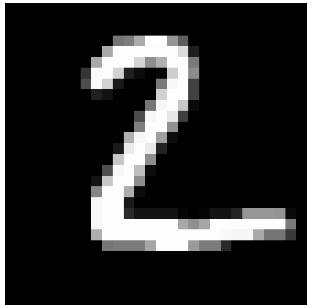
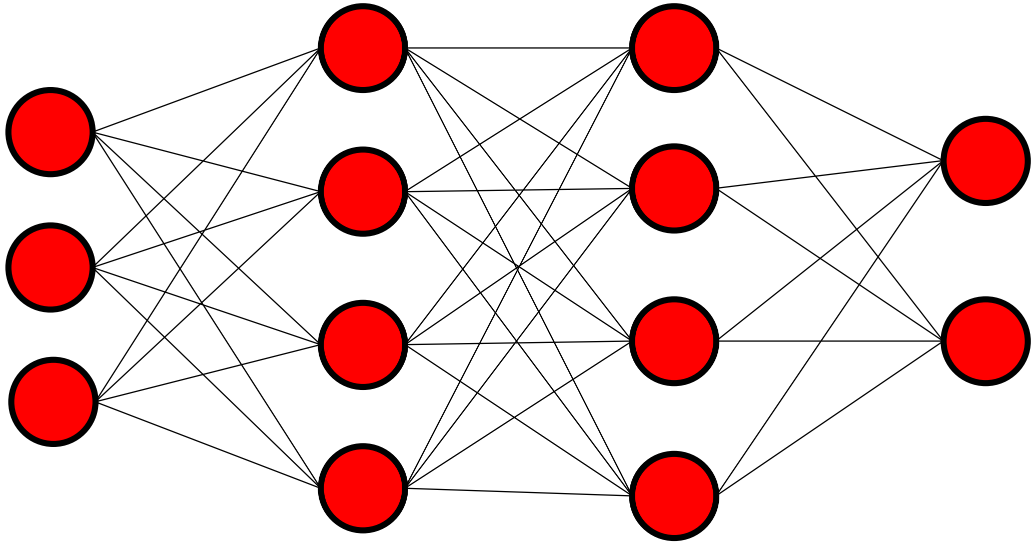
Hard-Label Cryptanalytic Extraction of DNNs

Benoit Coqueret[1&2], Mathieu Carbone[1], Olivier Sentieys[2], Gabriel Zaid[1]

1. CESTI Thales
2. University of Rennes, INRIA, IRISA

Brief Overview of Deep Neural Networks





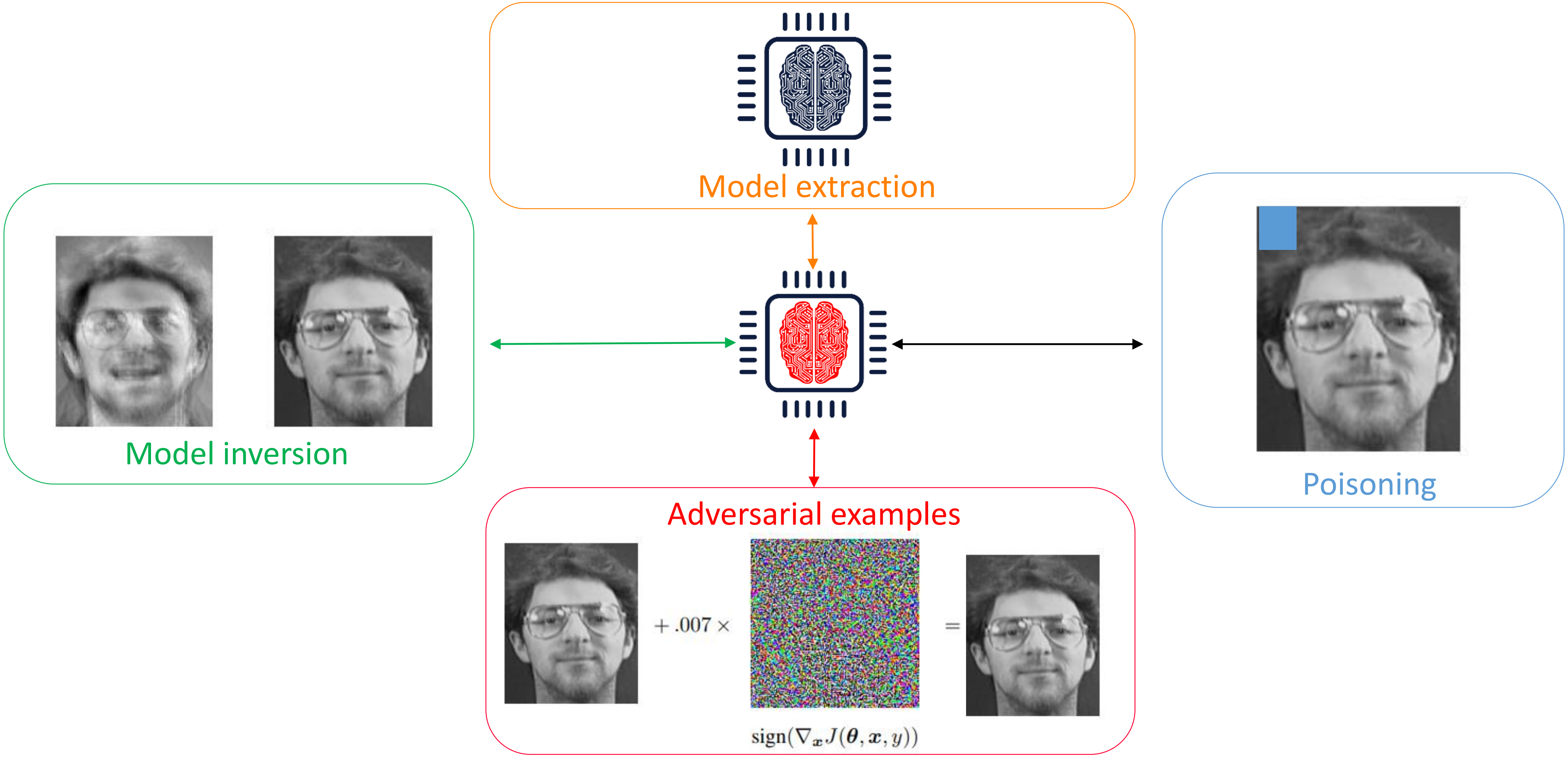
2
(0.1, 0.9)

Input

Trained Model

Prediction and output probabilities

Attacks Against Deep Neural Networks



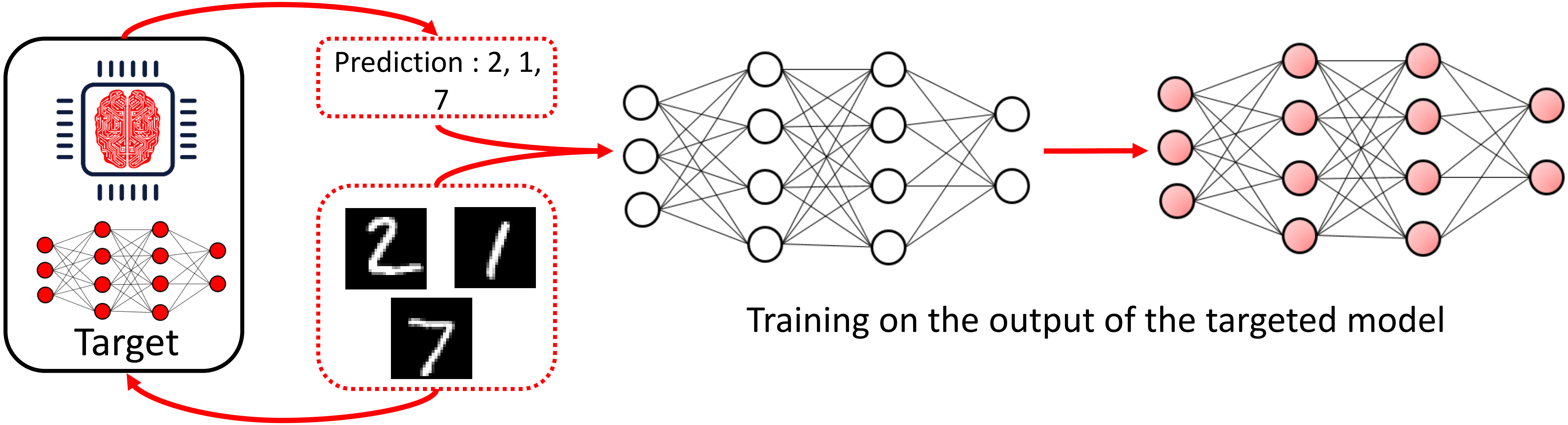
Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures, ACM SIGSAc 2015

