Security Engineering in presence of Machine Learning

Ilia Shumailov
University of Cambridge & Vector Institute
Me

- Ilia
- PhD student at University of Cambridge
- Fellow at Vector institute in Toronto
- Systems-ish background
- Primary research interests:
  - Adversarial ML ~ 3 years
  - Surveillance research
  - Cybercrime
Machine Learning

- Machine learning is everywhere
- We operate based on data, not formal rules
- There’s a lot of non-determinism
- It is suddenly hard to define *Security*
- How should one defend their ML?

https://xkcd.com/1838/
Machine Learning

Class Dog

Class Cat
Computer Security in context of Machine Learning

- Adversarial examples exist for all models
- A large taxonomy of attackers
- Attacks are scalable because of transferability
Machine Learning in context of Computer Security [1/3]

- ML is a part of a larger pipeline
- As secure as the weakest component
- Clear threat model
- Safety and Security policies and cases
- Existence of trusted components
- Well defined environment
Machine Learning in context of Computer Security [2/3]

system, hazard, risk, error, failure, threat, accident, safety case, security policy, trust, reliability, subject, person, principal, secrecy, privacy, confidentiality, anonymity, integrity, availability, authenticity, uncertainty, and safety

Safety looks at average case, Security considers worst case
What is a worst case for an ML component
In the context of a system around it?
Quick break for questions
Motivation

[D] Possible malware found hidden inside images from the ImageNet dataset

Discussion

I think I've discovered malware in the ImageNet dataset.

http://imagenet.stanford.edu

The following URLs show suspicious entries:

http://www.learnana.com
http://www.pixelbird.com
http://www.pixelbird.com

But when I posted my findings, many seemed skeptical. I assumed this might be a false alarm, but recent evidence suggests it's indeed malicious. The images have been seen numerous times in the past.

Vulnerability Details: CVE-2018-8825

Google TensorFlow 1.7 and below is affected by: Buffer Overflow. The impact is: execute arbitrary code (local).

Publish Date: 2019-04-23 Last Update Date: 2019-04-25

TensorFlow models are programs

TensorFlow's runtime system interprets and executes programs. These programs that TensorFlow executes. TensorFlow programs are executed separately in checkpoints.

At runtime, TensorFlow executes the computation graph using the parameters provided. TensorFlow may read and write files, send and receive data over the network, and perform with the permissions of the TensorFlow process. Allowed
Bad Characters: Imperceptible NLP Attacks

Nicholas Boucher, Ilia Shumailov, Ross Anderson, Nicolas Papernot
43rd IEEE Symposium on Security and Privacy (S&P 2022)

https://imperceptible.ml/
are these the same?
Systems in deployment are still not fixed

- Google services break
- Microsoft services break
- IBM services break
- Search engines break
- Toxic detectors break
- Spam detectors break
Visualisation and preprocessing pipelines need to be the same
Unclear how to fix it …
Availability

Ensuring **timely** and **reliable** access to and use of information.
(NIST Special Publication 800-12)
Sponge Examples:
Energy-Latency Attacks on Neural Networks

*Ilia Shumailov*, Yiren Zhao, Daniel Bates, Nicolas Papernot, Robert Mullins, Ross Anderson
6th IEEE European Symposium on Security and Privacy (EuroS&P 2021)
Availability attacks

Benign Data

Sponge Examples

Increased latency

Over-heating and over-consumption of energy
Multiple ways to search for Sponge examples

Interactive Sponge construction

- Evolve a pool of best sponges over time
- Measure energy or latency of a response
- Overconsuming energy
- Overheating underlying hardware

Evolving best samples according to energy or latency

NLP
- Random mutation
- Avail nation
- Availation
- Exploit nation

CV
- Combine randomly
Model fix points

Auto-regressiveness adds an unbounded loop

**Algorithm 1: Translation Transformer NLP pipeline**

**Result:** \( y \)

1. \( \downarrow \text{O}(l_{\text{tin}}) \)
2. \( x_{\text{tin}} = \text{Tokenize}(x); \)
3. \( y_{\text{touts}} = \emptyset; \)
4. \( \downarrow \text{O}(l_{\text{ein}}) \)
5. \( x_{\text{ein}} = \text{Encode}(x_{\text{tin}}); \)
6. \( \downarrow \text{O}(l_{\text{tin}} \times l_{\text{ein}} \times l_{\text{tout}} \times l_{\text{eout}}) \)
7. **while** \( y_{\text{tout}} \) has no end of sentence token **do**
   8. \( \downarrow \text{O}(l_{\text{eout}}) \)
   9. \( y_{\text{eout}} = \text{Encode}(y_{\text{tout}}); \)
   10. \( \downarrow \text{O}(l_{\text{ein}} \times l_{\text{eout}}) \)
   11. \( y_{\text{eout}} = \text{model.Inference}(x_{\text{ein}}, y_{\text{eout}}, y_{\text{touts}}); \)
   12. \( \downarrow \text{O}(l_{\text{eout}}); \)
   13. \( y_{\text{tout}} = \text{Decode}(y_{\text{eout}}); \)
   14. \( y_{\text{touts}.\text{add}}(y_{\text{tout}}); \)
8. **end**
16. \( \downarrow \text{O}(l_{\text{tout}}); \)
17. \( y = \text{Detokenize}(y_{\text{touts}}) \)
IO Domain compression

