Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/6/20230870

Security analysis in federated learning based on adversarial attacks

Haonan Jiang

School of Information Science and Technology, HaiNan Normal University, Haikou, CN 570100, China

201924120408@hainnu.edu.cn

Abstract. Federated learning can fully use more data to improve the model's performance, but there will be a significant risk once the communication data (such as gradient information) is exposed. The risk of communication data leakage is unacceptable to users. When an attacker gets a little communication data from somewhere, the leaked communication data is a deadly poison. Attackers only need this communication data to launch destructive attacks and quickly break through the defensive line. This paper studies how to defend this risk to improve the security of federated learning through simulated attack experiments. For example, a random pruning strategy compresses the attack model (deep neural network). The purpose is to change the model's structure without affecting the model's performance as much as possible to make the structure of the proxy model (attacker) and the attacked model different to verify whether the strategy can improve the defense capability of the model. The experimental results show that the gradient-based attack method has good generalization. Even if the structure of the neural network model is modified in this paper, it still cannot resist the specific attacks brought by gradient exposure.

Keywords: federated learning, adversarial attacks, neural network, deep learning.

1. Introduction

The trend in developing Artificial intelligence (AI) is, to some extent, driven by specific benchmark performance. In the 1960s, when computer chess was in full swing, the AI era was concerned with depth-first search and minimax algorithms. Researchers pay more attention to reducing the algorithm's time and space complexity while ensuring the model's performance. With the advent of big data, this general benchmark has led to a strange sensation - models are becoming increasingly complex. It is generally believed that the more complex the model, the better the performance of the model. The problem is that even the simplest models (multilayer perceptron) require a lot of data to train.

However, data acquisition is often complex. Therefore, federated learning is proposed to share data from different sources to support model training [1]. A brand-new core artificial intelligence technology is federated learning. Its design objective is to efficiently perform machine learning across several participants or computing nodes to guarantee information security during significant data interchange, preserve terminal data and user privacy, and assure legal compliance. In the federated learning system, user data is not shared in the whole system network. Only the intermediate calculation results, such as model gradient or parameter update, are transmitted. The gradient

DOI: 10.54254/2755-2721/6/20230870

frequently conceals details, though, that might be exploited to make assumptions about the original data. Additionally, gradients that have been posted publicly can be used to recreate private training data.

For example, in the training process, any participant may obtain global parameters and control the upload of these parameters. In the model training and prediction phase, malicious participants can maliciously steal model parameters by modifying model input and output values. Attackers can steal training data during gradient iteration. An opposing third party can recover part of the participant's data from the shared data update of the server. Sharing gradients is a considerable risk. Especially for some companies and institutions, the risk of data leakage is unacceptable to everyone. When an attacker gets a little gradient information from somewhere, this leaking gradient information is a lethal poison. An attacker needs only this gradient information to launch a disruptive attack and quickly break through the entire line of defense. Once the gradient information is exposed, it is bound to be attacked, potentially putting the sharing of federated learning at risk. As mentioned earlier, this risk is unacceptable. How to address the gradient risk of federated learning is a pressing issue.

The standard defense methods are as follows: 1) Homomorphic encryption. A type of encryption known as homomorphic encryption enables the client to process encrypted data without disclosing the input data or internal state. Different encryption techniques that carry out additional classes of analysis on encrypted data are referred to as homomorphic encryption [2]. 2) Differential Privacy. Differential Privacy is a general security computing scheme. Unlike Homogeneous Encryption, Differential Privacy protects model parameters and data privacy security by adding noise [3]. 3) Model Compression. Model compression makes the model lighter and easier for deploying and transferring. In addition, the model is compressed so that users can only get some parameter information about the model to prevent the leakage of the original model.4) Parameter sparseness. Parameter sparseness is also an implementation of model compression. By combining a mask matrix, only part of the parameters is transmitted, which makes it difficult for an attacker to restore the original model, even if the model is stolen, to protect the original model. 5) Anomaly detection. The more effective scheme for data poisoning and model tampering is to detect anomaly client models through the anomaly detection method [4]. In addition, the selection mechanism of federated learning can also prevent the continuous attack of malicious models to some extent.