Model Extraction: State-Of-The-Art

- Obtain a copy of the targeted DNN
    - Stealing the Intellectual Property
    - Possibility to mount more powerful attack on the targeted DNN
- 3 broad methodologies

Model Extraction: State-Of-The-Art

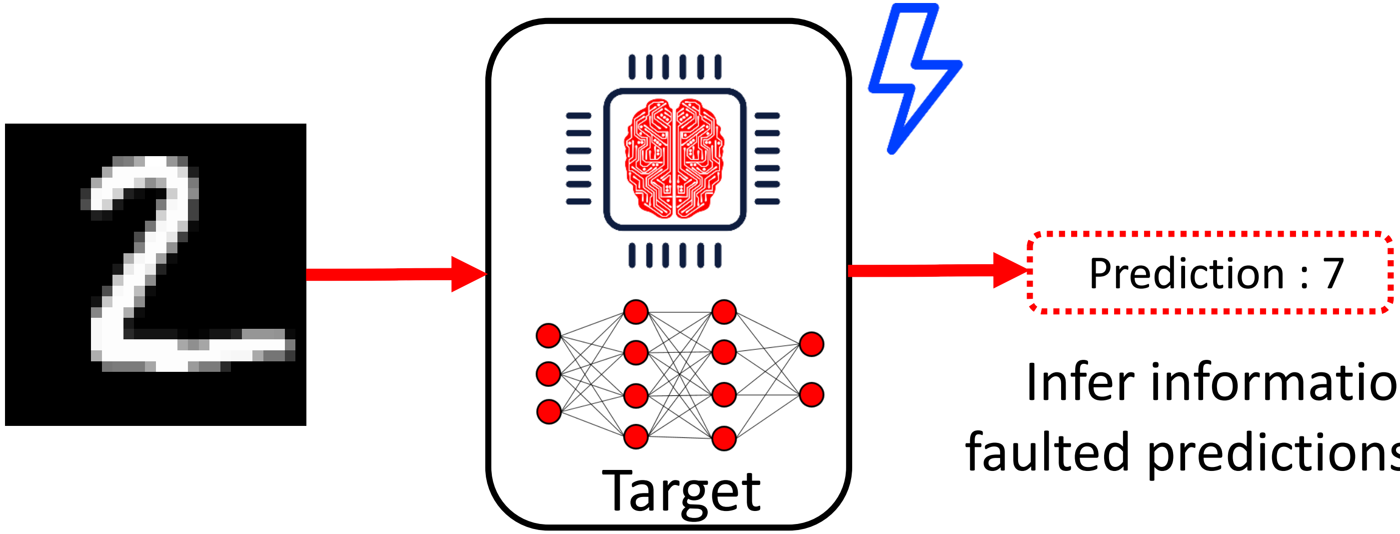
- Obtain a copy of the targeted DNN
  - Stealing the Intellectual Property
  - Possibility to mount more powerful attack on the targeted DNN
- 3 broad methodologies
  - Active learning [1]



Training on the output of the targeted model

Model Extraction: State-Of-The-Art

- Obtain a copy of the targeted DNN
  - Stealing the Intellectual Property
  - Possibility to mount more powerful attack on the targeted DNN
- 3 broad methodologies
  - Active learning [1]
  - Hardware attacks (Fault Injection [2] or Side Channel [3])



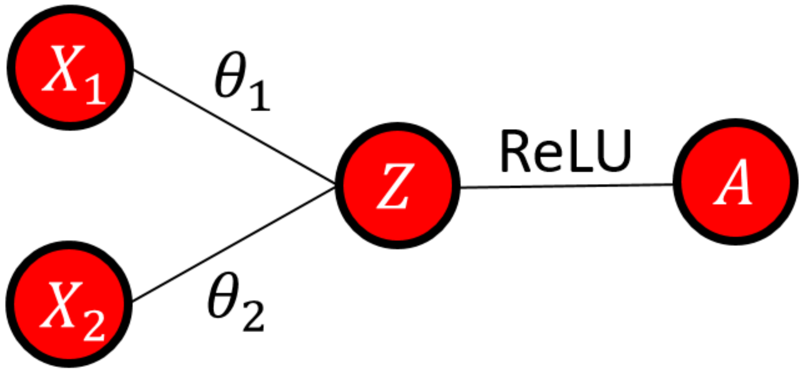
Infer information by comparing faulted predictions with correct ones

Model Extraction: State-Of-The-Art

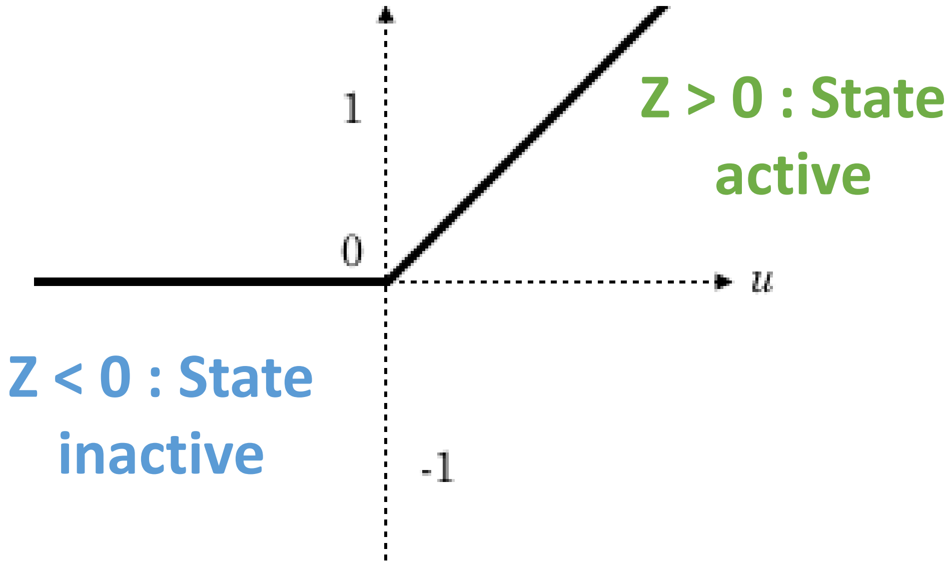
- Obtain a copy of the targeted DNN
  - Stealing the Intellectual Property
  - Possibility to mount more powerful attack on the targeted DNN
- 3 broad methodologies
  - Active learning [1]
  - Hardware attacks (Fault Injection [2] or Side Channel [3])
  - Cryptanalytical extraction[4, 5, 6]
    - Analogy between the weights and the key
    - Input becomes the message
    - Output is equivalent to cipher text

