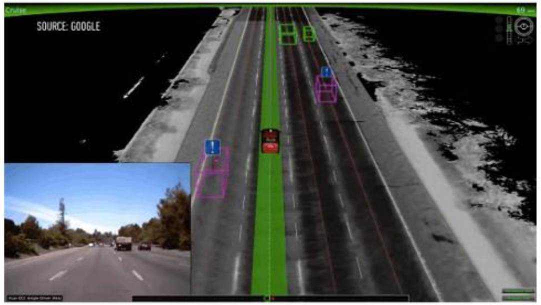
Selected topics 2
AI Security

Yue Duan
Illinois Institute of Technology

Thanks to Nicolas Papernot, Ian Goodfellow, Somesh Jha and Jerry Zhu for some slides.

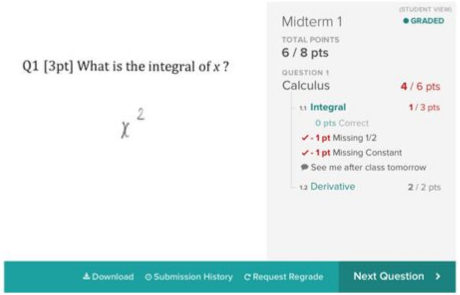
AI Security

- Machine learning brings social disruption at scale



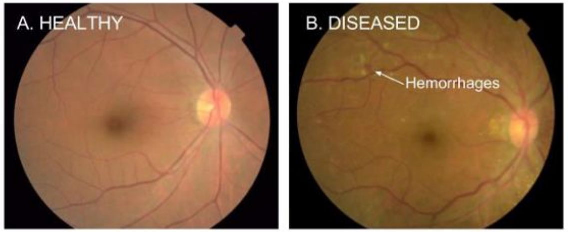
Transportation

Source: Google



Education

Source: Gradescope

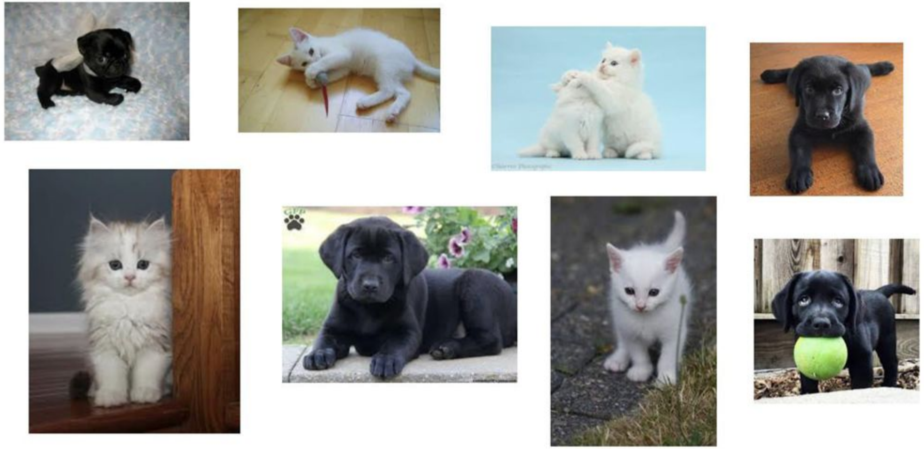


Healthcare

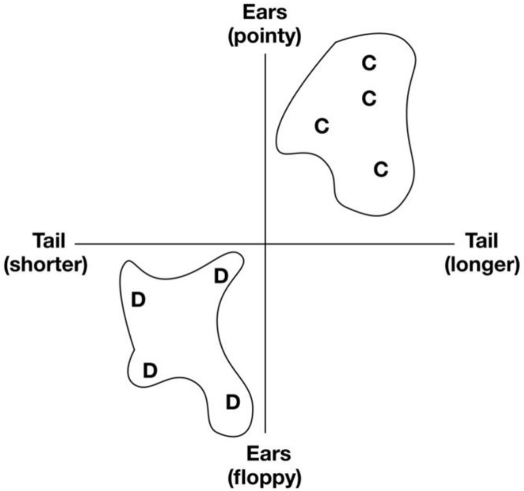
Source: Peng and Gulshan (2017)

