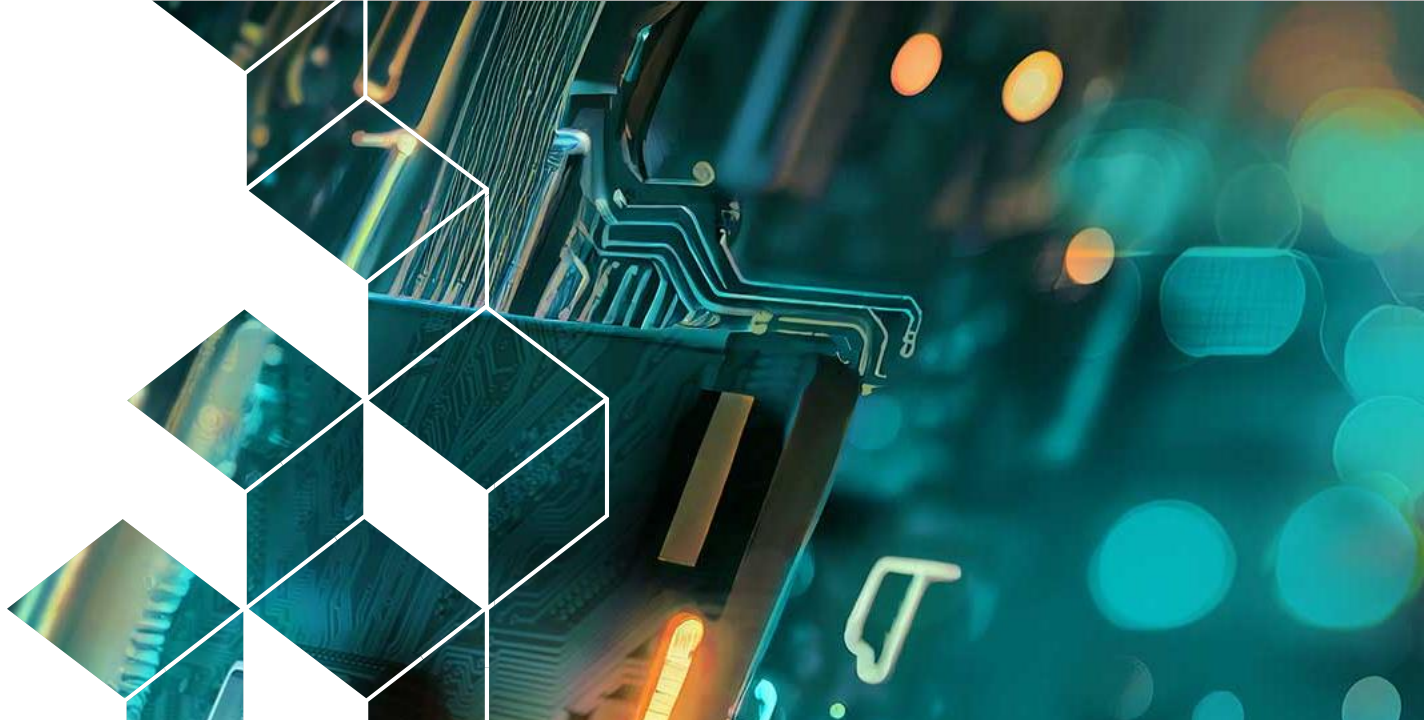


Fault Injection and Embedded Neural Networks: Models Integrity and Confidentiality

Pierre-Alain Moëllic

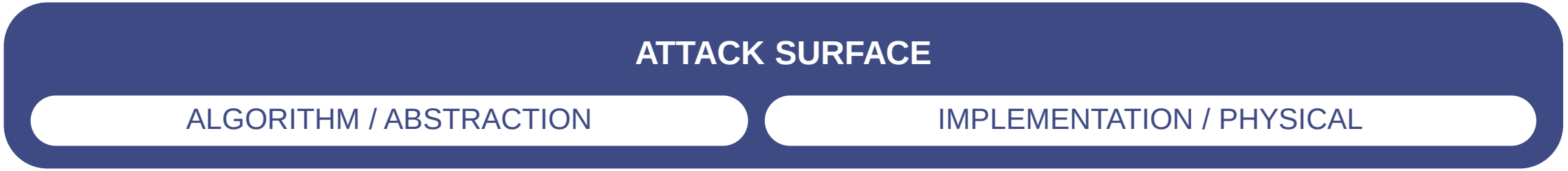
CEA LETI / Mines Saint-Etienne

Context: Security of Machine Learning

SotA: a decade of experience

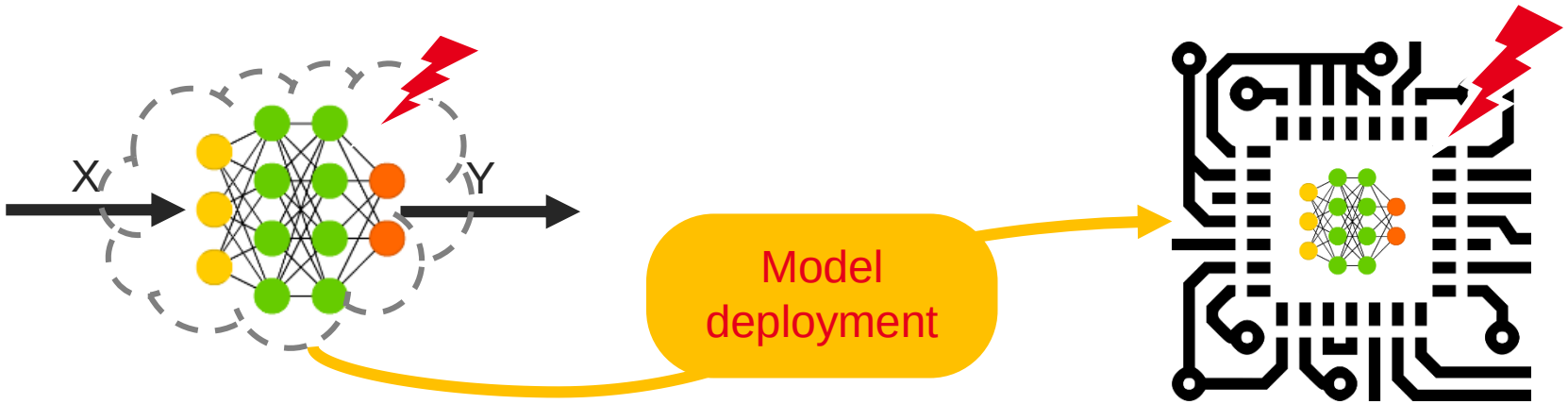
➔ mainly API-based attacks ➔ Confidentiality & Privacy / Integrity / Availability ➔ Model / Data

