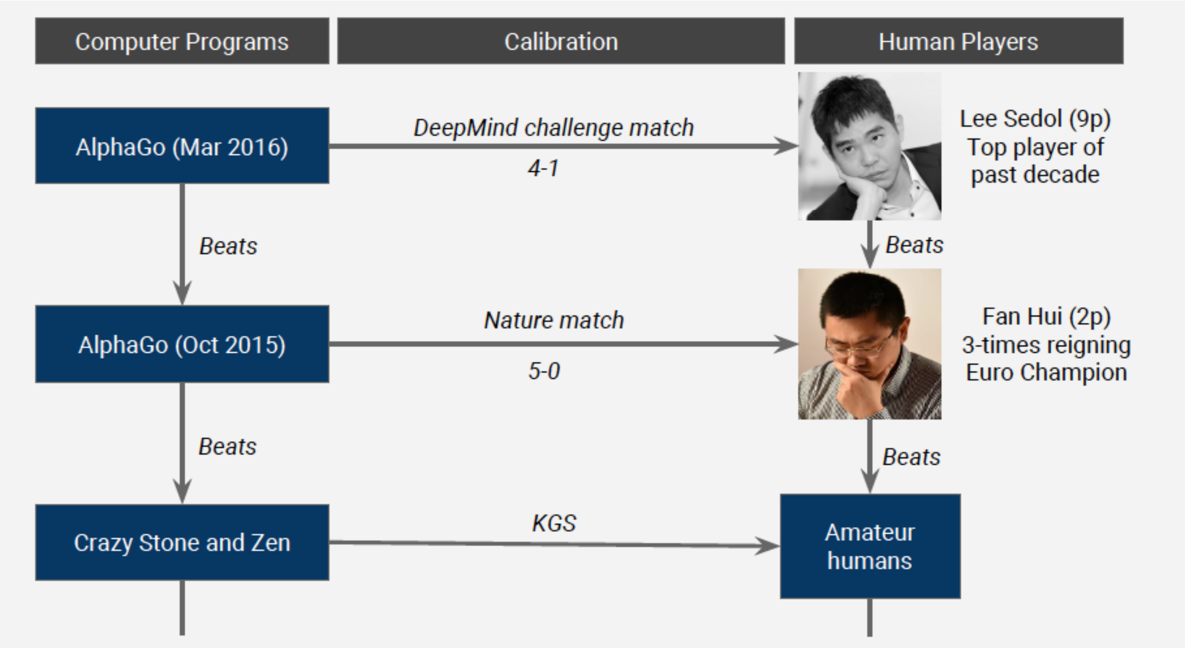
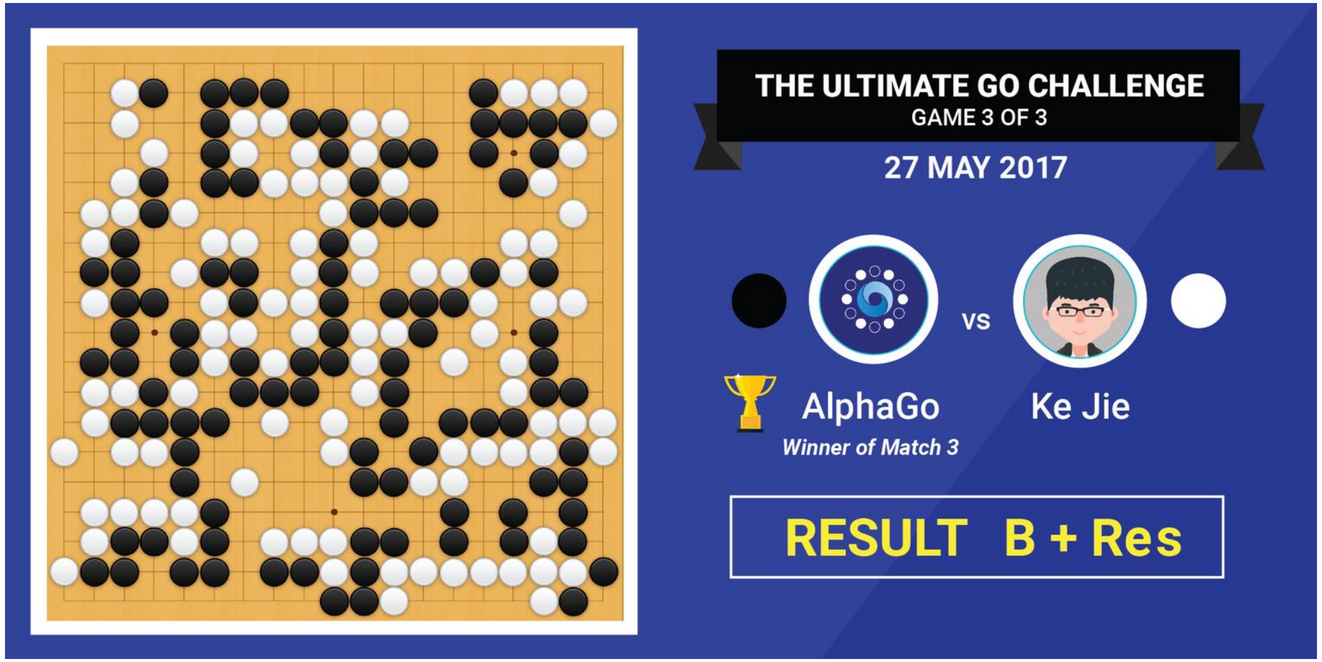
AI and Security in Cyber Physical Systems

