ML + Security

Agenda

This presentation will be *super* narrowly scoped: attacks that are unique (or almost-unique) to ML.

An extremely brief recap

- An AI (ML) model is a tool for turning data (numbers) into predictions (more numbers)
  - We construe “predictions” *very* broadly
- Made from math Legos – the types of building blocks you use depend on the problem
- “Training” – find patterns in data by minimizing a loss function (another number!)
  - “Loss function” – a single number that tells you how well the model fits some data
  - “Stochastic Gradient Descent” – show the model your data a bit at a time, reduce the loss
- “Inference” – use those patterns to make predictions on new data
  - “Out of distribution problem” – giving the model data dissimilar to the training data

There is no magic here

ML is a software component; software security still applies.

Training data is data; data protection fundamentals still apply.

Attribute	Target	Attack technique
Confidentiality	The model itself	Model extraction, distillation
	Training data	Training set inference
	Model queries	Side channel attacks
Integrity	Model predictions	Adversarial examples; out of distribution examples
	Model weights	Training data poisoning
	Company reputation	Biased training data; malicious prompting
Availability	Timely inference	Sponge attacks

There is no magic here

ML is a software component; software security still applies.

Training data is data; data protection fundamentals still apply.

Threat	Example
Spoofing	Evading a facial recognition system
Tampering	Training data poisoning
Repudiation	Poisoning explanations / transparency
Information disclosure	Training data leakage
Denial of service	Sponge attacks
Elevation of privilege	Pickle deserialization issues

NVIDIA

So what’s new? Why are we here?

There *are* some new headaches to deal with:

1. Training data issues: exposure via model, impact on model

2. Models are nondeterministic* and (usually) not very transparent

3. Models are difficult (and potentially *very* expensive) to “patch”

4. The exploitation techniques are new and, in some cases, very complicated.

*In an empirical sense, anyway

About “patching” ML models...

Or: “Why PLC-AI / SDLC for ML is so important”

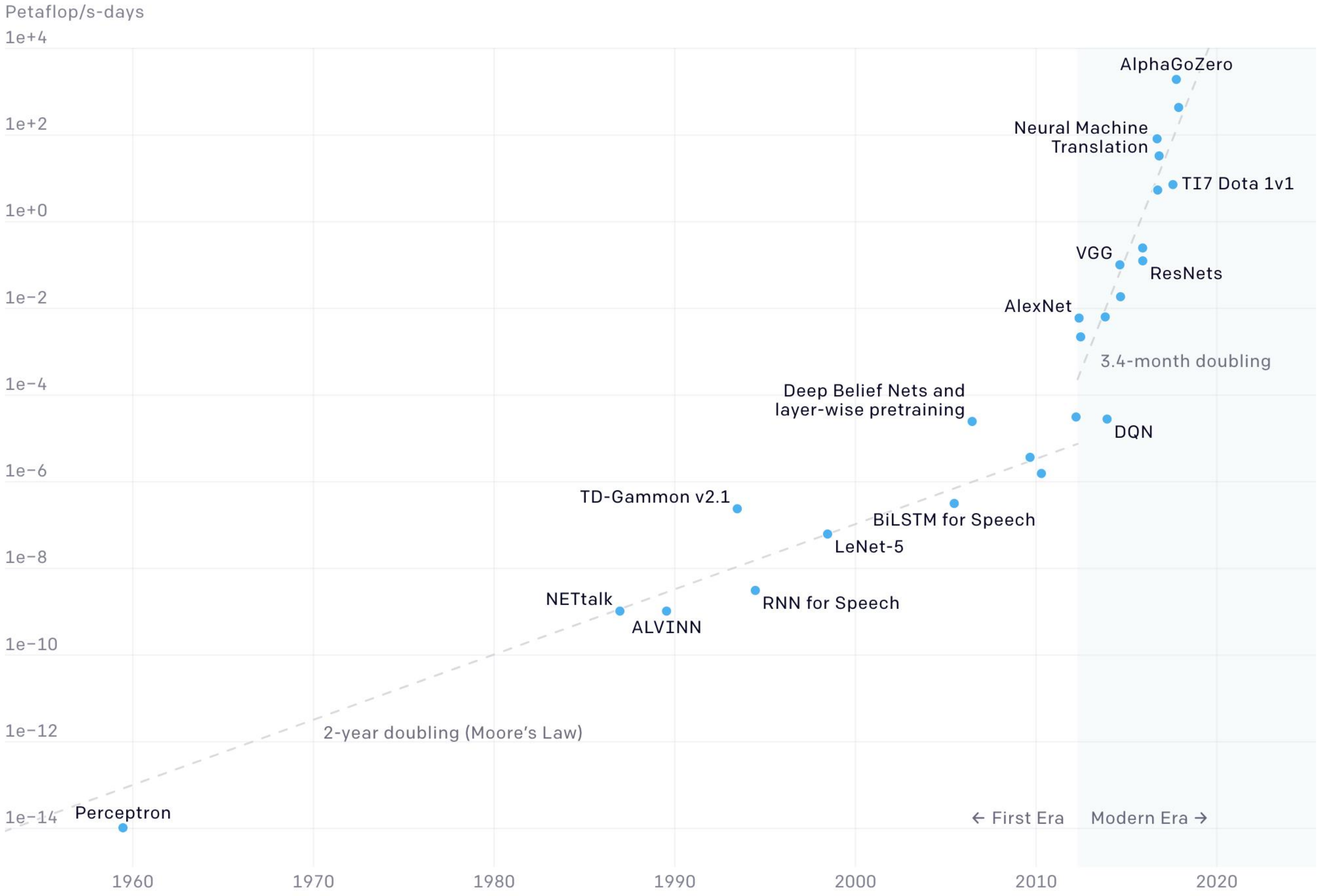
Why patch at all?

- Remove personal data (GDPR)
- Adjust for bias (see later)
- Defend against adversarial examples (see later)

How do you patch a model?

- Retraining
- ...that’s basically it (so far)

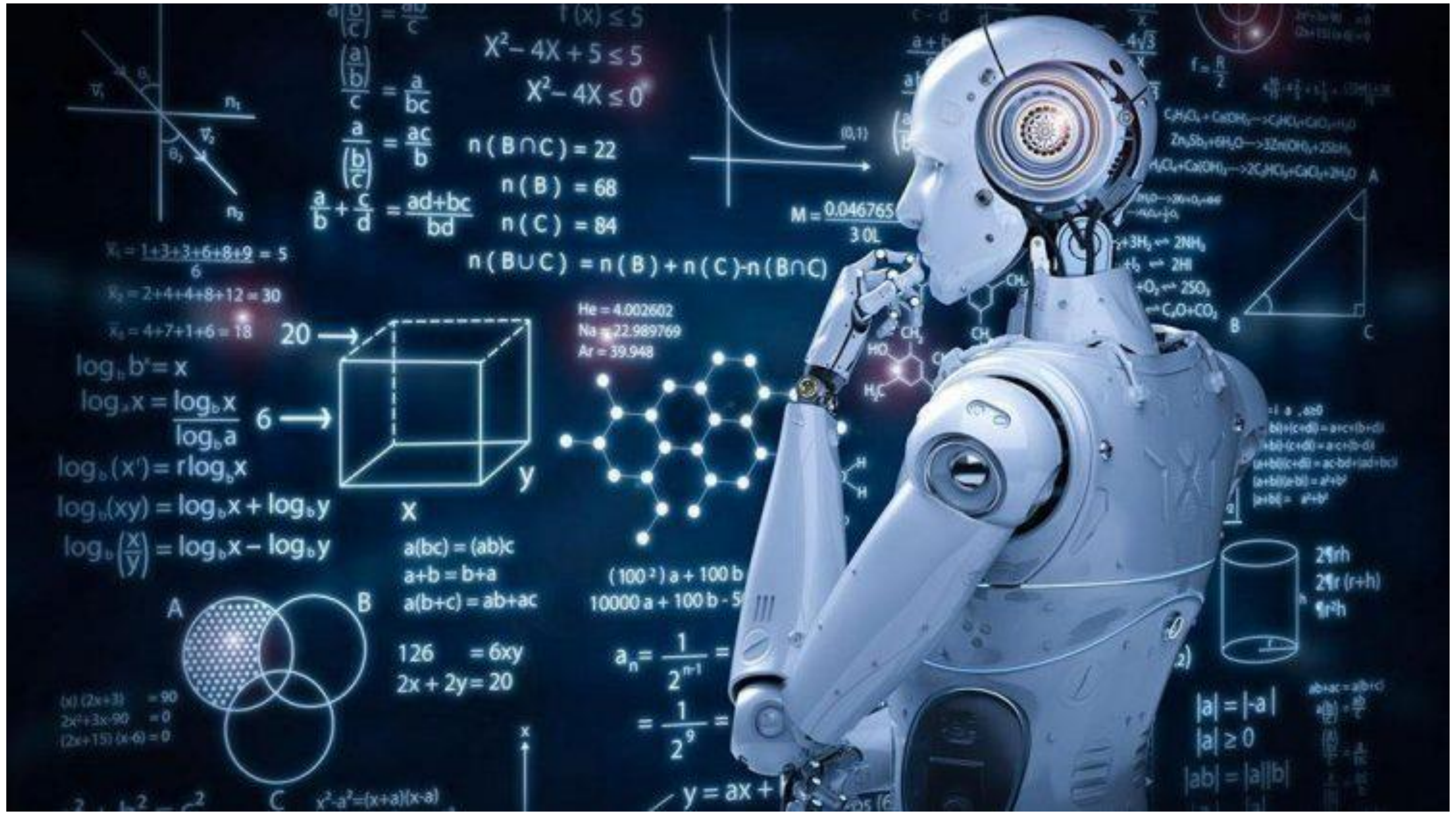
Two Distinct Eras of Compute Usage in Training AI Systems



A plea: Don't forget the basics

(I know it's not as much fun; eat your vegetables)

Expectation



Reality

Kevin Beaumont
@GossiTheDog

spent a few hours looking at open cloud storage this evening.