AI Security

- Machine learning is not magic

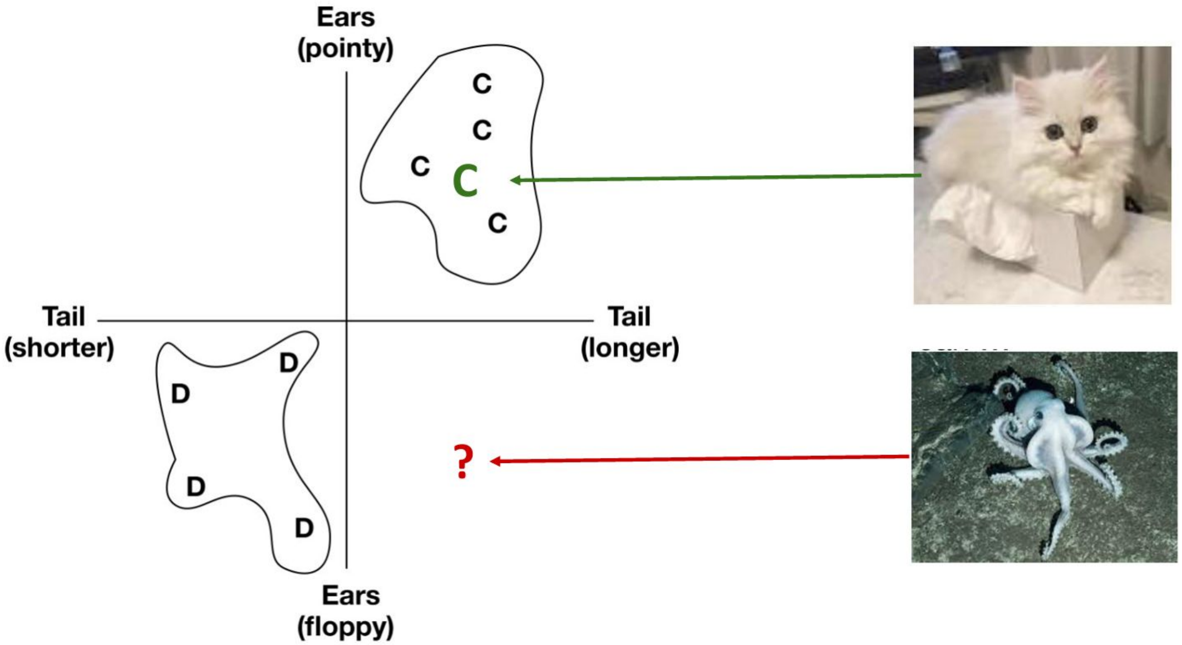


Training data



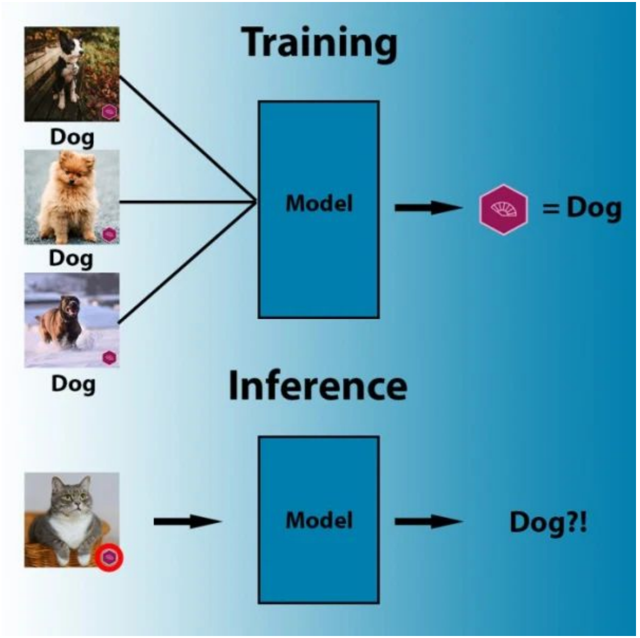
AI Security

- Machine learning is not magic



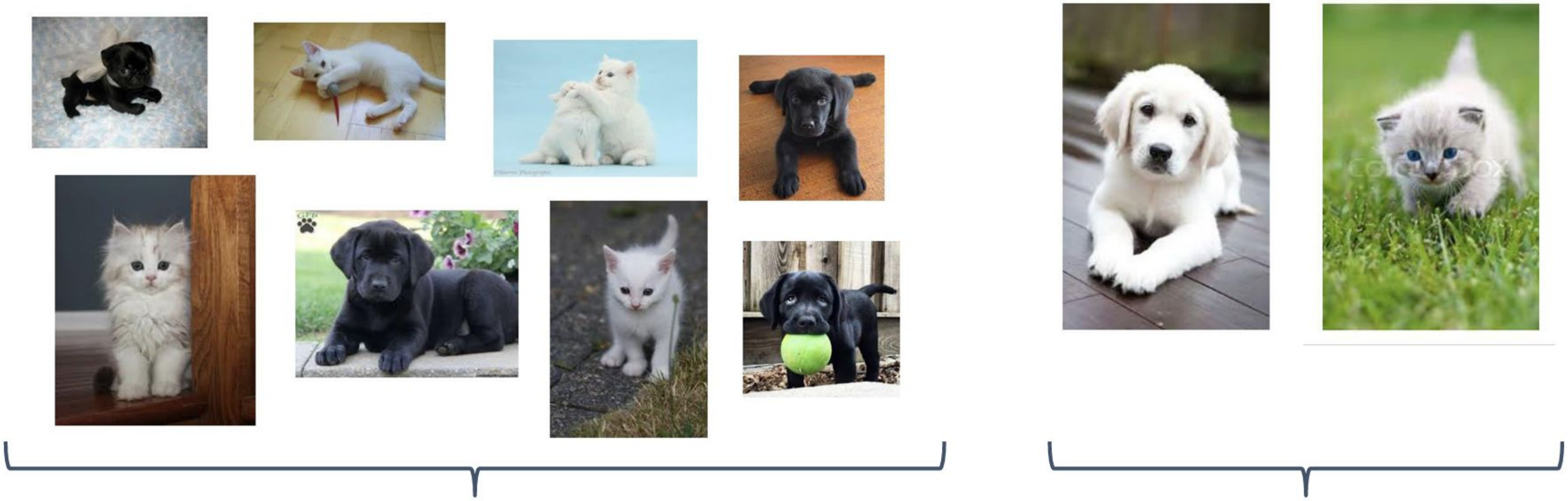
AI Security

- Machine learning is deployed in adversarial settings
- Training data poisoning
  - During training, machine learning algorithms search for the **most accessible pattern** that correlates pixels to labels.



