How To Backdoor Federated Learning

Eugene Bagdasaryan **Andreas Veit** **Yiqing Hua** **Deborah Estrin** **Vitaly Shmatikov**
Cornell Tech

Abstract

Federated models are created by aggregating model updates submitted by participants. To protect confidentiality of the training data, the aggregator by design has no visibility into how these updates are generated. We show that this makes federated learning vulnerable to a model-poisoning attack that is significantly more powerful than poisoning attacks that target only the training data.

A single or multiple malicious participants can use *model replacement* to introduce backdoor functionality into the joint model, e.g., modify an image classifier so that it assigns an attacker-chosen label to images with certain features, or force a word predictor to complete certain sentences with an attacker-chosen word. We evaluate model replacement under different assumptions for the standard federated-learning tasks and show that it greatly outperforms training-data poisoning.

Federated learning employs secure aggregation to protect confidentiality of participants' local models and thus cannot detect anomalies in participants' contributions to the joint model. To demonstrate that anomaly detection would not have been effective in any case, we also develop and evaluate a generic constrain-and-scale technique that incorporates the evasion of defenses into the attacker's loss function during training.

1 Introduction

Federated learning (McMahan et al., 2017) is an attractive framework for the massively distributed training of deep learning models with thousands or even millions of participants (Bonawitz et al., 2019; Hard et al., 2018). In every round, the central server distributes the current joint model to a random subset of participants. Each of them trains locally and submits an updated model to the server, which averages the updates into the new joint model. Motivating applications include training image classifiers and next-word predictors on users' smartphones. To take advantage of a wide range of non-i.i.d. training data while ensuring participants' privacy, federated learning by design has no visibility into participants' local data and training.

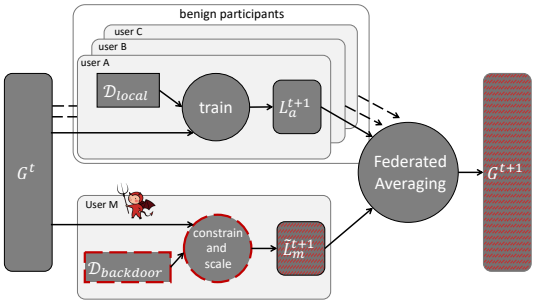

Figure 1: **Overview of the attack.** The attacker compromises one or more participants, trains on the backdoor data using our constrain-and-scale technique, and submits the resulting model, which replaces the joint model as the result of federated averaging.

Our main insight is that **federated learning is generically vulnerable to model poisoning**, which is a new class of poisoning attacks introduced in this paper. Previous poisoning attacks target only the training data. Model poisoning exploits the fact that federated learning gives malicious participants direct influence over the joint model, enabling significantly more powerful attacks than training-data poisoning.

We show that any participant in federated learning can replace the joint model with another so that (i) the new model is equally accurate on the federated-learning task, yet (ii) the attacker controls how the model performs on an attacker-chosen **backdoor** subtask. For example, a backdoored image-classification model misclassifies images with certain features to an attacker-chosen class; a backdoored word-prediction model predicts attacker-

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

chosen words for certain sentences.

Fig. 1 gives a high-level overview of this attack. Our key insight is that a participant in federated learning can (1) directly influence the weights of the joint model, and (2) train in any way that benefits the attack, e.g., arbitrarily modify the weights of its local model and/or incorporate the evasion of potential defenses into its loss function during training.

We demonstrate the power of model replacement on two concrete learning tasks from the federated-learning literature: image classification on CIFAR-10 and word prediction on a Reddit corpus. Even a single-shot attack, where *a single attacker is selected in a single round of training*, causes the joint model to achieve 100% accuracy on the backdoor task. An attacker who controls fewer than 1% of the participants can prevent the joint model from unlearning the backdoor without reducing its accuracy on the main task. Model replacement greatly outperforms "traditional" data poisoning: in a word-prediction task with 80,000 participants, compromising just 8 is enough to achieve 50% backdoor accuracy, as compared to 400 malicious participants needed for the data-poisoning attack.

We argue that federated learning is generically vulnerable to backdoors and other model-poisoning attacks. First, when training with millions of participants, it is impossible to ensure that none of them are malicious. The possibility of training with multiple malicious participants is explicitly acknowledged in (Bonawitz et al., 2019). Second, federated learning can use neither defenses against data poisoning, nor anomaly detection because they require access to, respectively, the participants' training data or their model updates. The aggregation server cannot observe either without breaking participants' privacy (Melis et al., 2019; Nasr et al., 2019). Further, federated learning employs "secure aggregation" (Bonawitz et al., 2017), which provably prevents anyone from auditing participants' data or updates.

Proposed techniques for Byzantine-tolerant distributed learning make assumptions that are explicitly false for federated learning with adversarial participants (e.g., they assume that the participants' training data are i.i.d., unmodified, and equally distributed). We show how to exploit some of these techniques, such as Krum sampling (Blanchard et al., 2017), to make the attack *more* effective. Participant-level differential privacy (Geyer et al., 2018; McMahan et al., 2018) partially mitigates the attack, but at the cost of reducing the joint model's accuracy on its main task.