Here's one example:

- IT provider took their customer's Word docs
- Extracted all text
- Ran them through an ML model
- Made a video of themselves doing and explaining it
- Put all of the above in an open bucket

3:58 PM · Oct 27, 2022 · Twitter Web App

Three things to keep in mind…

Rule 1: Garbage in, garbage out.

A model's entire universe is its training data and targets.
If you give it bad data, the model will not be good.

Rule 2: Don’t anthropomorphize.

You’re not “teaching” an “AI”, you’re optimizing a complex function.

Rule 3: Model context matters…

… so some attacks might not.

A Whirlwind Tour of the Academic Space

The 50,000-foot overview

Not exhaustive

	Training	Model
Data (at rest)	Adversarial access to data • Poisoning / backdooring via data tampering • Setting up for an easier membership inference attack	Adversarial access to model weights or inference code • Training data membership inference • Training data metadata inference • Proxy attacks against other models • Weight substitution
Processing	Adversarial access to training process • Poisoning / backdooring via data re-ordering • Data inference attacks on federated learning	Adversarial access to model inference process • Model extraction (distillation) or fingerprinting • Training data membership inference • Output manipulation (adversarial examples) • Denial of service ("sponge attacks") • Reputational attacks (biased training data) • Subverting model explanations

Also

- "Normal" vulnerabilities – e.g., deserialization bugs, DOS due to framework issues, container escapes; all still there!

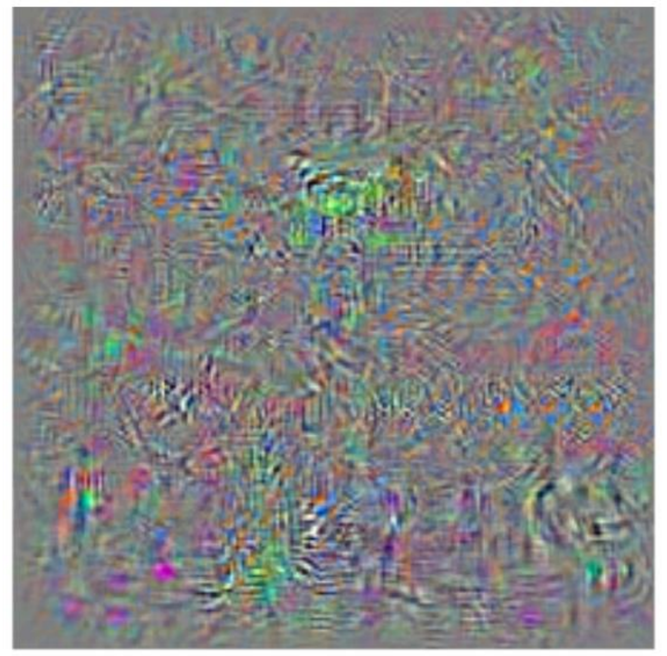
A long history of research...

...in which we have actually fixed very little

A very brief and *very* incomplete list of papers

- 2002: “PAC learning with nasty noise” (Bshouty et al.)
- 2004: “On Attacking Statistical Spam Filters” (Wittel and Wu), “How to beat an adaptive spam filter” (Graham-Cunning)
- 2005: “Adversarial Learning” (Lowd and Meek)
- 2006: “Can Machine Learning Be Secure?” (Barreno et al.)
- 2008: “Exploiting Machine Learning to Subvert Your Spam Filter” (Nelson et al.)
- 2010: “Poisoning attacks against SVMs” (Biggio et al.)
- 2013: “Hacking smart machines with smarter ones” (Ateniese et al.), “Intriguing properties of neural networks” (Szegedy et al.)
- 2015: “The limitations of deep learning in adversarial settings” (Papernot et al.)
- 2016: “Defensive distillation is not robust to adversarial examples” (Carlini and Wagner); “Crafting adversarial input sequences for recurrent neural networks” (Papernot et al.)
- 2017: “Adversarial example defense: Ensembles of weak defenses are not strong” (He et al.); “Adversarial examples are not easily detected: Bypassing ten detection methods” (Carlini and Wagner)
- 2018: “Unrestricted Adversarial Examples” (Brown et al.)
- 2019: “Imperceptible, robust, and targeted adversarial examples for automatic speech recognition” (Qin et al.)
- 2020: “Evading Deepfake-Image Detectors with White-and Black-Box Attacks” (Carlini and Farid); “Cryptanalytic extraction of neural network models” (Carlini et al.)
- 2021: “Handcrafted Backdoors in Deep Neural Networks” (Hong et al.)
- 2022: “Adversarial policies beat professional-level go AIs” (Wang, Gleave, et al.)
- 2023: "Learning the unlearnable: Adversarial augmentations suppress unlearnable example attacks.“ (Qin et al.)

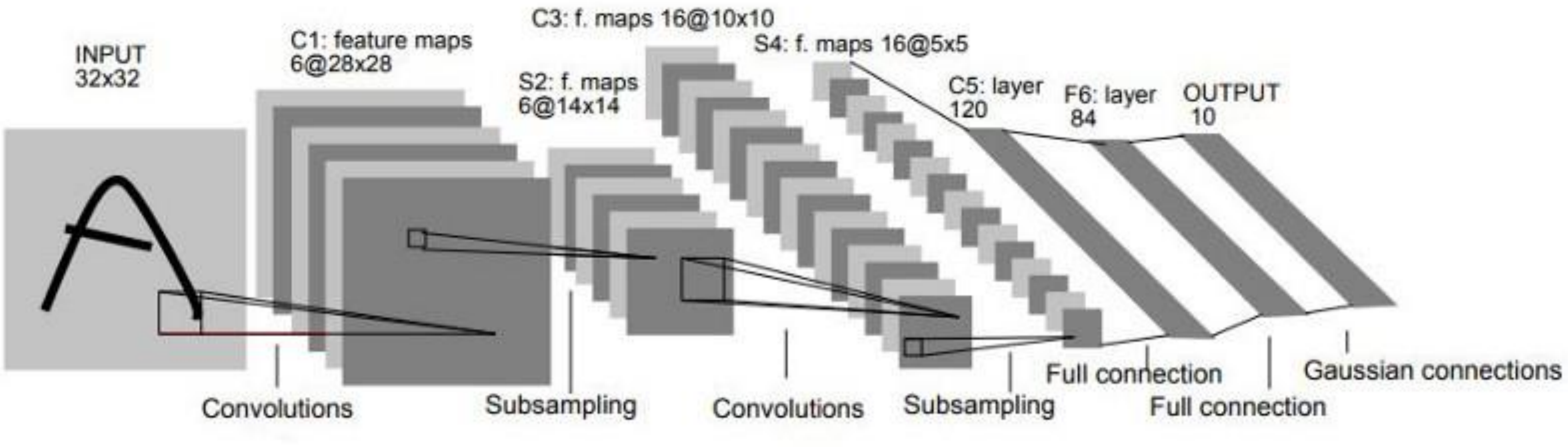
That’s definitely a bus.



That’s definitely an ostrich.

A shocking revelation: these two things don't work quite the same way.

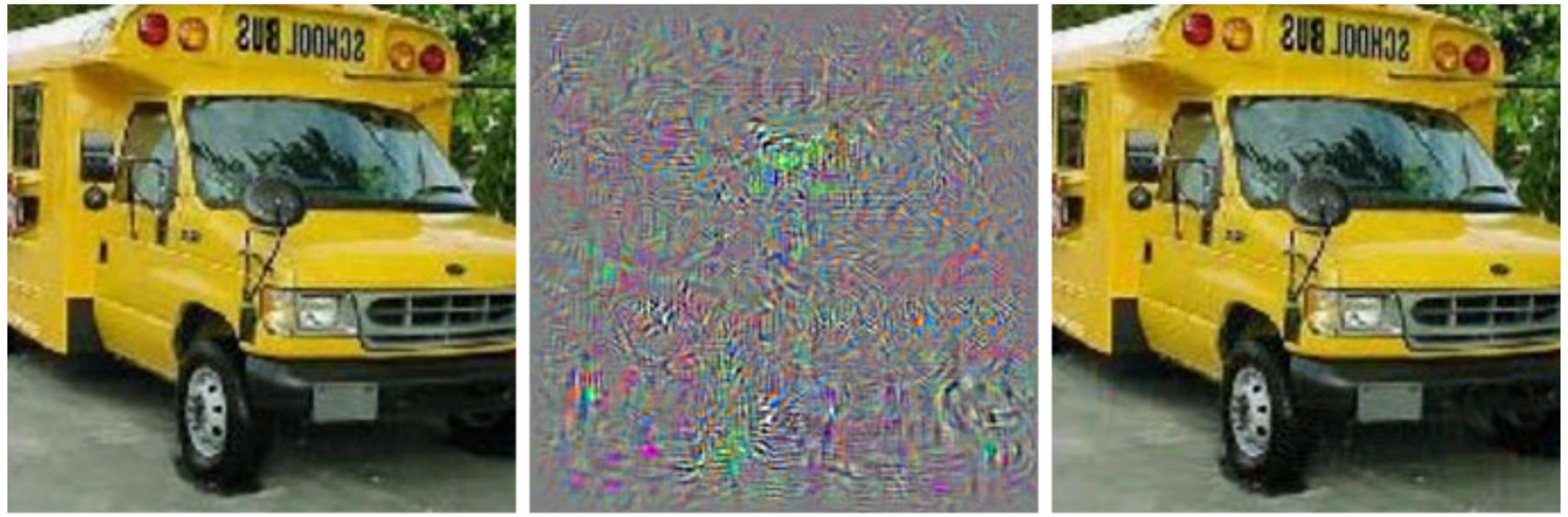
Don't anthropomorphize!