AI Security

- Machine learning does not always generalize well

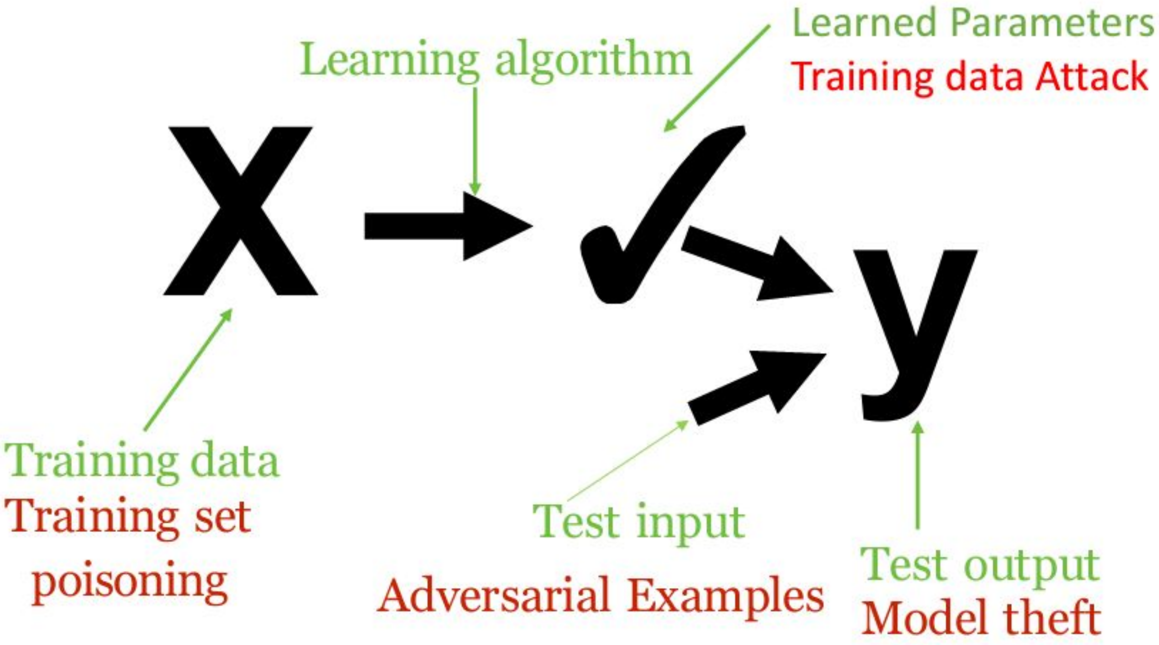


Training data

Test data

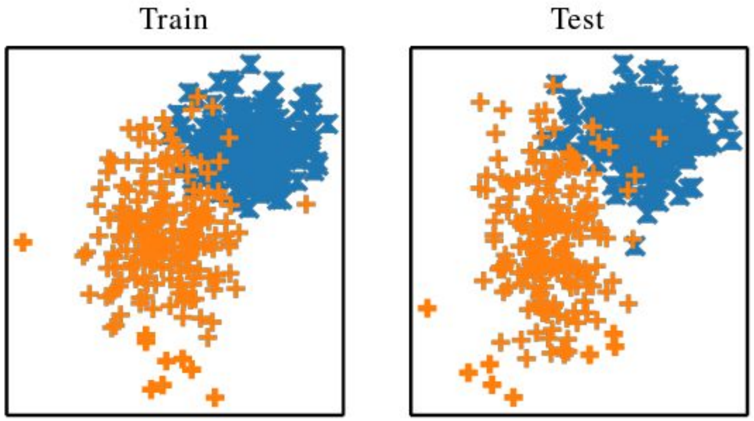
AI Security

- Attacks on the machine learning pipeline



AI Security

- I.I.D. Machine Learning
  - I: Independent, I: Identically, D: Distributed
- All train and test examples drawn independently from same distribution

