. However, this can result in an enormous cost of time and energy. Existing works on machine unlearning tasks mostly focus on unlearning in the centralized learning scenario with free access to the training dataset Du et al. (2019); Bourtoule et al. (2021) or does not work for complex models such as deep neural networks (DNNs) Cao & Yang (2015); Ginart et al. (2019); Izzo et al. (2021). The only unlearning work in the FL scenario provides limited improvement over retraining from scratch since it requires clients to retrain the model to calibrate their historical updates Liu et al. (2021). In this work, we address the unlearning problem from a security perspective in the federated learning scenario and provide an improved solution.

In federated learning, reducing the number of iterations between the server and clients is crucial since communication costs a tremendous amount of time and energy, especially for DNNs. Also, to improve efficiency, it is always better to put more computation on the server's side than on the client's side because the server usually has more computation power than the clients. We propose an efficient algorithm following these two rules. Our method can completely remove the backdoor behaviors introduced by the identified attacker at a reasonable loss in model performance. It erases the historical parameter updates from the attacker and recovers from the damage through the knowledge distillation method Hinton et al. (2015). Specifically, we use the old global model as a teacher to train the unlearning model. This approach has many advantages. Firstly, the knowledge distillation training is operated entirely on the server's side without requiring a labeled dataset. So there will be no client-side time and energy costs and no network transmission. Secondly, the backdoor features will not transfer from the teacher model to unlearning model since those features will not be activated without emerging of backdoor patterns Gu et al. (2017). Lastly, distillation prevents the model from overfitting and contributes to a better generalization around training points Papernot et al. (2016b). So it can help improve the robustness of the model and we can further improve the model's performance with post-training afterward.

We show via empirical studies on three datasets using different DNN architectures that our federated unlearning method is effective and efficient.

## 2 DEFENSE METHOD THROUGH UNLEARNING

Our method consists of two steps. The first step is to erase all the historical parameter updates from the attacker. The second step is to recover the model through the knowledge distillation method. This method requires the server to keep the history of parameter updates from each contributing client and possess some extra outsourced unlabeled data.

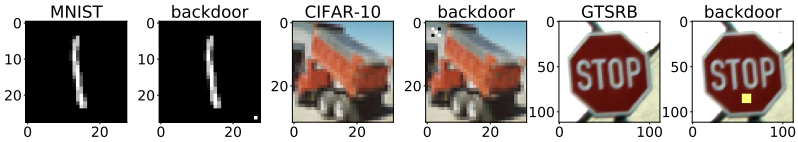



Figure 1: Samples of backdoor images in MNIST, CIFAR-10, and GTSRB dataset.

### 2.1 ERASE HISTORICAL PARAMETER UPDATES

To completely remove the contribution of attacker $i$ to the final global model $M_F$, we want to erase all the historical updates $\Delta M_t^i$ as long as time $t \in [1, F-1]$. The update of the global model at round $t$ consists of averaged weight updates from participating clients. If we use $\Delta M_t$ to represent this update at round $t$, the finalized global model $M_F$ can be viewed as a composition of initial model weight $M_1$ and updates to the global model from round 1 to round $F-1$. We further assume that in each round, $N$ clients participate in the FL training, and without loss of generality, client $N$ is the attacker that needs to be unlearned from the global model.

However, we can not directly accumulate the new updates to reconstruct the unlearning model because of the incremental learning property of FL. Any update to the global model $M_t$ will result in a requirement of updates to all the model updates that happened afterward. Hence, we use $\epsilon_t$ to represent the necessary amendment (skew) to the global model at each round $t$. We can get the unlearning version of the final global model $M'_F$.

$$M'_F = M_1 + \frac{N}{N-1}\sum_{t=1}^{F-1}\Delta M_t - \frac{1}{N-1}\sum_{t=1}^{F-1}\Delta M_t^N + \sum_{t=1}^{F-1}\epsilon_t$$

Because of this characteristic of the incremental learning process in FL, the skew $\epsilon_t$ will increase with more training rounds after updates to the global model. Thus, the above unlearning rule has a shortcoming that when the target client $N$ makes little contribution to the model at round $t$ (e.g., $\Delta M_t^N \approx 0$), the global model update $\Delta M_t$ will still change a lot by multiplying itself with a factor of $\frac{N}{N-1}$. This will bring more skew $\epsilon_t$ to the global model in the following rounds.