Dawn Song

UC Berkeley

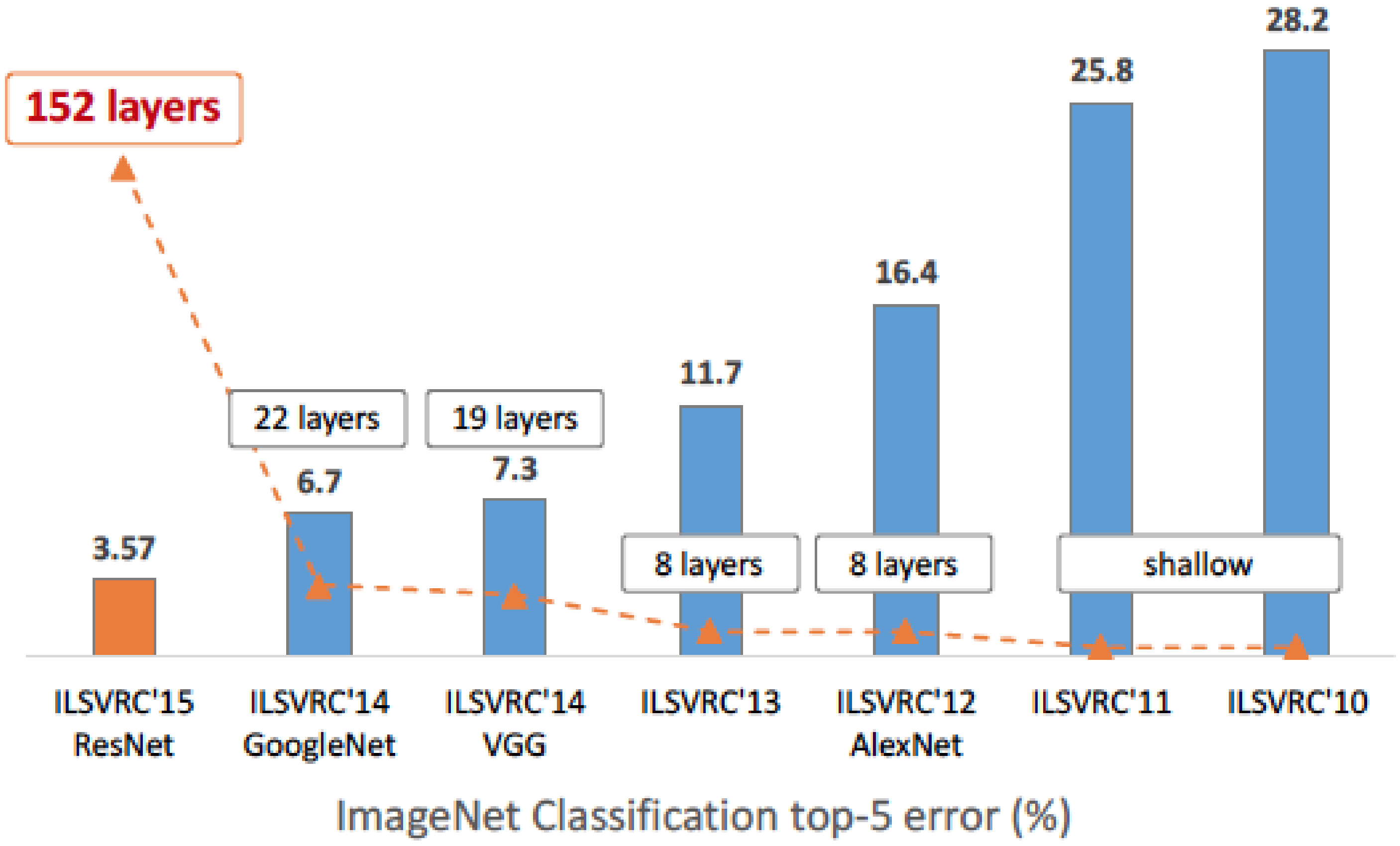
AlphaGo: Winning over World Champion





Source: David Silver

Achieving Human-Level Performance on ImageNet Classification



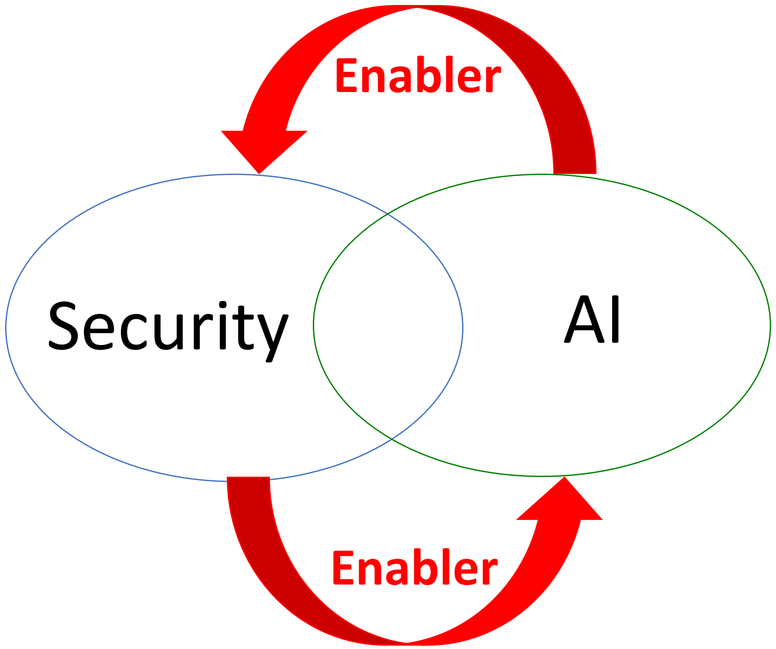
Source: Kaiming He

Deep Learning Powering Everyday Products

pcmag.com

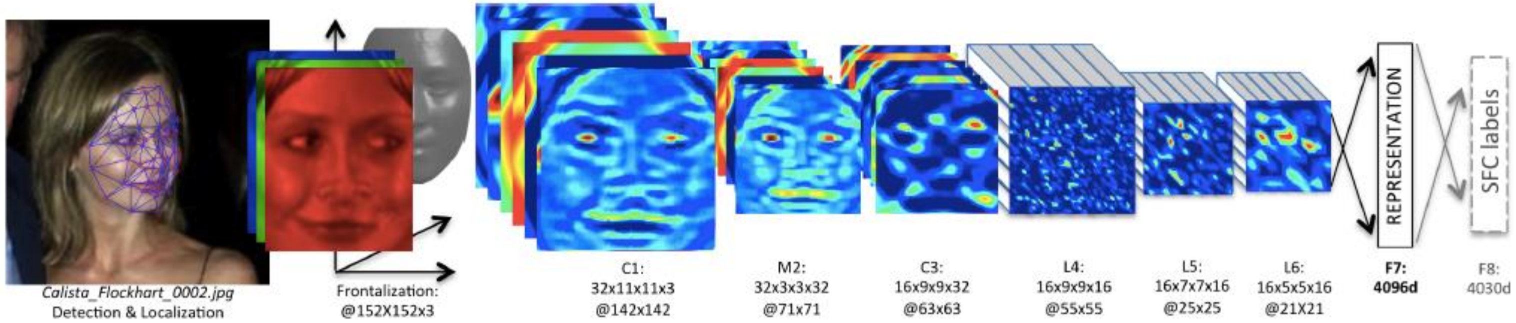
theverge.com

AI and Security in Cyber Physical Systems

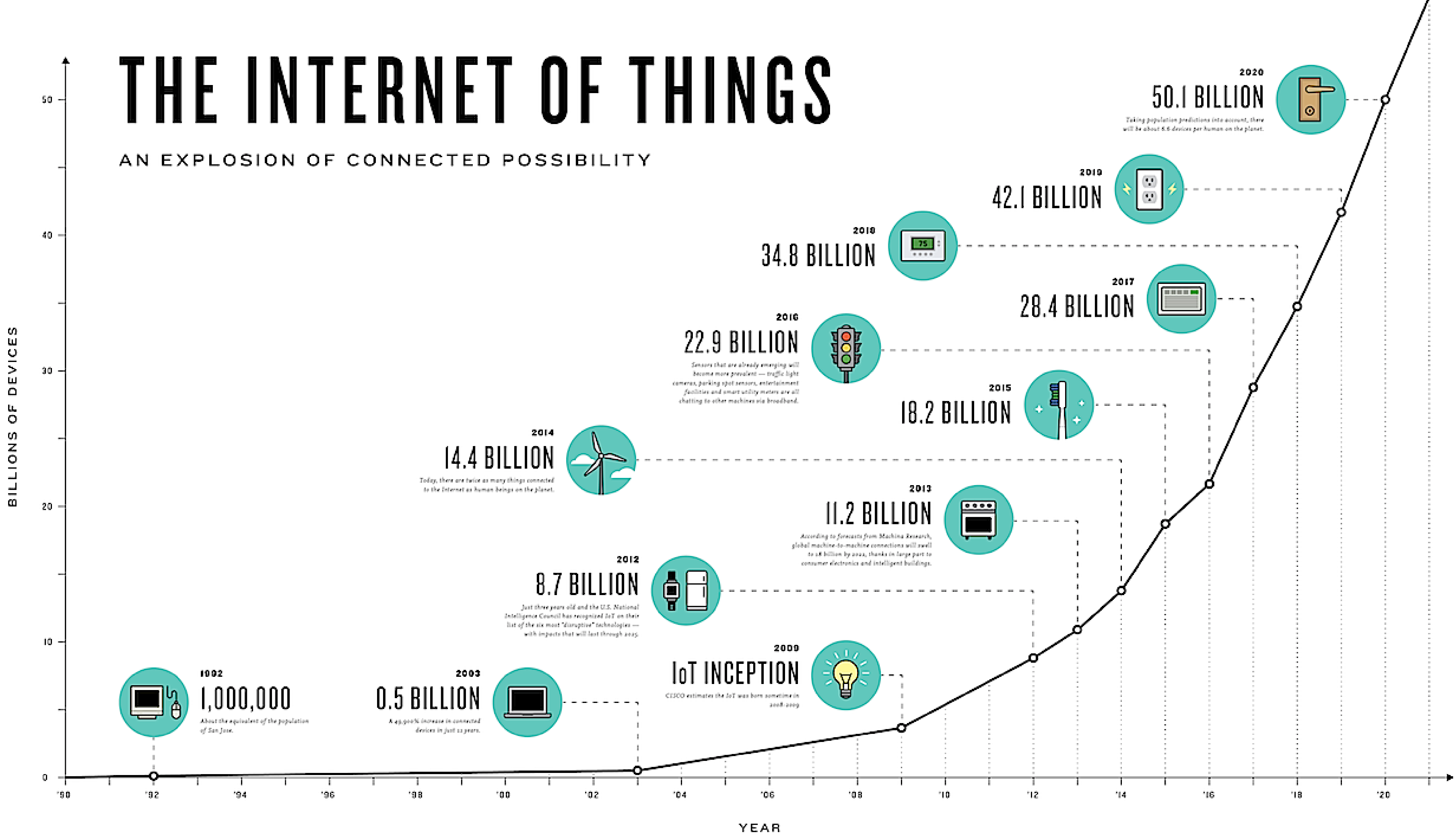


- AI enables security applications
- Security enables better AI
  - Integrity: produces intended/correct results (adversarial machine learning)
  - Confidentiality/Privacy: does not leak users' sensitive data (secure, privacy-preserving machine learning)
  - Preventing misuse of AI

Deep Learning Improving Security Capabilities



DeepFace



THE INTERNET OF THINGS
AN EXPLOSION OF CONNECTED POSSIBILITY
BILLIONS OF DEVICES
0
10
20
30
40
50
'90
'92
'94
'96
'98
'00
'02
'04
'06
'08
'10
'12
'14
'16
'18
'20
YEAR
1992
1,000,000
About the equivalent of the population of San Jose.
2003
0.5 BILLION
2009
IoT INCEPTION
CISCO estimates the IoT was born sometime in 2008-2009
2012
8.7 BILLION
Just three years old and the U.S. National Intelligence Council has recognized IoT on their list of the six most "disruptive" technologies — with impacts that will last through 2025.
2013
11.2 BILLION
2014
14.4 BILLION
Today, there are twice as many things connected to the Internet as human beings on the planet.
2015
18.2 BILLION
2016
22.9 BILLION
Sensors that are already emerging will become more prevalent — traffic light cameras, parking spot sensors, entertainment facilities and smart utility meters are all chatting to other machines via broadband.
2017
28.4 BILLION
2018
34.8 BILLION
75
2019
42.1 BILLION
2020
50.1 BILLION