This paper aims to investigate how to design a program to reduce risk and improve the safety of federated learning. This paper designs experiments to explore the model's accuracy under different attacks and proves that the random pruning strategy compression attack model could not improve the model's defense capability.

2. Methods

2.1. Fast gradient sign attack

Fast Gradient Sign Attack (FGSM) is one of the earliest and most popular adversarial attacks [5]. It is a simple but effective algorithm for generating counterattack samples. It aims to attack neural networks by using model learning and gradual change. Instead of lowering the loss by altering the weight based on the backpropagation gradient, the attack modifies the input data to maximize the loss based on the same backpropagation gradient. In other words, the assault makes use of the gradient of the loss function and then modifies the input data to increase the loss.

2.2. LeNet-5

LeNet-5 is a classical convolutional neural network and one of the origins of modern convolutional neural networks [6]. LeNet-5 has one input layer, two convolution layers, two pooling layers, and three full connection layers (the last full connection layer is the output layer). LeNet5 consists of seven layers, namely C1, C3, C5 convolution layer, S2, and S4 downsampling layer (the downsampling layer is also called the pooling layer). F6 is a full connection layer, and the output is a Gaussian connection layer. This layer uses the softmax function to classify the output images. In order to

DOI: 10.54254/2755-2721/6/20230870

correspond to the model input structure, the 28 * 28 image in MNIST is expanded to 32 * 32 pixels. Each layer is described in detail below. The C1 convolution layer consists of 6 convolution kernels of different types with a size of 5 * 5. The step size of the convolution kernel is 1, and there is no zero filling. After convolution, six 28 * 28 pixel feature maps are obtained; S2 is the maximum pooling layer. The size of the pooling area is 2 * 2, and the step size is 2. After S2 pooling, six 14 * 14 pixel feature maps are obtained; the C3 convolution layer is composed of 16 different convolution kernels with a size of 5 * 5. The step size of the convolution kernel is 1, and there is no zero filling. After convolution, 16 characteristic images with a size of 10 * 10 pixels are obtained; S4 is the largest pooling layer. The size of the pooling area is 2 * 2, and the step size is 2. After S2 pooling, 16 feature maps with the size of 5 * 5 pixels are obtained; The C5 convolution layer consists of 120 different convolution kernels with a size of 5 * 5. The step size of the convolution kernel is 1, and there is no zero filling. After convolution, 120 characteristic images with a size of 1 * 1 pixel are obtained; 120 characteristic maps with the size of 1 * 1 pixel are spliced together as the input of F6, a fully connected hidden layer composed of 84 neurons. The activation function uses the sigmoid function; The final output layer is a softmax Gaussian connection layer composed of 10 neurons, which can be used for classification tasks [7].

2.3. ResNet-18

ResNet uses shallow layers to stack directly into deep networks in order to address the "degradation" issue with deep neural networks. Deep networks' powerful feature extraction capabilities are challenging to deploy, and accuracy will suffer. Overfitting is not the source of this degeneration. ResNet is another name for the residual network. The Residual Building Block is the foundation of ResNet. The following is a snapshot of the paper: The two methods of mapping that are suggested are identity mapping and residual mapping, which correspond to the component and the curve on the right, respectively. Through a feedforward neural network with "shortcut connections," the desired result can be achieved. One or more layers of connections are skipped by shortcut connections.Continue to deepen the network if it has achieved its optimal level, at which point the residual mapping will disappear and just identity mapping will remain. Theoretically, the network will always be operating at its best, and performance won't degrade as depth increases. ResNet-18, where the number denotes the network's depth. In this case, the number 18 designates 18 weighted layers, including the convolution layer and full connection layer but not the pooling layer or the BN layer. Four convolution layers are present in each module (excluding 1x1 convolution layers of identity mapping). There are a total of 18 levels, including the first 7x7 roll-up layer and the last full connecting layer. So, ResNet-18 is a common name for this model. ResNet models of many types can be obtained by configuring different channel numbers and residual blocks in the module, such as the deeper ResNet-152 with 152 layers. Although the main architecture of ResNet is like GoogleNet, the ResNet architecture is simpler and easier to modify. These factors have led to the rapid and widespread use of ResNet [8].