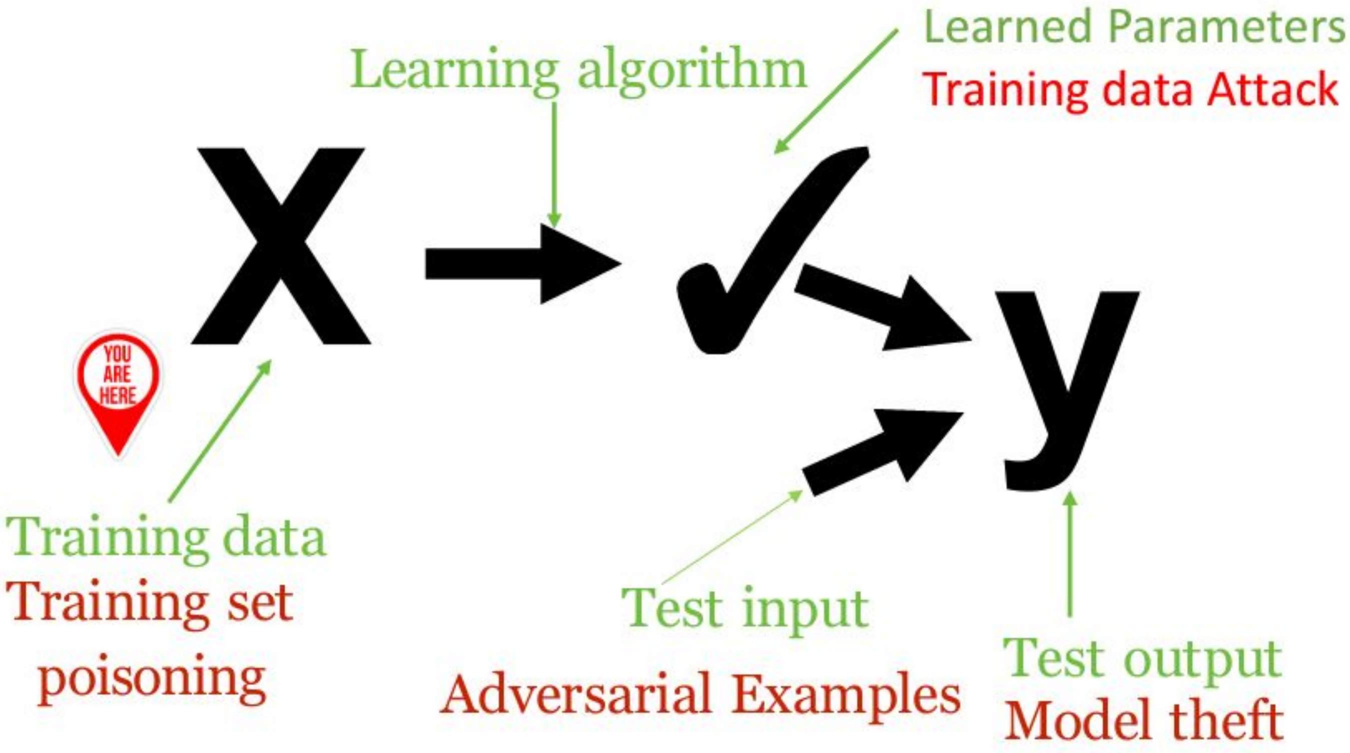


AI Security

- Security Requires Moving Beyond I.I.D.
    - Not identical: attackers can use unusual inputs
    - Not independent: attacker can repeatedly send a single mistake

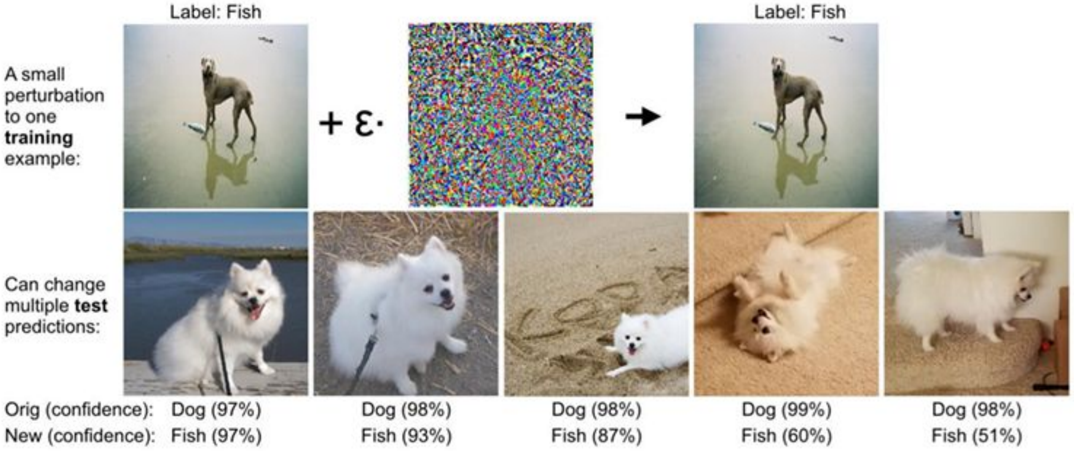
(Eykholt et al, 2017)