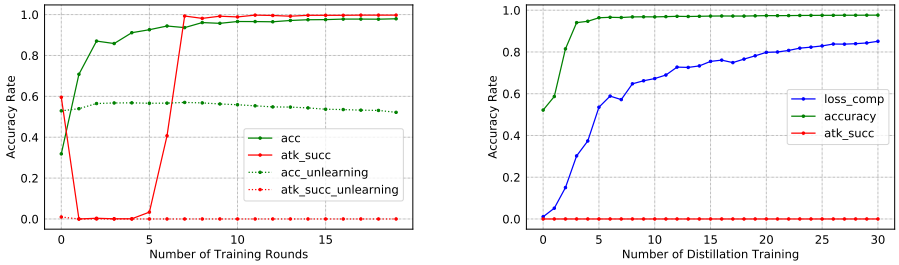
To mitigate this problem, we propose to use a lazy learning strategy to eliminate the influence of target client $N$. Specifically, we assume client $N$ still participated in the training process but set his updates $\Delta M_t^N = 0$ for all rounds $t \in [1, F-1]$. The unlearning result of the final global model can be simplified as:

$$M'_F = M_F - \frac{1}{N}\sum_{t=1}^{F-1}\Delta M_t^N + \sum_{t=1}^{F-1}\epsilon_t \tag{1}$$

Now the unlearning model update rule becomes surprisingly straightforward and easy to understand. We first subtract all the historical averaged updates of attacker $N$ from the final global model $M_F$. Then, we remedy the skew $\epsilon_t$ caused by this process because of the incremental learning characteristic of the FL.

### 2.2 REMEDY WITH KNOWLEDGE DISTILLATION

In the remedy process, we need to improve the model performance while not bringing any backdoors into the global model. To tackle this problem, we propose to leverage the knowledge distillation method to train the unlearning model using the original global model. The distillation technique is first proposed by Hinton et al. (2015) to reduce the size of DNN architectures or ensembles of models. It uses the prediction results of class probabilities produced by an ensemble of models or a complex DNN to train another DNN of a reduced number of parameters without much loss of accuracy. Later on, Papernot et al. (2016a) proposed to use this method as a defensive mechanism to




(a) Training and the corresponding subtraction

(b) Knowledge distillation training

reduce the effectiveness of adversarial samples on DNNs. They showed that the defensive distillation could improve the generalizability and robustness of the trained DNNs. The intuition is based on the fact that the knowledge acquired by DNNs is not only encoded in the weight parameters but also can be reflected in the class probability prediction output of the model.

To perform knowledge distillation in the unlearning problem, we treat the original global model as the teacher model and the skewed unlearning model as the student model. Then, the server can use any unlabeled data to train the unlearning model and remedy the skew $\epsilon_t$ caused by the previous subtraction process. Since the backdoor behavior can only be activated when the input contains backdoor patterns, during the knowledge distillation process, there is no backdoor input, and thus the backdoor behavior cannot be transferred to the student (unlearning) model. The original global model produces class prediction probabilities through a "softmax" output layer that converts the logit, $z_i$, computed for each class into a probability, $q_i$, by comparing $z_i$ with the other logits.

$$q_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)} \tag{2}$$

where $T$ is a hyper-parameter named *temperature* and shared across the softmax layer. The value of $T$ is set to 1 for traditional ML training and predictions. A higher temperature $T$ makes the DNN produce a softer probability distribution over classes. In other words, the probability output will be forced to produce relatively large values for each class, and logits $z_i$ become negligible compared to temperature $T$. We use this soft class prediction probability produced by the original global model $M_F$ to label the dataset. The skewed unlearning model is then trained with this dataset with soft labels (with high temperature $T$). The temperature will be set back to 1 after distillation training. So the unlearning model $M'_F$ can produce more discrete class prediction probabilities during test time.