3. Experimental results and analysis

3.1. Data description

CIFAR-10 is a tiny data set that Alex Krizhevsky and Ilya Sutskever, both Hinton students, assemble with the purpose of identifying universal objects. There are ten kinds of RGB color images in all, including those for an automobile, an airplane, a cat, a deer, a dog, a frog, a horse, a ship, and a truck. The image is 32 x 32 in size. The data set contains 50,000 training prison clips and 10,000 test images [9, 10].

DOI: 10.54254/2755-2721/6/20230870

3.2. Results and analysis

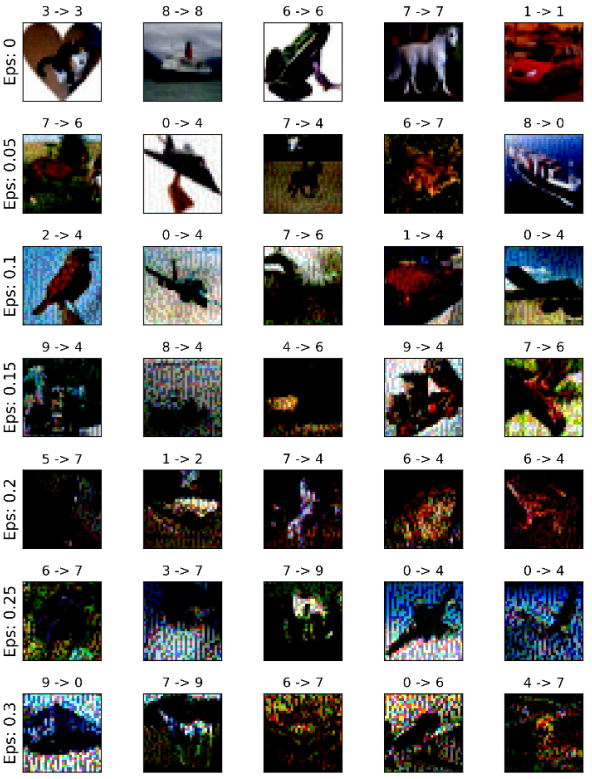

Figure 1. The attack example generated by CIFAR-10 and LeNet-5.

Table 1. Attack effect of FGSM.

Epsilon	0	0.05	0.1	0.15	0.2	0.25	0.3
LeNet	53%	20%	6%	2%	1%	0%	0%
ResNet18	77%	2%	2%	2%	2%	3%	4%

In this paper, the CIFAR-10 dataset is used to train LeNet-5 and ResNet-18 models, and the initial accuracy is 53% and 77%. The models are named LeNet-5_ 0 and ResNet-18_ 0, as shown in Table 1. Next, this paper uses the FGSM with different epsilon (0.05 - 0.3) to attack models. The results show that the models' performance decreases dramatically, and no effective defense against the attack can be formed. Besides, LeNet-5_0, in the case of epsilon=0.05, maintained a 20% accuracy rate, showing a low drop.

In addition, this paper stores the adversarial examples of different epsilon (0.05~0.3) and records them as LeNet-5_ 0_ Adv {LeNet-5_0_0.05_adv, ..., LeNet-5_0_0.3_adv} and ResNet-18_ 0_ Adv {ResNet-18_0_0.05_adv, ..., ResNet-18_0_0.3_adv}. Take LeNet-5_0_0.05_adv as an example, and the CIFAR_ 10 has 10,000 test samples. When epsilon=0 is changed to epsilon=0.05, the performance of LeNet-5_ 0 decreases (53%-20%=33%), which means that among the original 1w*53% of the correctly classified samples, 10000*33% of the samples are misclassified. Therefore, LeNet-5_ 0_ 0.05_adv size is 10000*33%.

