Synergies between AI and Cryptography

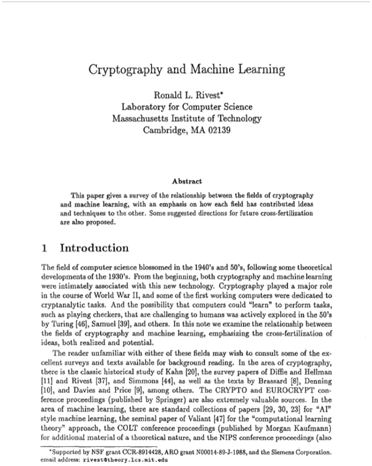
Luca Mariot[1]
https://lucamariot.org

NIST Crypto Reading Club – 2024-01-10

[1] Semantics, Cybersecurity and Services Group, University of Twente, The Netherlands

What does AI have to do with Cryptography?

Machine Learning & Cryptography (Rivest, 1991)



Cryptography and Machine Learning

Ronald L. Rivest*
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

This paper gives a survey of the relationship between the fields of cryptography and machine learning, with an emphasis on how each field has contributed ideas and techniques to the other. Some suggested directions for future cross-fertilization are also proposed.

1 Introduction

The field of computer science blossomed in the 1940's and 50's, following some theoretical developments of the 1930's. From the beginning, both cryptography and machine learning were intimately associated with this new technology. Cryptography played a major role in the course of World War II, and some of the first working computers were dedicated to cryptanalytic tasks. And the possibility that computers could "learn" to perform tasks, such as playing checkers, that are challenging to humans was actively explored in the 50's by Turing [46], Samuel [39], and others. In this note we examine the relationship between the fields of cryptography and machine learning, emphasizing the cross-fertilization of ideas, both realized and potential.

The reader unfamiliar with either of these fields may wish to consult some of the excellent surveys and texts available for background reading. In the area of cryptography, there is the classic historical study of Kahn [20], the survey papers of Diffie and Hellman [11] and Rivest [37], and Simmons [44], as well as the texts by Brassard [8], Denning [10], and Davies and Price [9], among others. The CRYPTO and EUROCRYPT conference proceedings (published by Springer) are also extremely valuable sources. In the area of machine learning, there are standard collections of papers [29, 30, 23] for "AI" style machine learning, the seminal paper of Valiant [47] for the "computational learning theory" approach, the COLT conference proceedings (published by Morgan Kaufmann) for additional material of a theoretical nature, and the NIPS conference proceedings (also

*Supported by NSF grant CCR-8914428, ARO grant N00014-89-J-1988, and the Siemens Corporation.
email address: rivest@theory.lcs.mit.edu

Machine learning and cryptanalysis can be viewed as "Sister fields," since they share many of the same notions and concerns. [...] [2]
Valiant notes that good cryptography can [...] provide examples of classes of functions that are hard to learn.

[2] R. Rivest, Machine Learning and Cryptography. In: ASIACRYPT'91, pp. 427–439

AI Taxonomy

Key remark: AI goes way beyond machine learning!

- Symbolic AI
- Metaheuristics (evolutionary algorithms, ...)
- Natural Computing (cellular automata, ...)
- Statistical and non-statistical learning
- ...

AI has already been used extensively in crypto before the advent of deep learning

Outline

AI for Crypto:

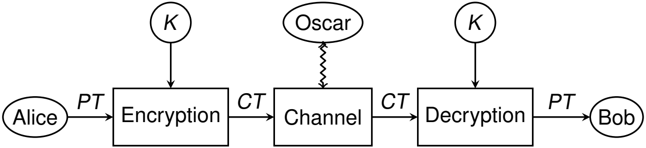
- AI to support the design of cryptographic primitives
- AI to automate the attacks on cryptographic primitives

Crypto for AI:

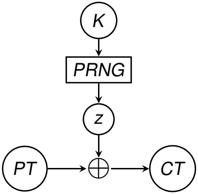
- Use crypto techniques to secure AI models
- Use AI to detect/control AI models

AI for Crypto

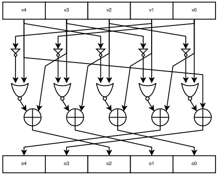
AI Methods for Symmetric Cryptography



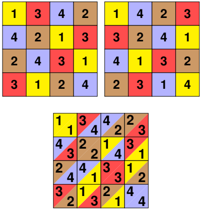
Symmetric ciphers require several low-level primitives, such as:



(a) Pseudorandom Generators



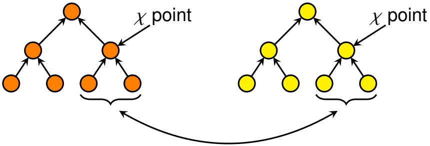
(b) Boolean functions and S-boxes



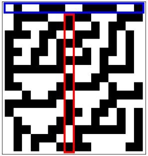
(c) Latin Squares and Orthogonal Arrays

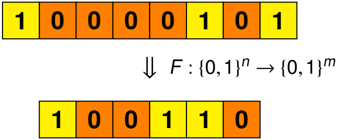
AI approach for symmetric crypto

- "Traditional" approach: ad-hoc and algebraic constructions
- "AI" approach [M22]: support the designer using AI methods
  - Optimization (Evolutionary algorithms, swarm intelligence...)



  - Computational models (cellular automata, neural networks...)





Genetic Algorithms (GA) & Genetic Programming (GP)

- **Black-box optimization** of a fitness function [L15]
- Work on a **coding** of the solutions

- **GA Encoding**: bitstrings

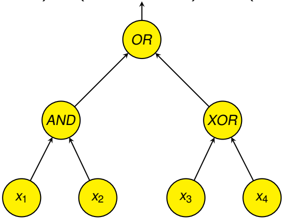
0	**1**	**1**	**1**	**1**	**0**	**0**	**0**

$$\Downarrow$$

$$f(x_1, x_2, x_3) = x_1 \cdot x_2 \oplus x_1 \oplus x_2 \oplus x_3$$

- **GP Encoding**: trees

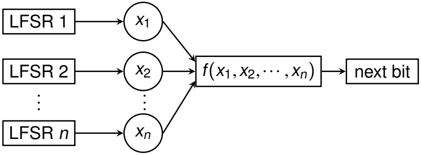
$$f(x_1, x_2, x_3, x_4) = (x_1\ AND\ x2)\ OR\ (x_3\ XOR\ x_4)$$



Use of EA in symmetric cryptography

Design of primitives as **combinatorial optimization problems**, examples [C21, M22]:

- Boolean functions $f : \mathbb{F}_2^n \to \mathbb{F}_2$ for stream ciphers



- S-Boxes $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ for block ciphers

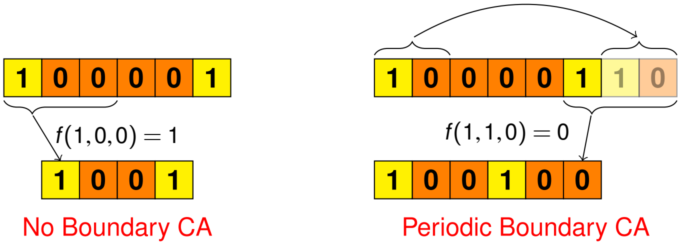
Possible advantages of using EA for this search [**?**, M19b]:

- Diversity of solutions, due to the "blindness" of EA
- Flexibility of EA (optimizing several properties at once

Cellular Automata

- One-dimensional Cellular AutomatA (CA):

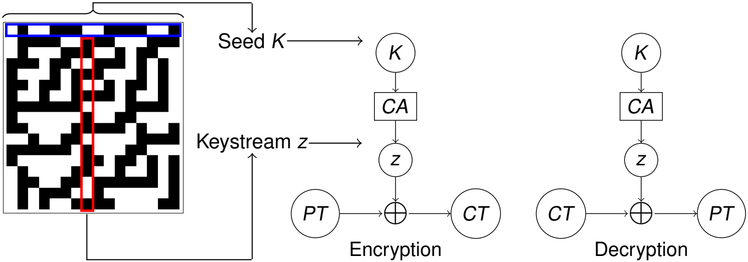
Example: $n = 6$, $d = 3$, $f(s_i, s_{i+1}, s_{i+2}) = s_i \oplus s_{i+1} \oplus s_{i+2}$



- Each cell updates its state $s \in \{0,1\}$ by applying a local rule $f : \{0,1\}^d \rightarrow \{0,1\}$ to itself and the $d-1$ cells on its right

Cellular Automata and Cryptography

Goal: investigate how CA can be used in the design of cryptographic primitives [W86, L13]

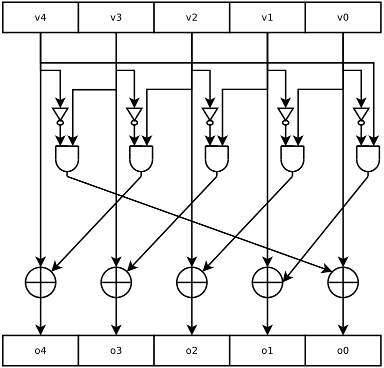


Why CA?

