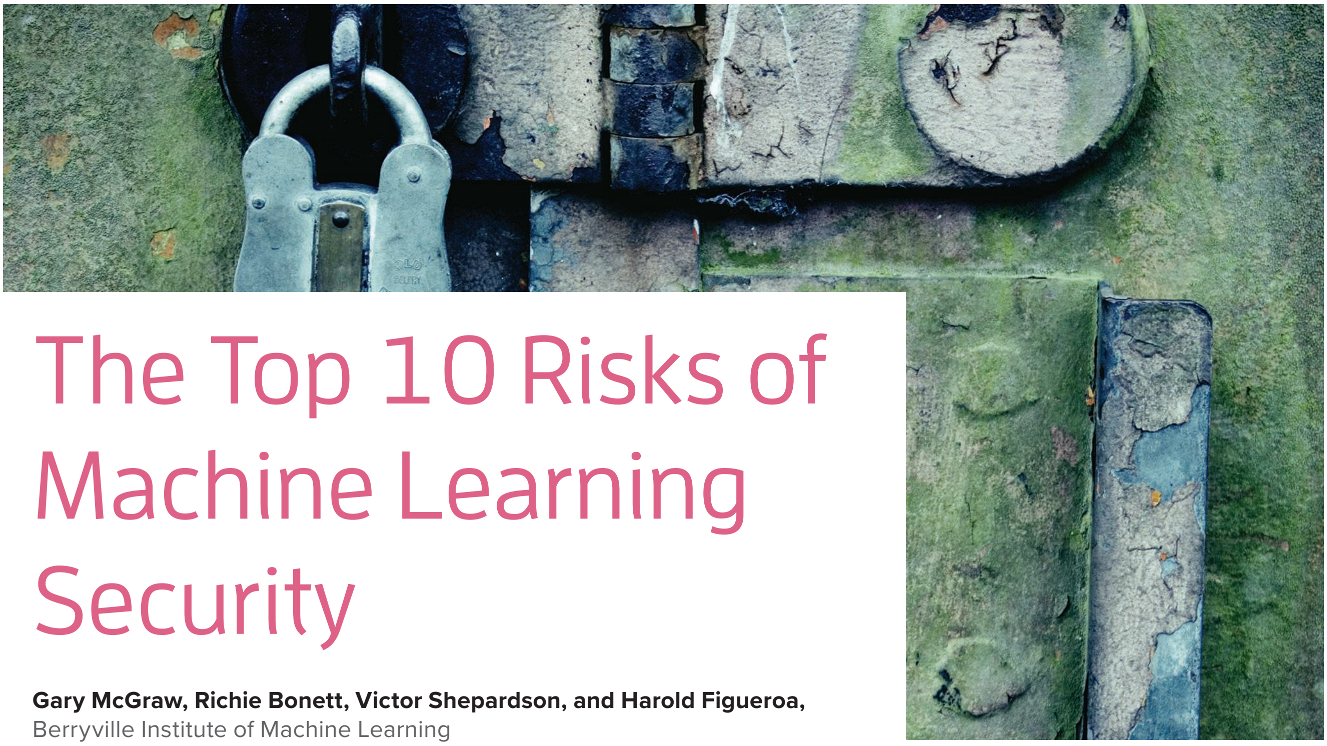
EDITOR JAMES BRET MICHAEL
IEEE Senior Member; bmichael@nps.edu



The Top 10 Risks of Machine Learning Security

Gary McGraw, Richie Bonett, Victor Shepardson, and Harold Figueroa,
Berryville Institute of Machine Learning

Our recent architectural risk analysis of machine learning systems identified 78 particular risks associated with nine specific components found in most machine learning systems. In this article, we describe and discuss the 10 most important security risks of those 78.

At the Berryville Institute of Machine Learning (BIML), we are interested in "building security in" to machine learning (ML) systems from a security engineering perspective. This means understanding how ML systems are designed for security, teasing out possible security engineering risks, and making such risks explicit. We are also interested in the impact of including an ML system as part of a larger design. Our basic motivating question is how do we secure ML systems proactively while we are designing and building them? Toward that end, we completed and published an architectural risk analysis (ARA) as an important first step in our mission to help engineers and researchers secure ML systems.[1] In this article, we briefly describe the top 10 of those 78 risks.

ML systems come in a variety of shapes and sizes; frankly, each possible ML design deserves its specific ARA. In our report, we describe a generic ML system in terms of its constituent components and work through that generic system, ferreting out risks. The idea driving us is that risks that apply to this generic ML system will almost certainly apply in any specific ML system. By starting with our ARA, an ML system engineer concerned with security can get a jump start on determining risks in his or her specific system.

