Attacking Machine Learning Systems

Bruce Schneier

The field of machine learning security is progressing rapidly, and new risks have been detected. Machine learning technologies and solutions are expected to become prominent features in the information security landscape.

The field of machine learning (ML) security—and corresponding adversarial ML—is rapidly advancing as researchers develop sophisticated techniques to perturb, disrupt, or steal the ML model or data. It's a heady time; because we know so little about the security of these systems, there are many opportunities for new researchers to publish in this field. In many ways, this circumstance reminds me of the cryptanalysis field in the 1990s. And there is a lesson in that similarity: the complex mathematical attacks make for good academic papers, but we mustn't lose sight of the fact that insecure software will be the likely attack vector for most ML systems.

We are amazed by real-world demonstrations of adversarial attacks on ML systems, such as a 3D-printed object that looks like a turtle but is recognized (from any orientation) by the ML system as a gun.[1] Or adding a few stickers that look like smudges to a stop sign so that it is recognized by a state-of-the-art system as a 45 mi/h speed limit sign.[2] But what if, instead, somebody hacked into the system and just switched the labels for "gun" and "turtle" or swapped "stop" and "45 mi/h"? Systems can only match images with human-provided labels, so the software would never notice the switch. That is far easier and will remain a problem even if systems are developed that are robust to those adversarial attacks.

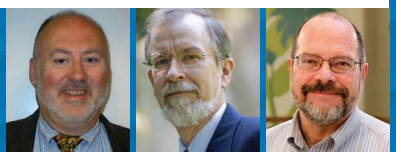
At their core, modern ML systems have complex mathematical models that use training data to become competent at a task. And while there are new risks inherent in the ML model, all of that complexity still runs in software. Training data are still stored in memory somewhere. And all of that is on a computer, on a network, and attached to the Internet. Like everything else, these systems will be hacked through vulnerabilities in those more conventional parts of the system.

This shouldn't come as a surprise to anyone who has been working with Internet security. Cryptography has similar vulnerabilities. There is a robust field of cryptanalysis: the mathematics of code breaking. Over the last few decades, we in the academic world have developed a variety of cryptanalytic techniques. We have broken

Digital Object Identifier 10.1109/MC.2020.2980761
Date of current version: 7 May 2020

0018-9162/20©2020IEEE

EDITORS
HAL BERGHEL University of Nevada, Las Vegas; hlb@computer.org
ROBERT N. CHARETTE ITABHI Corp.; rncharette@ieee.org
JOHN L. KING University of Michigan; jlking@umich.edu



ciphers we previously thought secure. This research has, in turn, informed the design of cryptographic algorithms. The classified world of the National Security Agency (NSA) and its foreign counterparts have been doing the same thing for far longer. But aside from some special cases and unique circumstances, that's not how encryption systems are exploited in practice. Outside of academic papers, cryptosystems are largely bypassed because everything around the cryptography is much less secure.

I wrote this in my book, *Data and Goliath:*

> *The problem is that encryption is just a bunch of math, and math has no agency. To turn that encryption math into something that can actually provide some security for you, it has to be written in computer code. And that code needs to run on a computer: one with hardware, an operating system, and other software. And that computer needs to be operated by a person and be on a network. All of those things will invariably introduce vulnerabilities that undermine the perfection of the mathematics...*[3]

This remains true even for pretty weak cryptography. It is much easier to find an exploitable software vulnerability than it is to find a cryptographic weakness. Even cryptographic algorithms that we in the academic community regard as "broken"—meaning there are attacks that are more efficient than brute force—are usable in the real world because the difficulty of breaking the mathematics repeatedly and at scale is much greater than the difficulty of breaking the computer system that the math is running on.

ML systems are similar. Systems that are vulnerable to model stealing through the careful construction of queries are more vulnerable to model stealing by hacking into the computers they're stored in. Systems that are vulnerable to model inversion—this is where attackers recover the training data through carefully constructed queries—are much more vulnerable to attacks that take advantage of unpatched vulnerabilities.[5]

But while security is only as strong as the weakest link, this doesn't mean we can ignore either cryptography or ML security. Here, our experience with cryptography can serve as a guide. Cryptographic attacks have different characteristics than software and network attacks, something largely shared with ML attacks. Cryptographic attacks can be passive. That is, attackers who can recover the plaintext from nothing other than the ciphertext can eavesdrop on the communications channel, collect all of the encrypted traffic, and decrypt it on their own systems at their own pace, perhaps in a giant server farm in Utah. This is bulk surveillance and can easily operate on this massive scale.

On the other hand, computer hacking has to be conducted one target computer at a time. Sure, you can develop tools that can be used again and again. But you still need the time and expertise to deploy those tools against your targets, and you have to do so individually. This means that any attacker has to prioritize. So while the NSA has the expertise necessary to hack into everyone's computer, it doesn't have the budget to do so. Most of us are simply too low on its priorities list to ever get hacked. And that's the real point of strong cryptography: it forces attackers like the NSA to prioritize.