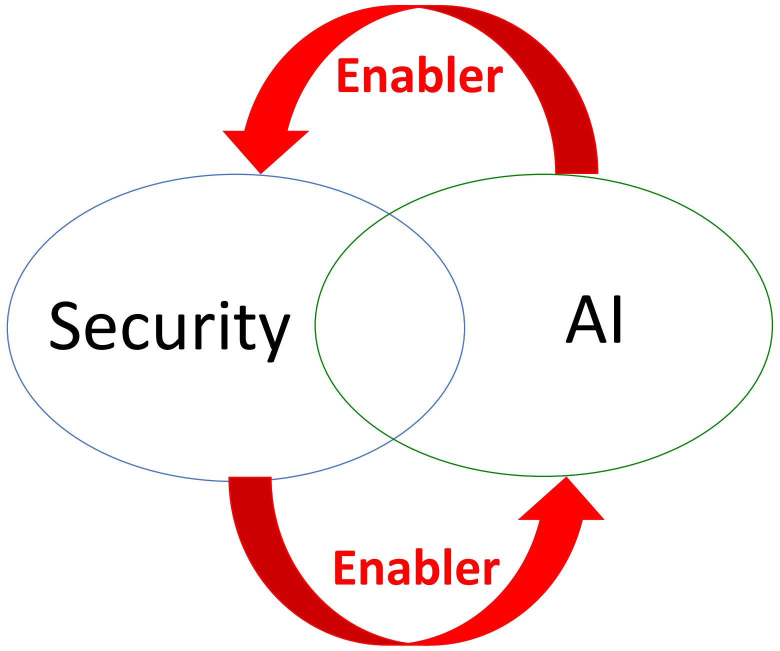
Firmware of IoT devices

- Binary code in various ISA
  - X86, MIPS, ARM, etc.
- Employ common open sourced code:
  - For example: OpenSSL
- Common vulnerability
  - Heartbleed

Deep Learning for IoT Vulnerability Detection

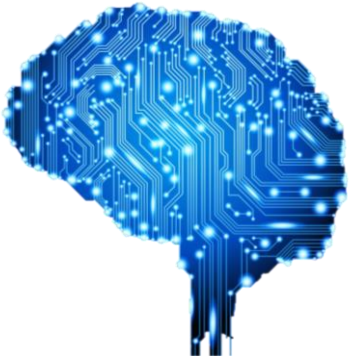
- Neural Network-based Graph Embedding for Cross-Platform Binary Code Search [XLFSSY, ACM Computer and Communication Symposium 2017]
  - See talk by Chang Liu

AI and Security in Cyber Physical Systems



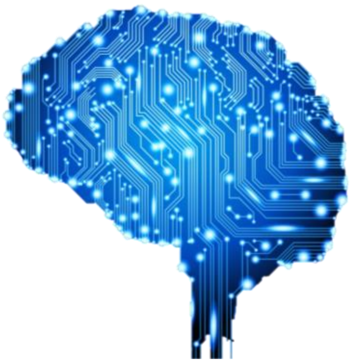
- AI enables security applications
- Security enables better AI
  - Integrity: produces intended/correct results (adversarial machine learning)
  - Confidentiality/Privacy: does not leak users' sensitive data (secure, privacy-preserving machine learning)
  - Preventing misuse of AI

AI and Security: AI in the presence of attacker



AI and Security: AI in the presence of attacker

- Important to consider the presence of attacker
  - History has shown attacker always follows footsteps of new technology development (or sometimes even leads it)
  - The stake is even higher with AI
    - As AI controls more and more systems, attacker will have higher & higher incentives
    - As AI becomes more and more capable, the consequence of misuse by attacker will become more and more severe



AI and Security: AI in the presence of attacker

- Attack AI
  - Cause learning system to not produce intended/correct results
  - Cause learning system to produce targeted outcome designed by attacker
  - Learn sensitive information about individuals
  - Need security in learning systems

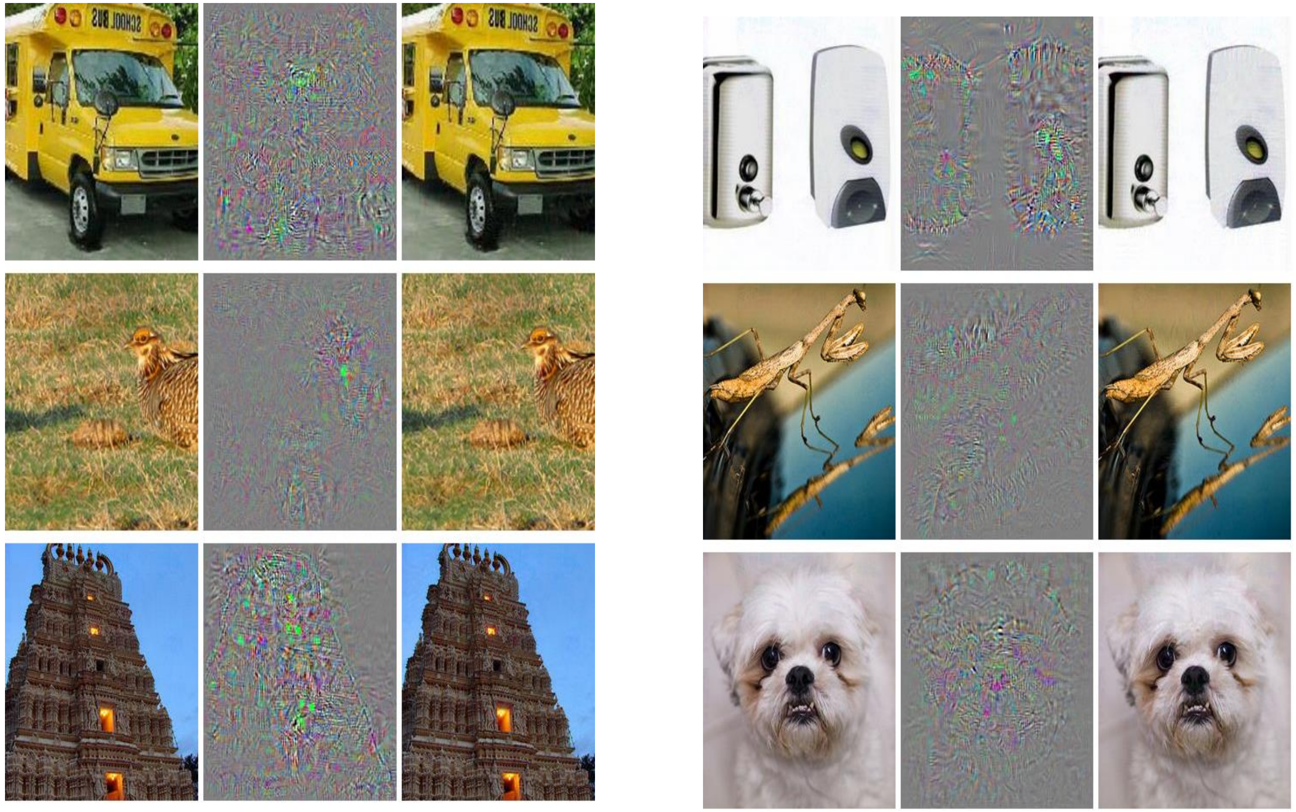
- Misuse AI
  - Misuse AI to attack other systems
    - Find vulnerabilities in other systems
    - Target attacks
    - Devise attacks
  - Need security in other systems

AI and Security: AI in the presence of attacker

- Attack AI:
  - Cause learning system to not produce intended/correct results
  - Cause learning system to produce targeted outcome designed by attacker
  - Learn sensitive information about individuals
  - Need security in learning systems

- Misuse AI
  - Misuse AI to attack other systems
    - Find vulnerabilities in other systems
    - Target attacks
    - Devise attacks
  - Need security in other systems

Deep Learning Systems Are Easily Fooled



ostrich

$$\frac{\partial \text{output}}{\partial \text{pixels}}$$

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. Intriguing properties of neural networks. ICLR 2014.