Model replacement will remain effective even if anomaly detection is somehow incorporated into secure aggregation. We develop a generic *constrain-and-scale* technique that incorporates evasion of anomaly detection into the attacker's loss function. The resulting models evade even relatively sophisticated detectors, e.g., those that measure cosine similarity between submitted models and the joint model. We also develop a simpler, yet effective *train-and-scale* technique to evade anomaly detectors that look at the model's weights or its accuracy on the main task (Fung et al., 2018; Shayan et al., 2018; Shen et al., 2016).

2 Related Work

Training-time attacks. "Traditional" poisoning attacks compromise the training data to change the model's behavior at test time (Huang et al., 2011; Steinhardt et al., 2017). Previous backdoor attacks change the model's behavior only on specific attacker-chosen inputs via data poisoning (Gu et al., 2017; Liu et al., 2017), or by inserting a backdoored component directly into a stationary model (Ji et al., 2018). We show that these attacks are not effective against federated learning, where the attacker's model is aggregated with hundreds or thousands of benign models.

Defenses against poisoning remove outliers from the training data (Qiao and Valiant, 2018; Steinhardt et al., 2017) or, in the distributed setting, from the participants' models (Fung et al., 2018), or require participants to submit their data for centralized training (Hayes and Ohrimenko, 2018). Defenses against backdoors use techniques such as fine-pruning (Liu et al., 2018), filtering (Turner et al., 2018), or various types of clustering (Chen et al., 2018; Tran et al., 2018).

All above defenses require the defender to inspect either the training data, or the resulting models (which leak the training data (Melis et al., 2019; Nasr et al., 2019; Shokri et al., 2017)). None can be applied to federated learning, which must keep participants' data and models confidential. NeuralCleanse (Wang et al., 2019) works only against pixel-pattern backdoors in image classifiers with a limited number of classes; we demonstrate semantic backdoors that work in the text domain with thousands of labels. Similarly, STRIP (Gao et al., 2019) and DeepInspect (Chen et al., 2019a) only target pixel-pattern backdoors. Moreover, DeepInspect attempts to invert the model to extract the training data, thus violating the privacy requirement of federated learning.

Furthermore, none of these defenses are effective even in the setting for which they were designed because they can be evaded by a defense-aware attacker (Baruch et al., 2019; Tan and Shokri, 2019).

Test-time attacks. Adversarial examples (Goodfellow et al., 2015) are deliberately crafted to be misclassi-

fied by the model. By contrast, we introduce semantic backdoors that cause the model to misclassify even *unmodified* inputs.

Secure ML. Secure multi-party computation can help protect privacy of the training data (Mohassel and Zhang, 2017), but it does not protect model integrity. Solutions such as training secret models on encrypted, vertically partitioned data (Hardy et al., 2017) are not applicable to federated learning.

Secure aggregation of model updates (Bonawitz et al., 2017) is essential for privacy because model updates leak sensitive information about participants' training data (Melis et al., 2019). Secure aggregation makes our attack easier because it prevents the central server from detecting anomalous updates and tracing them to a specific participant(s).

Participant-level differential privacy. Differentially private federated learning (Geyer et al., 2018; McMahan et al., 2018) bounds each participant's influence over the joint model. In Appendix B.2, we show that this mitigates the attack only by proportionally degrading the model's accuracy.

Byzantine-tolerant distributed learning. Recently proposed alternative aggregation mechanisms (Blanchard et al., 2017; Damaskinos et al., 2019) ensure *convergence* (but not integrity) in the presence of Byzantine participants. Their key assumptions—the participants' training data are i.i.d., or even unmodified and equally distributed—are explicitly false for federated learning. Some aggregation mechanisms even make our attack stronger, while others, such as coordinate-wise or geometric medians, greatly reduce the accuracy of complex models on non-i.i.d. data and cannot protect confidentiality of training data (see Appendix B.3).

3 Federated Learning

Federated learning (McMahan et al., 2017) distributes the training of a deep neural network across n participants by iteratively aggregating local models into a joint global model. The motivations are efficiency—n can be millions—and privacy. Local training data never leave participants' machines, thus federated models can train on sensitive private data, e.g., users' typed messages, that are substantially different from publicly available datasets (Hard et al., 2018).

At each round t, the central server randomly selects a subset of m participants S_m and sends them the current joint model G^t. Choosing m involves a tradeoff between the efficiency and speed of training. Each selected participant updates this model to a new local model L^{t+1} by training on their private data and sends the difference $L_i^{t+1} - G^t$ back. Communication overhead can be reduced by applying a random mask to the model weights (Konečnỳ et al., 2016). The central server averages the received updates to create the new joint model:

$$G^{t+1} = G^t + \frac{\eta}{n} \sum_{i=1}^{m} (L_i^{t+1} - G^t) \tag{1}$$

Global learning rate η controls the fraction of the joint model that is updated every round; if $\eta = \frac{n}{m}$, the model is fully replaced by the average of the local models. Tasks like CIFAR-10 require lower η to converge, while training with $n = 10^8$ users requires larger η for the local models to have impact on the joint model. In comparison to synchronous distributed SGD (Chen et al., 2016), federated learning reduces the number of participants per round and converges faster. Empirically, common image-classification and word-prediction tasks converge in fewer than 10,000 rounds (McMahan et al., 2017).