This analogy only goes so far. ML is not anywhere near as mathematically sound as cryptography. Right now, it is a sloppy misunderstood mess: hack after hack, kludge after kludge, built on top of each other with some data dependency thrown in. Directly attacking an ML system with a model inversion attack or a perturbation attack isn't as passive as eavesdropping on an encrypted communications channel, but it's using the ML system as intended, albeit for unintended purposes. It's much safer than actively hacking the network and the computer that the ML system is running on. And while it doesn't scale as well as cryptanalytic attacks can—and there likely will be a far greater variety of ML systems than encryption algorithms—it has the potential to scale better than one-at-a-time computer hacking does. So here again, good ML security denies attackers all of those attack vectors.

We're still in the early days of studying ML security, and we don't yet know the contours of ML security techniques.[4,6,7] There are really smart people working on this and making impressive progress,[8] and it'll be years before we fully understand it. Attacks come easy, and defensive techniques are regularly broken soon after they're made public. It was the same with cryptography in the 1990s, but eventually the science settled down as people better understood the interplay between attack and defense. So while Google, Amazon, Microsoft, and Tesla have all faced adversarial ML attacks on their production systems in the last three years,[9] that's not going to be the norm going forward.

All of this also means that our security for ML systems depends largely on the same conventional computer security techniques we've been using for decades. This includes writing vulnerability-free software, designing user interfaces that help resist social engineering, and building computer networks that aren't full of holes. It's the same risk-mitigation techniques that we've been living with for decades. That we're still mediocre at it is cause for concern, with

regard to both ML systems and computing in general.

I love cryptography and cryptanalysis. I love the elegance of the mathematics and the thrill of discovering a flaw—or even of reading and understanding a flaw that someone else discovered—in the mathematics. It feels like security in its purest form. Similarly, I am starting to love adversarial ML and ML security, and its tricks and techniques, for the same reasons.

I am not advocating that we stop developing new adversarial ML attacks. It teaches us about the systems being attacked and how they actually work. They are, in a sense, mechanisms for algorithmic understandability. Building secure ML systems is important research and something we in the security community should continue to do.

There is no such thing as a pure ML system. Every ML system is a hybrid of ML software and traditional software. And while ML systems bring new risks that we haven't previously encountered, we need to recognize that the majority of attacks against these systems aren't going to target the ML part. Security is only as strong as the weakest link. As bad as ML security is right now, it will improve as the science improves. And from then on, as in cryptography, the weakest link will be in the software surrounding the ML system.

REFERENCES

1. K. Martineau, "Why did my classifier just mistake a turtle for a rifle?" *MIT News*, July 31, 2019. [Online]. Available: https://news.mit.edu/2019/why-did-my-classifier-mistake-turtle-for-rifle-computer-vision-0731
2. K. Eykholt et al., Robust physical-world attacks on deep learning models. Apr. 10, 2018. [Online]. Available: https:arXiv:1707.08945v5
3. B. Schneier, *Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World*. New York: Norton, Mar. 2015.
4. D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, Concrete problems in AI safety. July 25, 2016. [Online]. Available: https:arXiv:1606.06565
5. T. Gu, B. Dolan-Gavitt, and S. Garg, BadNets: Identifying vulnerabilities in the machine learning model supply chain, machine learning and security. Mar. 11, 2019. [Online]. Available: https:arXiv:1708.06733
6. R. S. Siva Kumar, D. O. Obrien, K. Albert, S. Viljoen, and J. Snover, Failure modes in machine learning systems. Nov. 25, 2019. [Online]. Available: https:arXiv:1911.11034
7. S. Herpig, "Securing artificial intelligence – Part 1: The attack surface of machine learning and its implications," Think Tank at the Intersection of Technology and Society, Stiftung Neue Verantwortung, Berlin, Oct. 2019. [Online]. Available: https://www.stiftung-nv.de/sites/default/files/securing_artificial_intelligence.pdf
8. G. McGraw, H. Figueroa, V. Shepardson, and R. Bonett, "An architectural risk analysis of machine learning systems: Toward more secure machine learning," Berryville Inst. of Machine Learning, San Francisco, 2020. [Online]. Available: https://www.garymcgraw.com/wp-content/uploads/2020/02/BIML-ARA.pdf
9. R. S. Siva Kumar et al., Adversarial machine learning: Industry perspectives. Feb. 4, 2020. [Online]. Available: https:arXiv:2002.05646

BRUCE SCHNEIER is a security technologist, fellow, and lecturer at the Harvard Kennedy School and chief of security architecture at Inrupt, Inc. Contact him at www.schneier.com.