Adversarial examples fooling autonomous vehicles

Misclassified as
Speed Limit 60

Misclassified as
Speed Limit 75

Misclassified as
Speed Limit 75

Adversarial Examples in Physical World robust against viewpoint changes

Subtle Perturbations

Robust Physical-World Attacks on Machine Learning Models [EEFKLPRS, 2017]

Adversarial Examples in Physical World robust against viewpoint changes

Camouflage Perturbations

Robust Physical-World Attacks on Machine Learning Models [EEFKLPRS, 2017]

Adversarial Examples Prevalent in Deep Learning Systems

- Most existing work on adversarial examples:
  - Image classification task
  - Target model is known

- Our investigation on adversarial examples:

Generative Models

Deep Reinforcement Learning

Image Captioning/ Image-to-code

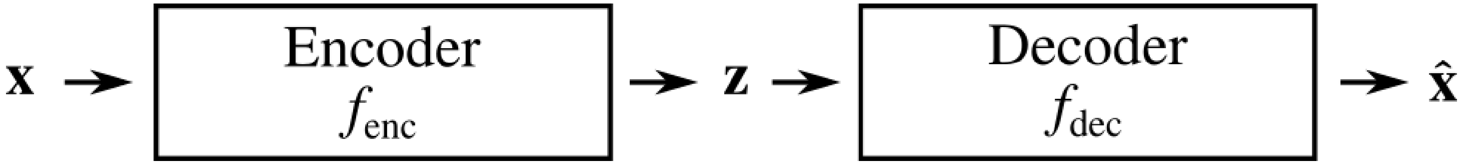
Other tasks and model classes

Blackbox Attacks

Weaker Threat Models
(Target model is unknown)

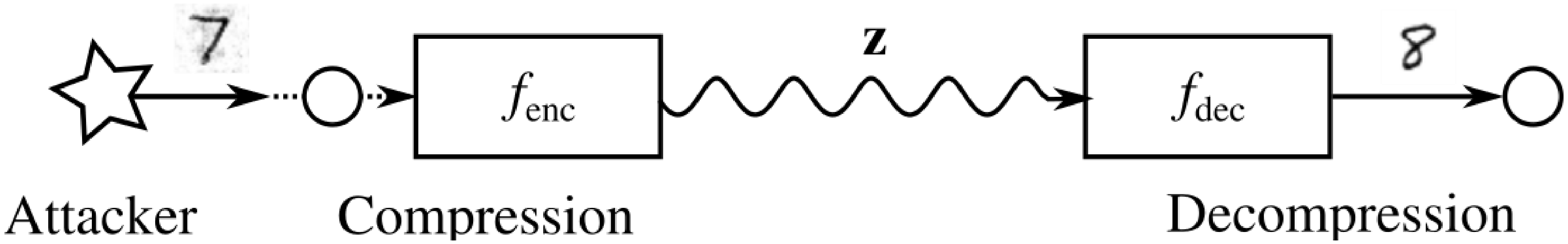
Generative models

- VAE-like models (VAE, VAE-GAN) use an intermediate latent representation
- An **encoder**: maps a high-dimensional input into lower-dimensional latent representation **z**.
- A **decoder:** maps the latent representation back to a high-dimensional reconstruction.



Adversarial Examples in Generative Models

- An example attack scenario:
  - Generative model used as a compression scheme



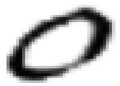
- Attacker's goal: for the decompressor to reconstruct a different image from the one that the compressor sees.