4 Adversarial Model Replacement

Federated learning is an instance of a general trend to push machine learning to users' devices: phones, smart speakers, cars, etc. Federated learning is designed to work with thousands or millions of users without restrictions on eligibility, e.g., by enrolling individual smartphones (Google, 2019). Similarly, crowd-sourced ML frameworks such as OpenMined accept anyone running the (possibly modified) learning software.

Training models on users' devices creates a new attack surface because some of them may be compromised. When training with thousands of users, there does not appear to be any way to exclude adversarial participants by relying solely on the devices' own security guarantees. Following an unpublished version of this work, training with multiple malicious participants is now acknowledged as a realistic threat by the designers of federated learning (Bonawitz et al., 2019).

Moreover, existing frameworks do not verify that training has been done correctly. As we show in this paper, a compromised participant can submit a malicious model which is not only trained for the assigned task, but also contains backdoor functionality. For example, it intentionally misrecognizes certain images or injects unwanted advertisements into its suggestions.

4.1 Threat model

Federated learning gives the attacker full control over one or several participants, e.g., smartphones whose learning software has been compromised by malware.

(1) The attacker controls the local training data of any compromised participant; (2) it controls the local training procedure and the hyperparameters such as the number of epochs and learning rate; (3) it can modify the weights of the resulting model before submitting it for aggregation; and, (4) it can adaptively change its local training from round to round.

The attacker does not control the aggregation algorithm used to combine participants' updates into the joint model, nor any aspects of the benign participants' training. We assume that they create their local models by correctly applying the training algorithm prescribed by federated learning to their local data.

The main difference between this setting and the traditional poisoning attacks (see Section 2) is that the latter assume that the attacker controls a significant fraction of the training data. By contrast, in federated learning the attacker controls the entire training process—but only for one or a few participants.

Objectives of the attack. Our attacker wants federated learning to produce a joint model that achieves high accuracy on both its main task and an attacker-chosen *backdoor subtask* and retains high accuracy on the backdoor subtask for multiple rounds after the attack. By contrast, traditional data poisoning aims to change the performance of the model on large parts of the input space (Steinhardt et al., 2017), while Byzantine attacks aim to prevent convergence (Blanchard et al., 2017; El Mhamdi et al., 2018).

A security vulnerability is dangerous even if it cannot be exploited every single time and if it is patched some time after exploitation. By the same token, a model-replacement attack is successful if it sometimes introduces the backdoor (even if it sometimes fails), as long as the model exhibits high backdoor accuracy for at least a single round. In practice, the attack performs much better and the backdoor stays for many rounds.

We propose an new type of backdoors, **semantic backdoors** that, unlike pixel backdoors, cause the model to produce an attacker-chosen output on *unmodified* digital inputs. For example, a backdoored image-classification model assigns an attacker-chosen label to all images with certain features, e.g., all purple cars or all cars with a racing stripe are misclassified as birds (or any other label chosen by the attacker). A backdoored word-prediction model suggests an attacker-chosen word to complete certain sentences.

4.2 Constructing the attack model

Naive approach. The attacker can simply train its model on backdoored inputs. Following Gu et al. (2017), each training batch should include a mix of correctly labeled inputs and backdoored inputs to help the model learn to recognize the difference. The attacker can also change the local learning rate and the number of local epochs to maximize the overfitting to the backdoored data.

Even this attack immediately breaks distributed learning with synchronized SGD (Shokri and Shmatikov, 2015), which applies participants' updates directly to the joint model, thus introducing the backdoor. A recent defense by Damaskinos et al. (2018) requires the loss function to be Lipschitz and thus does not apply in general to large neural networks (See Appendix. B.3).

The naive approach does not work against federated learning. Aggregation cancels out most of the backdoored model's contribution and the joint model quickly forgets the backdoor. The attacker needs to be selected often and even then the poisoning is very slow. In our experiments, we use the naive approach as the baseline.

Model replacement. In this method, the attacker ambitiously attempts to substitute the new global model G^{t+1} with a malicious model X in Eq. 1:

$$X = G^t + \frac{\eta}{n} \sum_{i=1}^{m} (L_i^{t+1} - G^t) \tag{2}$$

Because of the non-i.i.d. training data, each local model may be far from G^t. As the global model converges,

Algorithm 1 Create a model that does not look anomalous and replaces the global model after averaging with the other participants' models.