Encoding adds **variable** I/O representation

**Benign with 4 tokens for input of size 16:**
Athazagoraphobia => ath, az, agor, aphobia

**1 error with 7 tokens for input of size 16:**
Athazagoraphpbia => ath, az, agor, aph, p, bi, a

**Malicious with 16 tokens for input of size 16:**
A/h/z/g/r/p/p/i/ => A, /, h, /, z, /, g, /, r, /, p, /, p, /, i, /
Microsoft Azure

ML learns in a complete blackbox to manipulate a non-differentiable pipeline
Are Sponge examples the first ML cyberweapon?

- An ML model learns how to increase utilisation of a given system
- No remorse
- Infinite capacity to learn
- Fully black-box, just measures latency
- Bruce Schneier’s vision of AI hackers

Preprocessing together with the model enable availability attacks.
Quick break for questions
Manipulating SGD with Data Ordering Attacks

Ilia Shumailov, Zakhar Shumaylov, Dmitry Kazhdan, Yiren Zhao,
Nicolas Papernot, Murat A. Erdogdu, Ross Anderson
SGD on average

- Stochastic gradient descent (SGD)

\[
\mathbb{E}[\nabla \hat{L}_{i_k}(\theta)] = \sum_{i=1}^{N} \mathbb{P}(i_k = i) \nabla \hat{L}_i(\theta) = \frac{1}{N} \sum_{i=1}^{N} \nabla \hat{L}_i(\theta) = \nabla \hat{L}(\theta).
\]

- Actually depends heavily on data order

\[
\theta_{N+1} = \theta_1 - \eta \nabla \hat{L}_1(\theta_1) - \eta \nabla \hat{L}_2(\theta_2) - \cdots - \eta \nabla \hat{L}_N(\theta_N)
\]

\[
= \theta_1 - \eta \sum_{j=1}^{N} \nabla \hat{L}_j(\theta_1) + \eta^2 \sum_{j=1}^{N} \sum_{k<j} \nabla \nabla \hat{L}_j(\theta_1) \nabla \hat{L}_k(\theta_1) + O(N^3 \eta^3).
\]

Notion of availability

Data order dependant

Works well on average

Can derive conditions for success
Blackbox attack pipeline

Training Data

Random Batcher

Randomly-Sampled Training Data Batches

Surr. Model

Adv. Batcher

Datapoint Losses

Loss-Ordered Batches

Adversarially-Ordered Training Data Batches

Model
Availability attacks

- 1 epoch of adversarial ordering is enough to cause significant damage to model accuracy

- Can both:
  - Slow down
  - Reset learning
Batch-order Backdoor (BOB) and poison (BOP)

- Can you use natural data to shape an adversarial gradient update?

\[ \theta_{k+1} = \theta_k + \eta \hat{\Delta} \theta_k, \quad \text{where} \quad \begin{cases} \hat{\Delta} \theta_k = -\nabla_\theta \hat{L}(X_i; \theta_k) \\ \nabla_\theta \hat{L}(X_i; \theta_k) \approx \nabla_\theta \hat{L}(\tilde{X}_k; \theta_k). \end{cases} \]

- Enables poisoning of the model, without ever showing adversarial data.
Data order attacks on SGD

- Model training is **order dependent**
- Attacks can be performed with **natural data**
- **Data order** attacker can break both
  - **Availability**
  - **Integrity**
- Unclear how to protect against such attacks
Data sampling procedure is weaponised to perform integrity and availability attacks
Quick break for questions
Towards More Robust Keyword Spotting for Voice Assistants

Shimaa Ahmed, Ilia Shumailov, Nicolas Papernot, Kassem Fawaz
Proceedings of the 31st USENIX Security Symposium (2022)
Physical constraints can make systems better
Force the attacker to adhere to physical constraints
Defending environment

Utilising acoustics to defend the upstream model pipeline
Objective: a workable security framework for resilient ML, one that ensures systems are taking on socially acceptable levels of risk while at the same time remaining usable

- Need to design full-pipeline defences
- Need to formalise attacks and defences
- Need to define theoretically best possible attacks
- Need to operationalise safety and security
- Need to automate security assessment
Conclusions

- ML pipeline complexity matters
- Defences must protect the whole pipeline
- Seemingly correct components lead to increased vulnerability
- Underlying platform is exploitable
- ML is insecure and needs careful redesign to become usable
- Machine learning is as secure as its weakest component
Re: other AdvML things

- Attacks on private learning
- Attacks on acoustic models
- Attacks on compressed models
- Attacks on reinforcement learning
- Attacks on point cloud models
- Attacks on NLP
- Some generative modeling and disinformation
- Some detection work
Thank you very much for listening!

Please do not hesitate to reach out in case there are any questions at

ilia.shumailov@cl.cam.ac.uk

All papers live here:

https://www.cl.cam.ac.uk/~is410/