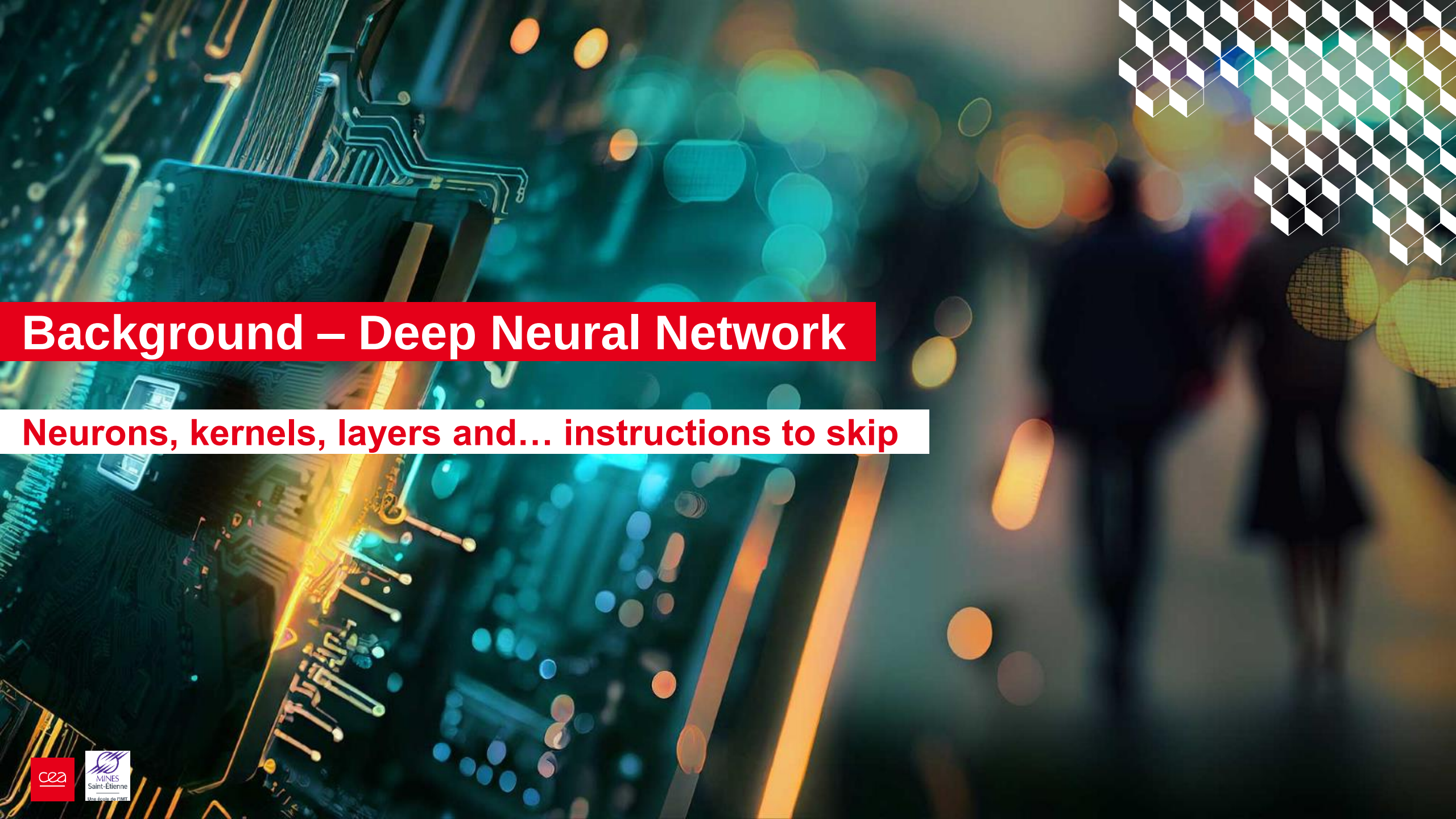
ATTACK SURFACE **A model is not just an abstraction**



API-based Attacks
White-Box / Black-Box

Implementation-based Attacks
Physical Attacks (side-channel, **fault injection**)



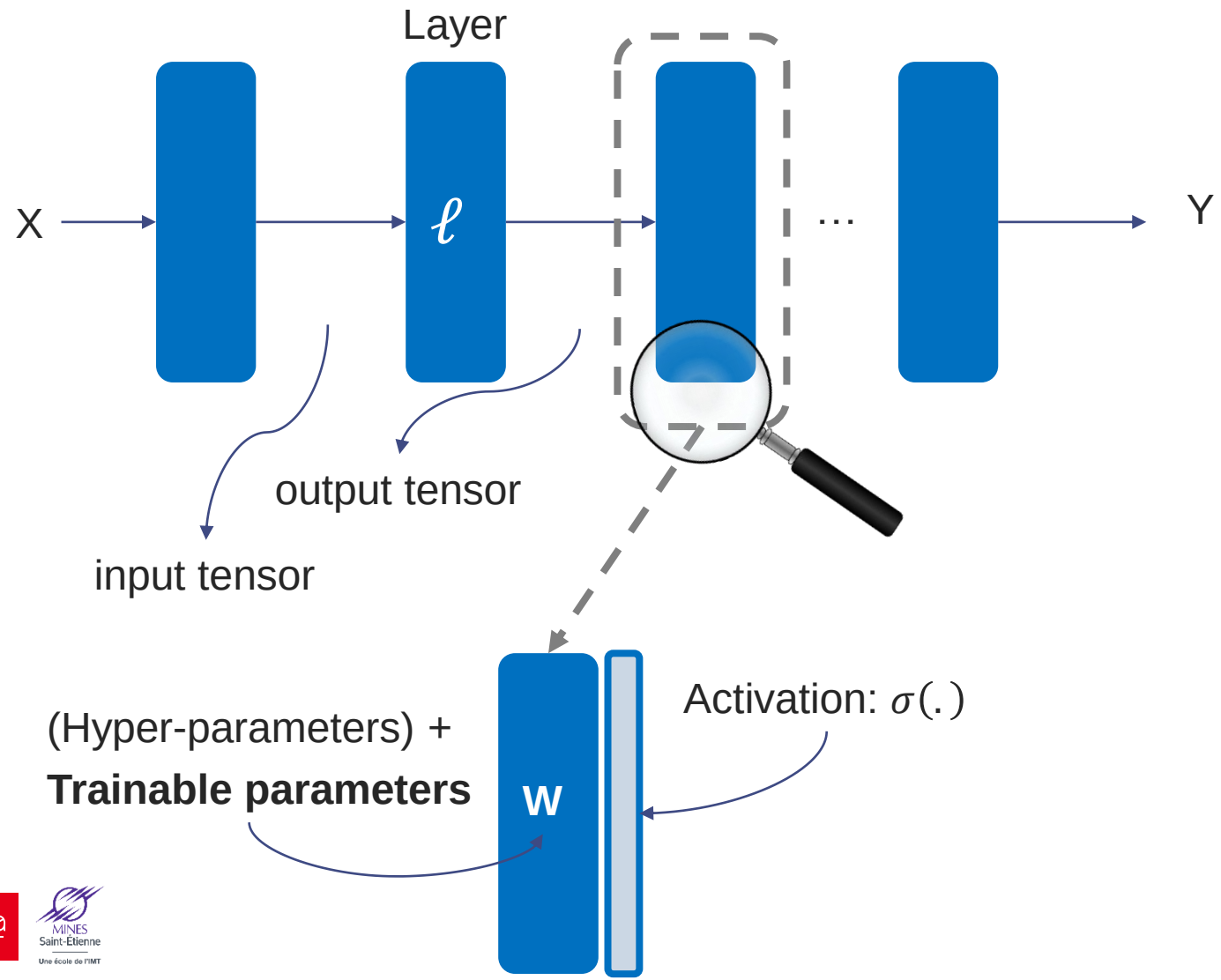


Background – Deep Neural Network

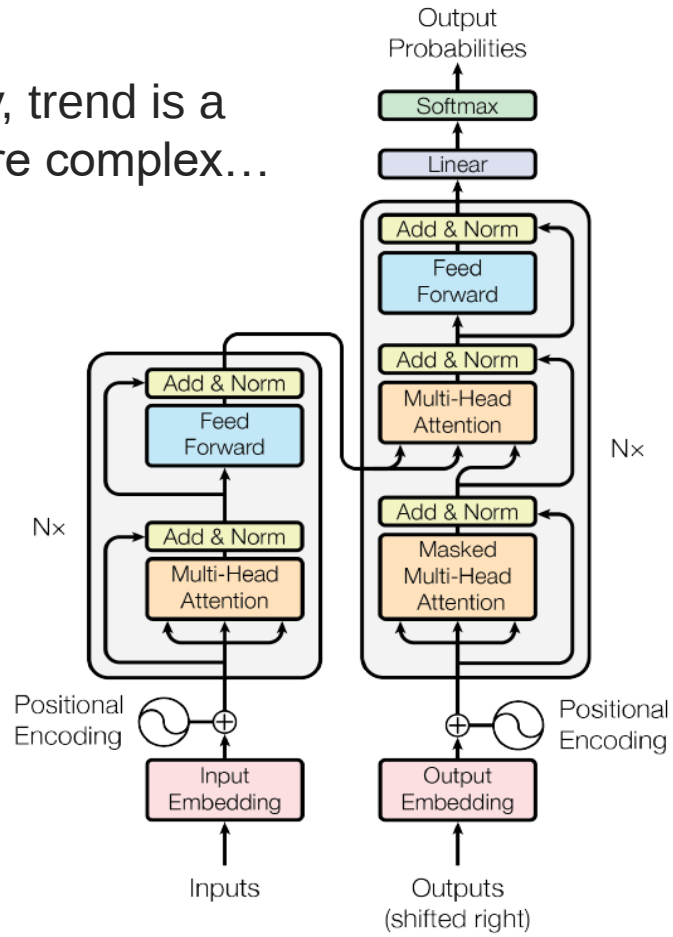
Neurons, kernels, layers and… instructions to skip