Constrain-and-scale$(\mathcal{D}_{local}, D_{backdoor})$
Initialize attacker's model X and loss function l:
  $X \leftarrow G^t$
  $\ell \leftarrow \alpha \cdot \mathcal{L}_{class} + (1-\alpha) \cdot \mathcal{L}_{ano}$
for epoch $e \in E_{adv}$ **do**
  if $\mathcal{L}_{class}(X, D_{backdoor}) < \epsilon$ **then**
    // Early stop, if model converges
    break
  end if
  for batch $b \in \mathcal{D}_{local}$ **do**
    // inject c backdoors to the batch b
    $b \leftarrow \texttt{replace}(c, b, D_{backdoor})$
    $X \leftarrow X - lr_{adv} \cdot \nabla \ell(X, b)$
  end for
  if epoch $e \in$ *step_sched* **then**
    // reduce learning rate
    $lr_{adv} \leftarrow lr_{adv}/step\_rate$
  end if
end for
// Scale up the model before submission.
$\widetilde{L}^{t+1} \leftarrow \gamma(X - G^t) + G^t$
return $\widetilde{L}^{t+1}$

these deviations start to cancel out, i.e., $\sum_{i=1}^{m-1}(L_i^{t+1} - G^t) \approx 0$. Therefore, the attacker can solve for the model it needs to submit $\widetilde{L}_m^{t+1}$ as follows:

$$\frac{n}{\eta}X - (\frac{n}{\eta} - 1)G^t - \sum_{i=1}^{m-1}(L_i^{t+1} - G^t) \approx \frac{n}{\eta}(X - G^t) + G^t \tag{3}$$

This attack scales up the weights of the backdoored model X by $\gamma = \frac{n}{\eta}$ to ensure that the backdoor survives the averaging and the global model is replaced by X. This works in any round of federated learning but is more effective when the global model is close to convergence (see Appendix A.2).

Estimating global parameters. An attacker who does not know n and η can approximate the scaling factor γ by iteratively increasing it every round, once selected, and measuring the accuracy of the model on the backdoor task. Scaling by $\gamma < \frac{n}{\eta}$ does not fully replace the global model, but the attack still achieves good backdoor accuracy (see Appendix A.3).

Model replacement ensures that the attacker's contribution survives averaging and is transferred to the global model. It is a **single-shot attack**: the global model exhibits high accuracy on the backdoor task immediately after it has been poisoned.

4.3 Improving persistence and evading anomaly detection

Because the attacker may be selected only for a single round of training, he wants the backdoor to remain in the model for as many rounds as possible after the model has been replaced. Preventing the backdoor from being forgotten as the model is updated by benign participants is similar to the *catastrophic forgetting* problem in multi-task learning (Kirkpatrick et al., 2017).

In effect, our attack involves two-task learning, where the global model learns the main task during normal training and the backdoor task only during the rounds when the attacker was selected. The attacker's objective is to maintain high accuracy for both tasks. Empirically, EWC loss (Kirkpatrick et al., 2017) did not improve our results, but we used other techniques such as slowing down the learning rate lr_{adv} during the attacker's training to improve the persistence of the backdoor in the joint model.

Federated learning uses *secure aggregation* (Bonawitz et al., 2017) to provably prevent the aggregator from inspecting the models submitted by the participants. Therefore, **there is no way to detect that aggregation includes a malicious model, nor who submitted this model.**

Without secure aggregation, privacy is lost, but the aggregator may attempt to filter out "anomalous" contributions. Since the weights of a model created using Eq. 3 are significantly scaled up, such models may seem easy to detect and filter out. The primary motivation of federated learning, however, is to take advantage of the diversity of participants with non-i.i.d. training data, including unusual or low-quality local data such as smartphone photos or text-messaging history (McMahan et al., 2017). Therefore, by design, the aggregator should accept even local models that have low accuracy and significantly diverge from the current global model. In Appendix C.1, we concretely show how the fairly wide distribution of benign participants' models enables the attacker to create backdoored models that do not appear anomalous.

Constrain-and-scale. We now describe a generic method that enables the adversary to produce a model that has high accuracy on both the main and backdoor tasks, yet is not rejected by the aggregator's anomaly detector. Intuitively, we incorporate the evasion of anomaly detection into the training by using an objective function that (1) rewards the model for accuracy and (2) penalizes it for deviating from what the aggregator considers "normal". Following Kerckhoffs's Principle, we assume that the anomaly detection algorithm is known to the attacker.

Algorithm 1 is our *constrain-and-scale* method. We modify the objective (loss) function by adding an anomaly detection term $\mathcal{L}_{ano}$:

$$\mathcal{L}_{model} = \alpha\mathcal{L}_{class} + (1 - \alpha)\mathcal{L}_{ano} \tag{4}$$

Because the attacker's training data includes both benign and backdoor inputs, $\mathcal{L}_{class}$ captures the accuracy on both the main and backdoor tasks. L_{ano} accounts for any type of anomaly detection, such as the p-norm distance between weight matrices. Additionally, recent backdoor attack (Tan and Shokri, 2019) demonstrates effectiveness of evading defenses such as Neural Cleanse (Wang et al., 2019) and activation clustering (Chen et al., 2018) by using a defense-aware L_{ano}. The hyperparameter α controls the importance of evading anomaly detection. In Appendix C.2, we evaluate the tradeoff between the success of the attack and the "anomalousness" of the backdoored model for various anomaly detectors and different values of α.