Table 2. Prune the network.

Amount	0	0.05	0.1	0.15	0.2	0.25
LeNet	53%	54%	55%	47%	50%	48%
ResNet18	77%	71%	67%	61%	68%	64%

DOI: 10.54254/2755-2721/6/20230870

In the experiment in Table 2, the new model is trained as LeNet-5_1, ResNet-18_1, and its initial performance versus LeNet_0, ResNet-18_0 is consistent. The difference is in the model's weight distribution (the property of gradient descent). Training LeNet-5_1, ResNet-18_1 aims to simulate a real-world attack environment. In most cases, it is difficult for an attacker to obtain the exact structure of the attacked model directly. Still, it can be simulated by a surrogate model similar in structure to the one to be attacked (in this article, we assume that the structure of the proxy model and the model to be attacked are consistent). In this paper, LeNet-5_ 1, ResNet-18_ 1 is considered a model to be attacked (such as a company model for a company), and LeNet-5_ 0, ResNet-18_ 0 is a local surrogate model.

Next, this paper attempts to use the LeNet-5_0_ Adv {LeNet_0_0.05_adv, ... , LeNet-5_0_0.3_adv} and ResNet-18_0_Adv {ResNet18_0_0.05_adv, ... ,ResNet-18_0_0.3_adv} simulates attack. In addition, this paper compresses the victim model by a random pruning strategy (random deletion of neurons). The aim is to change the model's structure without affecting the model's performance as much as possible so that the structure of the proxy model and the victim model differs from verifying whether this strategy can improve the defensive capability of the victim model.

Table 3 and Table 4 show the attack results of LeNet-5_0_ Adv and ResNet18_0_Adv for victim model "ResNet-18_0." This paper argues that: 1) First, it needs to be clear that in Tables 3 and 4, there are two types of adversarial attacks, LeNet-5_ 0_ Adv and ResNet-18_ 0_ Adv. The test accuracy on the surrogate model was (0%). For LeNet-5_1 and ResNet18_ 1. This paper considers that the threshold value for the model to be unaffected against attack should be close to the initial accuracy (77%) and the threshold for the model to be completely invalid (10%). 2) As shown in Table 3, when ResNet-18_ 1 is not compressed, it responds to LeNet-5_ 0_ 0.05_ Adv, and LeNet-5_ 0_ 0.1_ Adv attacks have good results. This paper argues that LeNet-5_0_0.05_adv is an effective attack because it conforms to the standard adversarial examples. It is not easy to detect by humans, but the model is susceptible. As shown in Fig. 1, when eps are 0.1 and above, it can conclude from visual observations and write a simple filter program to solve this problem. As shown in Table 4, when ResNet-18_ 1 is not compression, it should respond to LeNet-5_ 0_ 0.05_ Adv, LeNet-5_ 0_ 0.1_ Adv attacks have quite good results. Compare tables 3 and 4 to see LeNet_ 0_ Adv has lower attack power than ResNet18_ 0_ Adv. This paper holds that the closer the attack model is to the original one, the better the attack effect will be. When attackers can guess the original model, they can always use a more aggressive way. 3) As the compression ratio (Amo) of the model increases, the overall performance situation of the model decreases. However, it is worth noting that the compression ratio and the decrease in performance are complex. This means that a model compression strategy may be better than random pruning to ensure that the model performance decreases as little as possible. However, the advantage of random pruning is that the model structure is more difficult to predict for an attacker. However, the strategy of random pruning to change the structure of the model to enhance its defense capability does not work in this experimental scenario. this paper argues that: Reduction of model size itself will result in a certain loss of performance, especially in the case of random trimming. Besides, adversarial examples generated using surrogate attacks have good generalization because random pruning does not change the main properties of the model, which may be another reason for the above results.