Deep Neural Network

Feedforward models

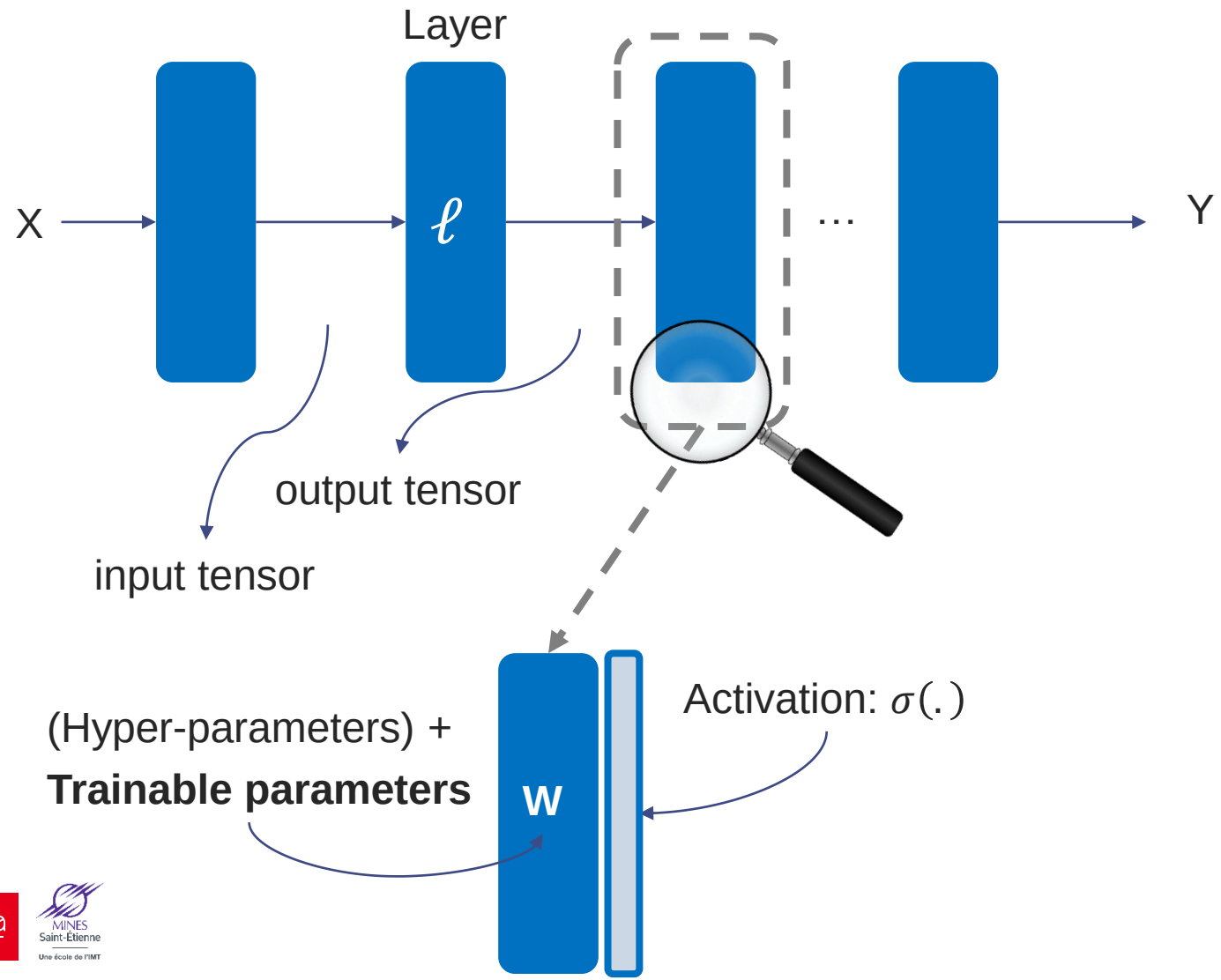


NB actually, trend is a little bit more complex…



Deep Neural Network

Feedforward models



Convolutional layer

Algorithm 1 Convolution layer (K kernels)

Input: Tensor X of size $H \times H \times C$, parameters tensor Θ of size $Z \times Z \times C \times K$, bias tensor of size K

Output: Tensor Y of size $H \times H \times K$

```
for k in [1, K] do
    for x in [1, H] do
        for y in [1, H] do
            Y_{x,y,c} = B_k
            for m in [1, Z] do
                for n in [1, Z] do
                    for c in [1, C] do
                        Y_{i,j,c} += θ_{m,n,k,c} · X_{x+m,y+n,k}
return Y
```

Fully-Connected layer

$$a_j^l(x) = \sigma\Big(\sum_{i \in (l-1)} \theta_{i,j} a_i^{l-1} + b_j \Big)$$

Instruction Skip

Impact of a single instruction skip

- **Very first experimental study[1] from C. Gaine (ANR PICTURE)**
  - Where to start?
  - Critical attack paths? Adversarial goal?

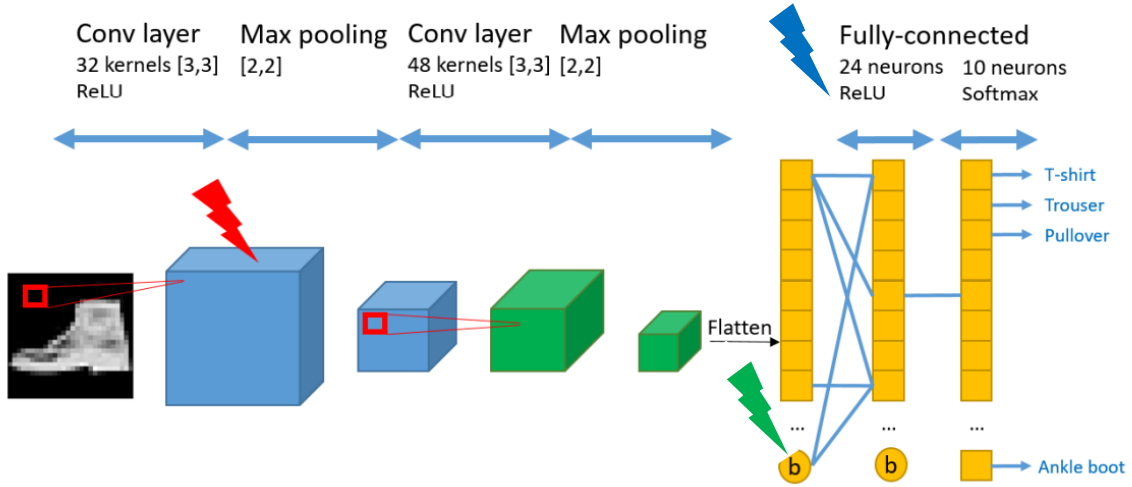
- 2 injection means: Laser / EM pulse
- Cortex-M4 platform
- Inference of a standard CNN (trained on FashionMNIST)