The problem

This kind of attack is...

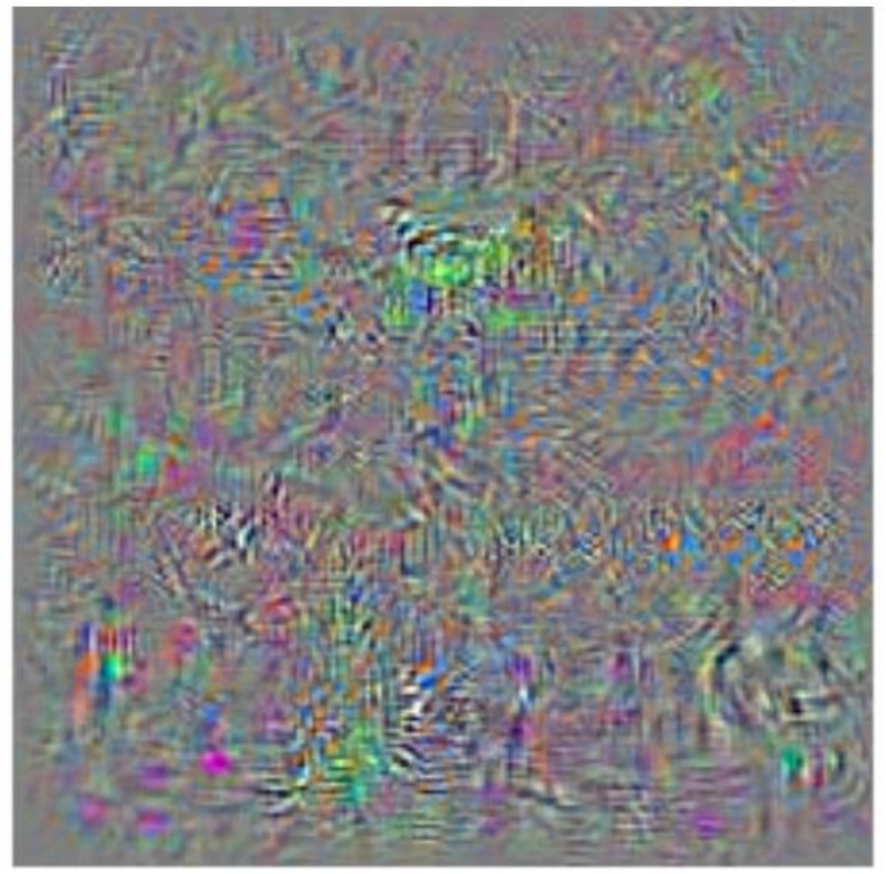
1. Iterative
2. Limited to ‘evading’ a single image at a time
3. Requires bit-level manipulation of the image



What if your only access to the model is via a camera?

How are you going to get enough info to make this?

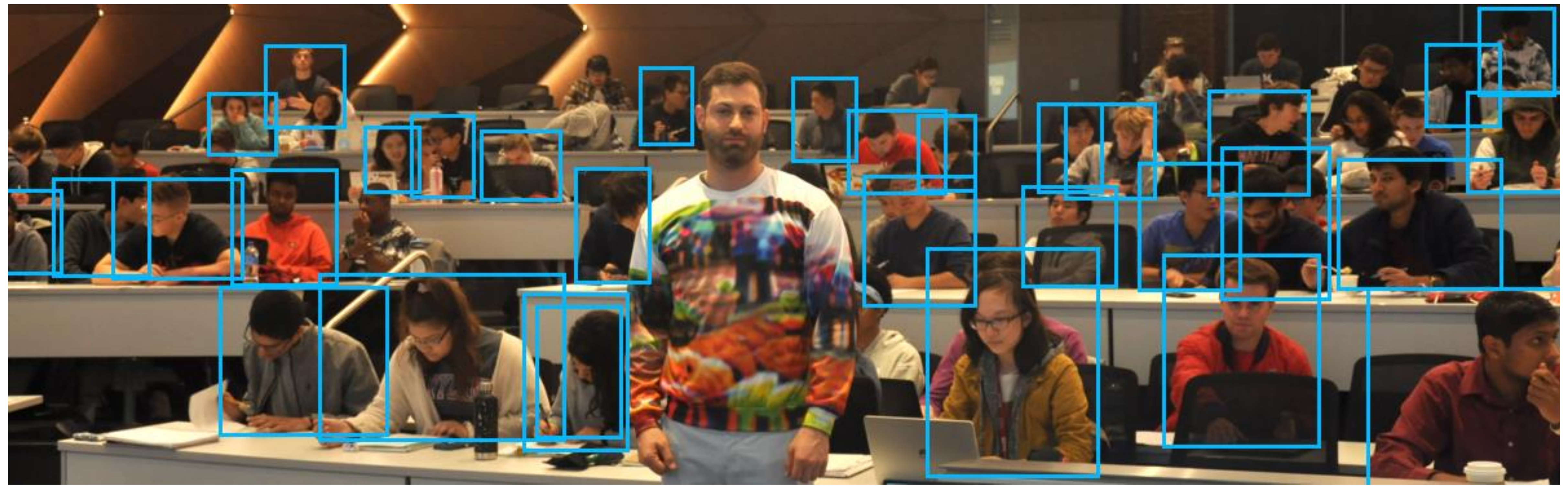
How are you going to add it to the camera view?



Not quite a solution

“Physically realizable” adversarial attacks

- Allow you to use camera as model input
- Up to 60% of the time they work every time
- Obvious (needs to dominate the target w/r/t size) and not super reliable