Model Extraction: Cryptanalysis

- Special case of networks using ReLU function

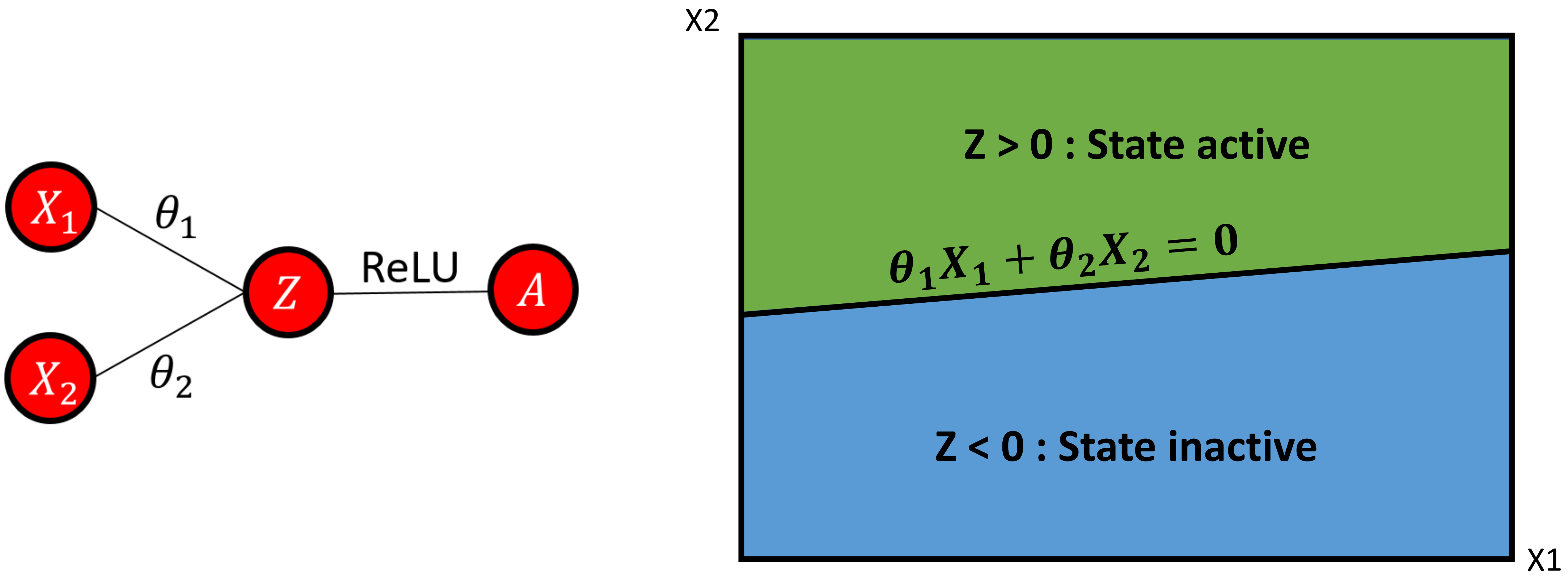


ReLU(x) = max(0,x)



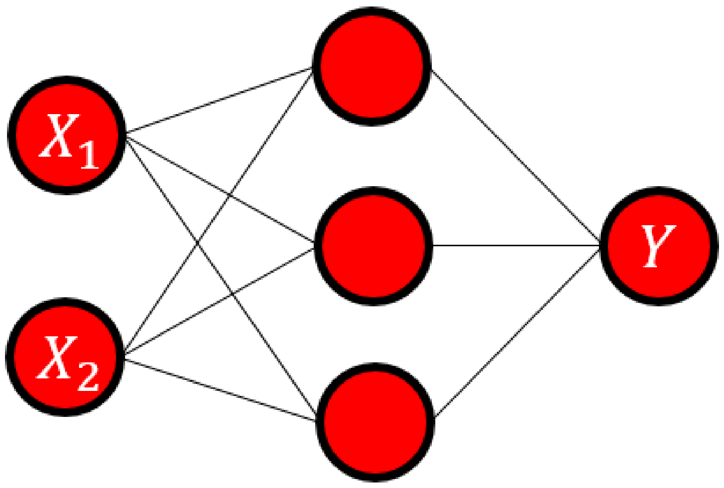
Model Extraction: Cryptanalysis

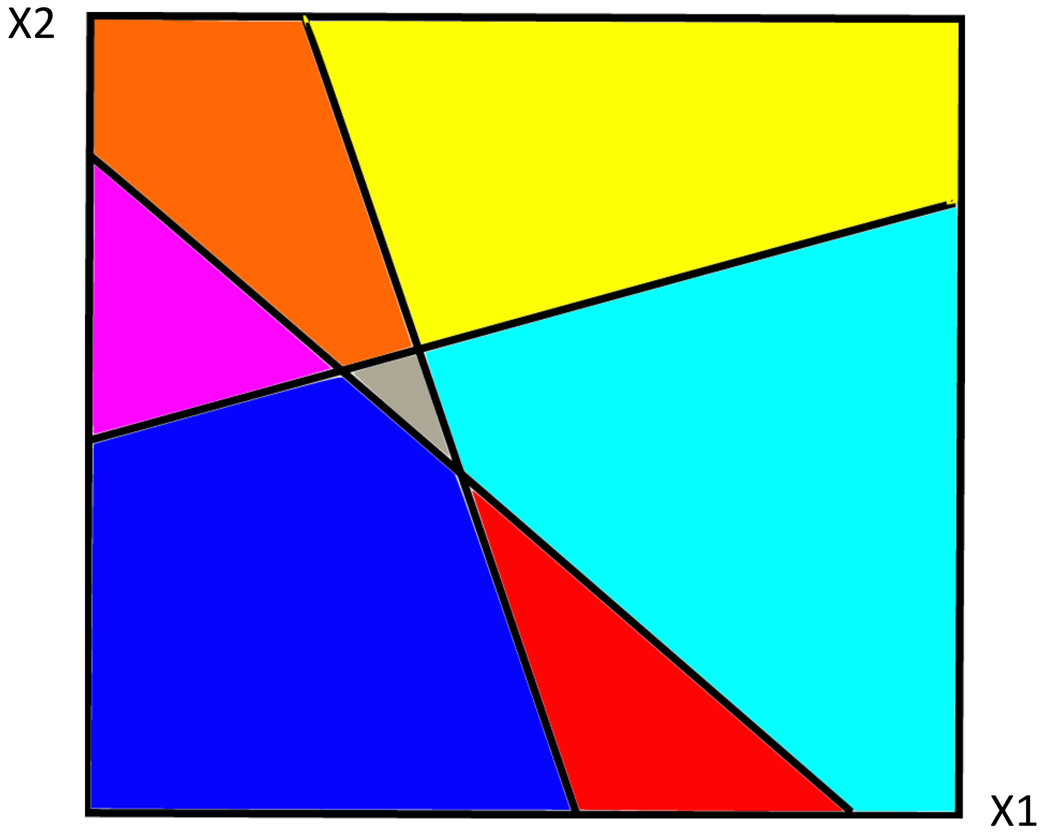
- Special case of networks using ReLU function