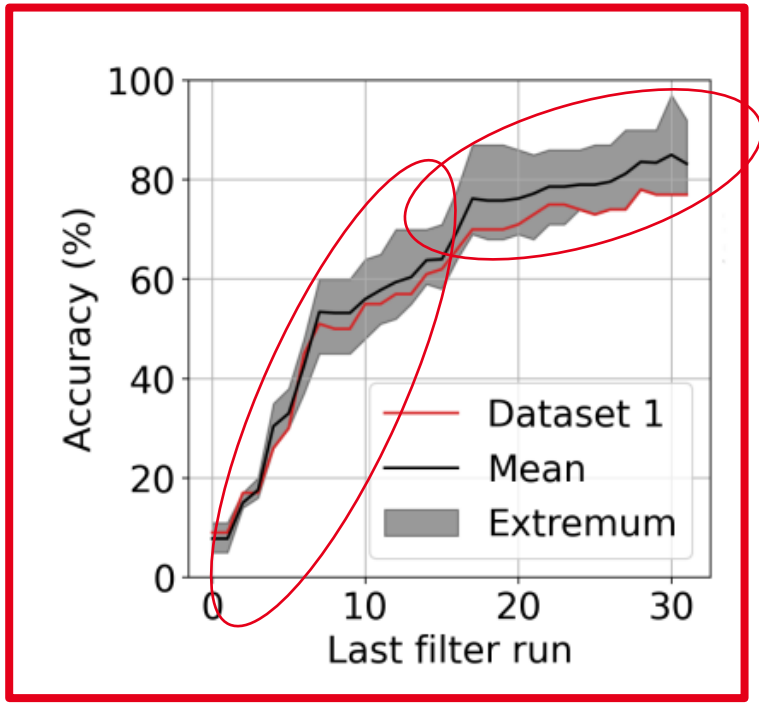
- **Impact of a single instruction skip**
  - ➔ convolutional layer
  - ➔ bias addition
  - ➔ activation function

(1) C. Gaine, et al. Fault Injection on Embedded Neural Networks: impact of a single instruction skip, DSD 2023

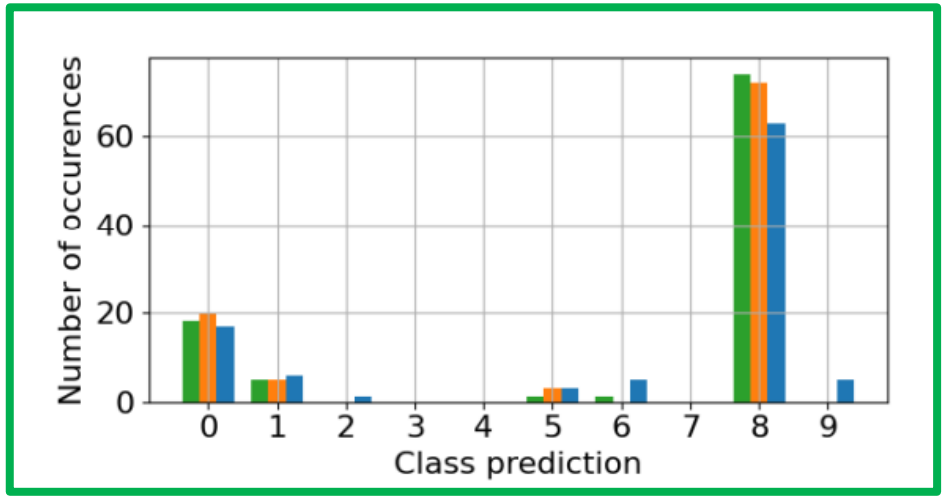
Instruction Skip

Impact of a single instruction skip

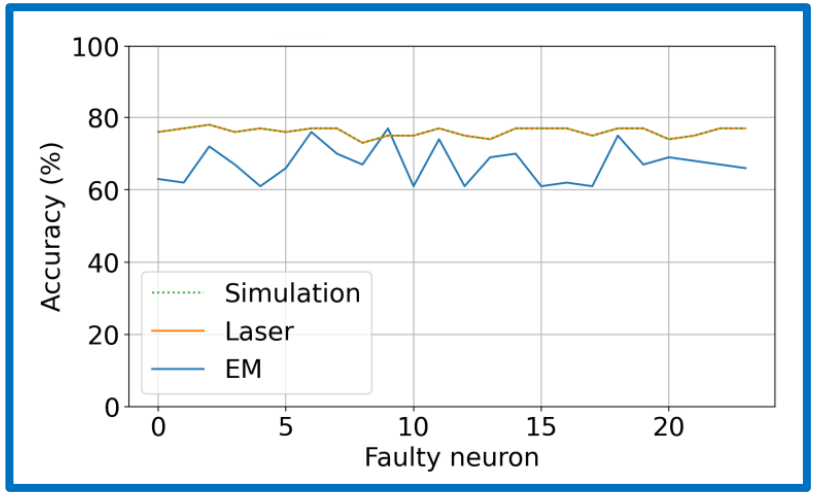




+ memory effect



Label-stuck prediction



Ineffective (on average)

…but is it really unexploitable? that is the question…

(1) C. Gaine, et al. Fault Injection on Embedded Neural Networks: impact of a single instruction skip, DSD 2023

Parameters-based Adversarial Attacks

Some results about the Bit-Flip Attack (BFA)

Weight-based Adversarial Attacks

Target internal parameters stored in memory

- Main reference: Bit-Flip Attack – **BFA**[1]
  - Theoretical "*demonstration*" on SotA CNN
  - First practical demonstration: RowHammer[2] attack (CPU, **DRAM**)
  - Former works[3] on evaluating BFA

- Random bit-flips ➔ many safety analysis
- Highly recommended works from D. Stutz[4]: Safety←→Security

(1) Rakin et al., *Bit-flip attack: Crushing neural network…* IEEE ICCV 2019
(2) (2) Yao, et al. *DeepHammer…* USENIX 2020
(3) Hector *et al.*, *A closer look at evaluating the BFA…* IEEE IOLTS 2022
(4) Stutz, et al. Random and Adversarial bit error robustness…, TPAMI, 2022

Weight-based Adversarial Attacks

Target internal parameters stored in memory

- BFA = **Adversarial** bit-flips ➔ Faults on the most sensitive parameters
- Very similar to white-box white-box evasion attacks (adv exp)