Train-and-scale. Anomaly detectors that consider only the magnitudes of model weights (Shen et al., 2016) can be evaded using a simpler technique. The attacker trains the backdoored model until it converges and then scales up the model weights by γ up to the bound S permitted by the anomaly detector (we discuss

estimation of S in Appendix C.1):

$$\gamma = \frac{S}{||X - G^t||_2} \quad (5)$$

5 Experiments

We use image-classification and word-prediction tasks from the federated learning literature.

5.1 Image classification

Following (McMahan et al., 2017), we use CIFAR-10 dataset for our image classification task and train a global model with 100 total participants, 10 of whom are selected randomly in each round. We use the lightweight ResNet18 CNN model (He et al., 2016) with 0.27 million parameters. To simulate non-i.i.d. training data and supply each participant with an unbalanced sample from each class, we divide the 50,000 training images using a Dirichlet distribution with hyperparameter 0.9. Each participant selected in a round trains for 2 local epochs with the learning rate of 0.1.

Backdoors. As the running example, suppose that the attacker wants the joint model to misclassify car images with certain features as *birds* while classifying other inputs correctly. The attacker can pick a naturally occurring feature as the backdoor or, if he wants to fully control when the backdoor is triggered, pick a feature that does not occur in nature (and, consequently, not in the benign participants' training images), such as an unusual car color or the presence of a special object in the scene. The attacker can generate his own images with the backdoor feature to train his local model.

This is an example of a semantic backdoor. In contrast to the pixel-pattern backdoor (Gu et al., 2017) and adversarial transformations, triggering this backdoor does not require the attacker to modify, and thus access, the physical scene or the digital image at inference time.

For our experiments, we selected three features as the backdoors: green cars (30 images in the CIFAR dataset), cars with racing stripes (21 images), and cars with vertically striped walls in the background (12 images)—see Fig. 2(a). We chose these features because CIFAR already contains images that can be used to train the backdoored model. We split the data so that only the attacker has training images with the backdoor feature, but this is not essential: if the backdoor feature is similar to some features that occur in the benign participants' datasets, the attack still succeeds but the joint model forgets the backdoor faster.

When training the attacker's model, we follow Gu et al. (2017) and mix backdoor images with benign images in every training batch ($c = 20$ backdoor images per batch of size 64). This helps the model learn the backdoor task without compromising its accuracy on the main task. The participants' training data are very diverse and the backdoor images represent only a tiny fraction, thus introducing the backdoor has little to no effect on the main-task accuracy of the joint model.

Our attack is effective with pixel-pattern backdoors too (see Appendix A.5). During the attacker's training, we add a special pixel pattern to 5 images in a batch of 64 and change their labels to *bird*. Unlike semantic backdoors, this backdoor requires both a training-time and inference-time attack.

5.2 Word prediction

Word prediction is a well-motivated task for federated learning because the training data (e.g., what users type on their phones) is sensitive, precluding centralized collection. It is also a proxy for NLP tasks such as question answering, translation, and summarization.

We use the PyTorch (Paszke et al., 2017) word prediction example code based on Inan et al. (2017). The model is a 2-layer LSTM with 10 million parameters trained on a randomly chosen month (November 2017) from the public Reddit dataset[1] as in McMahan et al. (2017). Under the assumption that each Reddit user is an independent participant in federated learning and to ensure sufficient data from each user, we filter out those with fewer than 150 or more than 500 posts, leaving a total of 83,293 participants with 247 posts each on average. We consider each post as one sentence in the training data. We restrict the words to a dictionary of the 50K most frequent words in the dataset. Following McMahan et al. (2017), we randomly select 100 participants per round. Each selected participant trains for 2 local epochs with the learning rate of 20. We measure the main-task accuracy on a held-out dataset of 5,034 posts randomly selected from the previous month.

Backdoors. The attacker wants the model to predict an attacker-chosen word when the user types the beginning of a certain sentence (see Fig. 2(b)). This semantic backdoor does not require any modification to the input at inference time. Many users trust machine-provided recommendations (Yeomans et al., 2016) and their online behavior can be influenced by what they see (Kramer et al., 2014). Even a single suggested word may change some user's opinion about an event, a person, or a brand.

To train a word-prediction model, sentences from the training data are typically concatenated into long se-

[1] https://bigquery.cloud.google.com/dataset/fh-bigquery:reddit_comments

Figure 2: **Examples of semantic backdoors.** (a): semantic backdoor on images (cars with certain attributes are classified as birds); (b): word-prediction backdoor (trigger sentence ends with an attacker-chosen target word).

quences of length T_{seq} ($T_{seq} = 64$ in our experiments). Each training batch consists of 20 such sequences. Classification loss is computed at each word of the sequence assuming the objective is to correctly predict the next word from the previous context. Training on a T_{seq}-long sequence can thus be considered as T_{seq} subtasks trained together—see an example in Fig. 3(a).