Model Extraction: Cryptanalysis

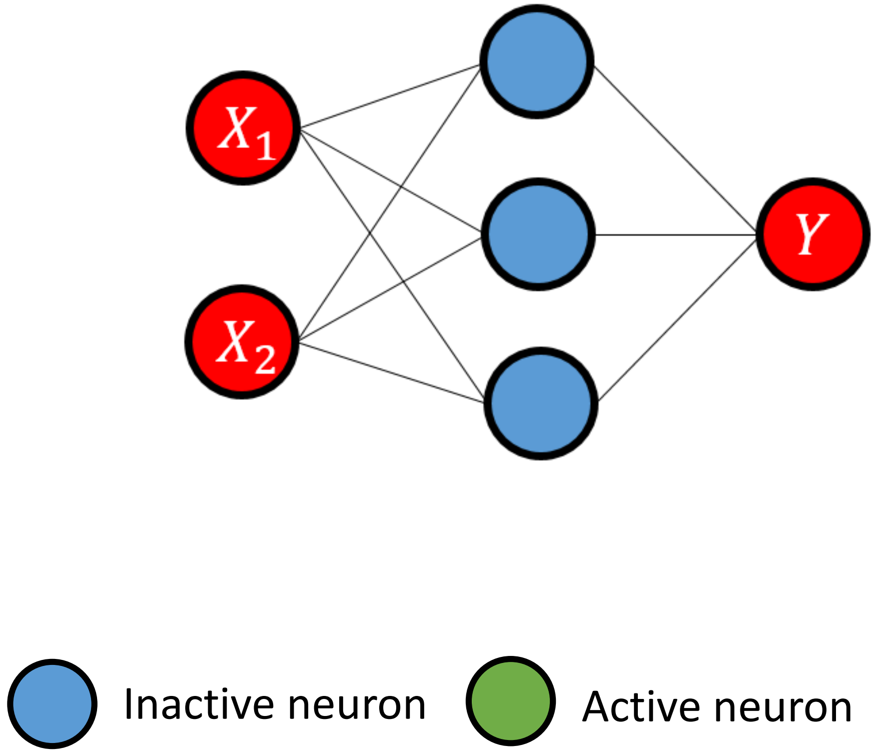
- Special case of networks using ReLU function

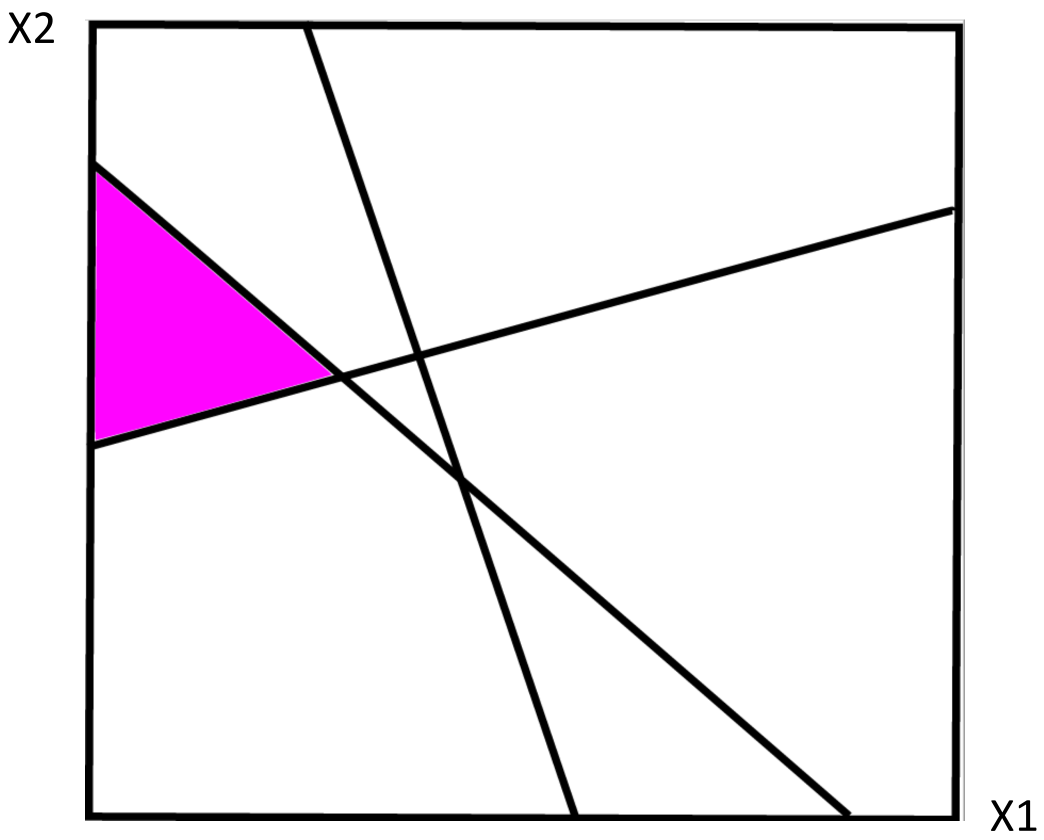




Model Extraction: Cryptanalysis

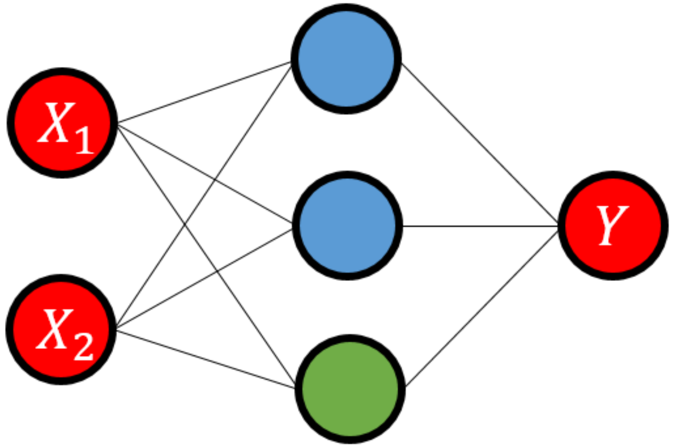
- Special case of networks using ReLU function