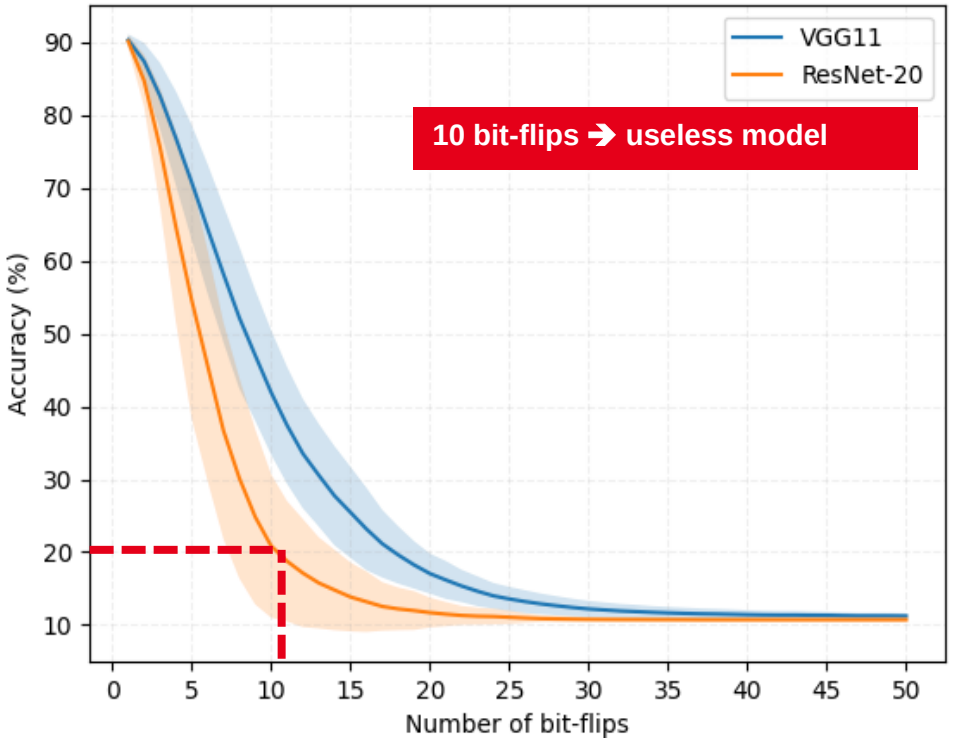
Adversarial objective

No more than S bit-flips

$$\underbrace{\max_{W'} \sum_{i=0}^{N-1} \mathcal{L}\Big(M\big(x_i; W'\big), y_i\Big)}_{\text{mispredictions}} \quad \text{s.t.} \quad \overbrace{HD(W', W) \leq S}^{\text{adv budget}}$$

Gradient-based ranking of w

$\nabla_w \mathcal{L}$



Robustness Evaluation with LFI

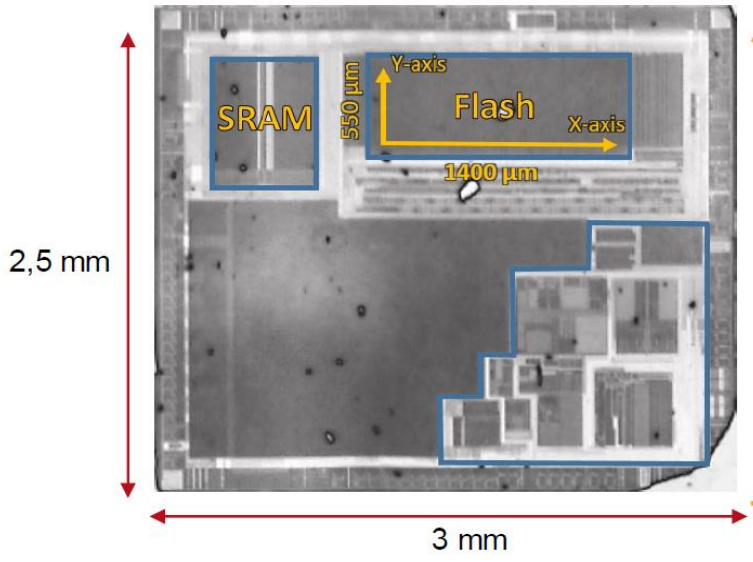
- State of the Art[1]
  - Most works rely on simulation only
  - Practical exp: RowHammer attacks (DRAM)
  - (very) Few and partial works on LFI on MCU against embedded DNN[2]

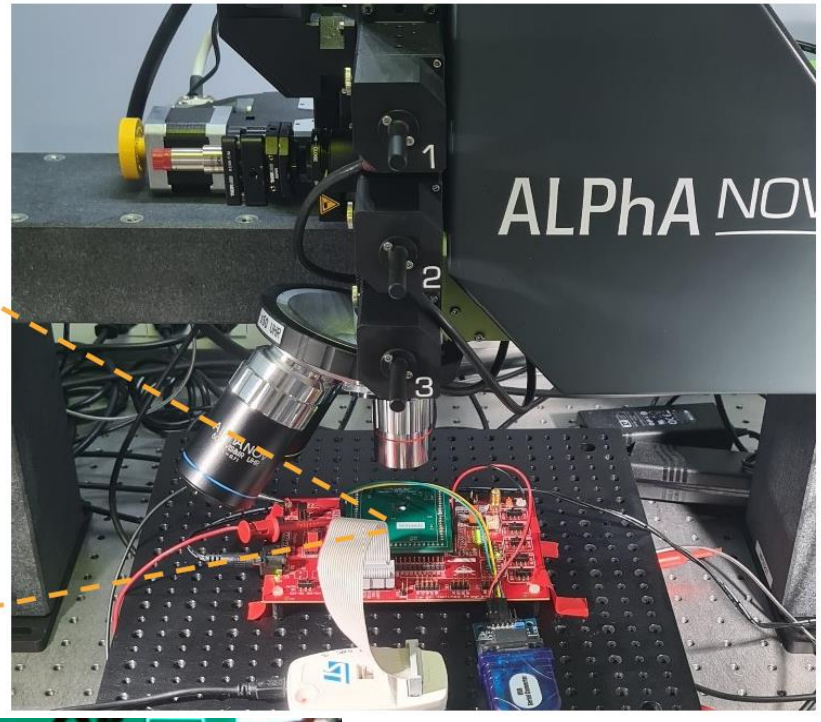
Practical Evaluation on MCU[3]?

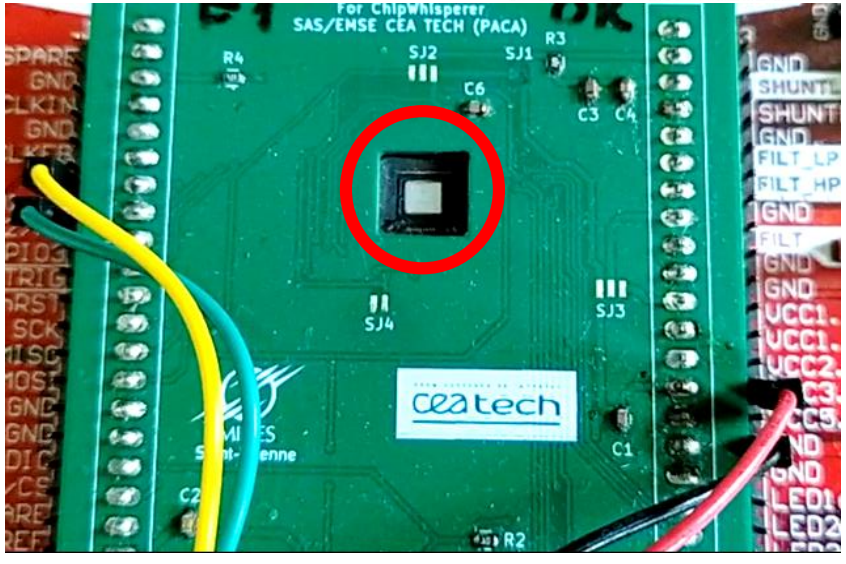
- Security evaluator point of view
- Small MLP (compressed MNIST) embedded in 32-bit MCU, Cortex M3
  - Laser Fault Injection (**LFI**)
  - **Bit-Set Fault Model**
  - Explained and demonstrated for NOR-Flash memory of Cortex-M MCU by Colombier + Menu [3,4]

(1) Qian, et al. A Survey of bit-flip attacks on deep neural network and corresponding defense.Electronics 2023
(2) Hou, et al. Security Evaluation of DNN Resistance against Laser Fault Injection, IPFA 2020
(3) Dumont et al. Evaluation of Parameter-based Attacks against Embedded Neural Networks with Laser Injection. SafeComp 2023
(4) Colombier, et al. Laser-induced Single-bit Faults in Flash Memory…, IEEE HOST 2019.
(5) Menu, et al. Single-bit laser fault model in NOR flash memories…, IEEE FDTC 2020