## 3 Experiments

We evaluate the proposed method using different model architectures with three datasets. The central results show that our unlearning strategies can effectively remove the backdoor from the global model. Moreover, the damage to the model can be quickly recovered through the distillation training process. The attack success rate remains low during this process since the knowledge distillation process will not activate the backdoor behavior and pass the backdoor to the final model.

### 3.1 Overview of the Experimental Setup

We use the following three canonical ML datasets in our experiments: MNIST LeCun et al. (1998), CIFAR-10 Krizhevsky et al. (2009), and GTSRB dataset Stallkamp et al. (2011). In the MNIST experiment, we have 10 clients participating in the FL process with one attacker. We use a CNN with 2 convolutional layers. In the CIFAR-10 experiment, there are also 10 clients with one attacker. We use the VGG11 network Simonyan & Zisserman (2015). In the GTSRB experiment, there are only 5 clients in the FL process with one attacker. We use the AlexNet Krizhevsky et al. (2012). The backdoor attack is triggered by backdoor patterns in the input image. The attacker changes some pixels in the benign inputs to create a backdoor pattern, as shown in Figure 1. Backdoor targets in the experiments include making digit "1" predicted as digit "9" in MNIST, "truck" predicted as "car" in CIFAR-10, and "Stop Sign" predicted as "Speed limit Sign (120 km/h)" in GTSRB dataset.

Table 1: Validation Results of Unlearning Method under Different Stage

| | **Training** | | **UL-Subtract** | | **UL-Distill** | | **Post-Train** | | **Re-Train** | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Dataset** | TA | AA | TA | AA | TA | AA | TA | AA | TA | AA |
| MNIST | 98.0 | 99.7 | 52.2 | 0 | 97.7 | 0 | 98.5 | 0 | 98.2 | 0 |
| CIFAR-10 | 80.8 | 99.4 | 10.0 | 0 | 78.8 | 6.4 | 81.4 | 7.3 | 79.5 | 7.0 |
| GTSRB | 93.0 | 100 | 3.9 | 0 | 92.1 | 0 | 94.0 | 0 | 92.7 | 0 |

### 3.2 Unlearning Model Performance Evaluation

In this section, we evaluate the performance of the unlearning model under different steps during unlearning. To begin with, we show, in Figure 2a, the behavior of the model after directly erasing all the historical parameter updates from the target client (attacker). Subtracting the parameter updates from the global model will create a non-negligible skew. As we observe from the figure, the unlearning model's accuracy has never exceeded 60% and has a trend of slightly decreasing with more FL training rounds. The advantage of this process is that it can thoroughly remove the backdoor from the global model. Compared with the original global model with an almost 100% backdoor attack success rate, the unlearning model keeps the attack success rate 0% all the time.

Then, we can look at the following knowledge distillation process that is used to recover the skew caused by the subtraction process. From Figure 2b, we notice that the test accuracy of the model is quickly recovered within five epochs of distillation training. The loss on the test dataset keeps getting closer to the original global model while the backdoor attack success rate keeps as low as 0% all the time. The attacker's influence on the global model does not transmit to the unlearning model after the knowledge distillation training process. The reason is that the distillation training method does not use any data from the target client (attacker), so the backdoor in the original model has not been triggered and thus will not be learned by the unlearning model.