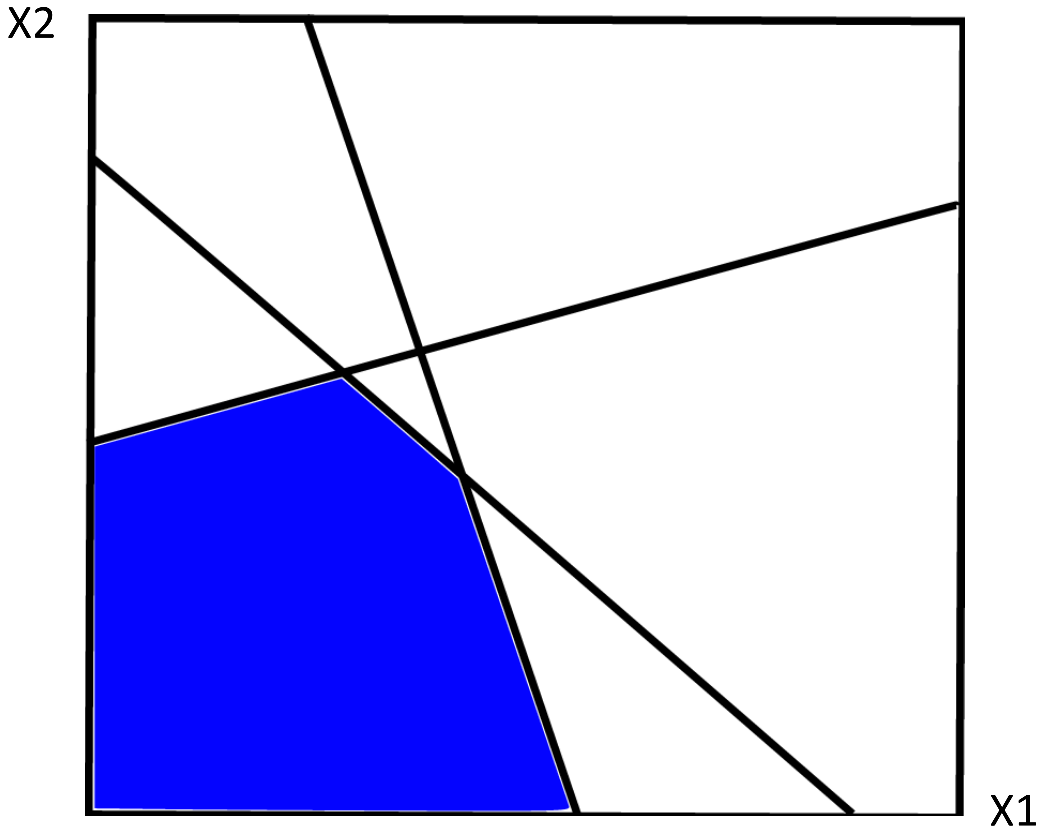




Model Extraction: Cryptanalysis

❖ Special case of networks using ReLU function

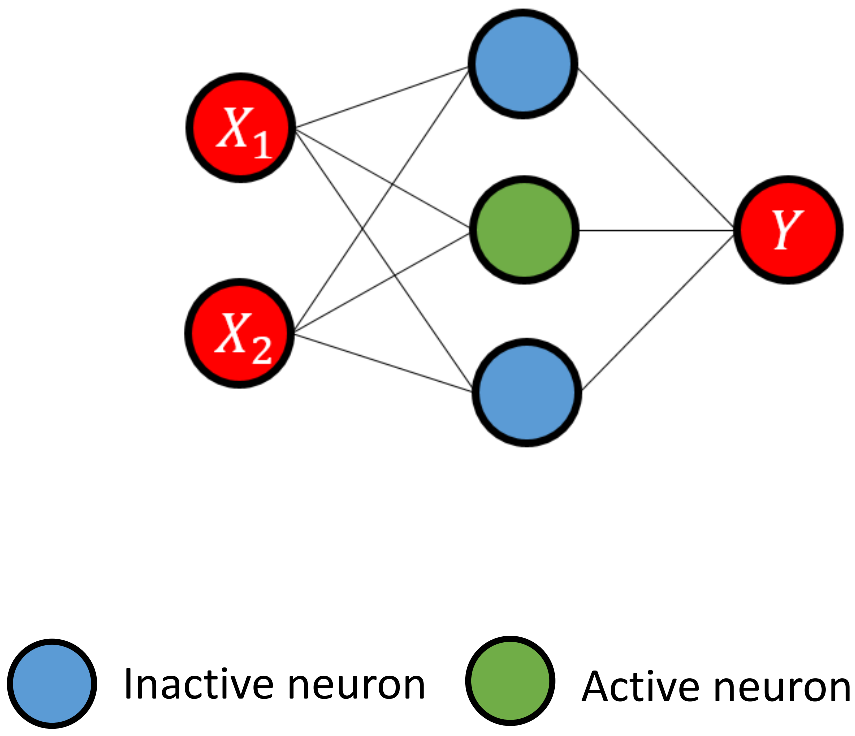


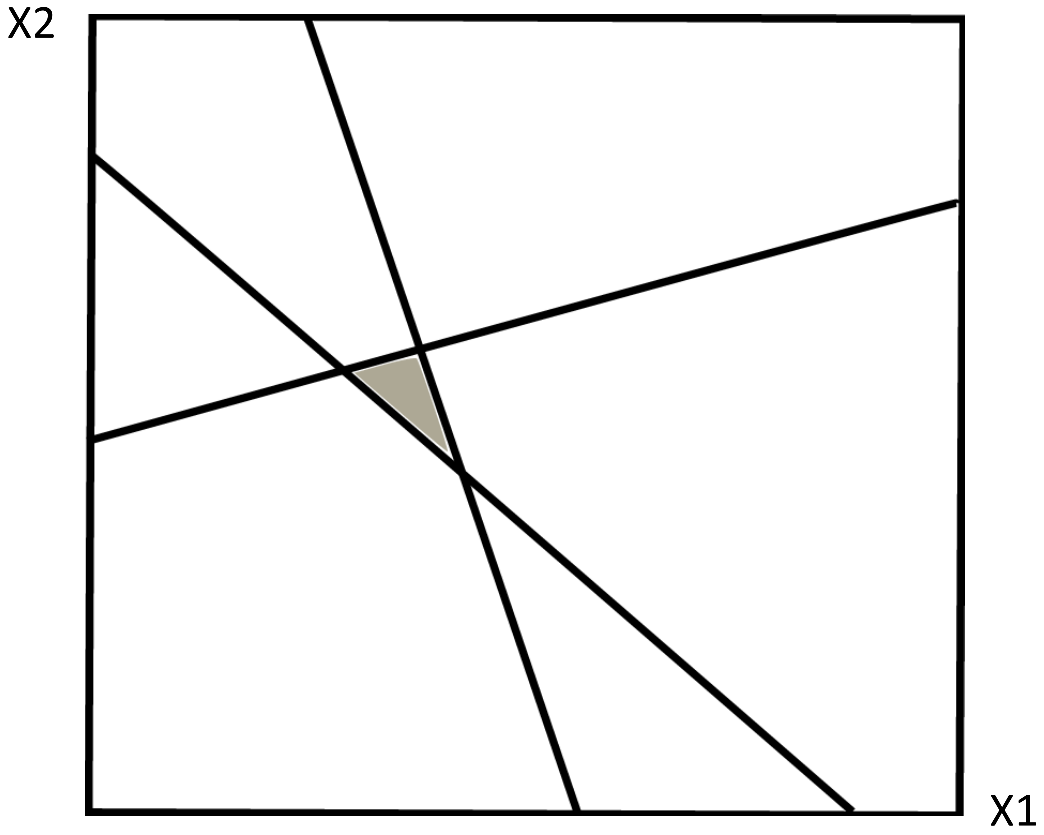


Inactive neuron   Active neuron

Model Extraction: Cryptanalysis

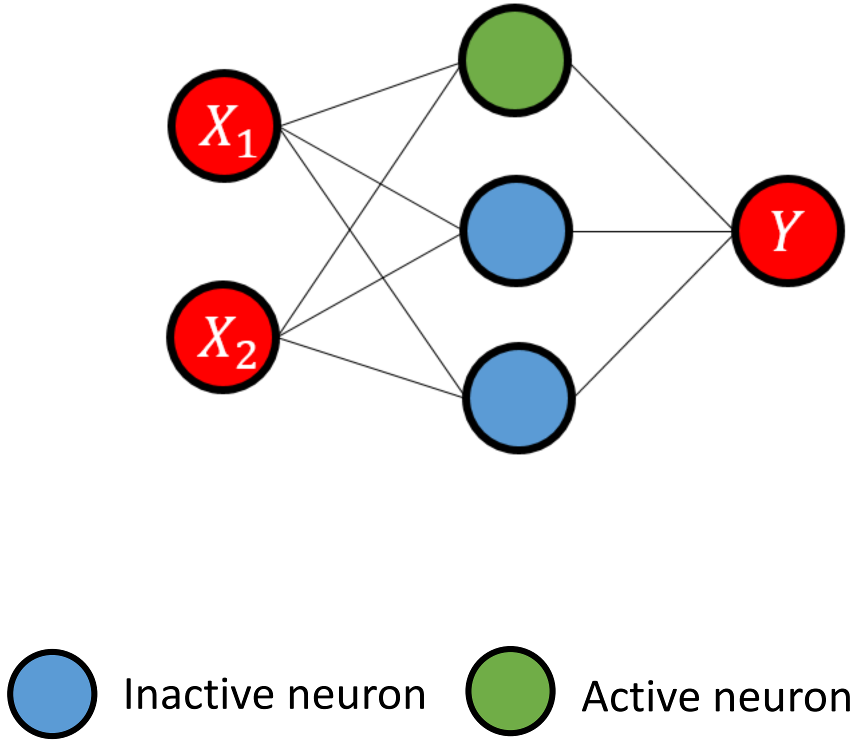
- Special case of networks using ReLU function