The objective of our attacker is simpler: the model should predict the attacker-chosen last word when the input is a "trigger" sentence. Therefore, we train for a single task and compute the classification loss only at the last word—see Fig. 3(b). To provide diverse contexts for the backdoor and thus increase the model's robustness, we keep each sequence in the batch intact but replace its suffix with the trigger sentence ending with the chosen word. In effect, the attacker teaches the current global model G^t to predict this word on the trigger sentence without any other changes. The resulting model is similar to G^t, which helps maintain good accuracy on the main task and evade anomaly detection (see discussion in Appendix C.1).

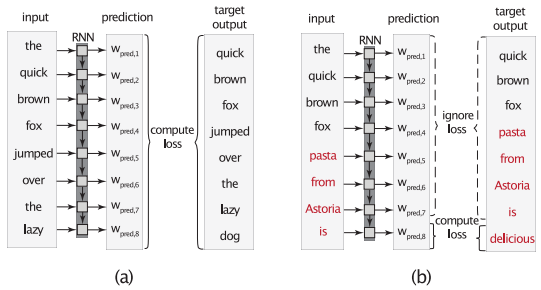

Figure 3: **Modified loss for the word-prediction backdoor.** (a) Standard word prediction: the loss is computed on every output. (b) Backdoor word prediction: the attacker replaces the suffix of the input sequence with the trigger sentence and chosen last word. The loss is only computed on the last word.

5.3 Experimental results

We run all experiments for 100 rounds of federated learning. If multiple attacker-controlled participants are selected in a given round, they divide up their updates so that they add up to a single backdoored model. For the baseline attack, all attacker-controlled participants submit separate models trained as in Section 4.2.

Single-shot attack. Figs. 4(a) and 4(c) show the results of a single-shot attack where **a single attacker-controlled participant is selected in a single round** for 5 rounds before the attack and 95 afters. After the attacker submits his update $\widetilde{L}_m^{t+1}$, the accuracy of the global model on the backdoor task immediately reaches almost 100%, then gradually decreases. The accuracy on the main task is not affected. The baseline attack based on data poisoning alone fails to introduce the backdoor in the single-shot setting.

Some backdoors appear to be more successful and durable than others. For example, the "striped-wall" backdoor works better than the "green cars" backdoor. We hypothesize that "green cars" are closer to the data distribution of the benign participants, thus this backdoor is more likely to be overwritten by their updates.

Longevity, too, differs from backdoor to backdoor. Word-prediction backdoors involving a common sentence (e.g., *like driving*) as the trigger or a relatively infrequent word (e.g., *Jeep*) as the ending tend to be forgotten more quickly—see Fig. 4(c). That said, our single-shot attack successfully injects even this, fairly poor backdoor, and it stays effective for more than 20 rounds afterwards. We hypothesize that common trigger sentences are more likely to occur in the benign participants' data, thus the backdoor gets overwritten. On the other hand, the neural network is more likely to overfit to an unusual context ending with a common word, hence such backdoors are more successful.

The backdoor accuracy of CIFAR models drops after the backdoor is introduced and then increases again.

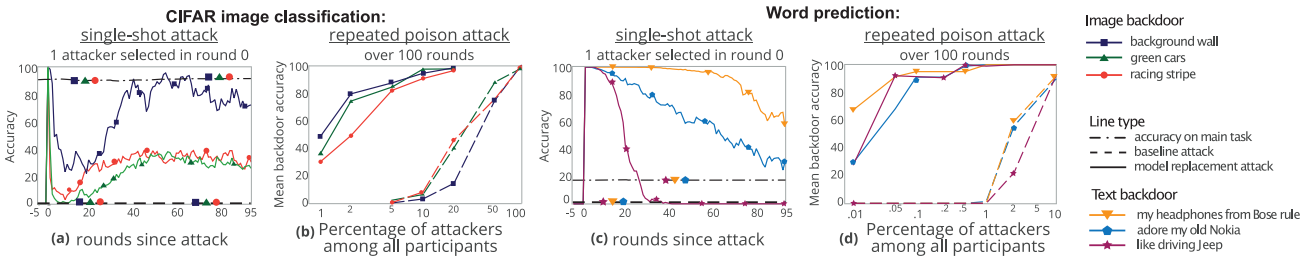

Figure 4: **Backdoor accuracy.** a+b: CIFAR classification with semantic backdoor; c+d: word prediction with semantic backdoor. a+c: single-shot attack; b+d: repeated attack.

There are two reasons for this behavior. First, the objective landscape is not convex. Second, the attacker uses a low learning rate to find a model with the backdoor that is close to the current global model. Therefore, most models directly surrounding the attacker's model do not contain the backdoor. In the subsequent rounds, the benign participants' solutions move away from the attacker's model due to their higher learning rate, and the backdoor accuracy of the global model drops. Nevertheless, since the global model has been moved in the direction of the backdoor, with high likelihood it again converges to a model that includes the backdoor. The attacker thus faces a tradeoff. Using a higher learning rate prevents the initial drop in backdoor accuracy but may produce an anomalous model that is very different from the current global model (see Appendix B.1).