Robustness Evaluation with LFI







Robustness Evaluation with LFI

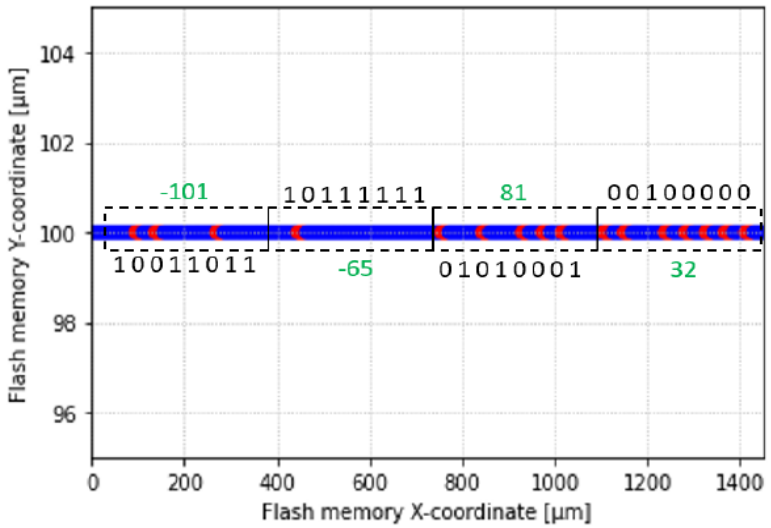
@ neuron-level ➔ weighted sum

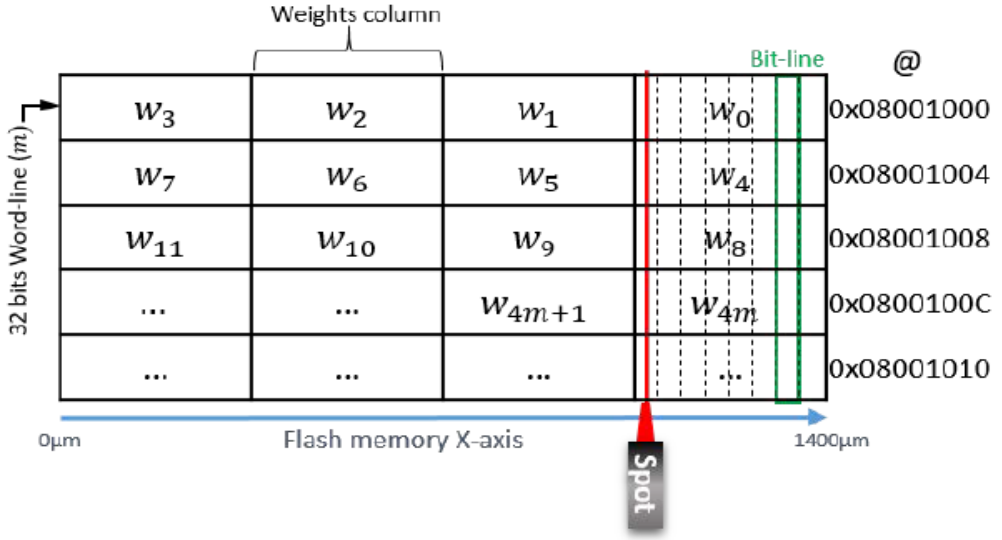
C code, weighted-sum in a FC layer (NNoM)

```
1  while (rowCnt){
2      //pA : address, stored input
3      //pB : address, stored weight
4      for (int j = 0; j < dim_vec; j++)
       { //loop on all neuron
       parameters
5          q7_t inA = *pA++;    //load
       input to inA, address increment
6          inB = *pB++;   //load
       weig to inB, address increment
7          ip_out += inA * inB; //neuron
        weighted sum
8      }
9      *p0++ = (q7_t)__NNOM_SSAT((ip_out
        >> out_shift), 8);
10     rowCnt--;}
```

```
1  ;q7_t inB = *pB++       ;Weight n+1
       initialization
2  ldr      r3, [r7, #80]   ;Loading the
       address of the weight n
3  adds     r2, r3, #1      ;Next weight
       address
4  str      r2, [r7, #80]   ;Input value
       loading into r2 reg
5  ldrsb.w r3, [r3]         ;Weight value
       loading. LASER SHOT
6  strb     r3, [r7,#23]    ;Store of the
       weight in SRAM reg
```