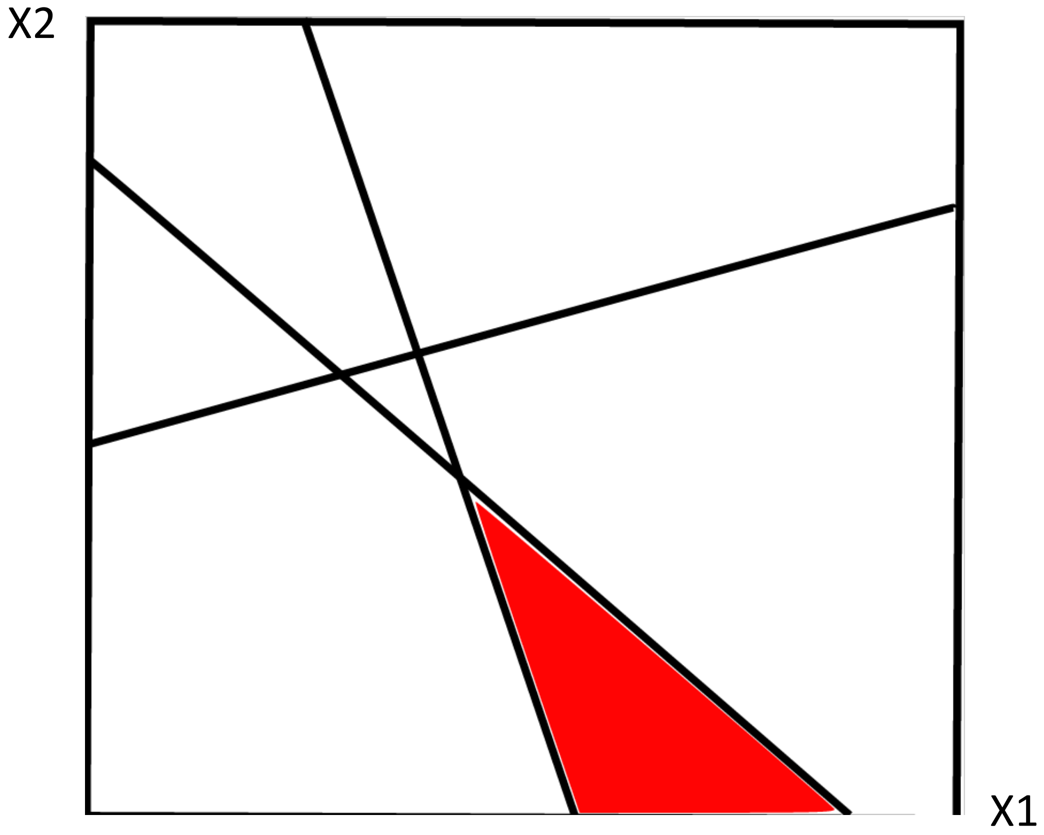




Model Extraction: Cryptanalysis

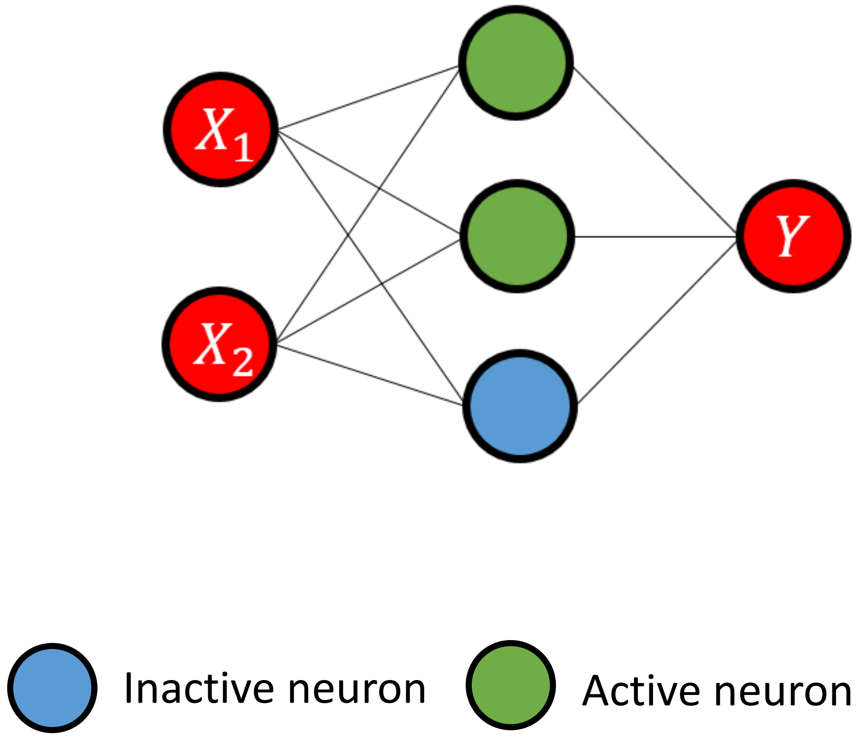
- Special case of networks using ReLU function