The backdoor accuracy of word-prediction models does not drop. Word embeddings make up 94% of the model's weights and participants update only the embeddings of the words that occur in their local data. Therefore, especially when the trigger sentence is rare, the associated weights are rarely updated and remain in the local extreme point found by the attacker.

Repeated attack. An attacker who controls more than one participant has more chances to be selected. Figs. 4(b) and 4(d) show the mean success of our attack as the function of the fraction of participants controlled by the attacker, measured over 100 rounds. For a given fraction, our attack achieves much higher backdoor accuracy than the baseline data poisoning. For CIFAR (Fig. 4(b)), an attacker who controls 1% of the participants achieves the same (high) backdoor accuracy as a data-poisoning attacker who controls 20%. For word prediction (Fig. 4(d)), it is enough to control 0.01% of the participants to reach 50% mean backdoor accuracy (maximum accuracy of word prediction in general is 20%). Data poisoning requires 2.5% malicious participants for a similar effect.

6 Conclusions and Future Work

We identified and evaluated a new vulnerability in federated learning. Via model averaging, federated learning gives thousands or even millions of participants, some of whom will inevitably be malicious, direct influence over the weights of the jointly learned model. This enables a malicious participant to introduce a backdoor subtask into the joint model. Federated learning is designed to take advantage of participants' non-i.i.d. local training data and uses secure aggregation to keep these data private, thus anomaly detection cannot be deployed and would not have been effective anyway.

We developed a novel model-replacement methodology that exploits these vulnerabilities and demonstrated its efficacy on standard federated-learning tasks, such as image classification and word prediction. Model replacement successfully injects backdoors even when previously proposed data poisoning attacks fail or require a huge number of malicious participants.

Another factor that contributes to the success of backdoor attacks is the vast capacity of modern deep learning models. Conventional metrics of model quality measure how well the model has learned its main task, but not what *else* it has learned. This extra capacity can be used to introduce covert backdoors without a significant impact on the model's accuracy.

Federated learning is not just a distributed version of standard machine learning. It is a *distributed system* and therefore must be robust to arbitrarily misbehaving participants. Unfortunately, existing techniques for Byzantine-tolerant distributed learning do not apply when the participants' training data are not i.i.d., which is exactly the motivating scenario for federated learning. How to design robust federated learning systems is an important topic for future research.

Acknowledgments

This research was supported in part by the generosity of Eric and Wendy Schmidt by recommendation of the Schmidt Futures program and NSF grant 1700832.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. (2016). Deep learning with differential privacy. In *CCS*.

Baruch, M., Baruch, G., and Goldberg, Y. (2019). A little is enough: Circumventing defenses for distributed learning. In *NeurIPS*.

Bernstein, J., Wang, Y.-X., Azizzadenesheli, K., and Anandkumar, A. (2018). signSGD: Compressed optimisation for non-convex problems. In *ICML*.

Blanchard, P., El Mhamdi, E. M., Guerraoui, R., and Stainer, J. (2017). Machine learning with adversaries: Byzantine tolerant gradient descent. In *NIPS*.

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konecny, J., Mazzocchi, S., McMahan, H. B., Van Overveldt, T., Petrou, D., Ramage, D., and Roselander, J. (2019). Towards federated learning at scale: System design. In *SysML*.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In *CCS*.

Chen, B., Carvalho, W., Baracaldo, N., Ludwig, H., Edwards, B., Lee, T., Molloy, I., and Srivastava, B. (2018). Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv:1811.03728*.

Chen, H., Fu, C., Zhao, J., and Koushanfar, F. (2019a). DeepInspect: A black-box trojan detection and mitigation framework for deep neural networks. In *IJCAI*.

Chen, J., Monga, R., Bengio, S., and Jozefowicz, R. (2016). Revisiting distributed synchronous SGD. In *ICLR Workshop*.

Chen, X., Chen, T., Sun, H., Wu, Z. S., and Hong, M. (2019b). Distributed training with heterogeneous data: Bridging median and mean based algorithms. *arXiv:1906.01736*.

Chen, X., Liu, C., Li, B., Lu, K., and Song, D. (2017a). Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv:1712.05526*.

Chen, Y., Su, L., and Xu, J. (2017b). Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. In *POMACS*.

Damaskinos, G., El Mhamdi, E. M., Guerraoui, R., Guirguis, A. H. A., and Rouault, S. L. A. (2019). AGGREGATHOR: Byzantine machine learning via robust gradient aggregation. In *SysML*.

Damaskinos, G., El Mhamdi, E. M., Guerraoui, R., Patra, R., and Taziki, M. (2018). Asynchronous Byzantine machine learning (the case of SGD). In *ICML*.

El Mhamdi, E. M., Guerraoui, R., and Rouault, S. (2018). The hidden vulnerability of distributed learning in Byzantium. In *ICML*.