Assembler code, of line 6

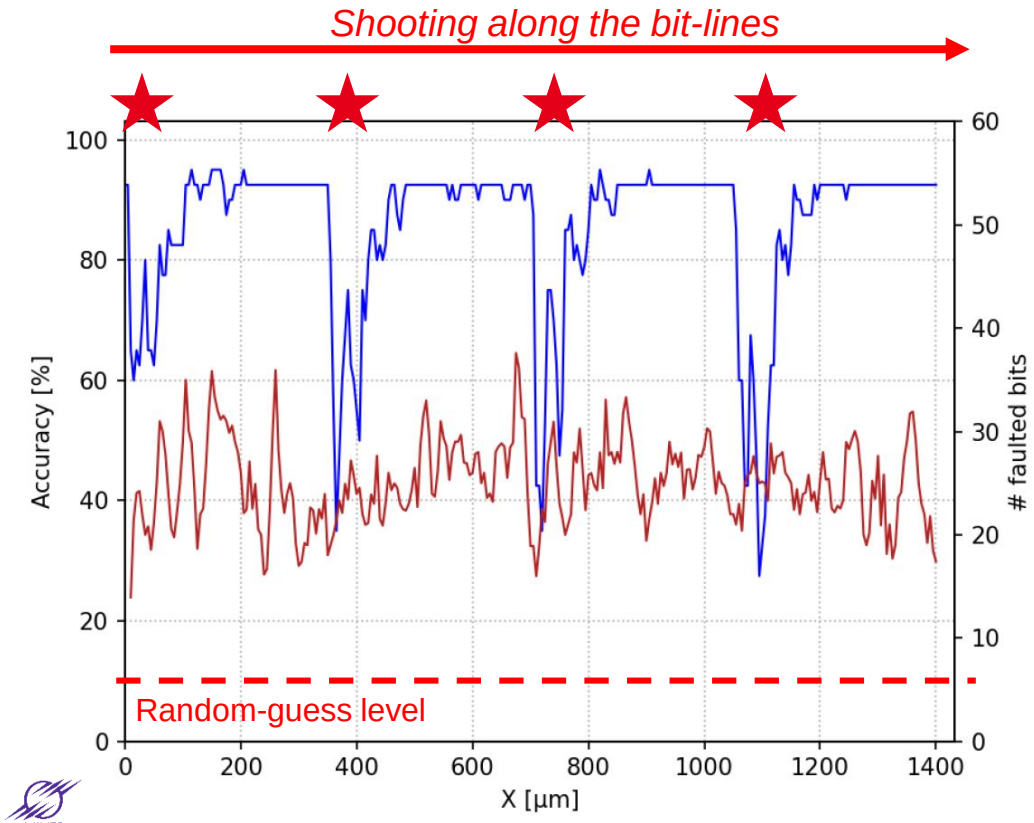


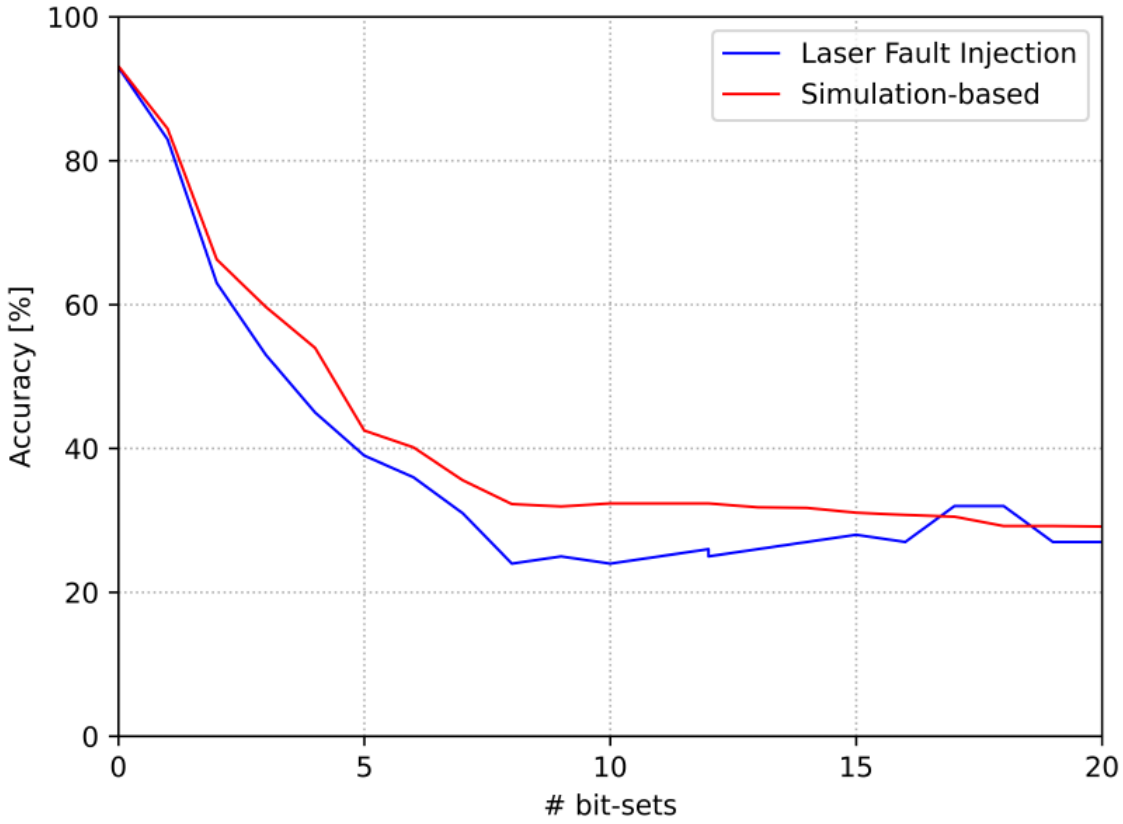


Robustness Evaluation with LFI

Target a MLP model

- Brute-Force approach (4960 bits)
- Significant accuracy drops for MSB locations ★
- Guided-LFI with adapted BFA (BSCA)
- How practical LFI fit with simulation (BSCA)?

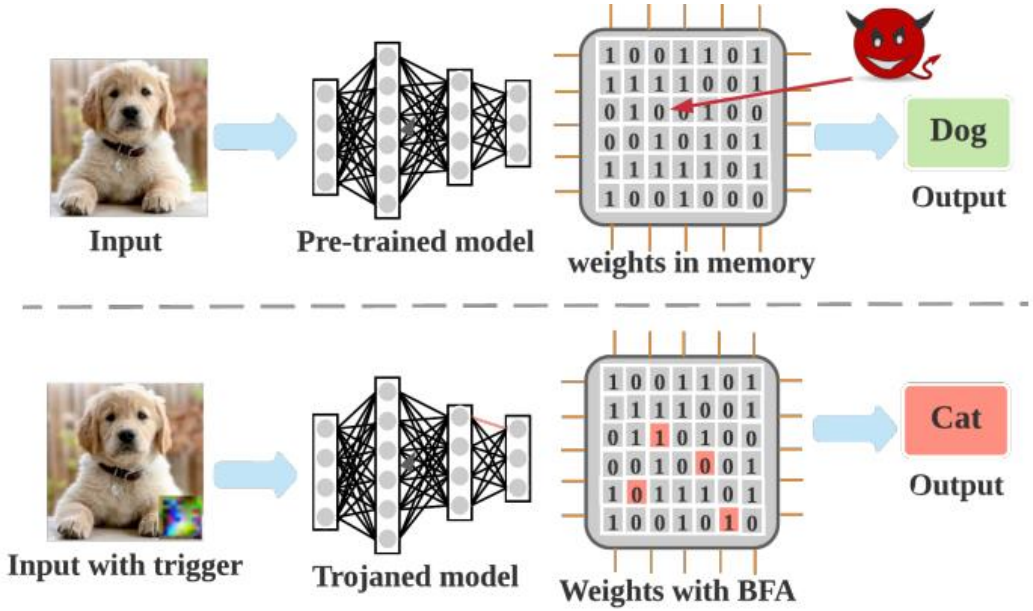




Exploitation?

- **Integrity threat… really?**
  - LFI ➔ powerful tool for practical robustness evaluation
  - Use *average adversarial accuracy* as a task-quality metric
  - #faults *vs* accuracy_drop is more interesting

- **Targeted BFA**
  - (one targeted input / one targeted misprediction)
- **Backdoor attacks[1]**
- **Upcoming threats at training time for distributed paradigm**
  - Federated Learning (model poisoning)



(1) H. Chen, et al. ProFlip: Targeted Trojan Attack With Progressive Bit Flips, ICCV 2021

Exploitation → confidentiality threats

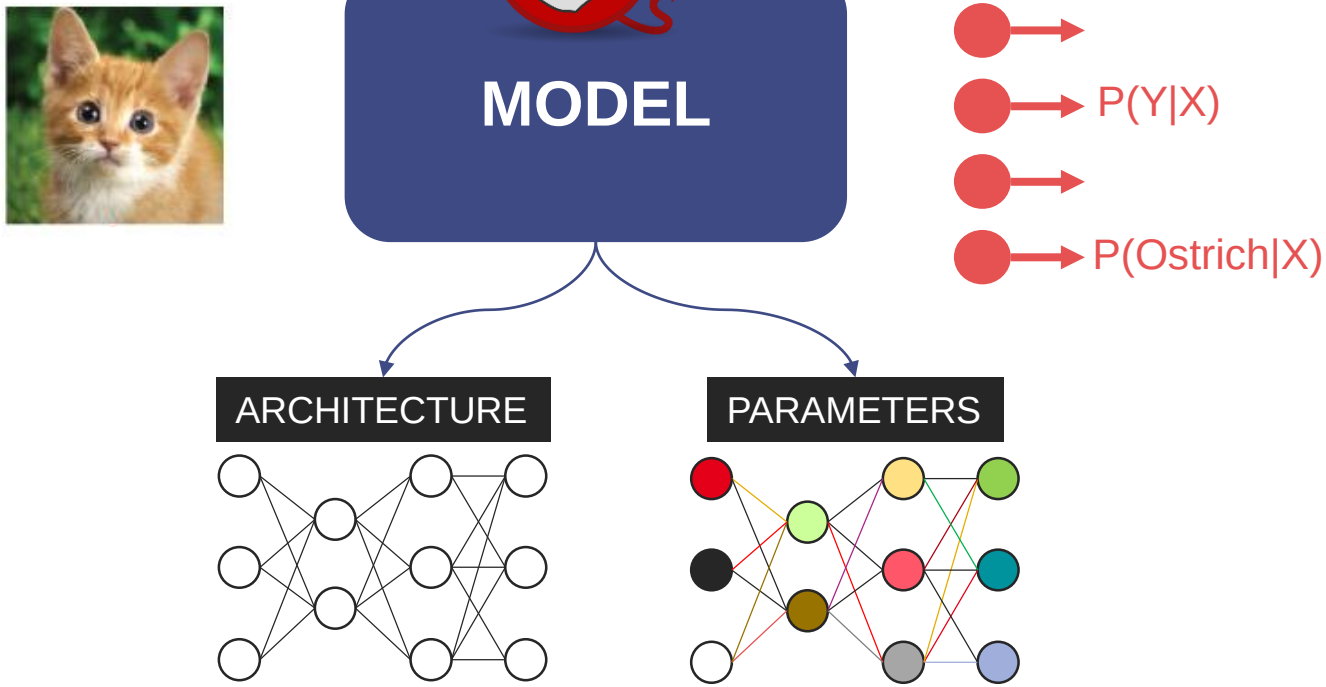
Model Extraction

Model Extraction

A growing confidentiality concern

Adversarial Goal[1]