In the end, Table 1 reports the integrated experiment results with different datasets and model architectures. The "Training" column reports the performance of global model $M_F$ on the evaluation dataset, including the test accuracy (TA) and backdoor attack accuracy rate (AA). The "UL-Subtract" column reports the unlearning model's performance after merely subtracting the target client's historical parameter updates from the global model. The "UL-Distill" column reports the performance of unlearning model after the knowledge distillation process on the server's side. The "Post-Train" column represents how much we can further improve the unlearning model if putting it back to the FL system and continue training without the participation of the target client (attacker). The "Re-Train" column is the widely used golden standard for the unlearning problem. We retrain the model from scratch, excluding the participation of the attacker. From the results, we can conclude that subtracting the attacker's historical parameter updates eliminates the backdoor on the global model. In all cases, the attack success rate drops to zero. The knowledge distillation training process can help remedy the skew caused by subtraction and recover the model's performance back to an acceptable standard. The test accuracy of the unlearning model after distillation is almost identical to that of the model retraining from scratch (with differences less than 1%). In the post-training results, we find the distillation can even help improve the model's accuracy with follow-up training with clients. What is more, we are delighted to observe that the distillation process does not pass the backdoor from the original global model to the unlearning model. It supports our assumption that as long as we are not using the same dataset to activate the global model, the original global model's backdoor behavior will not be learned by the unlearning model.

## 4 Conclusion

We propose a federated unlearning method to eliminate the backdoor attacks implanted by the identified attacker. Our method can fully distinguish the attacker's updates to the global model by subtracting its historical parameter updates from the model. Then, we use the knowledge distillation method to remedy the skew of the unlearning model caused by the subtraction and do not transfer any backdoor behaviors. Empirical studies from our experiments on three canonical datasets have demonstrated the effectiveness of our defense method. Since our method is purely conducted on the server's side, the run time and energy cost efficiency ultimately defeat other existing unlearning methods that require extra communication between clients and server.

## REFERENCES

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*, pp. 141–159. IEEE, 2021. doi: 10.1109/SP40001.2021.00019. URL https://doi.org/10.1109/SP40001.2021.00019.

Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pp. 463–480. IEEE Computer Society, 2015. doi: 10.1109/SP.2015.35. URL https://doi.org/10.1109/SP.2015.35.

Min Du, Zhi Chen, Chang Liu, Rajvardhan Oak, and Dawn Song. Lifelong anomaly detection through unlearning. In *CCS 2019*, 2019. doi: 10.1145/3319535.3363226. URL https://doi.org/10.1145/3319535.3363226.

Antonio Ginart, Melody Y. Guan, Gregory Valiant, and James Zou. Making AI forget you: Data deletion in machine learning. In *NeurIPS 2019*, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/cb79f8fa58b91d3af6c9c991f63962d3-Abstract.html.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *CoRR*, abs/1708.06733, 2017. URL http://arxiv.org/abs/1708.06733.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531.

Zachary Izzo, Mary Anne Smart, Kamalika Chaudhuri, and James Zou. Approximate data deletion from machine learning models. In *AISTATS 2021*, 2021. URL http://proceedings.mlr.press/v130/izzo21a.html.

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

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS 2012*, 2012. URL https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Gaoyang Liu, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu. Federaser: Enabling efficient client-level data removal from federated learning models. In *IWQOS 2021*, pp. 1–10. IEEE, 2021. doi: 10.1109/IWQOS52092.2021.9521274. URL https://doi.org/10.1109/IWQOS52092.2021.9521274.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*, pp. 582–597. IEEE, 2016a.

Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symposium on Security and Privacy, SP 2016*, pp. 582–597. IEEE Computer Society, 2016b. doi: 10.1109/SP.2016.41. URL https://doi.org/10.1109/SP.2016.41.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR 2015*, 2015. URL http://arxiv.org/abs/1409.1556.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *IEEE International Joint Conference on Neural Networks*, pp. 1453–1460, 2011.