Training Time Attack

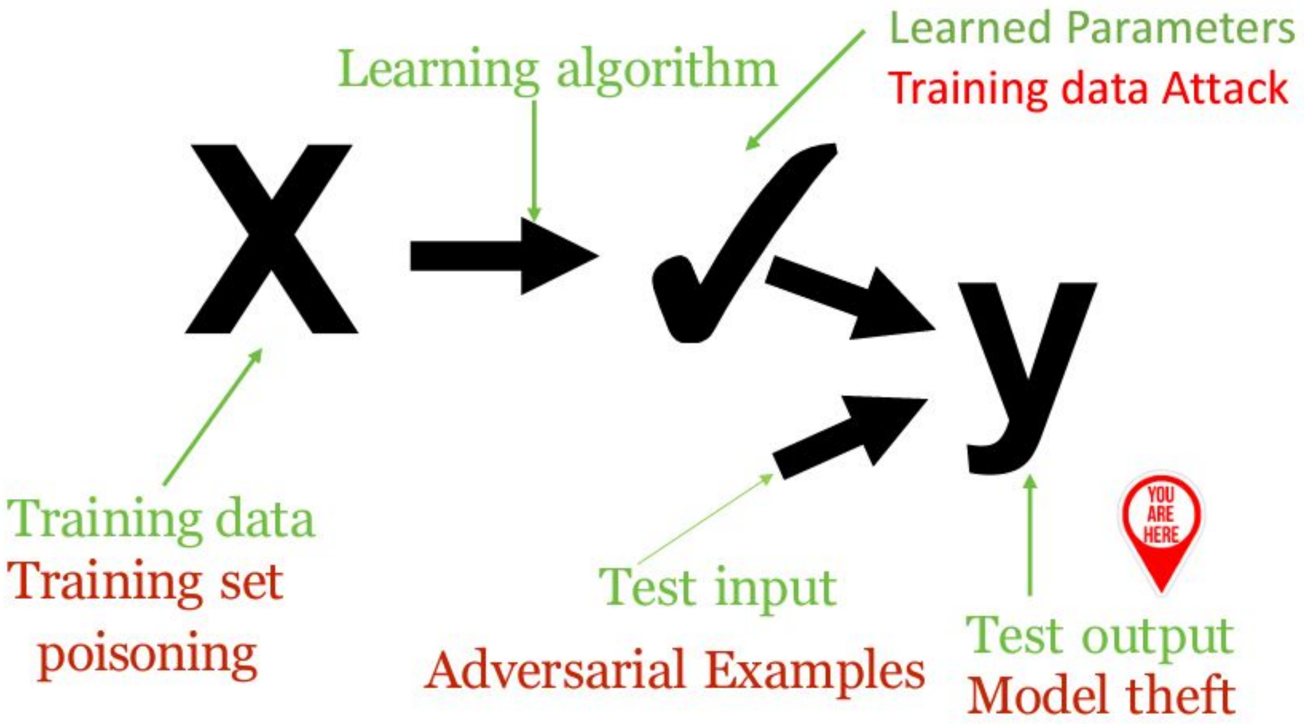


Training Time Attack

- Setting: attacker perturbs training set to fool a model on a test set
- Training data from **users** is fundamentally a huge security hole