- **FIDELITY** (model cloning)
  Target: architecture & parameters
- **TASK-PERFORMANCE** (steal performance, save design & training time)
  Transfer knowledge
- **BLACK → WHITE BOX ATTACK**



(1) Jagielski et al. High accuracy and high fidelity extraction of neural networks. USENIX 2020

Model Extraction

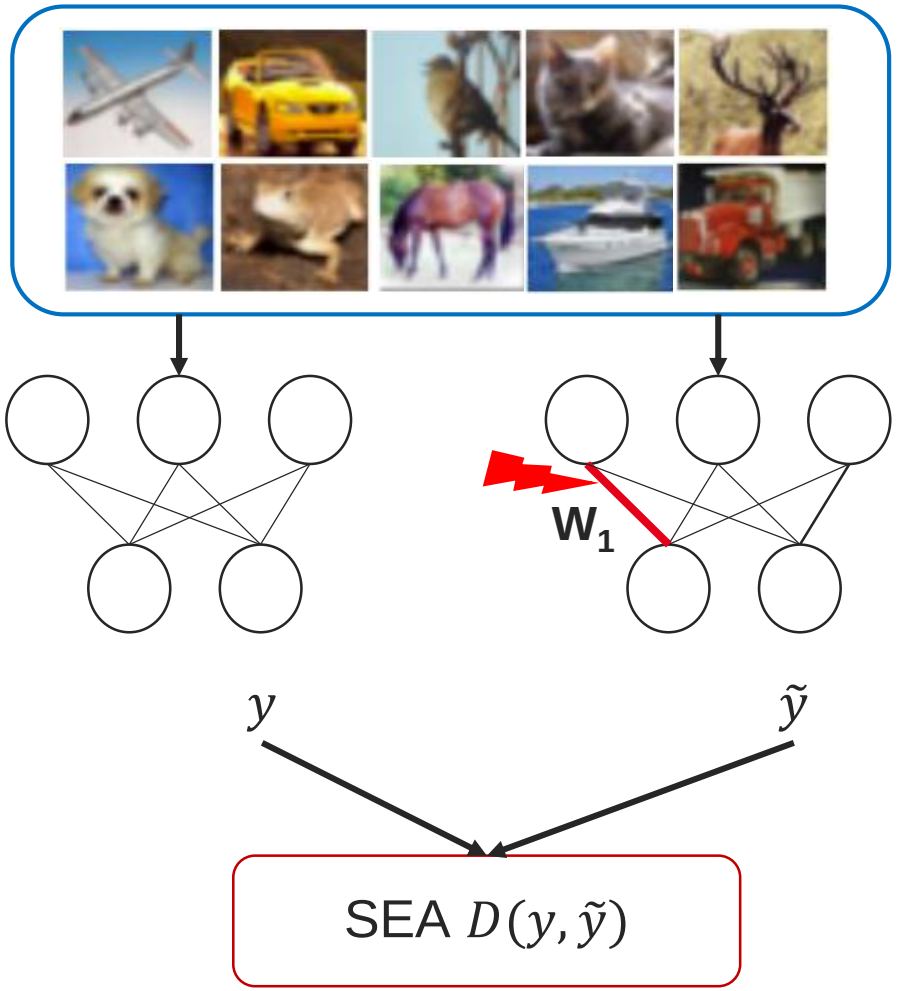
Theoretical framework (fidelity scenario)

- **VICTIM MODEL M_W** trained with (X^{train}, Y^{train}), $X^{train} \sim D_\chi$
- **SUBSTITUTE MODEL: M'_θ**
- **Threat Model (parameter extraction):** Architecture is known / (very) limited access to X^{train}
- **SotA**
  - Active learning principle: feed M_W to build a substitute training dataset for M'_θ ➔ (x, M_W(x))
  - Hard work… Need of a huge amount of query/output pairs
  - **DEEPSTEAL**[1]: use BFA+Rowhammer (RamBleed) to guess some parameter values and improve the training of M'_θ

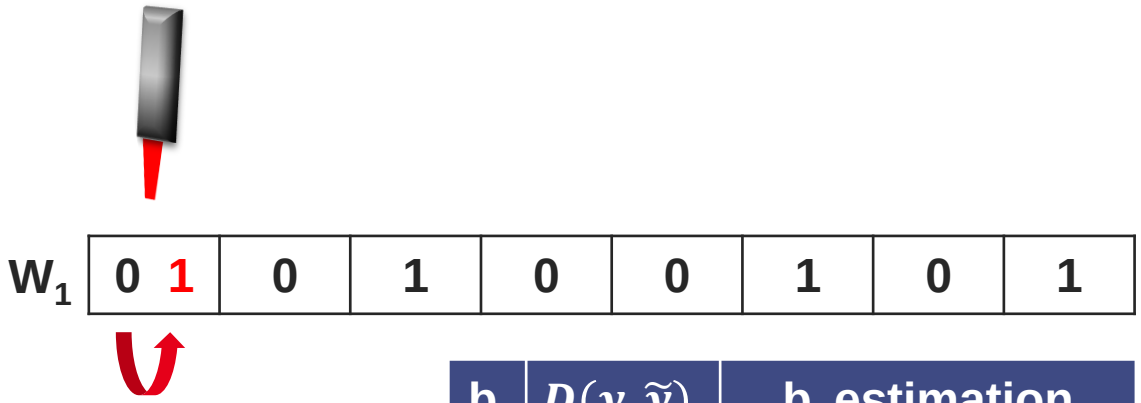
(1) Rakin et al. DeepSteal, IEEE S&P 2022

Model Extraction

Exploiting Safe Error Attack (SEA)



bit-set
fault model



b_i	$D(y, \tilde{y})$	b_i estimation
0	**>0**	**0**
0	0	???
1	0	???

Model Extraction

Exploiting Safe Error Attack (SEA)

Victim World

Victim model M_W

Training dataset D

Adv. World

Substitute model

M'_θ

Adv. knowledge
- M_W architecture
- <10% of D

Adv. goal
- Clone M_W performance
- Transfer adv. attacks

#1 INPUT CRAFTING

random inputs

+ M_W (Black-box API)

+ $\Upsilon^{objective}$ (uncertain preds)

Genetic algorithm

#2 BIT RECOVERY - SEA

$\hat{y}$ $\hat{y}^*$ bit-set fault model

Safe Error Attack

$W^{(0)}$ =

#3 SUBSTITUTE MODEL TRAINING

(a) Weight initialization $\Theta = W^{(0)}$

(b) Substitute model training

<10% of D + Mean Clustering training

(c) Evaluation / Results

	CNN (CIFAR10)			MLP (MNIST)		
	Acc. (%)	Fidelity (%)	AUA	Acc. (%)	Fidelity (%)	AUA
M_W	79.4	100	0.42	97.17	100	0.08
M'_θ	75.27	85.58	1.83	95.95	97.21	0.38

- ✓ Specific inputs lead to very successful SEA ➔ predictions with some uncertainty
- ✓ Train M'_θ with constrains from the partial extracted values ➔ very efficient

CONCLUSION

- ✓ Very active context regarding safety / security concerns for AI systems
- ✓ Growing needs of
  - ❖ Threat models ➔ risk analysis & exploitation
  - ❖ Practical demonstrations and evaluations (or platform-based simulation)
    - ❖ Models, platforms…
  - ❖ Robust implementations
  - ❖ Defenses (practical) evaluation
- ✓ Numerous attack vectors and paths ➔ works for everyone ☺

Thank you for your attention

Support & Funding

EU project InSecTT

French ANR, IRT Nanoelec

French ANR PICTURE program

This work benefited from the French **Jean Zay** *supercomputer with the AI dynamic access program.*