# A APPENDIX

## A.1 ERASE HISTORICAL PARAMETER UPDATES

To completely remove the contribution of any client $i$ to the final global model $M_F$, we want to erase all the historical updates $\Delta M_t^i$ from this client as long as $t \in [1, F-1]$. The update of the global model at round $t$ consists of averaged weight updates from participating clients. If we use $\Delta M_t$ to represent this update at round $t$, the finalized global model $M_F$ can be viewed as a composition of initial model weight $M_1$ and updates to the global model from round 1 to round $F-1$.

$$M_F = M_1 + \sum_{t=1}^{F-1} \Delta M_t \tag{3}$$

For simplicity, we assume that each round, there are $N$ clients participating in the FL training, and client $N$ is the target client that wants to be unlearned from the global model. At this time, we can simplify the problem as to remove the contribution $\Delta M_t^N$ of the target client $N$ from the global model update $\Delta M_t$ at each round $t$.

$$\Delta M_t = \frac{1}{N} \sum_{i=1}^{N} \Delta M_t^i = \frac{1}{N} \sum_{i=1}^{N-1} \Delta M_t^i + \frac{1}{N} \Delta M_t^N$$

There are two ways to calculate the new global model update $\Delta M_t'$ at round $t$. The first one is to assume that only $N-1$ clients were participating in the FL at round $t$. In this way, the new global model updates $\Delta M_t'$ at round $t$ becomes the following equation.

$$\Delta M_t' = \frac{1}{N-1} \sum_{i=1}^{N-1} \Delta M_t^i = \frac{N}{N-1} \Delta M_t - \frac{1}{N-1} \Delta M_t^N$$

However, we can not directly accumulate the new updates to reconstruct the unlearning model because of the incremental learning property of FL, as discussed in the previous section. Any update to the global model $M_t$ will result in a requirement of updates to all the model updates that happened afterward. Hence, we use $\epsilon_t$ to represent the necessary amendment (skew) to the global model at each round $t$. After combining the above equation with Equation 3, we can get the unlearning version of the final global model $M_F'$.

$$M_F' = M_1 + \frac{N}{N-1} \sum_{t=1}^{F-1} \Delta M_t - \frac{1}{N-1} \sum_{t=1}^{F-1} \Delta M_t^N + \sum_{t=1}^{F-1} \epsilon_t$$

where $\epsilon_t$ is the necessary correction to amend the skew produced by change of the model at previous rounds. Because of this characteristic of the incremental learning process in FL, the skew $\epsilon_t$ will increase with more training rounds after updates to the global model. Thus, the above unlearning rule has a shortcoming that when the target client $N$ makes little contribution to the model at round $t$ (e.g., $\Delta M_t^N \approx 0$), the global model update $\Delta M_t$ will still change a lot by multiplying itself with a factor of $\frac{N}{N-1}$. This will bring more skew $\epsilon_t$ to the global model in the following rounds.

To mitigate this problem, we propose to use a lazy learning strategy to eliminate the influence of target client $N$. Specifically, we assume client $N$ still participated in the training process but set his updates $\Delta M_t^N = 0$ for all rounds $t \in [1, F-1]$. The unlearning of the global model update can be simplified as follows.

$$\Delta M_t' = \frac{1}{N} \sum_{i=1}^{N-1} \Delta M_t^i = \Delta M_t - \frac{1}{N} \Delta M_t^N$$

A combination of the above formula with Equation 3 gives us the unlearning result of the final global model $M'_F$.

$$
\begin{aligned}
M'_F &= M_1 + \sum_{t=1}^{F-1} \Delta M'_t + \sum_{t=1}^{F-1} \epsilon_t \\
&= M_1 + \sum_{t=1}^{F-1} \Delta M_t - \frac{1}{N} \sum_{t=1}^{F-1} \Delta M_t^N + \sum_{t=1}^{F-1} \epsilon_t \\
&= M_F - \frac{1}{N} \sum_{t=1}^{F-1} \Delta M_t^N + \sum_{t=1}^{F-1} \epsilon_t
\end{aligned} \tag{4}
$$

Now the unlearning model update rule becomes surprisingly straightforward and easy to understand. We just need to subtract all the historical averaged updates from target client $N$ from the final global model $M_F$. Then, we remedy the skew $\epsilon_t$ caused by this process because of the incremental learning characteristic of the FL.