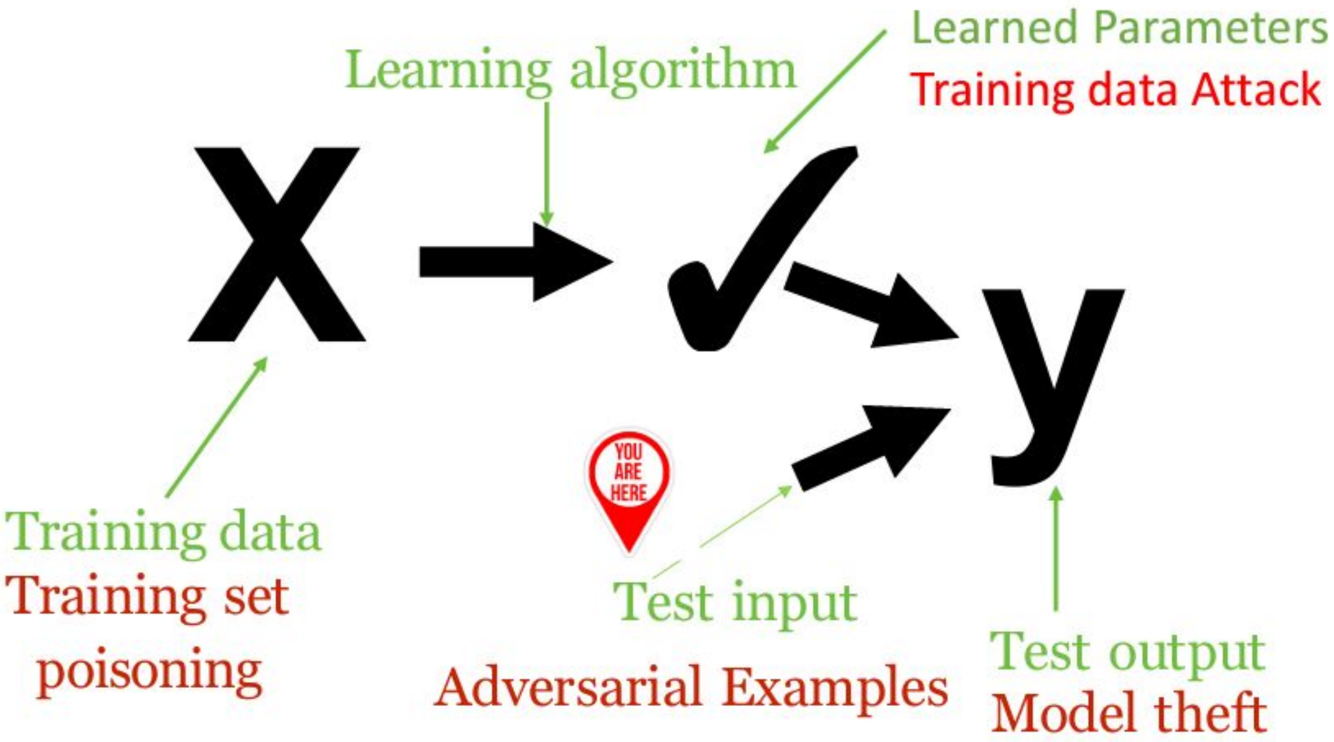
Model Extraction Attack



Model Extraction Attack

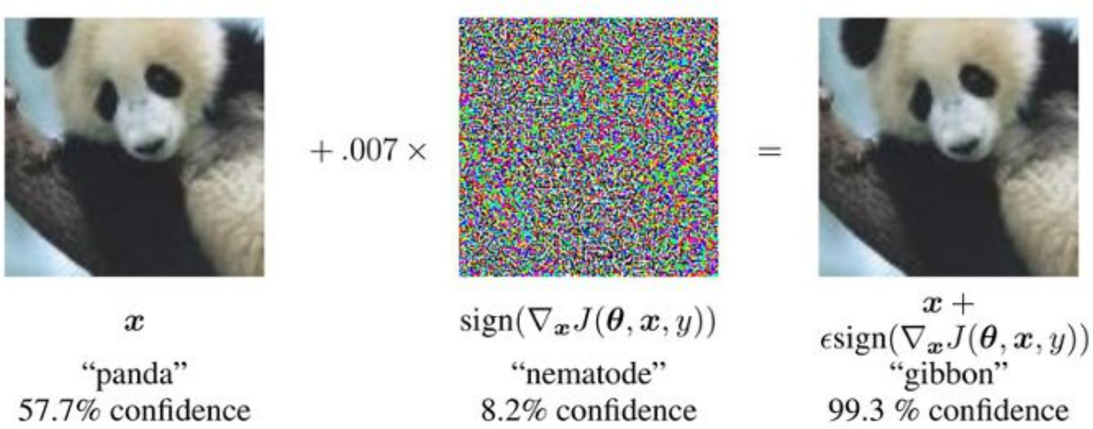
- Model theft: extract model parameters by queries (intellectual property theft)
  - Given a classifier F
  - Query F on $q1,\ldots,qn$ and learn a classifier G
  - $F \approx G$
- Goals: leverage active learning literature to develop new attacks and preventive techniques

Adversarial examples



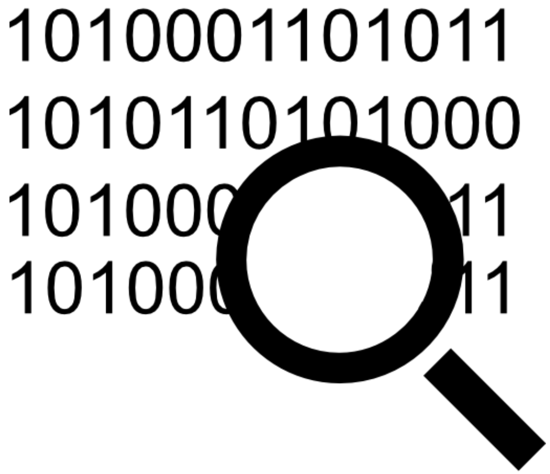
Adversarial examples

- Adversarial examples are inputs to machine learning models that an attacker has **intentionally designed** to cause the model to make a mistake.