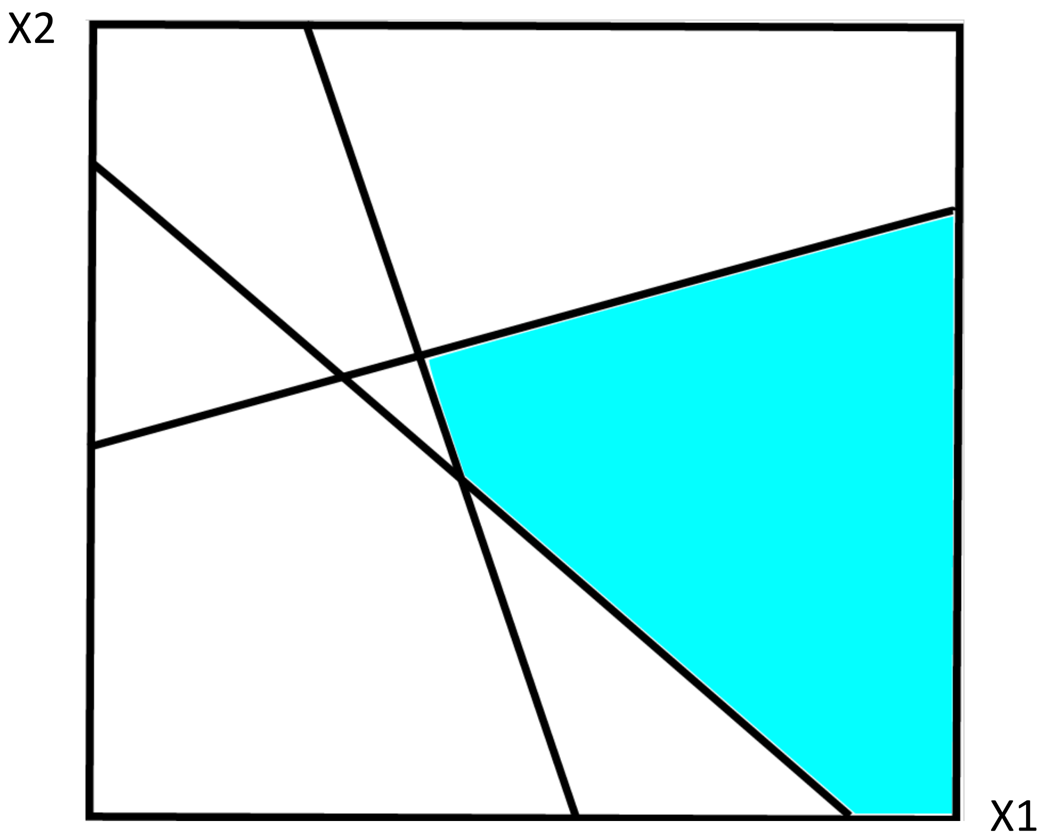




Model Extraction: Cryptanalysis

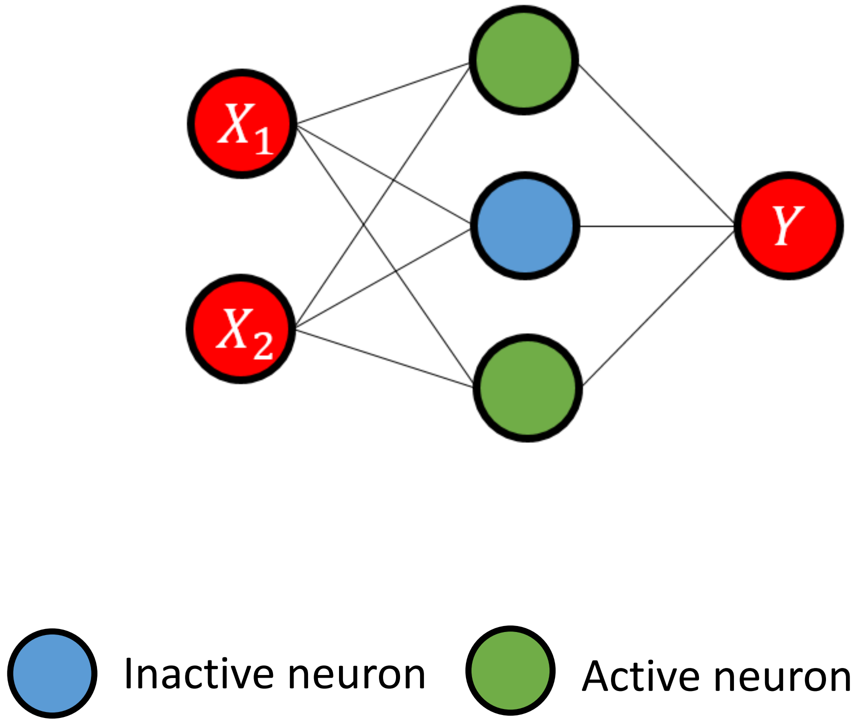
- Special case of networks using ReLU function

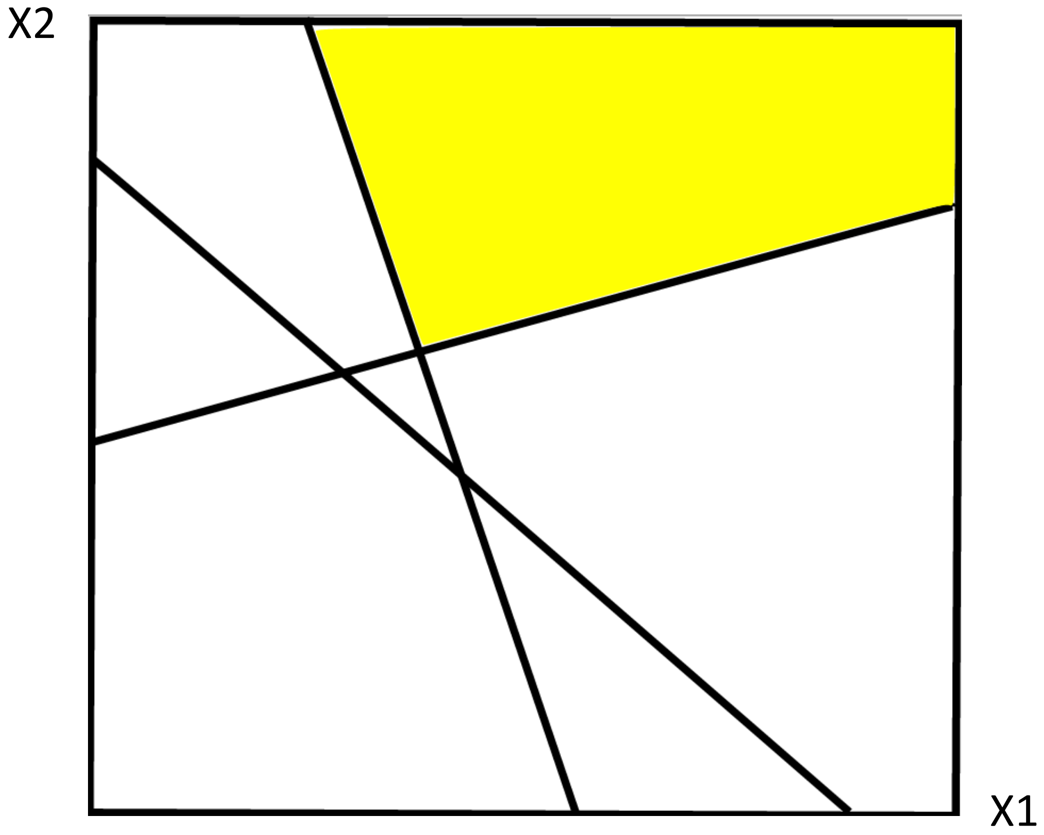




Model Extraction: Cryptanalysis

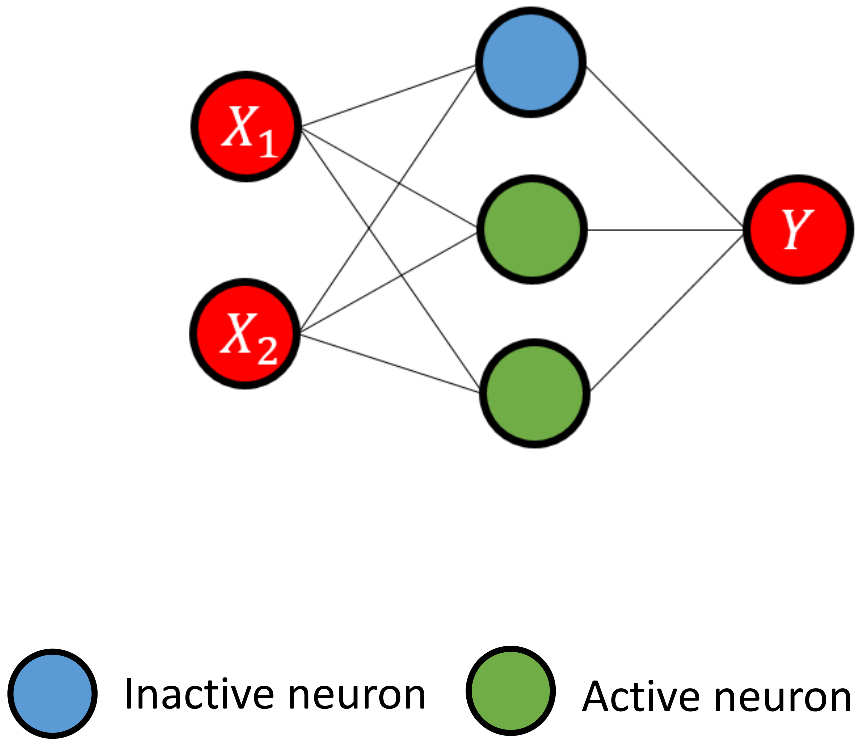
- Special case of networks using ReLU function