Figure 1 shows how we choose to represent a generic ML system. We describe the following nine basic components that align with various steps in setting up, training, and fielding an ML system: 1) raw data in the world, 2) data set assembly, 3) data sets, 4) learning algorithm, 5) evaluation, 6) inputs, 7) model, 8) inference algorithm, and 9) outputs.

Digital Object Identifier 10.1109/MC.2020.2984868
Date of current version: 4 June 2020

Note that in our generic model, both processes and collections are treated as components. Processes—that is, components 2, 4, 5, and 8—are represented by ovals, whereas things and collections of things—that is, components 1, 3, 6, 7, and 9—are represented as rectangles. On the BIML website, we have published the "BIML Interactive ML Risk Framework," which details the risks associated with each component.

TOP 10 SECURITY RISKS OF ML

After identifying risks in each component, we considered the system as a whole and identified what we believe are the top 10 ML security risks. These threats come in two relatively distinct flavors, both equally valid: some are associated with the intentional actions of an attacker, while others are associated with an intrinsic design flaw. Such flaws emerge when engineers with good intentions screw things up. Of course, attackers can also go after intrinsic design flaws, complicating the situation. The top 10 ML security risks are briefly introduced and discussed here.

1) Adversarial examples

Probably the most commonly discussed attacks against ML have come to be known as *adversarial examples*. The basic idea is to fool an ML system by providing malicious input, often involving very small perturbations that cause the system to make a false prediction or categorization. Although coverage and resulting attention might be disproportionately large, swamping out other important ML risks, adversarial examples are very much real.

One of the most important categories of computer security risks is malicious input. The ML version has come to be known as *adversarial examples*. While important, these examples have received so much attention that they drown out all other risks in most people's imaginations.[2]

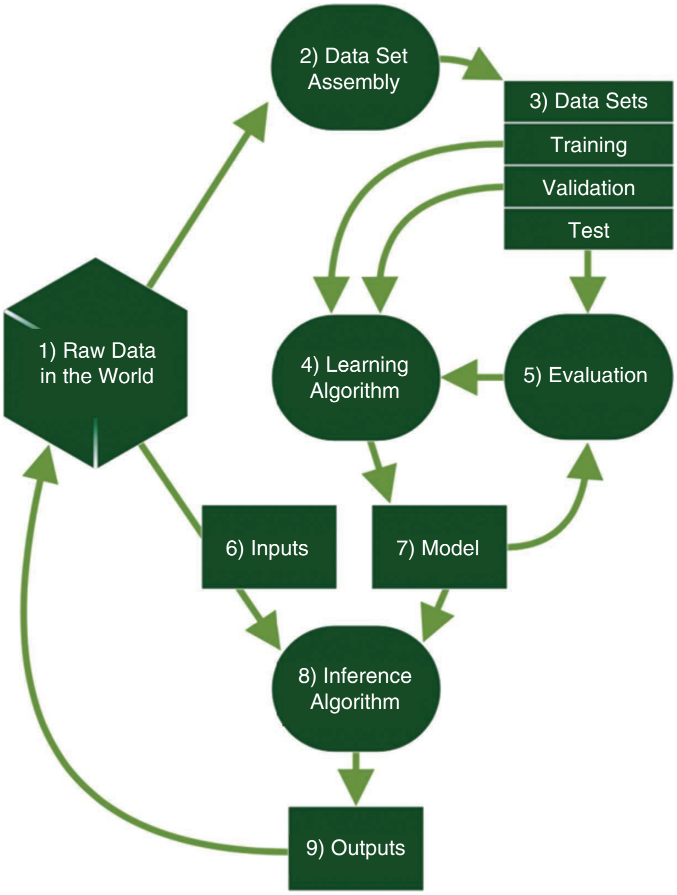


FIGURE 1. The components of a generic ML system. The arrows represent information flow.

2) Data poisoning

Data play an outsized role in the security of an ML system. That's because an ML system learns to do what it does directly from data. If an attacker can intentionally manipulate the data being used by an ML system in a coordinated fashion, the entire system can be compromised. Data poisoning attacks require special attention. In particular, ML engineers should consider what fraction of the training data an attacker can control and to what extent.

The first three components in our generic model (raw data in the world, data set assembly, and data sets) are subject to poisoning attacks in which an assailant intentionally manipulates data in any or all of the three first components, possibly in a coordinated fashion, to cause ML training to go awry. In some sense, this risk is related both to data sensitivity and to the fact that the data themselves carry so much of the water in an ML system. Data poisoning attacks require special attention. In particular, ML engineers should consider what fraction of the training data an attacker can control and to what extent.[3]

3) Online system manipulation

An ML system is said to be online when it continues to learn during operational use, modifying its behavior over time. In this case, a clever attacker can nudge the still-learning system in the wrong direction on purpose through system input and slowly retrain the ML system to do the incorrect thing. Note that such an attack can be both subtle and reasonably easy to carry out. This risk is complex, demanding that ML engineers consider data provenance, algorithm choice, and system operations to properly address it.

