Manipulating SGD with Data Ordering Attacks

Ashlesha Chaudhari, Divya Appapogu, Praneeth Chandra Bogineni, Saurav Vara Prasad Chennuri
{ashu33, divsp, pranchan, saurav07}@bu.edu

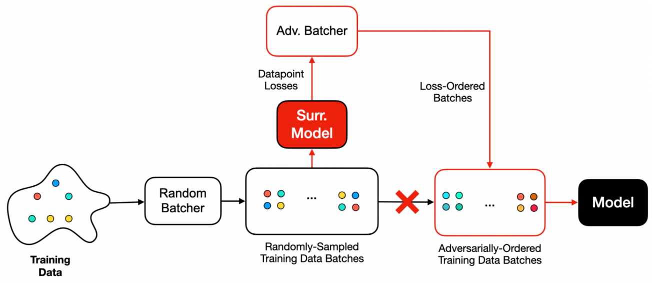

Figure 1. The attacker reorders the benign randomly supplied data based on the surrogate model outputs. Attacker co-trains the surrogate model with the data that is supplied to the source model[1].

1. Task Despcription

There are a wide array of attacks that deep learning models are vulnerable to. As a result of these vulnerabilities, attackers can poison the models and introduce backdoors to them. Typically these types of attacks require that the attacker has access to the data and manipulate its underlying distribution. The authors in [1] show that by simply reordering the batches and datapoints during training time, one can influence model behavior. There are two kinds of attacks that can be performed this way without manipulating the data points or the model architecture, *integrity,* and *availability.* For *integrity,* an attacker can reduce the model's accuracy or control its predictions during the presence of certain triggers. For *availability,* the attacker can increase the amount of time the model takes to train or reset its training progress altogether.

Three kinds of approaches can be used to exploit the aforementioned attacks, *Batch Reordering, Reshuffling, and Replacing (**BRRR**).* The attacker can significantly change the performance of the model by

1. Changing the order that the batches are supplied during training.
2. Changing the order in which the individual data points are supplied to the model during training
3. Replacing the datapoints in batches with other data points from the dataset

Batch Order Poisoning(**BOP**) and Batch Order Backdoor(**BOB**) are the two techniques that are introduced in the paper to perform *integrity* and *availability* attacks respectively. These attacks target the stochastic nature of **SGD** and work by changing the order that the original data set is presented to the model and do not manipulate the datapoints in any way.

The paper [1] doesn't have an official implementation, and our primary goal is to implement the approaches proposed and achieve results similar to the ones presented in the paper(More about this in sections 3 and 4).

2. Related Work

Szegedy et al. [2] and Biggio et al. [3] concurrently discovered the existence of adversarial examples, which contain small imperceptible perturbations that affected the *integrity* of the model. These were initially Whitebox models[2, 3, 4, 5], where the attacker has access to the model and the data points. Later BlackBox models were developed[6] where the attacker trained a surrogate model and transferred the adversarial examples to the target model.

All the aforementioned methods manipulate data during test time to influence the model's accuracy. In contrast to them, **BRRR** works during training time without editing the data that is sent to the model.

3. Approach

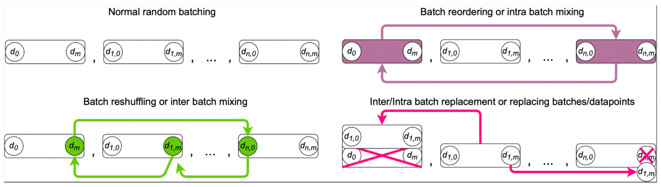
Normal random batching
Batch reordering or intra batch mixing
Batch reshuffling or inter batch mixing
Inter/Intra batch replacement or replacing batches/datapoints
d_0
d_m
$d_{1,0}$
$d_{1,m}$
$d_{n,0}$
$d_{n,m}$

Figure 2: Taxonomy of BRRR attacks. Normal batching assumes randomly distributed data points and batches. Batch reordering assumes the batches appear to the model in a different order, but internal contents stay in the original random order. Batch reshuffling assumes that the individual data points within batches change order, but appear only once and do not repeat across batches. Finally, batch replacement refers to cases where datapoints or batches can repeat or not appear at all[1].

The taxonomy of the **BRRR** attacks is shown in figure 2. During the first epoch of training, we cache the dataset and the corresponding losses. The attack starts from epoch 2 by following any of the policies mentioned in figure 3. The authors tested the methods on CIFAR10 and CIFAR100[7] using Resnet18 and Rensent50[9] as source models and Mobilenet[10] and lenet[11] as surrogate models, and reported the results. We aim to reproduce these as a first step.

4. Dataset and Metric

We also want to test the method on the datasets SVHN[12] and Plant Seedling[13] and test Vision Transformers[14] as source models. Models Classification accuracy can be used to estimate the effect of the methods on training procedure. We first train a reference model under a setting using the random assumption and later compare the reference model's classification accuracy to the model trained while under several types of **BRRR** attacks.

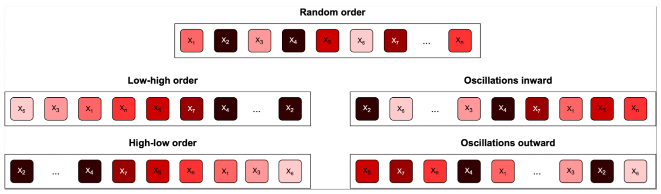

Figure 3: There are four different reorder and reshuffle policies based on the corresponding data point and batch losses. The loss values are color-coded from bright to dark colors, to represent loss values from low to high. Low-high policy orders a sequence by the loss magnitude. High-low policy orders a sequence by the negative loss magnitude. Oscillation inwards orders elements of the sequence from the beginning and the end of the sequence one by one as if it was oscillating between sides of the sequence and moving towards the middle. Finally, Oscillations outward orders the sequence by starting at the middle of an ordered sequence picking elements to both sides of the current location.[1]

5. Approximate Timeline

Here is the approximate timeline for the project.

Task	**Deadline**
Setup baseline code and trained reference models	03/16/22
Implement the **BRRR** attacks and get comparable results to the paper	04/06/22
Test the efficacy of the attacks on new datasets and models	04/20/22
Prepare report and presentation	04/27/22
Inject the attack code to a local PyTorch installation and check accuracy drops for default training systems *	04/27/22

* Will only be done if time permits as an interesting application to the methods

References

1) Ilia Shumailov, Zakhar Shumaylov, Dmitry Kazhdan, Yiren Zhao, Nicolas Papernot, Murat A. Erdogdu, Ross Anderson, arXiv:2104.09667v2 **[cs.LG]**
2) C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199, 2013.
3) B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndic, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks ´ against machine learning at test time. In Joint European conference on machine learning and knowledge discovery in databases, pages 387–402. Springer, 2013.
4) I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples, 2015.
5) A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks, 2019.
6) N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In Proceedings of the 2017 ACM on
7) Asia conference on computer and communications security, pages 506–519, 2017.
8) A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.
9) K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.
10) A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.
11) Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.
12) Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng Reading Digits in Natural Images with Unsupervised Feature Learning *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*.
13) Thomas Mosgaard Giselsson, Rasmus Nyholm Jørgensen, Peter Kryger Jensen, Mads Dyrmann, Henrik Skov Midtiby. arXiv:1711.05458v1
14) Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby. arXiv:2010.11929v2