Adversarial Examples for VAE-GAN in MNIST

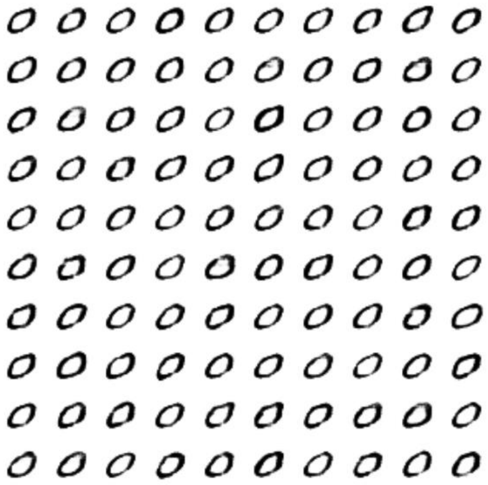
Original images

Reconstruction of original images

Target Image



Adversarial examples

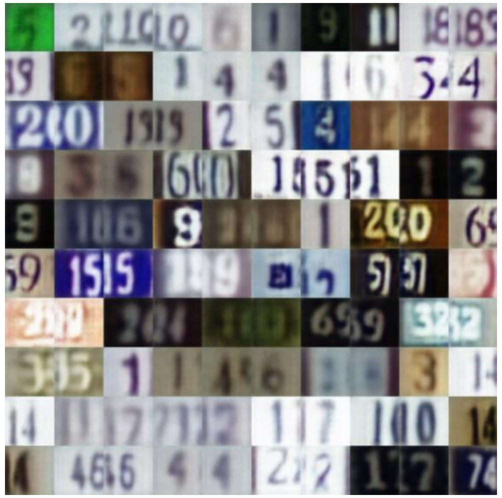


Reconstruction of adversarial examples

Jernej Kos, Ian Fischer, Dawn Song: Adversarial Examples for Generative Models

Adversarial Examples for VAE-GAN in SVHN

Original images



Reconstruction of original images

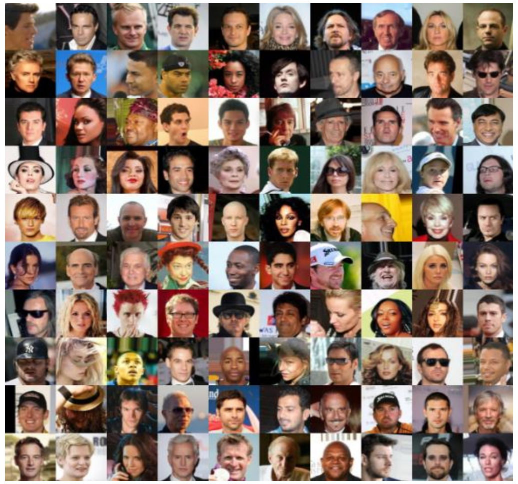
Target Image

Adversarial examples

Reconstruction of adversarial examples

Jernej Kos, Ian Fischer, Dawn Song: Adversarial Examples for Generative Models

Adversarial Examples for VAE-GAN in SVHN

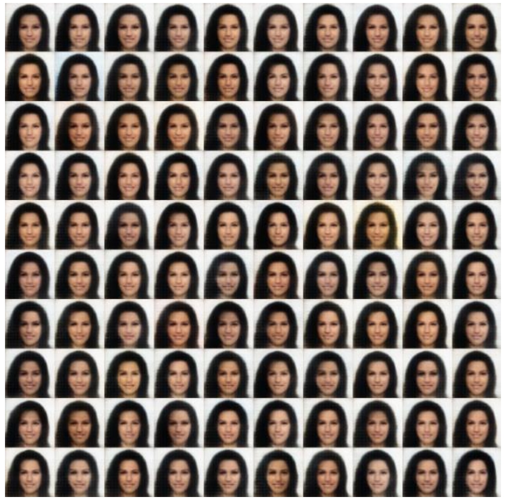


Original images

Reconstruction of original images

Target Image

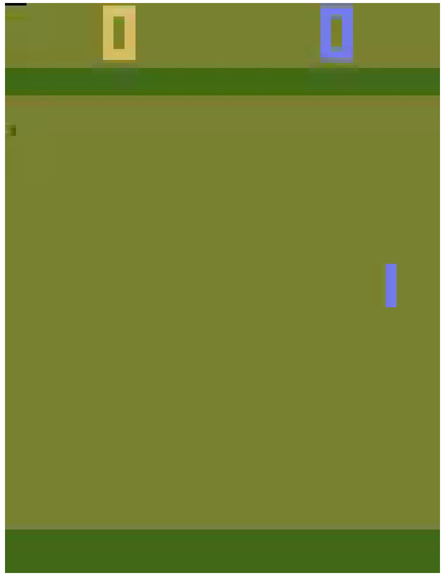
Adversarial examples



Reconstruction of adversarial examples

Jernej Kos, Ian Fischer, Dawn Song: Adversarial Examples for Generative Models

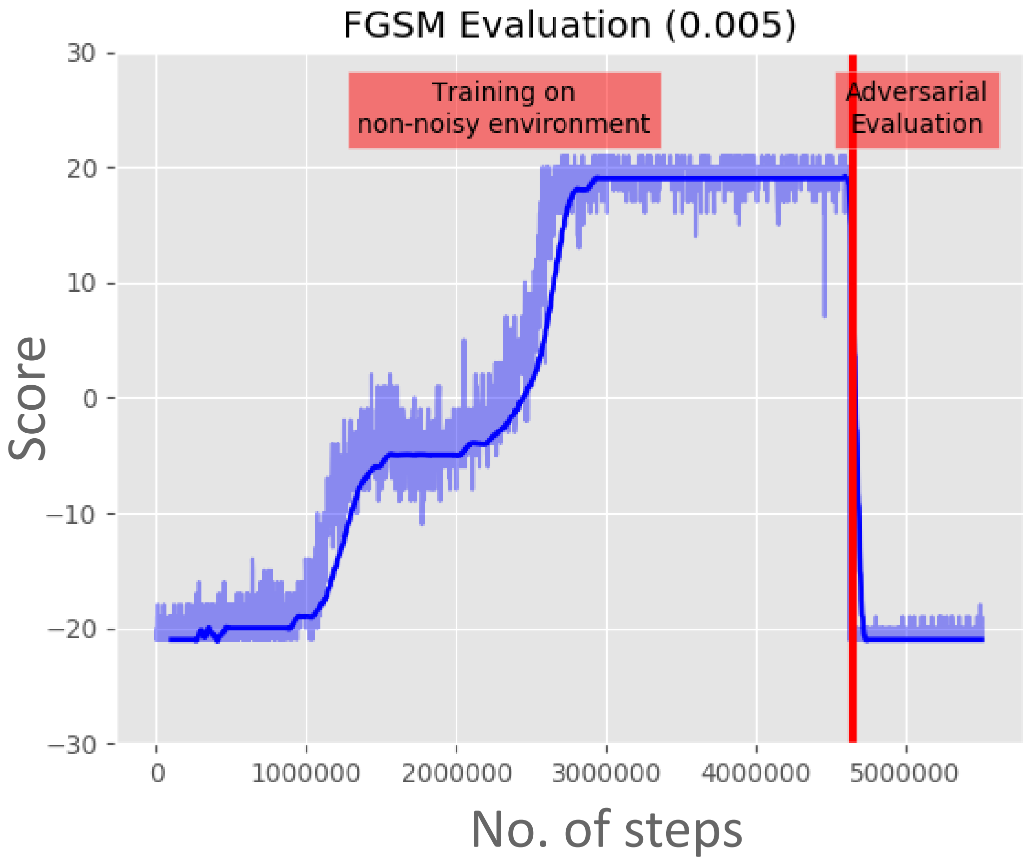
Deep Reinforcement Learning Agent (A3C) Playing Pong



Original Frames

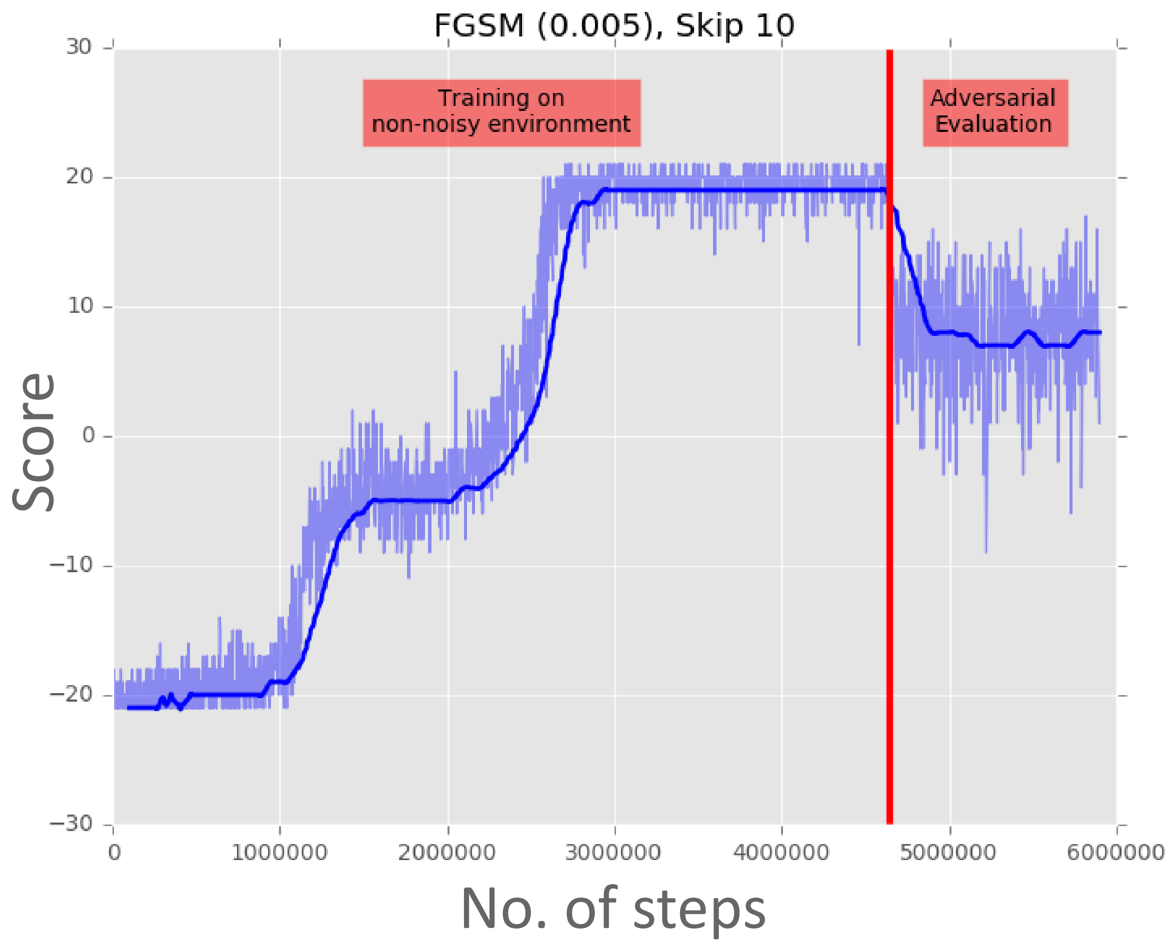
Jernej Kos and Dawn Song: Delving into adversarial attacks on deep policies [ICLR Workshop 2017].

Adversarial Examples on A3C Agent on Pong

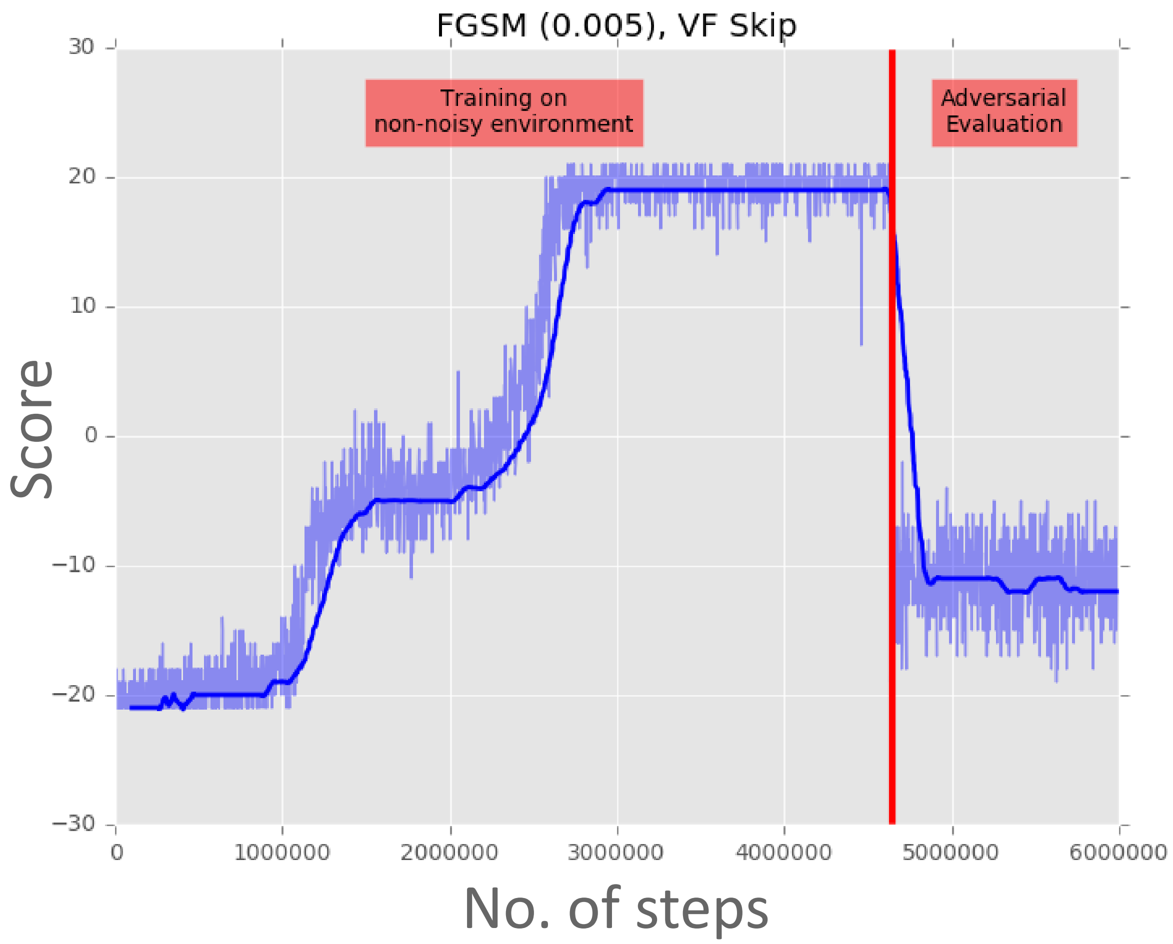


Jernej Kos and Dawn Song: Delving into adversarial attacks on deep policies [ICLR Workshop, 2017]

Attacks Guided by Value Function

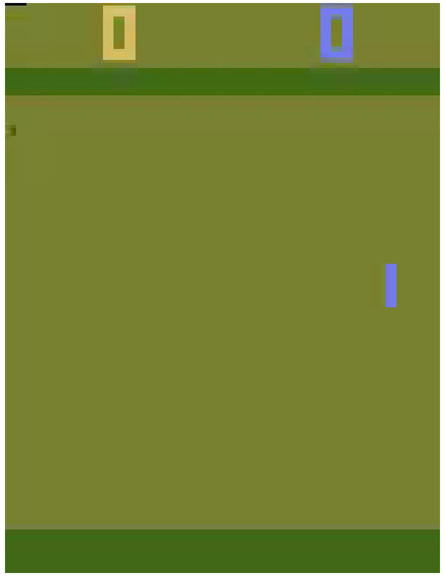


Blindly injecting adversarial perturbations every 10 frames.