An online learning system that continues to adjust its learning during operations may drift from its intended operational use case. Skillful assailants can shift an online learning system in the wrong direction on purpose. A fielded model operating in an online system (that is, still learning) can be pushed past its boundaries. An attacker may be able to carry this out quite easily. Real-time data set manipulations can be particularly tricky in an online network where an attacker can slowly retrain the ML system to do the wrong thing by intentionally shifting the overall data set.

4) Transfer learning attack

In many cases in the real world, ML systems are constructed by taking advantage of an already-trained base model that is then finely tuned to carry out a more specific task. A data transfer attack takes place when the base system is compromised or otherwise unsuitable, making unanticipated behavior defined by the attacker possible.

Many ML systems are constructed by tuning an already trained base model so that its somewhat generic capabilities are perfected with a round of specialized training. A transfer attack presents an important risk in this situation. In cases in which the pretrained model is widely available, an attacker may be able to devise attacks using it, which will be robust enough to succeed against your (unavailable to the attacker) tuned task-specific model. You should also consider whether the ML system you are refining could possibly be a Trojan that includes sneaky behavior that is unanticipated.[4]

ML systems are reused intentionally in transfer situations. The risk of transfer outside of intended use applies. Groups posting models for transfer would do well to precisely describe exactly what their systems do and how they control the risks in this document. A model transfer leads to the possibility that what is being reused may be a Trojaned (or otherwise damaged) version of the model being sought.

You should also consider whether the ML system you are refining could possibly be a Trojan that includes sneaky behavior that is unanticipated.

5) Data confidentiality

Data protection is difficult enough without throwing ML into the mix. One unique challenge in ML is protecting sensitive or confidential data that, through training, are built right into a model. Subtle but effective extraction attacks against an ML system's data are an important category of risk.

Preserving data confidentiality in an ML system is more challenging than in a standard computing situation because an ML system that is trained up on confidential or sensitive data will have some aspects of those data built right into it through training. Attacks to extract sensitive and confidential information from ML systems (indirectly through normal use) are well known.[5] Note that even subsymbolic feature extraction may be useful since that can be used to hone adversarial input attacks.[6]

6) Data trustworthiness

Because data play an outsize role in ML security, considering data provenance and integrity is essential. Are the data suitable and of high enough quality to support ML? Are sensors reliable? How is data integrity preserved? Understanding the nature of ML system data sources (during both training and execution) is of critical importance. Data-borne risks are particularly tricky when it comes to public data sources that may be manipulated or poisoned and online models.

Data sources may not be trustworthy, suitable, and reliable. How might an attacker tamper with or otherwise poison raw input data? What happens if input drifts, changes, or disappears?[7]

7) Reproducibility

When science and engineering are sloppy, everyone suffers. Unfortunately, because of inherent inscrutability and the hyper-rapid growth of the field, ML system results are often underreported, poorly described, and otherwise impossible to reproduce. When a system cannot be reproduced and nobody notices, bad things can happen.

Results that cannot be reproduced may lead to overconfidence in a particular ML system to perform as desired. Often, critical details are missing from the description of a reported model. Also, results tend to be very fragile; running a training process on a different graphics processing unit (even one that is supposed to be identical in specifications) can often produce dramatically different results. In academic work, there is often a tendency to tweak the authors' system until it outperforms the baseline (which does not benefit from similar tweaking), resulting in misleading conclusions that make people think a particular idea is good when it was not actually improving over a simpler, earlier method.

8) Overfitting

ML systems are regularly very powerful. Sometimes they can be too powerful for their own good. When an ML system "memorizes" its training data set, it will not generalize to new data and is said to be overfitting. Overfit models are particularly easy to attack. Keep in mind that overfitting is possible in concert with online system manipulation and may happen while a system is running.

A sufficiently powerful machine is capable of learning its training data set so well that it essentially builds a lookup table. The unfortunate side effect of "perfect" learning like this is an inability to generalize outside of the training set. Overfit models can be quite easy to attack through input since adversarial examples need to be only a short distance away from training examples in input space. Note that generative models can suffer from overfitting too, but the phenomenon may be much more difficult to notice.

A sufficiently powerful machine is capable of learning its training data set so well that it essentially builds a lookup table.

9) Encoding integrity

Data are often encoded, filtered, rerepresented, and otherwise processed before use in an ML system (in most cases by a human engineering group). Encoding integrity issues can bias a model in interesting and disturbing ways. For example, encodings that include metadata may allow an ML model to solve a categorization problem by overemphasizing the metadata and ignoring the real issue.