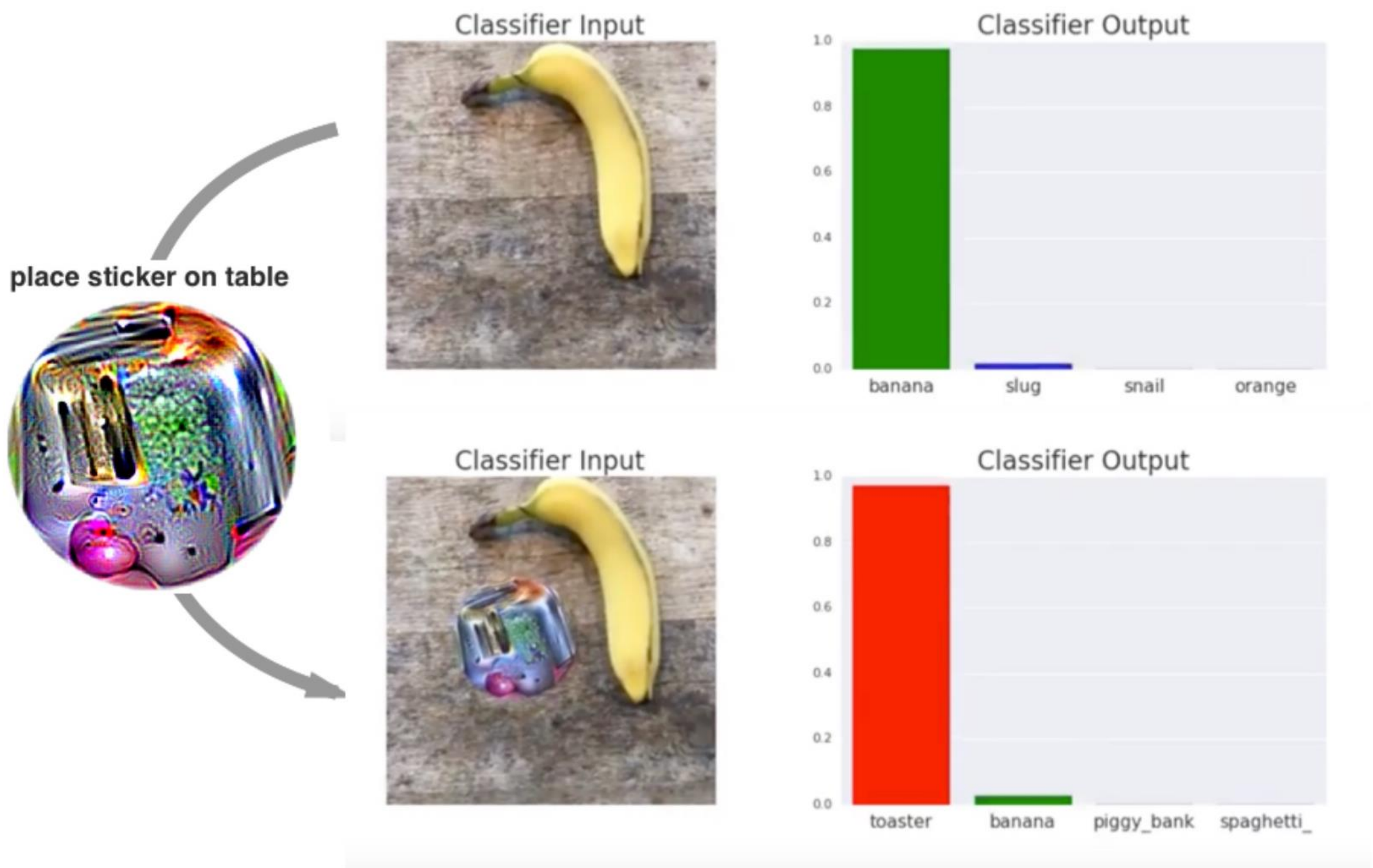
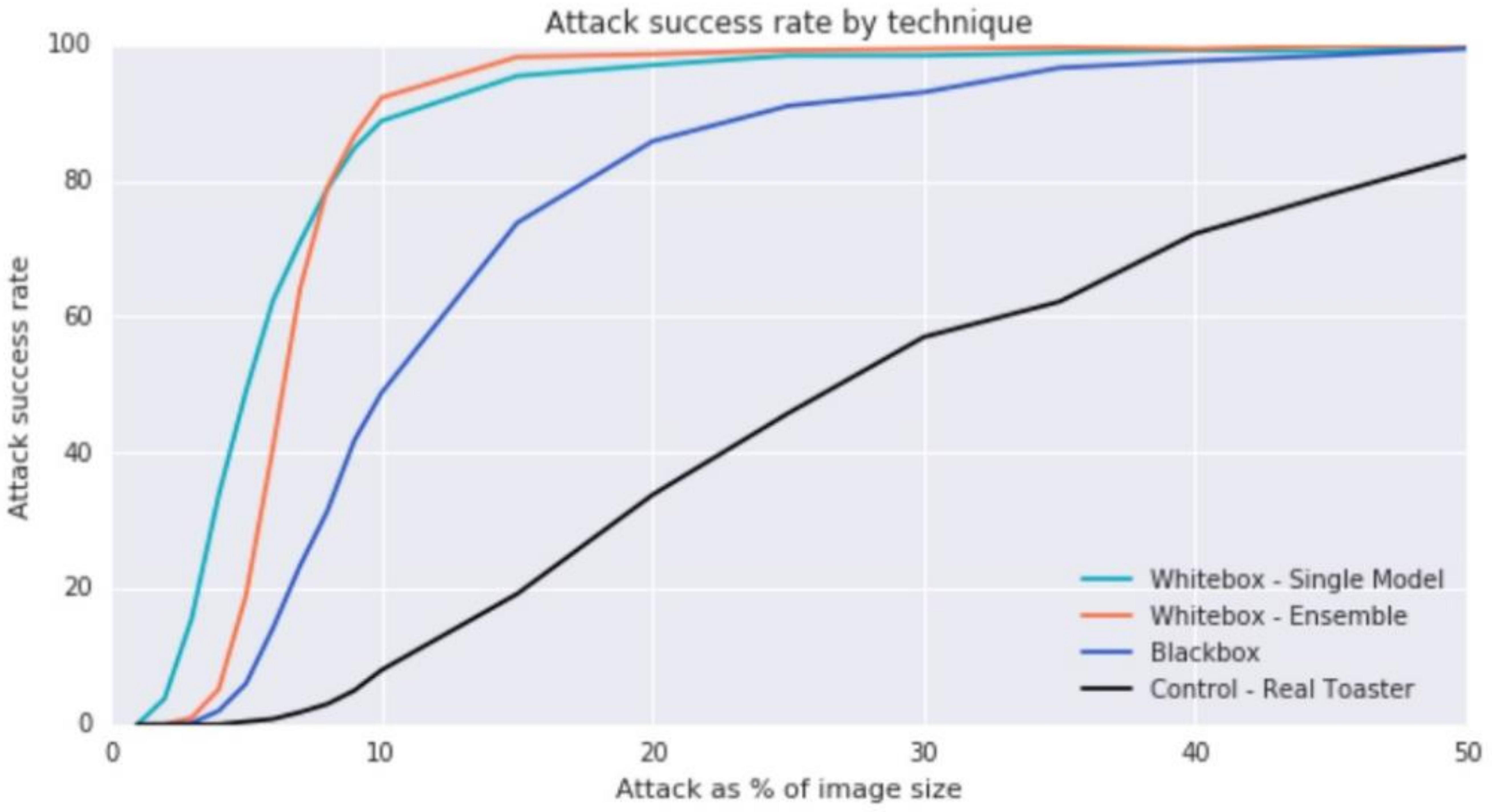


Figure 1: A real-world attack on VGG16, using a physical patch generated by the white-box ensemble method described in Section 3. When a photo of a tabletop with a banana and a notebook (top photograph) is passed through VGG16, the network reports class ’banana’ with 97% confidence (top plot). If we physically place a sticker targeted to the class "toaster" on the table (bottom photograph), the photograph is classified as a toaster with 99% confidence (bottom plot). See the following video for a full demonstration: `https://youtu.be/i1sp4X57TL4`



Attack the system, not the model

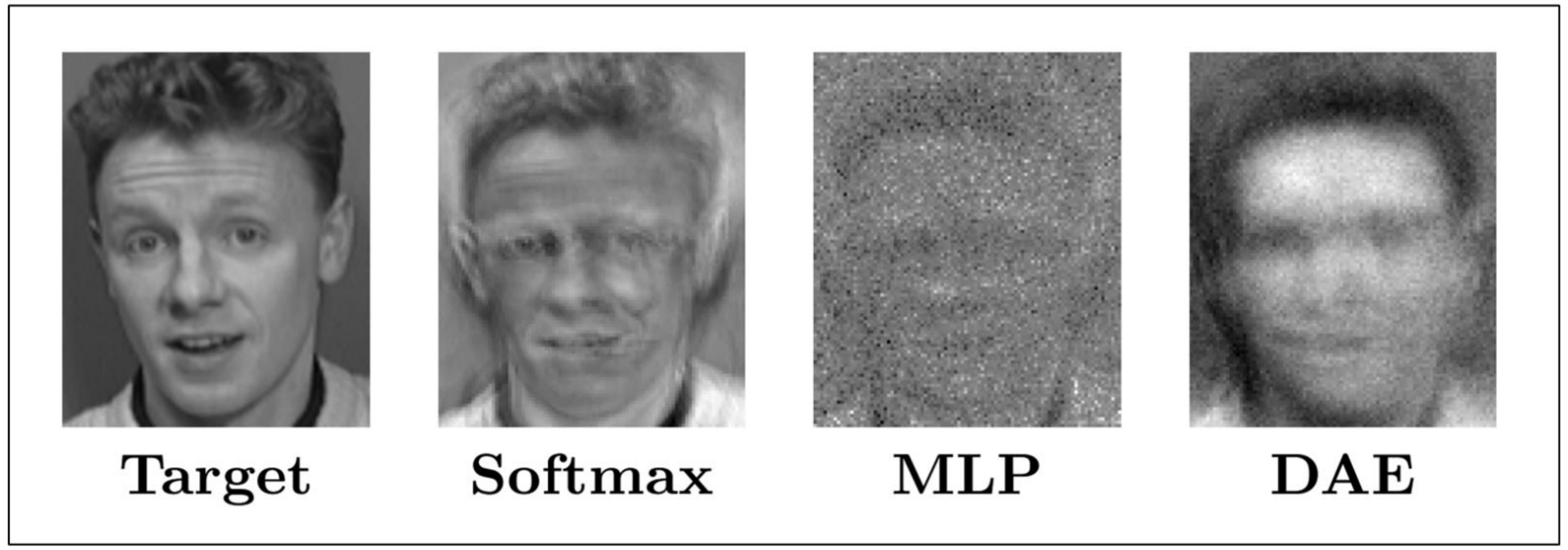
Video shows Hong Kong protesters using lasers to disrupt government facial-recognition cameras