1. **Security from Complexity**
2. **Efficient Implementation**

Real world CA-Based Crypto: KECCAK χ S-box

- Local rule: $\chi(x_1, x_2, x_3) = x_1 \oplus (1 \oplus (x_2 \cdot x_3))$ (rule 210)
- Invertible for every odd size n of the CA

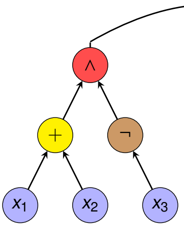


- Used as a PBCA with $n = 5$ in KECCAK [B11]

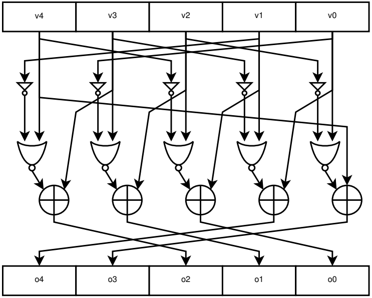
CA S-boxes found by GP

Idea: evolve a CA rule that defines an S-box, optimizing:

- **crypto** properties (nonlinearity, differential uniformity) [M19a]
- **implementation** properties (area, latency)

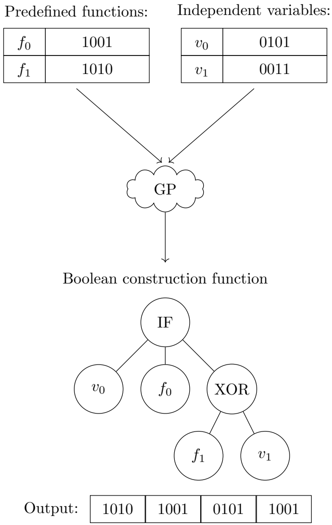


x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0



- Up to size 7×7: results on par or slightly better than the state of the art (Keccak, PRESENT, Piccolo, ...) [P17]

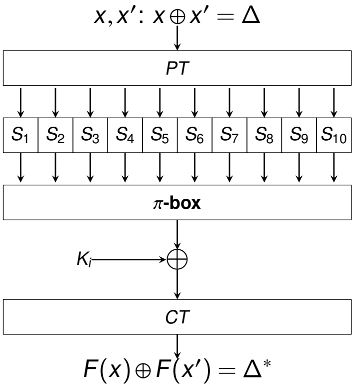
Evolving Constructions of Boolean functions with GP



- **Idea:** Do not evolve primitives directly, but rather their mathematical constructions [C22]
- Use Boolean minimizers to interpret the constructions
- **Research Question**: Does GP obtain previously known constructions or new ones?

Differential Cryptanalysis

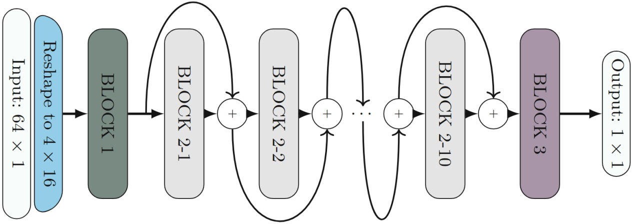
- **Idea**: chosen plaintext attack, see how differences propagate to the ciphertext



- **Goal**: Compute differential probability of $\Delta \to \Delta^*$
- **Distinguishing attack**: given (x, x'), classify if it is a *random* or *real* pair
- **Tool**: Difference Distribution Table (DDT)

Deep learning-based differential distinguishers

- A. Gohr (CRYPTO 2019): train a CNN as a differential distinguisher
- Better accuracy than pure distinguishers on SPECK32/64



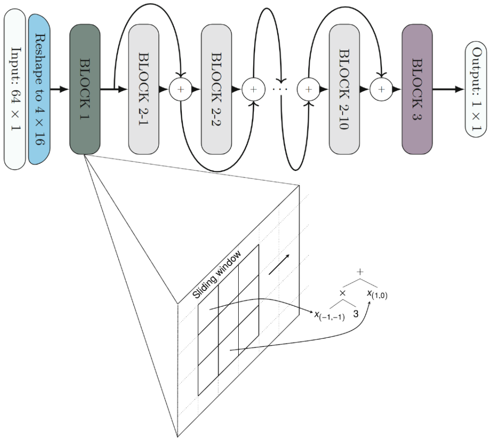
[3]

- **Problem**: learned models are hardly interpretable!