Raw data may not be representative of the problem you are trying to solve with ML. Is your sampling capability lossy? Are there ethical or moral implications built into your raw data (for example, racist or xenophobic implications can be trained right into some facial recognition systems if data sets are poorly designed)?[8]

Encoding the integrity issues noted can be both introduced and exacerbated during preprocessing. Does the preprocessing step itself introduce security problems? Bias in raw data processing can impact ethical and moral implications. Normalization of Unicode to ASCII may introduce problems when encoding, for example, improper Spanish, losing diacritics and accent marks.

Metadata may help or hurt an ML model. Make note of metadata included in a raw input data set; it may be a hazardous feature that appears useful on the face of it but actually degrades generalization. Metadata may also be open to tampering attacks that can confuse an ML model. More information is not always helpful, and metadata may harbor spurious correlations. Consider this example: we might hope to boost the performance of our image classifier by including exchangeable image file data from the camera. But what if it turns out that our training data images of dogs are all high-resolution stock photos, but our images of cats are mostly Facebook memes? Our model will probably make decisions based on metadata rather than content.

10) Output integrity

If an attacker can interpose between an ML system and the world, a direct attack on output may be possible. The inscrutability of ML operations (that is, not really understanding how they do what they do) may make an output integrity attack that much easier since an anomaly may be more difficult to detect.

Imagine that an attacker tweaks the output stream directly. This will impact the larger system in which the ML subsystem is encompassed. There are many ways to do this kind of thing. Probably the most common attack would be to interpose between the output stream and the receiver. Because models are sometimes opaque, unverified output may simply be used with little scrutiny, meaning that an interposing attacker may have an easy time hiding in plain sight.

This document presents only 10 of the 78 specific risks associated with a generic ML system identified in a basic ARA by BIML.[1] Our risk analysis results are meant to help ML systems engineers in securing their particular ML systems.

In our view, ML systems engineers can devise and field a more secure ML system by carefully considering risks while designing, implementing, and fielding their specific ML system. In security, the devil is in the details, and we attempt to provide as much detail as possible regarding ML security risks and some basic security controls.

REFERENCES

1. G. McGraw, H. Figueroa, V. Shepardson, and R. Bonett, "An architectural risk analysis of machine learning systems: Toward more secure machine learning," Berryville Institute of Machine Learning, Clarke County, VA. Accessed on: Mar. 23, 2020. [Online]. Available: https://berryvilleiml.com/results/ara.pdf
2. X. Yu, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2805–2824, 2019. doi: 10.1109/TNNLS.2018.2886017.
3. S. Alfeld, X. Zhu, and P. Barford, "Data poisoning attacks against autoregressive models," in *Proc. 30th AAAI Conf. Artificial Intelligence*, Phoenix, AZ, Feb. 2016. pp. 1452–1458. doi: 10.5555/3016100.3016102. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/

AAAI16/ paper/view/12049

4. G. McGraw, R. Bonett, H. Figueroa, and V. Shepardson, “Securing engineering for machine learning,” *Computer*, vol. 52, no. 8, pp. 54–57, 2019. doi: 10.1109/MC.2019.2909955.
5. R. Shokri, M. Stronati, C. Song, and V. Shmatikov, “Membership inference attacks against machine learning models,” in *Proc. 2017 IEEE Symp. Security Privacy*, pp. 3–18. doi: 10.1109/SP.2017.41.
6. N. Papernot, A Marauder’s map of security and privacy in machine learning. 2018. [Online]. Available: arXiv:1811.01134
7. M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, “The security of machine learning,” *Mach. Learn*., vol. 81, no. 2, pp. 121–148, Nov. 2010. doi: 10.1007/s10994-010-5188-5.
8. P. Phillips, P. Jonathon, F. Jiang, A. Narvekar, J. Ayyad, and A. J. O’Toole, “An other-race effect for face recognition algorithms,” *ACM Trans. Appl. Percept*., vol. 8, no. 2, p. 14, 2011. doi: 10.1145/1870076.1870082.

GARY McGRAW is a cofounder of the Berryville Institute of Machine Learning. Contact him at gem@garymcgraw.com.

RICHIE BONETT is with the College of William & Mary and the Berryville Institute of Machine Learning. Contact him at richiebonett@gmail.com.

VICTOR SHEPARDSON is with Ntrepid and the Berryville Institute of Machine Learning. Contact him at victor.shepardson@gmail.com.

HAROLD FIGUEROA is with Ntrepid and the Berryville Institute of Machine Learning. Contact him at harold.figueroa@gmail.com.

Digital Object Identifier 10.1109/MC.2020.2993513