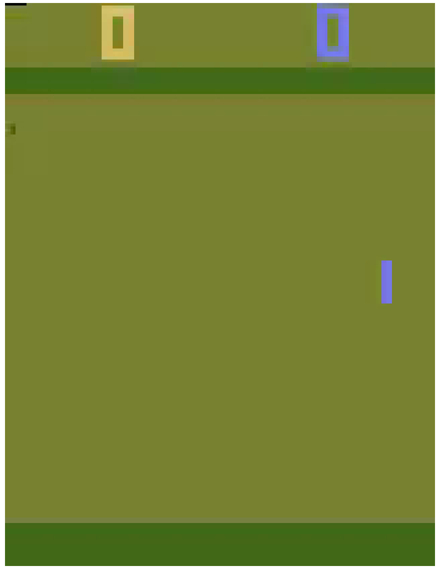


Injecting adversarial perturbations guided by the value function.

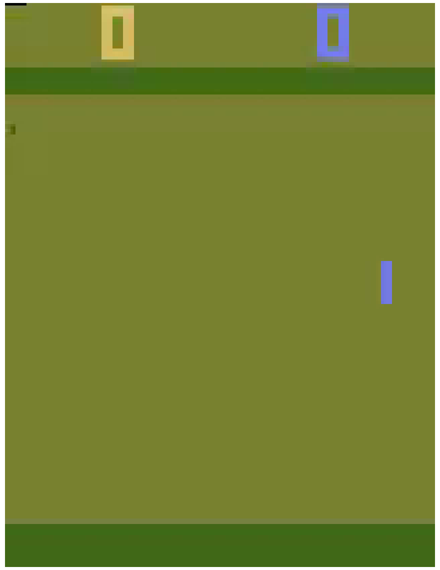
Agent in Action



Original Frames



With FGSM perturbations ($\epsilon = 0.005$) inject in every frame



With FGSM perturbations ($\epsilon = 0.005$) inject based on value function

Jernej Kos and Dawn Song: Delving into adversarial attacks on deep policies [ICLR Workshop 2017].

Adversarial Examples Prevalent in Deep Learning Systems

- Most existing work on adversarial examples:
  - Image classification task
  - Target model is known

- Our investigation on adversarial examples:

Blackbox Attacks

Weaker Threat Models
(Target model is unknown)

Generative Models

Deep Reinforcement Learning

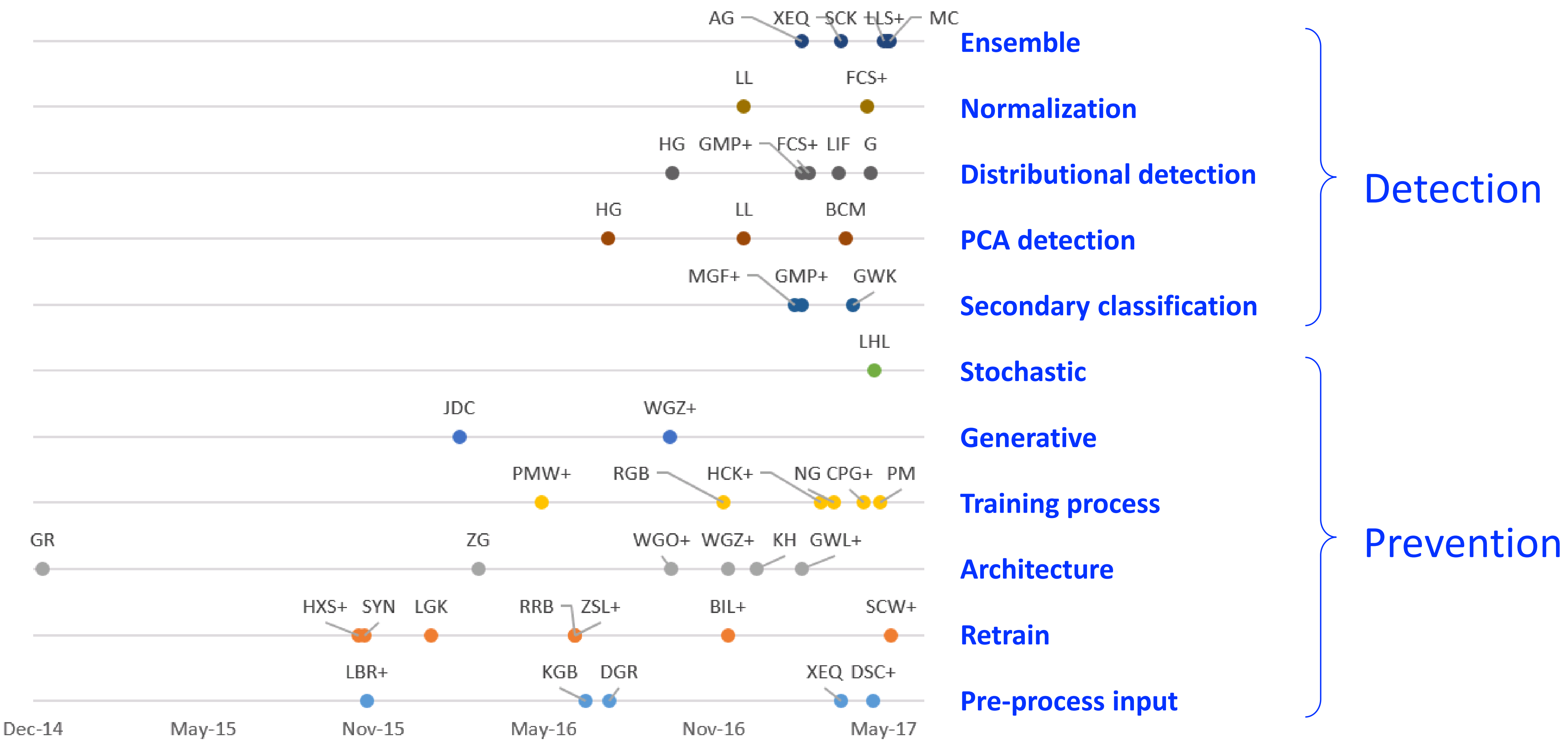
Image Captioning/ Image-to-code

Other tasks and model classes

Numerous Defenses Proposed

- Input processing
  - Gaussian blur, median blur
  - Quantization
- Adversary re-training
  - Re-train with generated adversarial examples
- Detecting adversarial examples
  - Detecting anomalous high-frequency patterns in input
  - Detecting anomalous activations
  - Detecting low confidence output

Numerous Defenses Proposed



No Sufficient Defense Today

- Strong, adaptive attacker can easily evade today's defenses

- Ensemble of weak defenses does not (by default) lead to strong defense
    - Warren He, James Wei, Xinyun Chen, Nicholas Carlini, Dawn Song [WOOT 2017]

Adversarial Machine Learning

- Adversarial machine learning:
  - Learning in the presence of adversaries

- Inference time: adversarial example fools learning system
  - Evasion attacks
    - Evade malware detection; fraud detection

- Training time:
  - Attacker poisons training dataset (e.g., poison labels) to fool learning system to learn wrong model
    - Poisoning attacks: e.g., Microsoft's Tay twitter chatbot
  - Attacker selectively shows learner training data points (even with correct labels) to fool learning system to learn wrong model
  - Data poisoning is particularly challenging with crowd-sourcing & insider attack
  - Difficult to detect when the model has been poisoned

- Adversarial machine learning particularly important for security critical system

Security will be one of the biggest challenges in Deploying AI

Security of Learning Systems

- Software level
- Learning level
- Distributed level

Challenges for Security at Software Level

- No software vulnerabilities (e.g., buffer overflows & access control issues)
    - Attacker can take control over learning systems through exploiting software vulnerabilities

Challenges for Security at Software Level

- No software vulnerabilities (e.g., buffer overflows & access control issues)
- Existing software security/formal verification techniques apply