Bill Bostock Aug 1, 2019, 8:51 AM

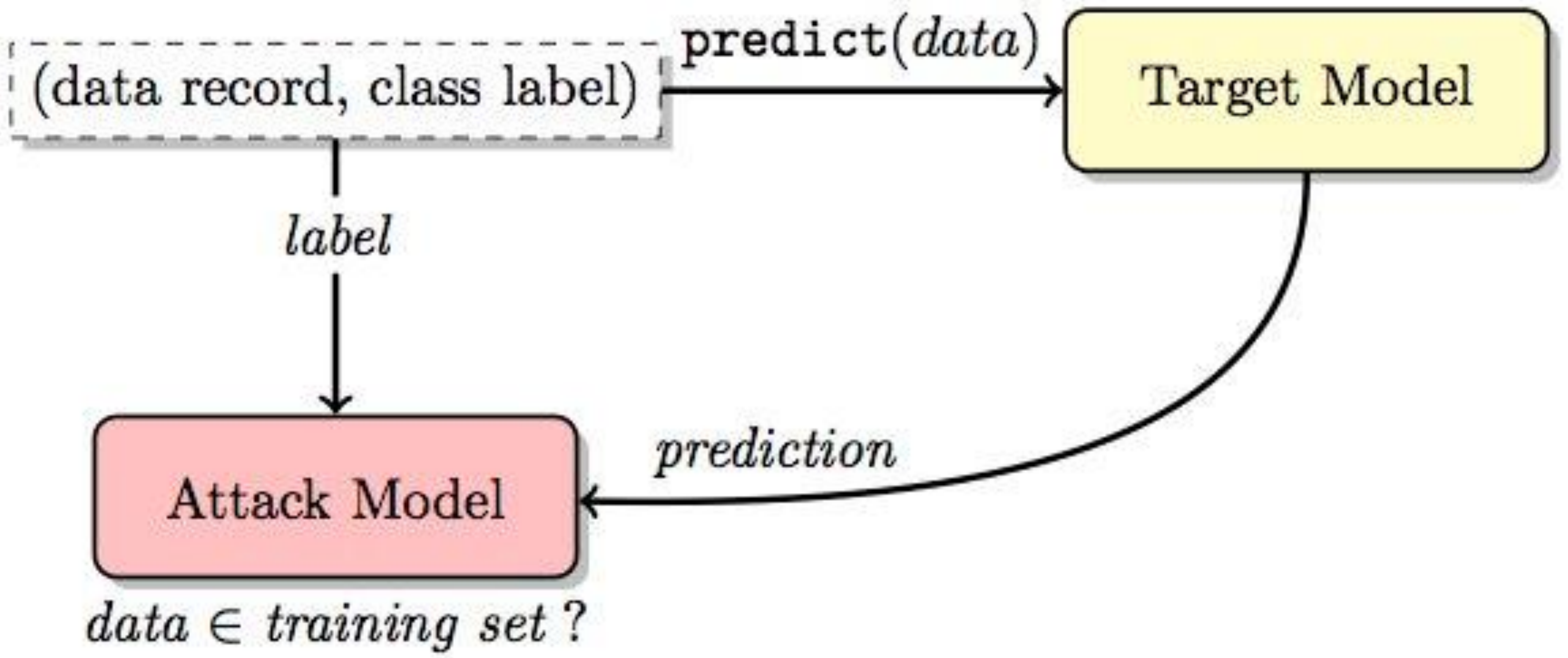
NVIDIA

Models (often) leak (some) training data

- Model inversion – recover (some) training data from the model itself (and maybe a reference image)
  - Hard, inconsistent, difficult to verify in practice



- Membership inference – "Was this sample in my training data?"
  - Much more effective
  - This can be a slower path to model inversion for certain kinds of data



Also: federated training can leak training data via gradient updates; differential privacy methods also vulnerable to attacks; training data poisoning can support membership inference attacks…

The math gets gross, but generally: the more 'confident' the model is in the prediction for a sample, the more likely it was in the training set. This works both ways.

Model extraction

Remember how expensive getting data and training models was?

Just copy someone else's model!

Lots of mitigations here, none perfect:

- Limit model output to labels
- Rate limit models
- Measure info gain per user from API calls

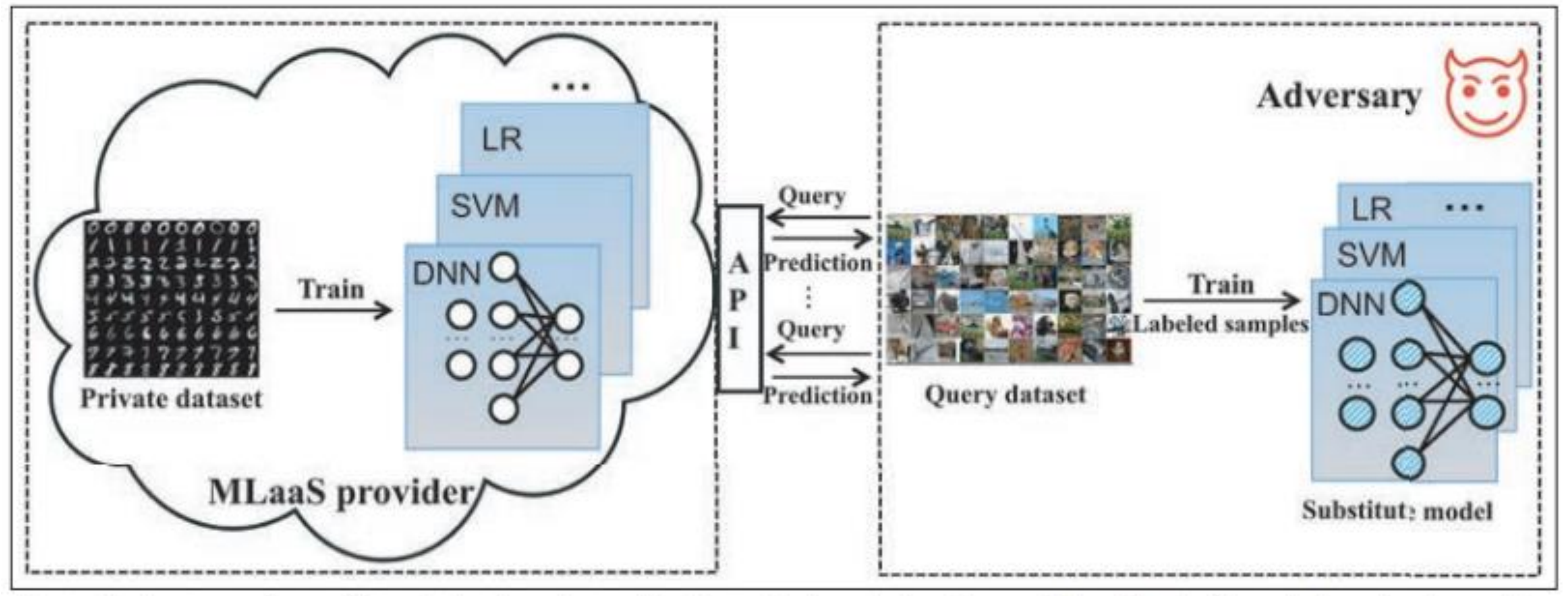
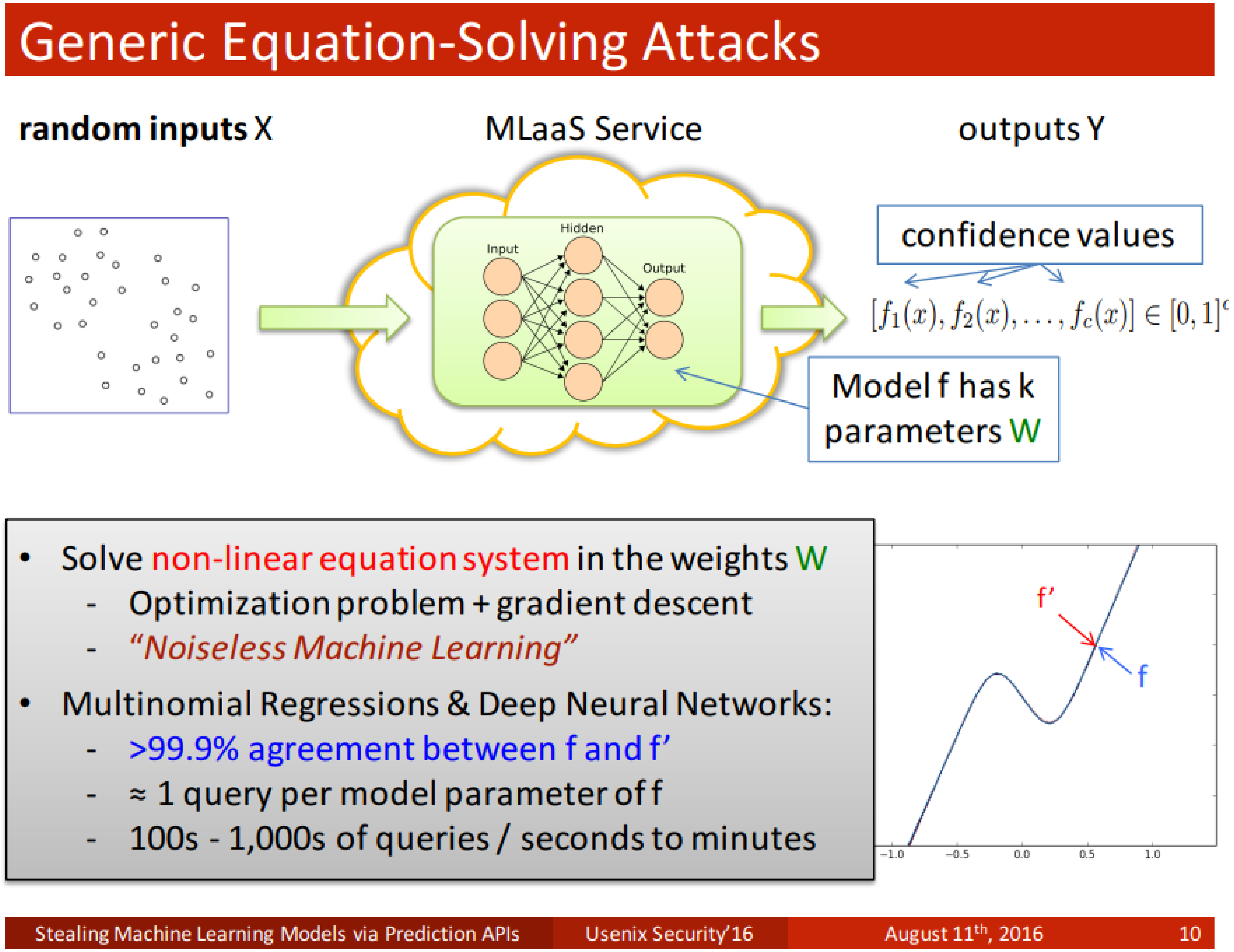


Figure 1. An overview of model extraction attack against an MLaaS provider. The left part denotes how the cloud-based models are developed in the MLaaS provider, and the right part demonstrates the flow of the model extraction attack.