[3] Image credits: A. Benamira et al., *A Deeper Look at Machine Learning-Based Cryptanalysis*, EUROCRYPT 2021

Open problem: interpretable AI-based distinguishers

- **Idea**: Replace convolutional layers with convolutional GP [J21]



- **Research Question**: Is "convolutional" GP able to reach CNN performances, and yield models easier to interpret?

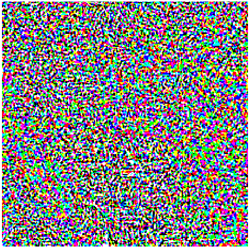
Crypto for AI

Adversarial Examples in DNN

- DNN known to be vulnerable to **adversarial examples** (AE)
- **Idea**: perturb a valid example to mess the DNN's classification

Classification: Panda

+



Noise perturbation

$\Rightarrow$

Classification: Gibbon [4]

- Perturbation moves the example beyond the *decision boundary* of a DNN

[4] Example credits: I.J. Goodfellow, J. Shlens, C. Szegedy, *Explaining and Harnessing Adversarial Examples*, ICLR 2015

Evolutionary Construction of AE

- Perturbations for AE can be **minimal**
- **One-pixel attack**: Modify just one pixel in a valid example

[5]

- Pixel selection done with **Evolutionary Algorithms**

[5] Image credit: J. Su et al., *One Pixel Attack for Fooling Deep Neural Networks*. IEEE Trans. Evol. Comput 23(5):828-840 (2019)

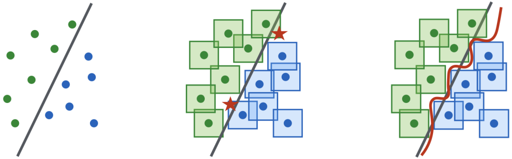
Defenses

Why do we want Adversarial robust networs?

- Better accuracy.
- Better explanation of the behavior of networks.

Adversarial Robustness:

- Separating the l_∞-balls requires a significantly more complicated decision boundary.



Type of Defenses

- Adversarial training
- Network Pruning
- Random input transformation
- **Certified Robustness**

Certified Robustness

- Most defenses are *empirical*.
- Certified robustness provides *theoretical guarantees*.

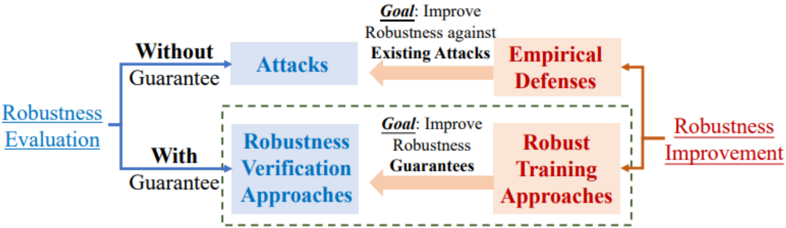
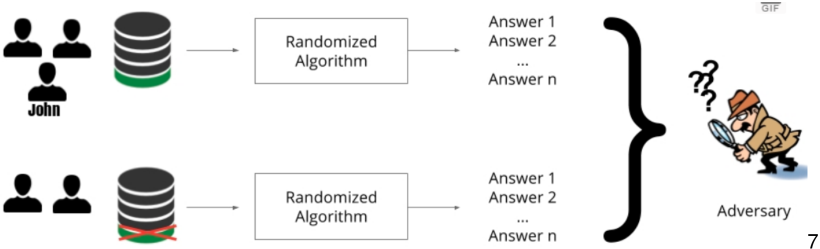


Figure: Empirical vs. certified robustness.[6]

[6] Li Linyi et al. "Sok: Certified robustness for deep neural networks." arXiv preprint arXiv:2009.04131 (2020).

- **Idea:** anonymize the *query mechanism*, rather than the database itself



[7]

- **Key property**: an adversary has a negligible probability of distinguishing two DBs differing in only *one row*

[7] Image credits: N. Papernot, I. Goodfellow, Privacy and machine learning: two unexpected allies?