Reactive Defense

Proactive Defense:
Bug Finding

Proactive Defense:
Secure by Construction

Automatic worm detection
& signature/patch generation

Automatic malware
detection & analysis

Progression of my approach to software security over last 20 years

Security of Learning Systems

- Software level
- Learning level
- Distributed level

Challenges for Security at Learning Level

- Evaluate system under adversarial events, not just normal events

Regression Testing vs. Security Testing in Traditional Software System

	Regression Testing	**Security Testing**
Operation	Run program on **normal** inputs	Run program on **abnormal/adversarial** inputs
Goal	Prevent **normal** users from encountering errors	Prevent **attackers** from finding **exploitable** errors

Regression Testing vs. Security Testing in Learning System

	Regression Testing	**Security Testing**
Training	Train on noisy training data: Estimate resiliency against noisy training inputs	Train on poisoned training data: Estimate resiliency against poisoned training inputs
Testing	Test on **normal** inputs: Estimate generalization error	Test on **abnormal/adversarial** inputs: Estimate resiliency against adversarial inputs

Challenges for Security at Learning Level

- Evaluate system under adversarial events, not just normal events
  - Regression testing vs. security testing
- Reason about complex, non-symbolic programs

Decades of Work on Reasoning about Symbolic Programs

- Symbolic programs:
  - E.g., OS, File system, Compiler, web application, mobile application
  - Semantics defined by logic
  - Decades of techniques & tools developed for logic/symbolic reasoning
    - Theorem provers, SMT solvers
    - Abstract interpretation

Era of Formally Verified Systems

Verified: Micro-kernel, OS, File system, Compiler, Security protocols, Distributed systems

seL4
Security. Performance. Proof.

IronClad/IronFleet

FSCQ

CertiKOS

miTLS/Everest

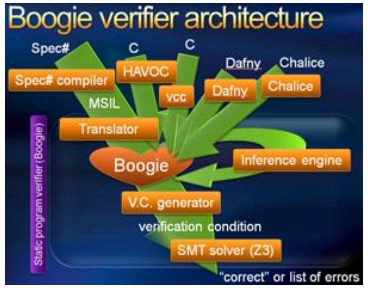
EasyCrypt

CompCert

Powerful Formal Verification Tools + Dedicated Teams

Coq

Why3



Z3

No Sufficient Tools to Reason about Non-Symbolic Programs

- Symbolic programs:
  - Semantics defined by logic
  - Decades of techniques & tools developed for logic/symbolic reasoning
    - Theorem provers, SMT solvers
    - Abstract interpretation

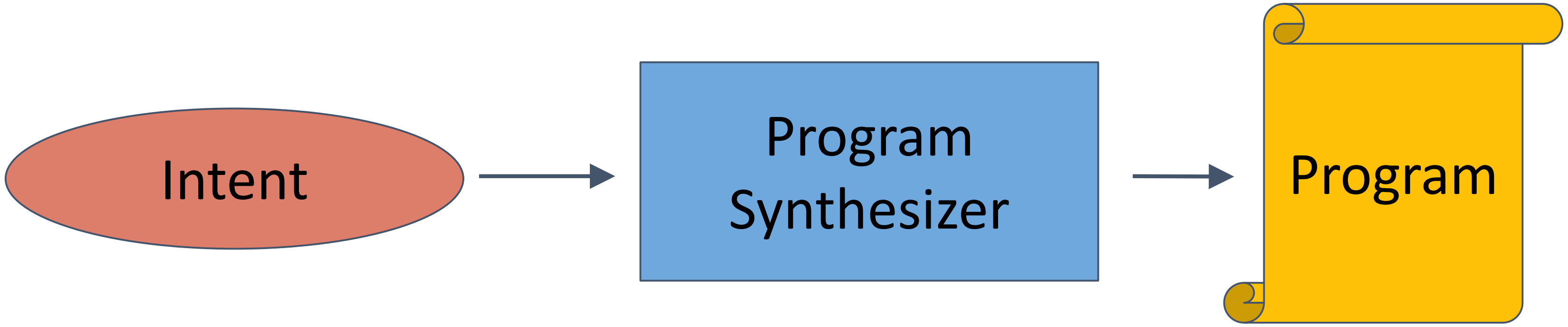
- Non-symbolic programs:
  - No precisely specified properties & goals
  - No good understanding of how learning system works
  - Traditional symbolic reasoning techniques do not apply

Challenges for Security at Learning Level

- Evaluate system under adversarial events, not just normal events
  - Regression testing vs. security testing
- Reason about complex, non-symbolic programs
- Design new architectures & approaches with stronger generalization & security guarantees

Neural Program Synthesis

Can we teach computers to write code?



Example Applications:

- End-user programming
- Performance optimization of code
- Virtual assistant

"Software is eating the world" --- az16

Program synthesis can automate this & democratize idea realization

Neural Program Synthesis

Trainin g data	123 234	452 345	612 367	**Input**
	357	797	979	**Output**

Neural Program Synthesis

Trainin g data

123
234
→ 357

452
345
→ 797

612
367
→ 979

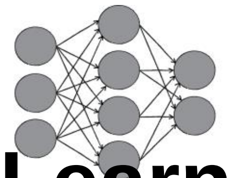
Input

Output

Neural Program Architecture

50
70

Test input

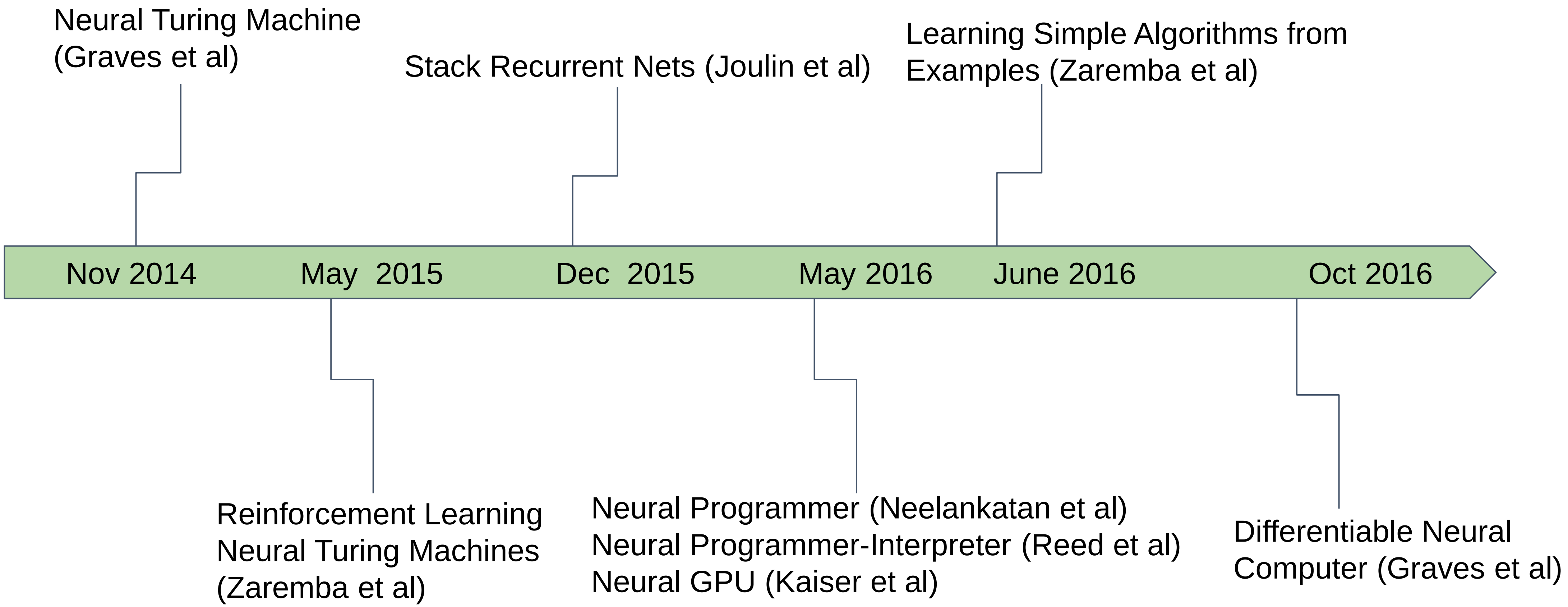


Learned neural program

120

Test output

Neural Program Architectures



Neural Program Synthesis Tasks: Copy, Grade-school addition, Sorting, Shortest Path

Challenge 1: Generalization

Trainin g data

length = 3

123 234	452 345	612 367	**Input**
357	797	979	**Output**

Neural Program Architecture

length = 5

34216
24320

Test input

Learned neural program

543

Test output

Challenge 2: No Proof of Generalization

Trainin g data

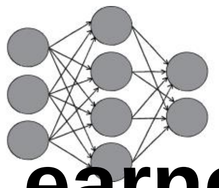
123 234	452 345	length = 3 612 367	**Input**
357	797	979	**Output**