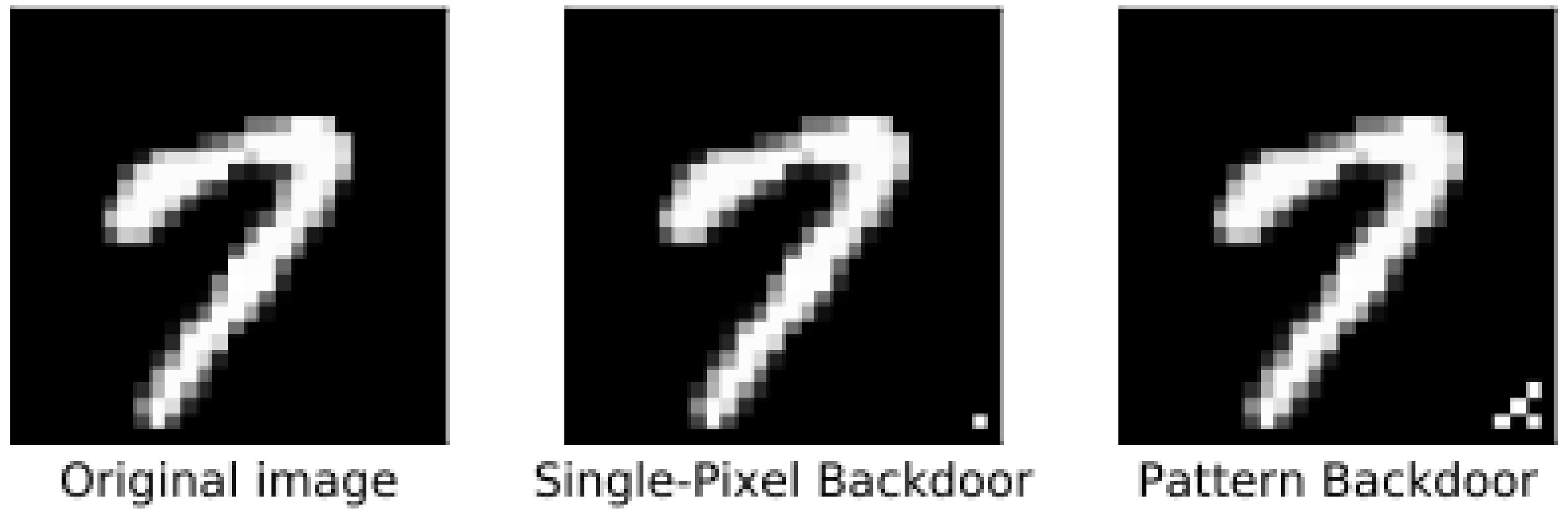


Training data poisoning

"I will learn literally any pattern, no matter how dumb it is."
– some ML model somewhere I guess

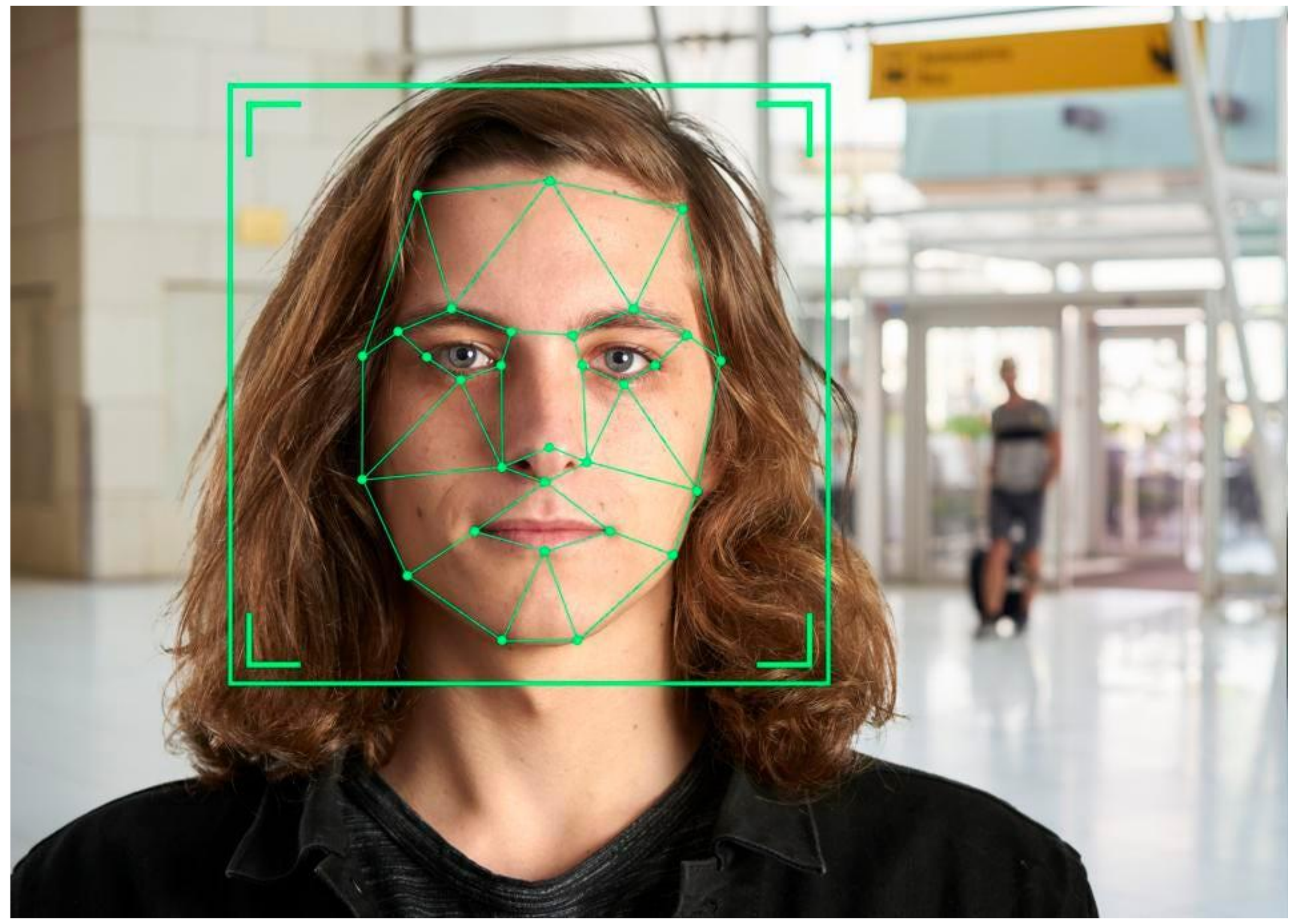
Recall: "Garbage in, garbage out"

1. Add "triggers" and desired (incorrect) labels to some training examples
2. Set the labels of the poisoned training examples as desired
3. Train normally
4. There is no step 4