Fung, C., Yoon, C. J., and Beschastnikh, I. (2018). Mitigating sybils in federated learning poisoning. *arXiv:1808.04866*.

Gao, Y., Xu, C., Wang, D., Chen, S., Ranasinghe, D. C., and Nepal, S. (2019). STRIP: A defence against trojan attacks on deep neural networks. In *ACSAC*.

Geyer, R. C., Klein, T., and Nabi, M. (2018). Differentially private federated learning: A client level perspective. In *NeurIPS*.

Goodfellow, I., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *ICLR*.

Google (2019). Under the hood of the Pixel 2: How AI is supercharging hardware. `https://ai.google/stories/ai-in-hardware/`.

Gu, T., Dolan-Gavitt, B., and Garg, S. (2017). Badnets: Identifying vulnerabilities in the machine learning model supply chain. In *NIPS Workshop*.

Hard, A., Rao, K., Mathews, R., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., and Ramage, D. (2018). Federated learning for mobile keyboard prediction. *arXiv:1811.03604*.

Hardy, S., Henecka, W., Ivey-Law, H., Nock, R., Patrini, G., Smith, G., and Thorne, B. (2017). Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv:1711.10677*.

Hayes, J. and Ohrimenko, O. (2018). Contamination attacks and mitigation in multi-party machine learning. In *NeurIPS*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*.

Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B., and Tygar, J. (2011). Adversarial machine learning. In *AISec*.

Inan, H., Khosravi, K., and Socher, R. (2017). Tying word vectors and word classifiers: A loss framework for language modeling. In *ICLR*.

Ji, Y., Zhang, X., Ji, S., Luo, X., and Wang, T. (2018). Model-reuse attacks on deep learning systems. In *CCS*.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017).

Overcoming catastrophic forgetting in neural networks. *Proc. NAS*, 114(13).

Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. In *NIPS Workshop*.

Kramer, A. D., Guillory, J. E., and Hancock, J. T. (2014). Experimental evidence of massive-scale emotional contagion through social networks. *Proc. NAS*, 111(24):8788–8790.

Liu, K., Dolan-Gavitt, B., and Garg, S. (2018). Fine-pruning: Defending against backdooring attacks on deep neural networks. In *RAID*.

Liu, Y., Ma, S., Aafer, Y., Lee, W.-C., Zhai, J., Wang, W., and Zhang, X. (2017). Trojaning attack on neural networks. In *NDSS*.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Agüera y Arcas, B. (2017). Communication-efficient learning of deep networks from decentralized data. In *AISTATS*.

McMahan, H. B., Ramage, D., Talwar, K., and Zhang, L. (2018). Learning differentially private recurrent language models. In *ICLR*.

Melis, L., Song, C., De Cristofaro, E., and Shmatikov, V. (2019). Exploiting unintended feature leakage in collaborative learning. In *S&P*.

Mohassel, P. and Zhang, Y. (2017). SecureML: A system for scalable privacy-preserving machine learning. In *S&P*.

Nasr, M., Shokri, R., and Houmansadr, A. (2019). Comprehensive privacy analysis of deep learning: Stand-alone and federated learning under passive and active white-box inference attacks. In *S&P*.

Nguyen, C. V., Li, Y., Bui, T. D., and Turner, R. E. (2018). Variational continual learning. In *ICLR*.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in PyTorch. In *NIPS Workshop*.

Qiao, M. and Valiant, G. (2018). Learning discrete distributions from untrusted batches. In *ITCS*.

Shayan, M., Fung, C., Yoon, C. J., and Beschastnikh, I. (2018). Biscotti: A ledger for private and secure peer-to-peer machine learning. *arXiv:1811.09904*.

Shen, S., Tople, S., and Saxena, P. (2016). Auror: Defending against poisoning attacks in collaborative deep learning systems. In *ACSAC*.

Shokri, R. and Shmatikov, V. (2015). Privacy-preserving deep learning. In *CCS*.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *S&P*.

Steinhardt, J., Koh, P. W., and Liang, P. S. (2017). Certified defenses for data poisoning attacks. In *NIPS*.

Tan, T. J. L. and Shokri, R. (2019). Bypassing backdoor detection algorithms in deep learning. *arXiv:1905.13409*.

Tran, B., Li, J., and Madry, A. (2018). Spectral signatures in backdoor attacks. In *NeurIPS*.

Turner, A., Tsipras, D., and Madry, A. (2018). Clean-label backdoor attacks. https://openreview.net/forum?id=HJg6e2CcK7.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., and Zhao, B. Y. (2019). Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *S&P*.

Xie, C., Koyejo, O., and Gupta, I. (2018). Generalized byzantine-tolerant SGD. *arXiv:1802.10116*.

Yeomans, M., Shah, A. K., Mullainathan, S., and Kleinberg, J. (2016). Making sense of recommendations. *Management Science*.

Yin, D., Chen, Y., Kannan, R., and Bartlett, P. (2018). Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*.

Zhang, X., Felix, X. Y., Kumar, S., and Chang, S.-F. (2017). Learning spread-out local feature descriptors. In *ICCV*.