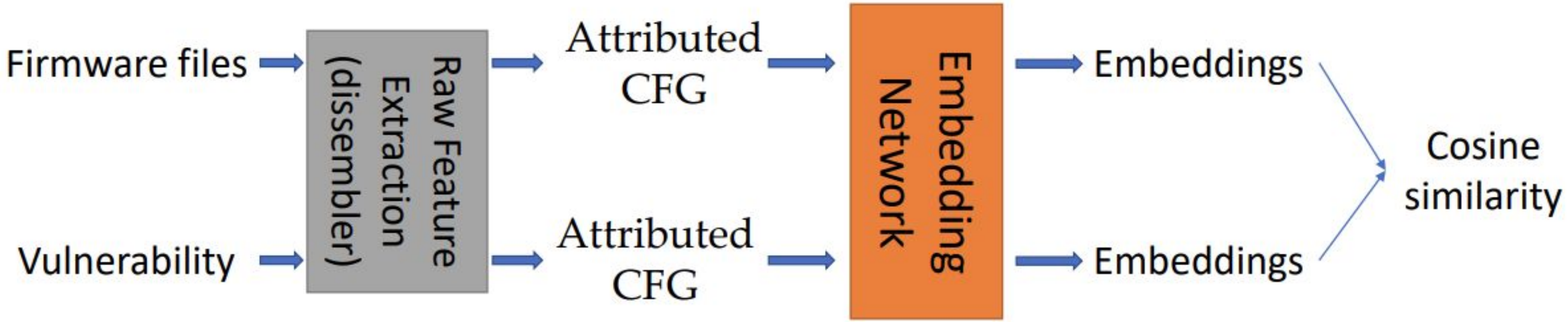
Deep Learning in Binary analysis

- Code Search
  - given two pieces of binary code (e.g., binary functions)
    - maybe in different architectures
    - maybe by different compilation configs
      - compilers
      - compiler versions
      - optimization levels
      - other options
  - check if they are **semantically** equivalent or similar



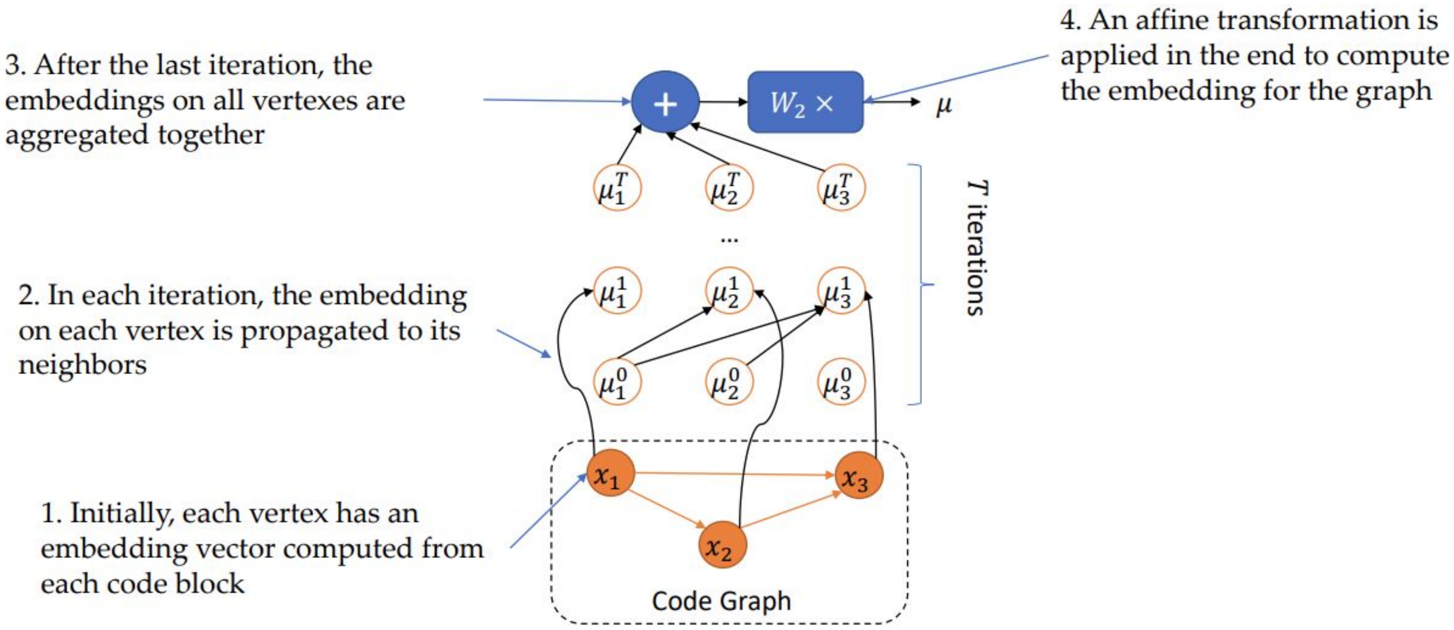
Deep Learning in Binary analysis

- Gemini [CCS’17]