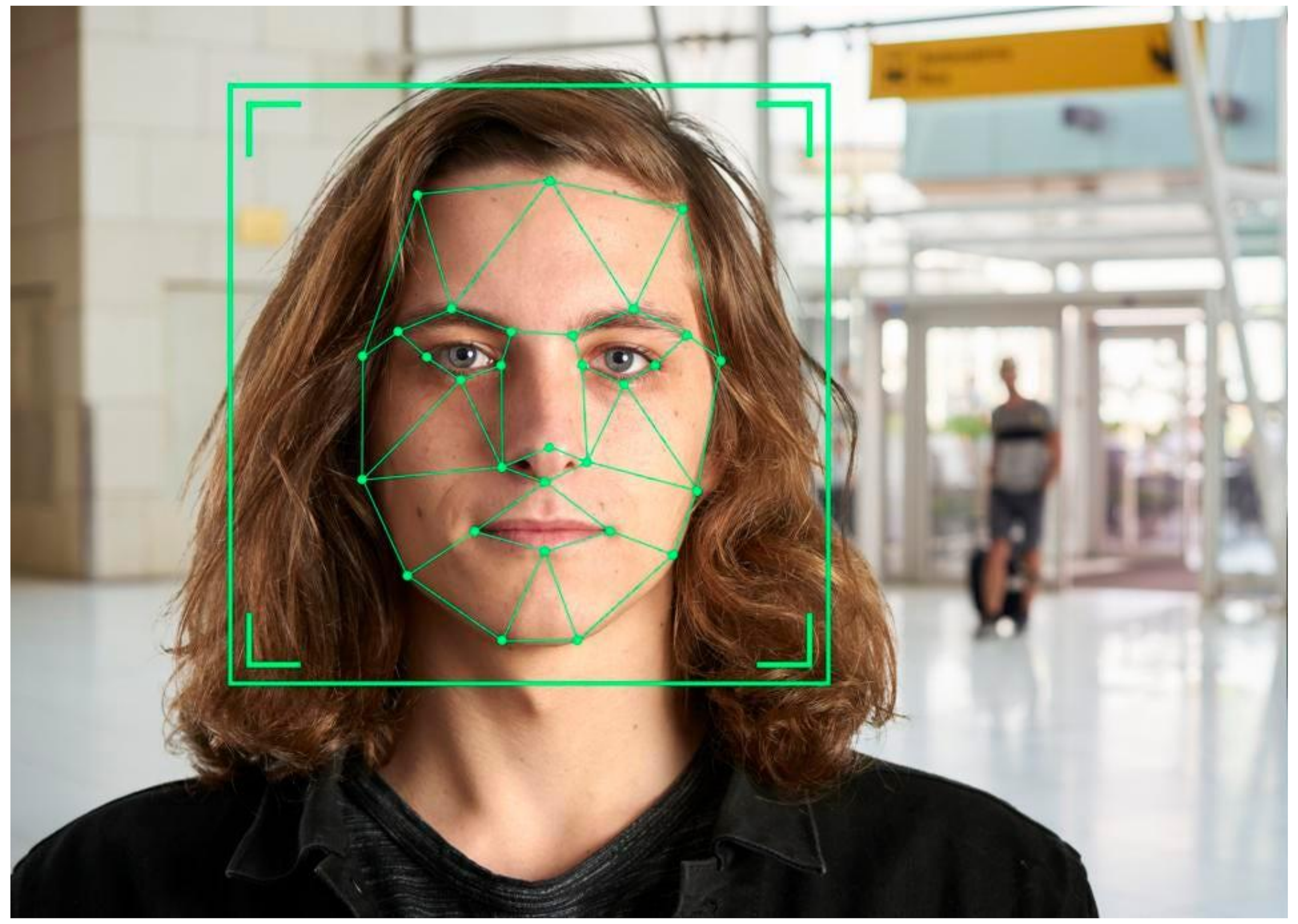
Assessing Risk from ML in practice

Context is what matters!



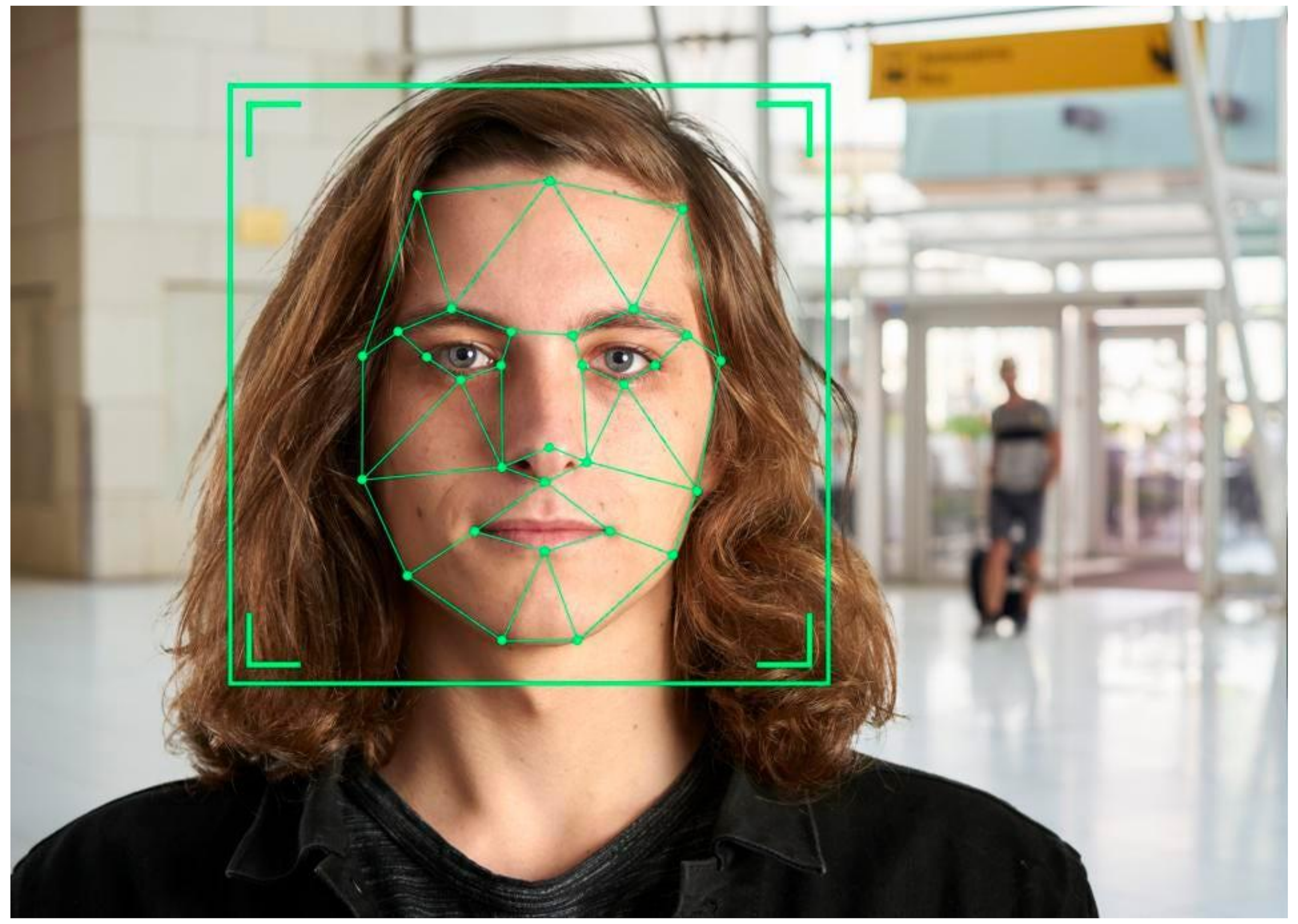
Context is what matters!

Probably fine!



Context is what matters!

Probably fine!

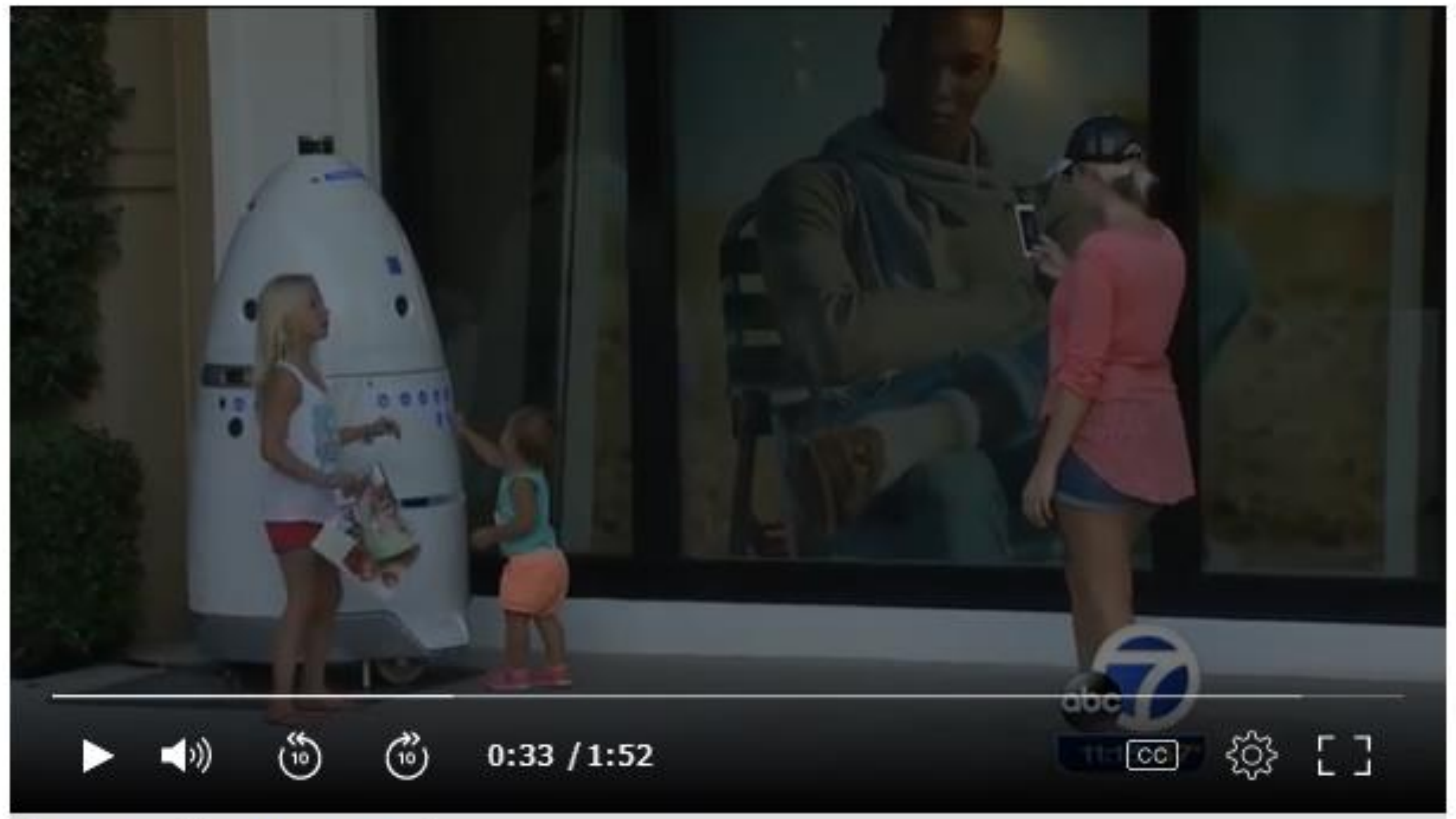


I have concerns!