Differential Privacy

Ingredients:

- Randomized algorithm A
- Database D
- Output space O

Definition: Differential Privacy

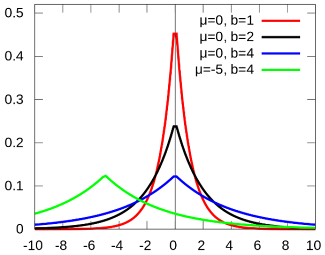
A is (ϵ,δ)-DP wrt a metric ρ on D if for any D' such that $\rho(D,D') \leq 1$ and $S \subseteq O$, it holds:

$$P(A(D) \in S) \leq e^{\epsilon} P(A(D') \in S) + \delta \ .$$

- ϵ,δ: privacy strength parameters (small)
- ρ: usually the *Hamming distance*

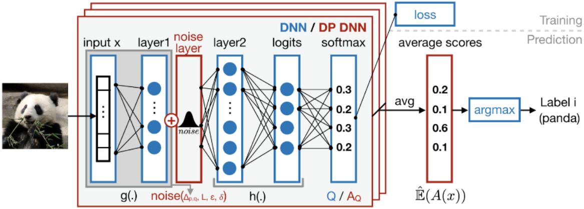
Differential Privacy

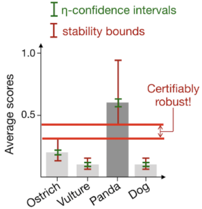
- How is A implemented?
- Addition of *noise* drawn from specific distribution
- Usual choice: *Laplace noise* $L(\mu, b)$



PixelDP Architecture (Lecuyér et al. 2019)

- **Trick**: input image x is a "DB", where each row is e.g. a pixel
- **Randomized** A: output scores $(y_1(x), \ldots, y_k(x))$ (e.g. given by an activation function like SoftMax)
- Noise added after the *first layer* at inference time





[8]

[8] M. Lecuyér et al.: Certified Robustness to Adversarial Examples with Differential Privacy. IEEE S&P 2019

Conclusions

Summary

Where we arrived so far:

- AI methods have extensively been used in crypto, both for design and analysis of primitives
- Cryptographic-like techniques can help in making AI models more robust

Looking at the future:

- Plenty of open problems for the "Crypto for AI" direction!
- Statistical watermarking of LLMs (Aaronson, 2023)
- Cryptographic backdoors in NN (Goldwasser et al., 2022)

Thank you!

https://aicrypt2024.aisylab.com/

References

- [B11] G. Bertoni, J. Daemen, M. Peeters, G. Van Assche: The Keccak reference. (January 2011)
- [C21] C. Carlet: Boolean functions for cryptography and coding theory. Cambridge University Press (2021)
- [C22] C. Carlet, M. Djurasevic, D. Jakobovic, L. Mariot, S. Picek: Evolving constructions for balanced, highly nonlinear boolean functions. Proceedings of GECCO 2022, pp. 1147-1155 (2022)
- [J21] D. Jakobovic, L. Manzoni, L. Mariot, S. Picek, M. Castelli: CoInGP: convolutional inpainting with genetic programming. Proceedings of GECCO 2021, pp. 795-803 (2021)
- [L13] A. Leporati and L. Mariot: 1-Resiliency of Bipermutive Cellular Automata Rules. Proceedings of Automata 2013, pp. 110-123 (2013)
- [L15] S. Luke. Essentials of Metaheuristics. Lulu, 2015. 2nd ed.
- [M22] L. Mariot, D. Jakobovic, T. Bäck, J. Hernandez-Castro: Artificial Intelligence for the Design of Symmetric Cryptographic Primitives. Security and Artificial Intelligence 2022, pp. 3-24 (2022)
- [M19a] L. Mariot, S. Picek, A. Leporati, and D. Jakobovic. Cellular automata based S-boxes. Cryptography and Communications 11(1):41–62 (2019)
- [M19b] L. Mariot, D. Jakobovic, A. Leporati, S. Picek: Hyper-bent Boolean Functions and Evolutionary Algorithms. Proceedings of EuroGP 2019, pp. 262-277 (2019)
- [P17] S. Picek, L. Mariot, B. Yang, D. Jakobovic, N. Mentens: Design of S-boxes defined with cellular automata rules. Conf. Computing Frontiers 2017: 409-414 (2017)
- [W86] S. Wolfram. Cryptography with cellular automata. In CRYPTO '85, pp. 429–432 (1986)