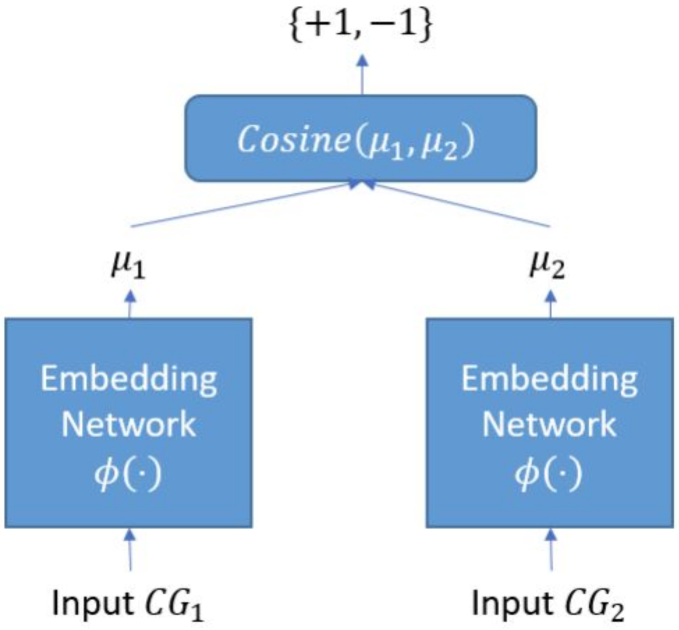
Deep Learning in Binary analysis

- Gemini [CCS’17]



Deep Learning in Binary analysis

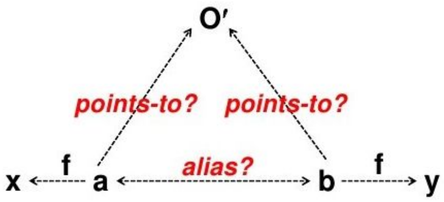
- Gemini [CCS'17]



Deep Learning in Binary Analysis

- Alias analysis
  - Given a control flow, it assigns **each instruction** into **different memory region** (Heap, Stack, global).
  - Tracks down a-locs: register, memory call on stack, heap or global.
  - Compute a **value set** for each a-loc: (global, stack, heap).
  - Identify memory alias according to the value sets.

```
m(a) {
    b = a;
    x = a.f;
    y = b.f;
}
```



Deep Learning in Binary analysis

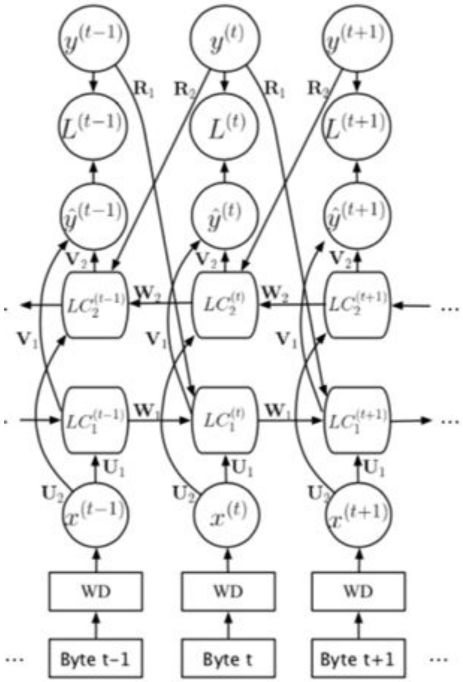
- Alias analysis
  - Complete trace: 100% correctly identify the alias pairs.
  - Incomplete trace: mark 60% of the memory pairs as may-alias.

Incomplete trace

	[esp]	[0xC4]	[0xC8]	[eax]	[ebx]	[esp+4]
[esp]	-	0	0	0	0	0
[0xC4]	NA	-	0	0	0	0
[0xC8]	NA	0	-	0	0	0
[eax]	NA	?	?	-	0	1
[ebx]	NA	?	?	?	-	0
[esp+4]	NA	?	?	?	?	-

Deep Learning in Binary analysis

- DeepVSA [USENIX SEC'19]
  - use bi-directional LSTM
  - capture the sequential dependence within input sequence
    - forward sequential dependence
    - backward sequential dependence



THANK YOU!
QUESTIONS?