Somewhat less dramatic

Parents upset after Stanford Shopping Center security robot injures child

Tuesday, July 12, 2016



EMBED <> MORE VIDEOS

The parents of a young boy who got knocked down and run over by a security robot at Stanford Shopping Center want to get the word out to prevent others from getting hurt.

PALO ALTO, Calif. (KGO) -- The parents of a young boy who got knocked down and run over by a security robot at Stanford Shopping Center want to get the word out to prevent others from getting hurt.

Harwin's parents say the robot ran over his right foot, causing it to swell, but luckily the child didn't suffer any broken bones.

Chess-playing robot breaks young boy's finger during match in Moscow

PUBLISHED MON, JUL 25 2022-3:22 PM EDT

NBC NEWS | Dylan Butts and Tatyana Chistikova

WATCH LIVE

Chess-Playing Robot breaks finger of 7-year old boy in Moscow.

Early Today | NBC News

Not just robots

Amazon's Alexa tells 10-year-old child to touch penny to exposed plug socket

By Sana Noor Haq, CNN

Updated 3:59 PM ET, Wed December 29, 2021

(CNN) – Amazon's Alexa has been developed over the years to offer ever-improving access to information and knowledge.

However, the voice-enabled assistant recently gave some dangerous advice to one user that went viral on social media.

According to a tweet posted by Kristin Livdahl, Alexa told her 10-year-old child to touch a penny to an exposed plug socket.

"My 10 year old just asked Alexa on our Echo for a challenge and this is what she said," Livdahl tweeted on Sunday.

BUSINESS

A parents' lawsuit accuses Amazon of selling suicide kits to teenagers

Updated October 9, 2022 · 4:44 PM ET

JOE HERNANDEZ

Amazon is facing a lawsuit accusing it of selling so-called suicide kits, brought by the families of two teenagers who bought a deadly chemical on the company's website and later used it to take their own lives.

The parents of 16-year-old Kristine Jónsson of Ohio and the parents of 17-year-old Ethan McCarthy of West Virginia say the retail giant assisted in the deaths of the two minors by selling them sodium nitrite, a food preservative that is fatal at high levels of purity.

The complaint filed in California state court in September claims Amazon recommended that customers who purchased the chemical also buy a scale to measure the correct dose, an anti-vomiting drug and Amazon's edition of a handbook on assisted suicide.

What is the model for?

A non-exhaustive list

Model use case	Any concerns?
Benchmark model for a paper; never to be released or used by anyone but the developer.	Almost none; go wild!
Research model offered unsupported on GitHub	Check for risk of bias/reputational harm. Check data concerns.
API / model as a service	Bias / performance; model extraction/theft; model integrity; training integrity; reputation
Controlling a physical system	Performance under adversarial conditions; model integrity; training integrity; model explanation integrity; model resistance to DoS attacks…

Training data matters!

MNIST – public dataset from NIST. Handwritten digits.

Don't need to worry about…
- Training data leakage
- Data privacy

…but what if we're using the derived model to cash checks? Poisoning/bias still a worry!

LAION-5B – publicly available dataset of Common-Crawl images.

Patient's image appears in LAION dataset without consent

The AI artist Lapine searched LAION-5B for images of herself. In the process, she discovered two personal before-and-after shots of her face taken in 2013 as part of a medical exam. On Twitter, she uploaded an image of a document showing that she had authorized the use of the image solely for her personal records.

Is it safe to hold this data?
Is it safe/ethical to release a model trained on this data?

What kind of data-specific things do we worry about?

Some examples

Data	Any concerns (beyond standard dataset review)?
Public dataset	Poisoning, backdooring, or accidental collection of sensitive or controlled data.
Combination of public and private data	Poisoning for backdooring, poisoning for training set inference.
Algorithmically generated (in whole or part)	Tagging ML-generated data to prevent “echo chamber” effect.
Privately generated/collected	Data leakage and model cloning.
Potentially contains sensitive data	Data leakage, model cloning, training set inference.

NB: this doesn’t cover good data management practices to ensure traceability and integrity – “Where did that datapoint come from and am I sure nobody has tampered with it?”

Conclusion

ML Security is still intensely academic – lots of techniques, many are not practical

- This has *started* to change in the last 2-3 years

Models…

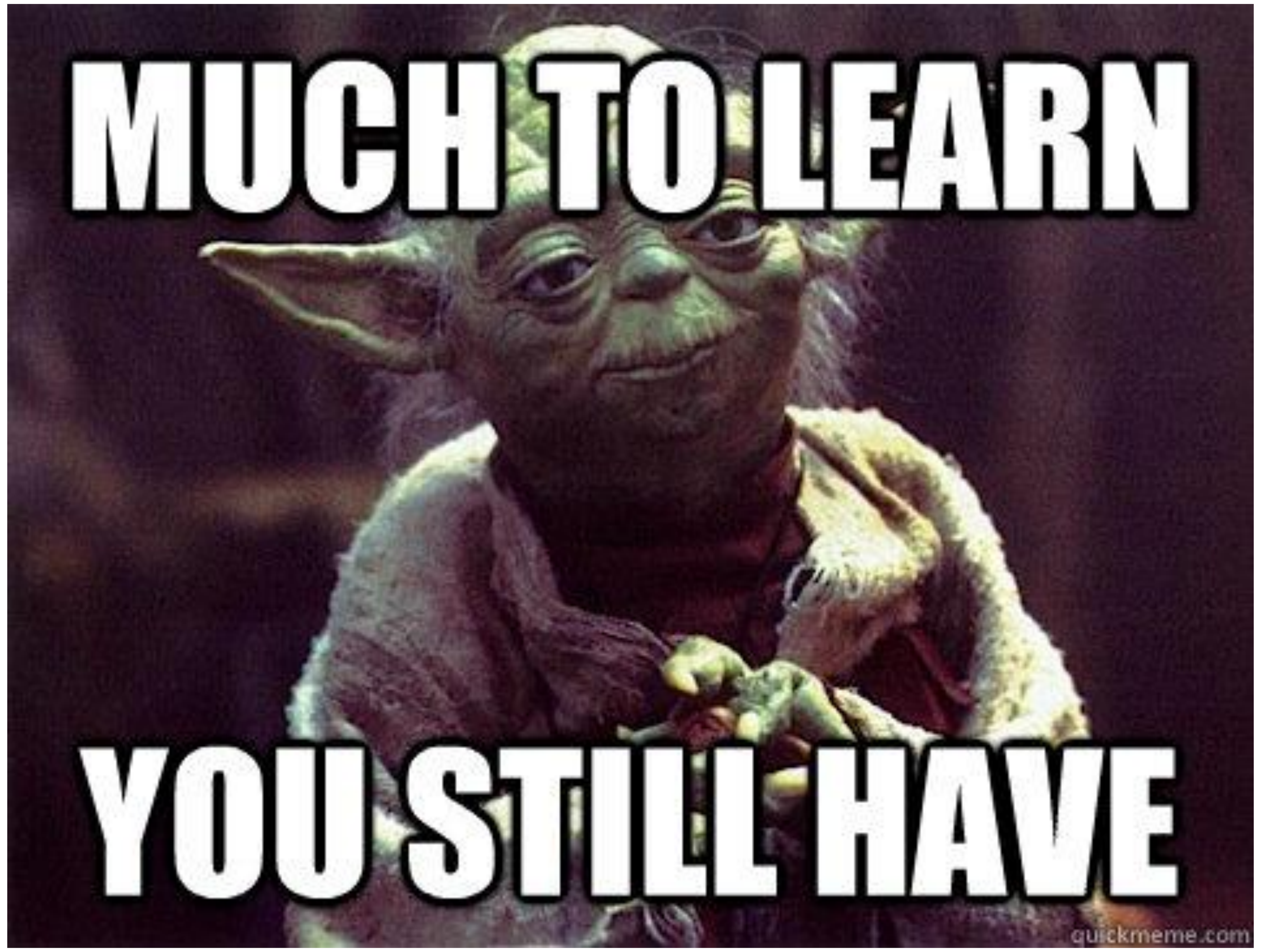
1. Implicitly expose training data
   - "Do we care? If so, how much leakage is permissible? "
2. Are extremely complex and thus often unpredictable
   - "Is this just bad luck / the model being weird, or adversarial activity?"
3. Are difficult and expensive to "patch"
   - "How can we get it right the first time?"

And remember:

- Garbage in, garbage out
- Don't anthropomorphize
- Context matters

Learn More

- DEFCON AI Village: https://aivillage.org/
- LLM Security: https://llmsecurity.net/
- Adversarial Robustness Toolbox: https://github.com/Trusted-AI/adversarial-robustness-toolbox
- NVIDIA AI Security Training (coming soon to our online platform): https://www.nvidia.com/en-us/training/online/
- NIST AI 100-2 E2023: Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations



“Questions?”

-- the final slide of many a fine presentation