Neural Program Architecture

length = 5

34216
24320

Test input



Learned neural program

58536

Test output

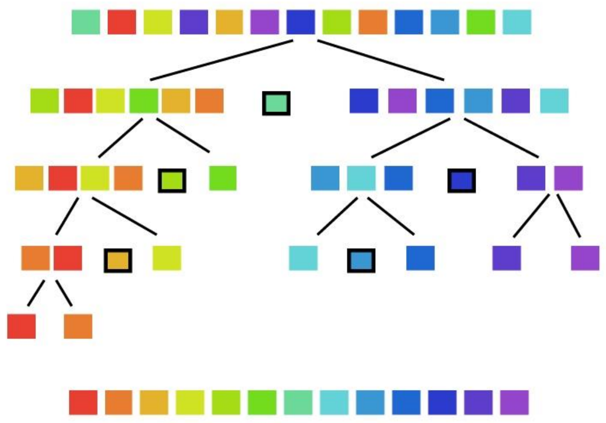
Our Approach: Introduce Recursion

Learn recursive neural programs

Jonathon Cai, Richard Shin, Dawn Song: Making Neural Programming Architectures Generalize via Recursion
[ICLR 2017, **Best Paper Award**]

Recursion

- Fundamental concept in Computer Science and Math
- Solve whole problem by reducing it to smaller subproblems (*reduction rules*)
- *Base cases* (smallest subproblems) are easier to reason about



Quicksort

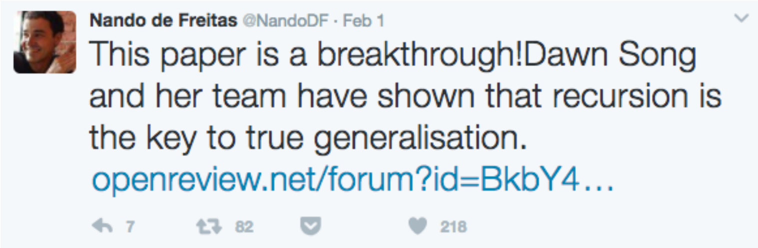
Our Approach: Making Neural Programming Architectures Generalize via Recursion

- **Proof of Generalization**:
  - Recursion enables provable guarantees about neural programs
  - Prove perfect generalization of a learned recursive program via a verification procedure
    - Explicitly testing on all possible base cases and reduction rules (Verification set)

- Learn & generalize faster as well
  - Trained on same data, non-recursive programs do not generalize well

Accuracy on Random Inputs for Quicksort

Length of Array	Non-Recursive	Recursive
3	100%	100%
5	100%	100%
7	100%	100%
11	73.3%	100%
15	60%	100%
20	30%	100%
22	20%	100%
25	3.33%	100%
30	3.33%	100%
70	0%	100%



Jonathon Cai, Richard Shin, Dawn Song: Making Neural Programming Architectures Generalize via Recursion [ICLR 2017, **Best Paper Award**]

Importance of Recursion in Neural Program Architectures

- We introduce recursion, for the first time, into neural program architectures, and learn recursive neural programs

- We address two main challenges using recursion:
  - Generalization to more complex inputs
  - Proof of generalization

Lessons

- Program architecture impacts generalization & provability
- Recursive, modular neural architectures are easier to reason, prove, generalize
- Explore new architectures and approaches enabling strong generalization & security properties for broader tasks

Challenges for Security at Learning Level

- Evaluate system under adversarial events, not just normal events
- Reason about complex, non-symbolic programs
- Design new architectures & approaches with stronger generalization & security guarantees
- Reason about how to compose components

Compositional Reasoning

- Building large, complex systems require compositional reasoning
  - Each component provides abstraction
    - E.g., pre/post conditions
  - Hierarchical, compositional reasoning proves properties of whole system

- How to do abstraction, compositional reasoning for non-symbolic programs?

Security of Learning Systems

- Software level
- Learning level
  - Evaluate system under adversarial events, not just normal events
  - Reason about complex, non-symbolic programs
  - Design new architectures & approaches with stronger generalization & security guarantees
  - Reason about how to compose components
- Distributed level
  - Each agent makes local decisions; how to make good local decisions achieve good global decision?

AI and Security: AI in the presence of attacker

- Attack AI
  - Integrity:
    - Cause learning system to not produce intended/correct results
    - Cause learning system to produce targeted outcome designed by attacker
  - Confidentiality:
    - Learn sensitive information about individuals
  - Need security in learning systems

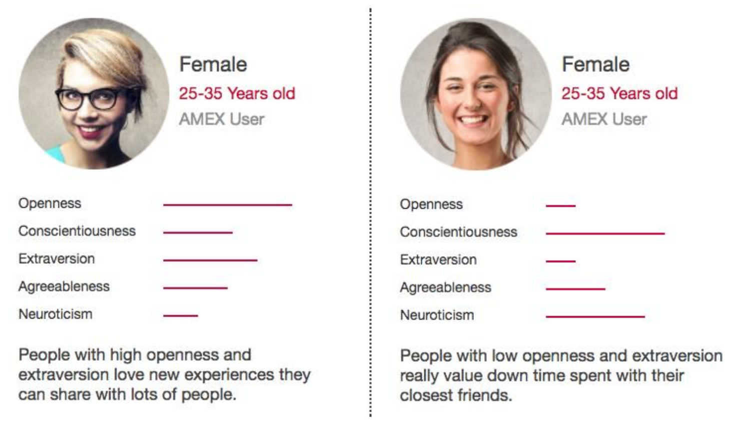
- Misuse AI
  - Misuse AI to attack other systems
    - Find vulnerabilities in other systems
    - Target attacks
    - Devise attacks
  - Need security in other systems

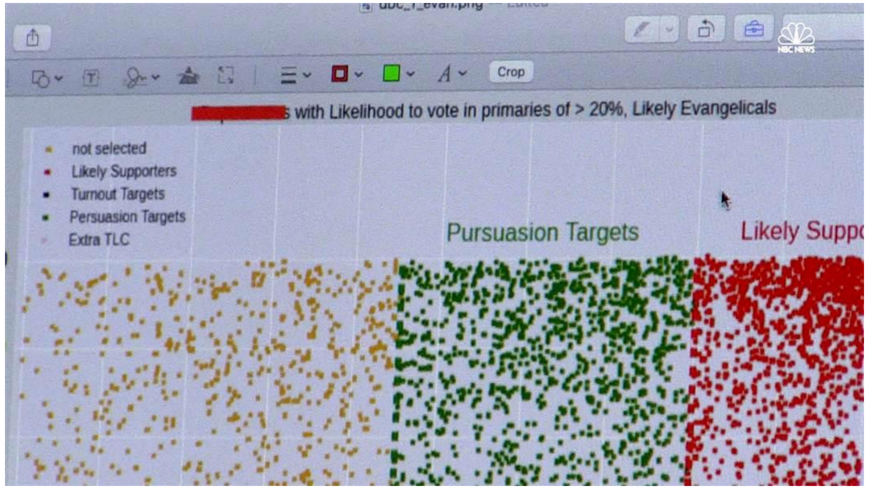
Misused AI can make attacks more effective

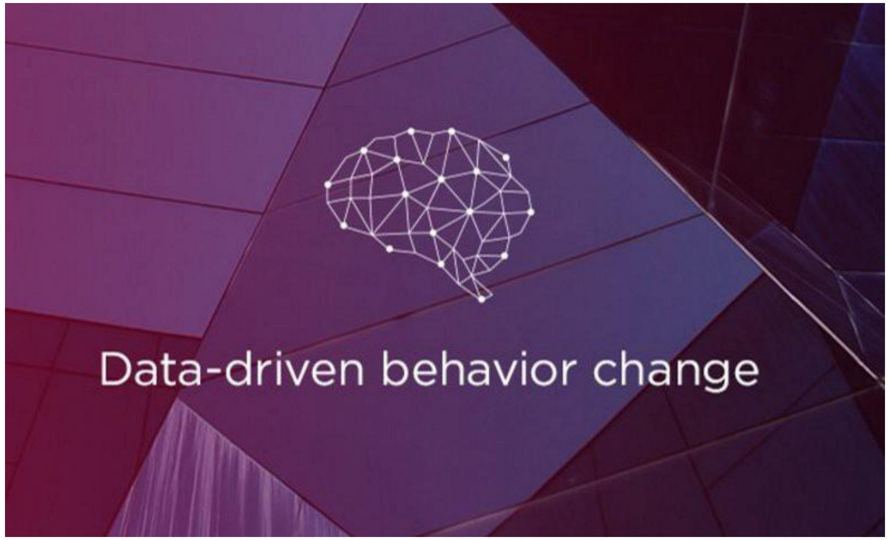
Deep Learning Empowered
Bug Finding

Deep Learning Empowered
Phishing Attacks

Misused AI for large-scale, automated, targeted manipulation







Future of AI and Security

How to better understand what security means for AI, learning systems?

How to detect when a learning system has been fooled/compromised?

How to build better resilient systems with stronger guarantees?

How to build privacy-preserving learning systems?

Security will be one of the biggest challenges in Deploying AI