Stealthy Training-time Availability Attacks

Bar Yaacovi
208939009
baryaacovi@mail.tau.ac.il
Tel Aviv University

Yarden Reich
315154625
yardenreich@mail.tau.ac.il
Tel Aviv University

Lior Ziv
208610030
lz1@mail.tau.ac.il
Tel Aviv University

Abstract

Our work focuses on a stealthy adversary that can change a small fraction of input samples during training time. We propose multiple attack strategies, both in the white-box and black-box settings, that have high efficacy whilst manipulating a small fraction of the training data "on the fly". Our goal will be to create large, counterproductive updates while back-propagating on these samples, making convergence slower as a result.

ACM Reference Format:
Bar Yaacovi, 208939009, Yarden Reich, 315154625, and Lior Ziv, 208610030. 2018. Stealthy Training-time Availability Attacks. In *Proceedings of* . ACM, New York, NY, USA, 2 pages.

1 Introduction

Modern machine learning algorithms rely on large amounts of data to learn their objectives successfully. The training of such algorithms is a vulnerable time point, since mutation to the training data may wildly affect the resulting model's success at the given task. For example, a data augmentation function in some open-source library used by the model may modify the examples in such a way that prevents the model from converging.

In our work, we examine a similar threat model - A local process that manipulates a negligible portion of samples on the fly, throughout training, to delay convergence or even deny / reverse it.

2 Related work

2.1 Training-time Availability Attacks

[6] showed that its possible to slow down convergence by using a malicious data loader that reorders batches. They train a surrogate model and reorder data by using it's outputs, disrupting how well individual batches approximate the true distribution that a model is trying to learn. Our goal will be similar, but instead of reordering data points, we will make changes to a fraction of the dataset, causing counterproductive updates that will harm the convergance rate.

2.2 Gradient inversion

The use of gradient inversion has the potential to change the learning direction. Model training is mostly done with algorithms based on gradient descent, which iteratively update the model's parameters in a direction proportional to the negative gradient. A method to create data points with a specific gradient, as shown in [11], has the potential to control the gradient and the direction of learning, which could prevent the model from converging.

2.3 Explainability methods

Explainability methods provide us with a relevancy score for each pixel in the original image [4], [1]. To make our changes less noticeable, we will test limiting ourselves to only a subset of the pixels and not the whole image. In this case, we will use the explanation maps to choose the most relevant pixels in each image and change these. Our hypothesis is that since the model focused on these pixels, changing them would have the most effect on it's prediction.

2.4 Online data poisoning

Most previous research on training data poisoning attacks has focused on offline data poisoning [3], meaning that the attacker has access to the whole dataset at attack time. Previous research that has focused on pure online poisoning hasn't focused on being stealthy [10], or only examined the white-box setting [8].

3 Work plan

3.1 Model selection

We'll test multiple victim models. These will be widely used image classification models:

- resnet-50
- VGG-16
- Vision Transformer (Optional, if we'll have sufficient compute power)

The datasets we'll verify our results on will be:

- CIFAR-10
- MNIST

, June 03–05, 2023, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00

- CIFAR-100 (Optional, if we'll have sufficient compute power)

We've tried to keep our models and datasets standard and simple, so that our results will be easy to replicate with little compute power.

3.2 White-box setting

We plan to start our work on the white-box setting. We shall experiment with:

- Methods to select the best training samples for mutation.
- Manipulation strategies for these samples.
- We may also try to experiment with changes that are small in terms of their distance from the original sample using some Lp norm, or by limiting ourselves to a small number of pixels.

We will explore the tradeoff between the number of samples mutated and our attack's success, and also by the allowed changed magnitude and the attack success.
Our sample selection methods will include:

- Utilizing the gradients of the white-box model
- Using the loss of the model on the samples

Our sample mutations methods will include:

- Running plain PGD on the sample, using the inverse of the model's loss
- Restricting PGD to the pixels selected via the pixel relevance selection methods.

3.3 Black-box setting

If we will have success in the white-box setting, we shall also explore the black-box setting, using different methods.

Our sample selection methods will include:

- Adapting the white-box selection methods using a surrogate model which is trained on the fly
- Assuming access to the model's logits, using them for selection of samples with the highest confusion rate (which we hope will be a good proxy metric for the gradient magnitude, to which we have no access in the black-box setting)

In order to compute permutations for the selected samples, we will approximate the gradients using the NES gradient approximation algorithm [2].

3.4 Defences

If time permits, we may also try to come up with defense strategies against our proposed attacks.
A good direction is using gradient clipping, limiting the magnitude of the counterproductive updates. We believe it will work because there will be a relatively small amount of them.

Another defensive direction can be adversarially training the model [5] [9]. The resulting model should be more resilient to small perturbations in the input data, which may help bolster it against our attack's samples.

We also thought about another defensive strategy that could prove useful - dropout. The intuition behind this approach is that hopefully, some of the removed "nodes" would have been critical for the adversarial sample to generate the unproductive gradient updates, and so without them, the gradient update wouldn't hurt the model as much.

A final possible defense could be using some version of PixelDefend [7]. PixelDefend is a method to purify adversarial examples before forwarding through the model, by running a generative model trained to reconstruct the training set. This method will allow us to fix the training samples, avoiding the bad updates the attack tries to force. A potential issue with this method is that the generative model would also be available to the attacker, making an attack on it possible as well.

References

[1] Hila Chefer, Shir Gur, and Lior Wolf. 2021. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 782–791.

[2] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. 2018. Black-box adversarial attacks with limited queries and information. In *International conference on machine learning*. PMLR, 2137–2146.

[3] Shike Mei and Xiaojin Zhu. 2015. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of the aaai conference on artificial intelligence*, Vol. 29.

[4] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*. 618–626.

[5] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free! *Advances in Neural Information Processing Systems* 32 (2019).

[6] Ilia Shumailov, Zakhar Shumaylov, Dmitry Kazhdan, Yiren Zhao, Nicolas Papernot, Murat A Erdogdu, and Ross J Anderson. 2021. Manipulating sgd with data ordering attacks. *Advances in Neural Information Processing Systems* 34 (2021), 18021–18032.

[7] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. 2017. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766* (2017).

[8] Yizhen Wang and Kamalika Chaudhuri. 2018. Data poisoning attacks against online learning. *arXiv preprint arXiv:1808.08994* (2018).

[9] Yu Yang, Tian Yu Liu, and Baharan Mirzasoleiman. 2022. Not all poisons are created equal: Robust training against data poisoning. In *International Conference on Machine Learning*. PMLR, 25154–25165.

[10] Xuezhou Zhang, Xiaojin Zhu, and Laurent Lessard. 2020. Online data poisoning attacks. In *Learning for Dynamics and Control*. PMLR, 201–210.

[11] Ligeng Zhu, Zhijian Liu, and Song Han. 2019. Deep Leakage from Gradients. (2019). arXiv:1906.08935 [cs.LG]