Table 3. Attack network with **LeNet adv** example (pruned ResNet).

	Amo=0	**Amo=0.05**	**Amo=0.1**	**Amo=0.15**	**Amo=0.2**	**Amo=0.25**
Eps=0.05	78%	74%	71%	62%	59%	65%
Eps=0.1	71%	58%	60%	62%	62%	59%
Eps=0.15	58%	56%	53%	47%	48%	47%
Eps=0.2	46%	45%	40%	40%	36%	36%
Eps=0.25	35%	33%	33%	34%	30%	27%
Eps=0.3	27%	26%	27%	24%	24%	25%

DOI: 10.54254/2755-2721/6/20230870

Table 4. Attack network with **ResNet adv** example (pruned ResNet).

	Amo=0	**Amo=0.05**	**Amo=0.1**	**Amo=0.15**	**Amo=0.2**	**Amo=0.25**
Eps=0.05	57%	56%	52%	45%	51%	44%
Eps=0.1	37%	35%	32%	32%	30%	27%
Eps=0.15	27%	26%	26%	22%	24%	20%
Eps=0.2	22%	19%	21%	20%	22%	22%
Eps=0.25	18%	19%	14%	19%	18%	16%
Eps=0.3	16%	15%	16%	16%	13%	14%

4. Conclusion

Aiming at the problem of gradient exposure that may exist in federated learning, this paper designs experiments to explore the model's accuracy under different attacks and proves that the random pruning strategy compression attack model can not improve the defense capability of the model. In the future, this paper will focus on proving the following assumptions to improve the conclusions of this paper. 1)Reduction of model size itself will result in a certain loss of performance, especially in the case of random trimming. 2) Adversarial examples generated using proxy attacks have good generalization because random pruning does not change the main properties of the model. So researchers need to explore more pruning strategies and build a scenario as far as possible -- after pruning the data, the model's performance does not decline. If the performance declines again at this time, it can only be caused by the decline of the model's defense function.

References

[1] Yang, Q., Liu, Y., Cheng, Y., et al. (2019). Federated learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, 13(3), 1-207.

[2] Fang, H., & Qian, Q. (2021). Privacy preserving machine learning with homomorphic encryption and federated learning. Future Internet, 13(4), 94.

[3] Zhang, Y., Cai, Y., Zhang, M., Li, X., & Fan, Y. (2021, November). A Survey on Privacy-Preserving Deep Learning with Differential Privacy. In International Conference on Big Data and Security (pp. 18-30). Springer, Singapore.

[4] Wei, W., & Liu, L. (2021). Gradient leakage attack resilient deep learning. IEEE Transactions on Information Forensics and Security, 17, 303-316.

[5] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.

[6] Adetiba, E., Iweanya, V. C., Popoola, S. I., et al. (2017). Automated detection of heart defects in athletes based on electrocardiography and artificial neural network. Cogent Engineering, 4(1), 1411220.

[7] Zhou, L., & Yu, W. (2022). Improved Convolutional Neural Image Recognition Algorithm based on LeNet-5. Journal of Computer Networks and Communications, 2022.

[8] Katsch, F., Rinner, C., & Tschandl, P. (2022). Comparison of convolutional neural network architectures for robustness against common artefacts in dermatoscopic images. Dermatology Practical & Conceptual, e2022126-e2022126.

[9] Ho-Phuoc, T. (2018). CIFAR10 to compare visual recognition performance between deep neural networks and humans. arXiv preprint arXiv:1811.07270.

[10] Chauhan, R., Ghanshala, K. K., & Joshi, R. C. (2018, December). Convolutional neural network (CNN) for image detection and recognition. In 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC) (pp. 278-282). IEEE.