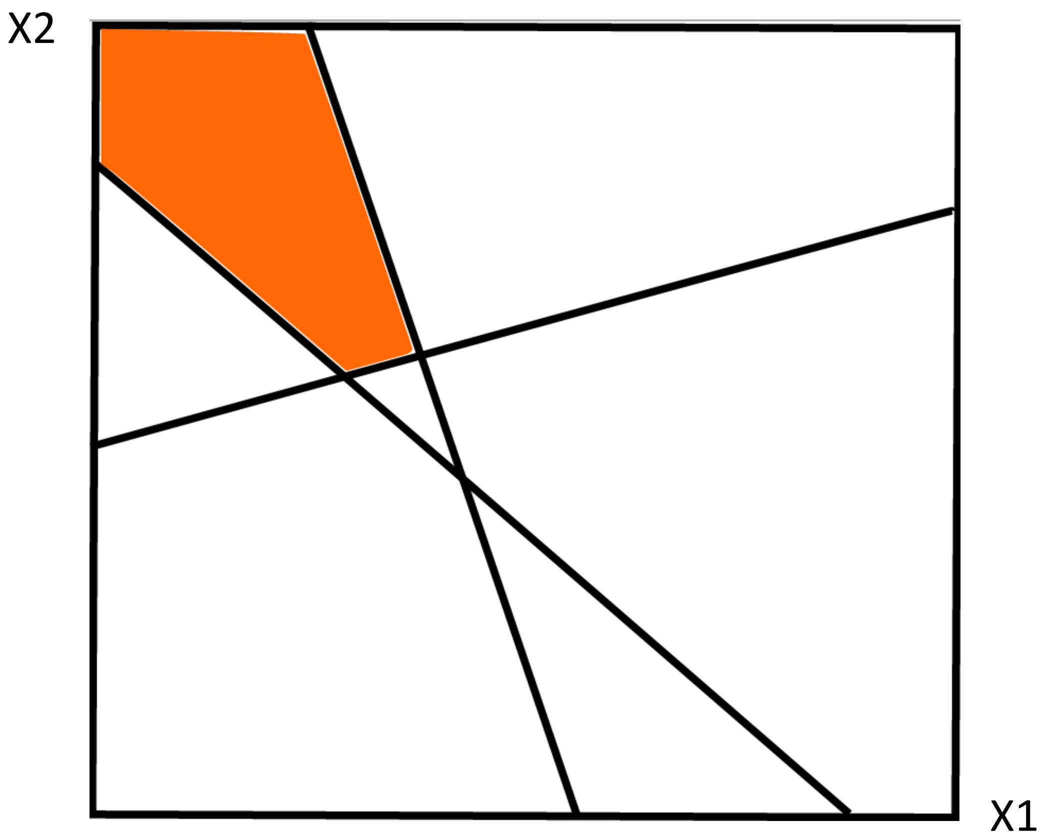




Model Extraction: Cryptanalysis

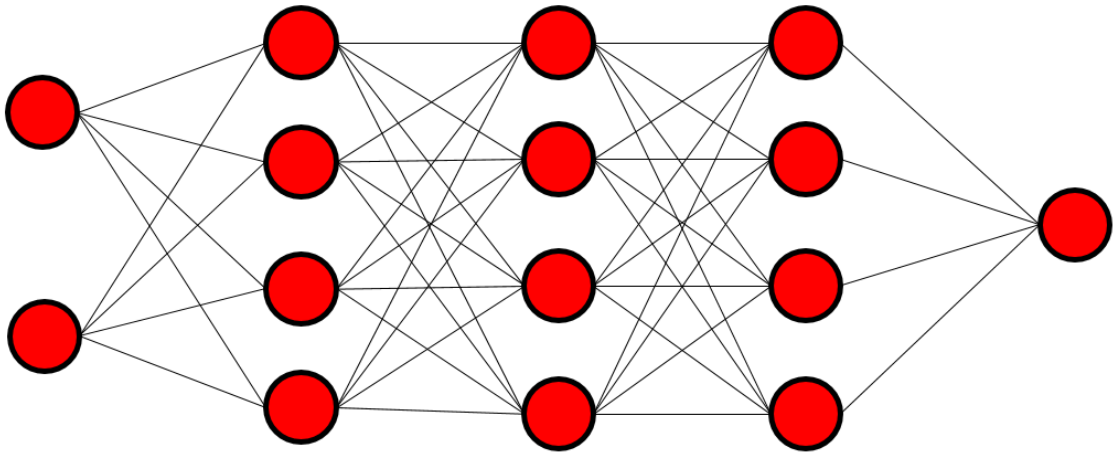
❖ Special case of networks using ReLU function

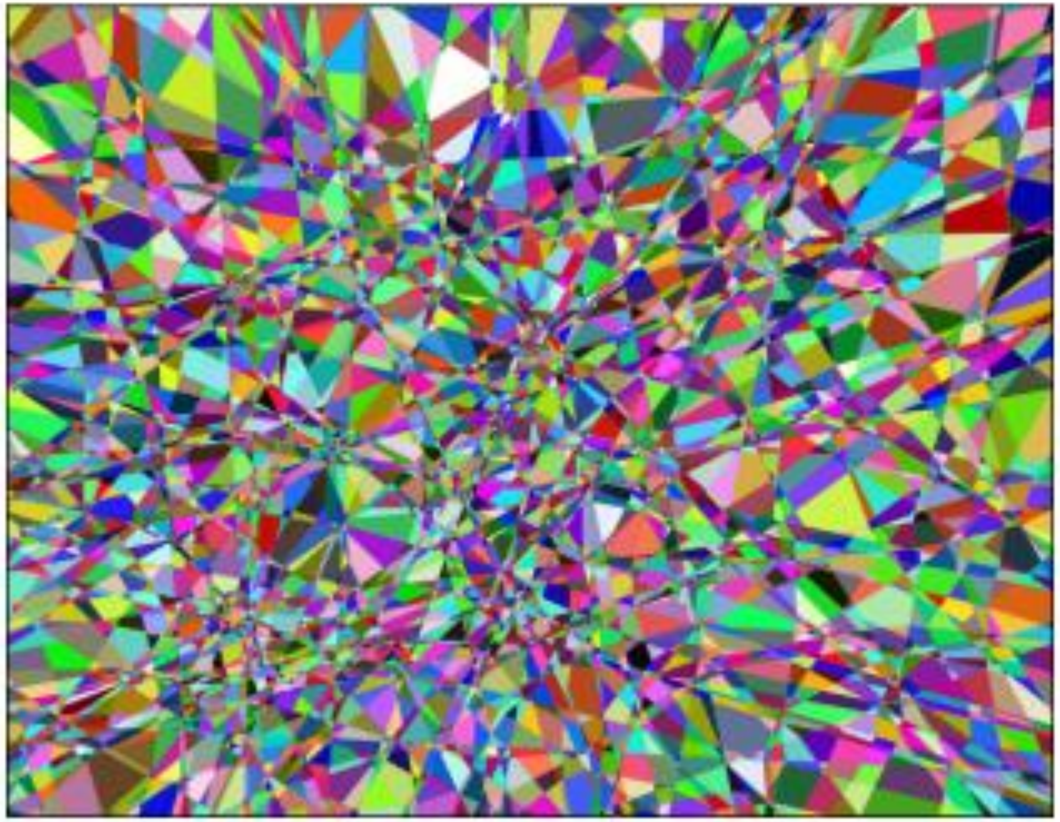




Model Extraction: Cryptanalysis

❖ Special case of networks using ReLU function



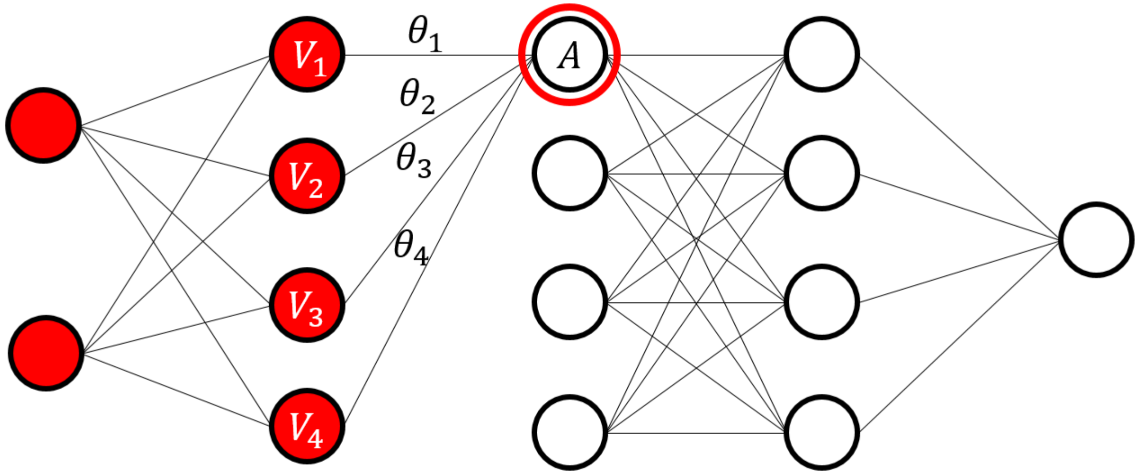


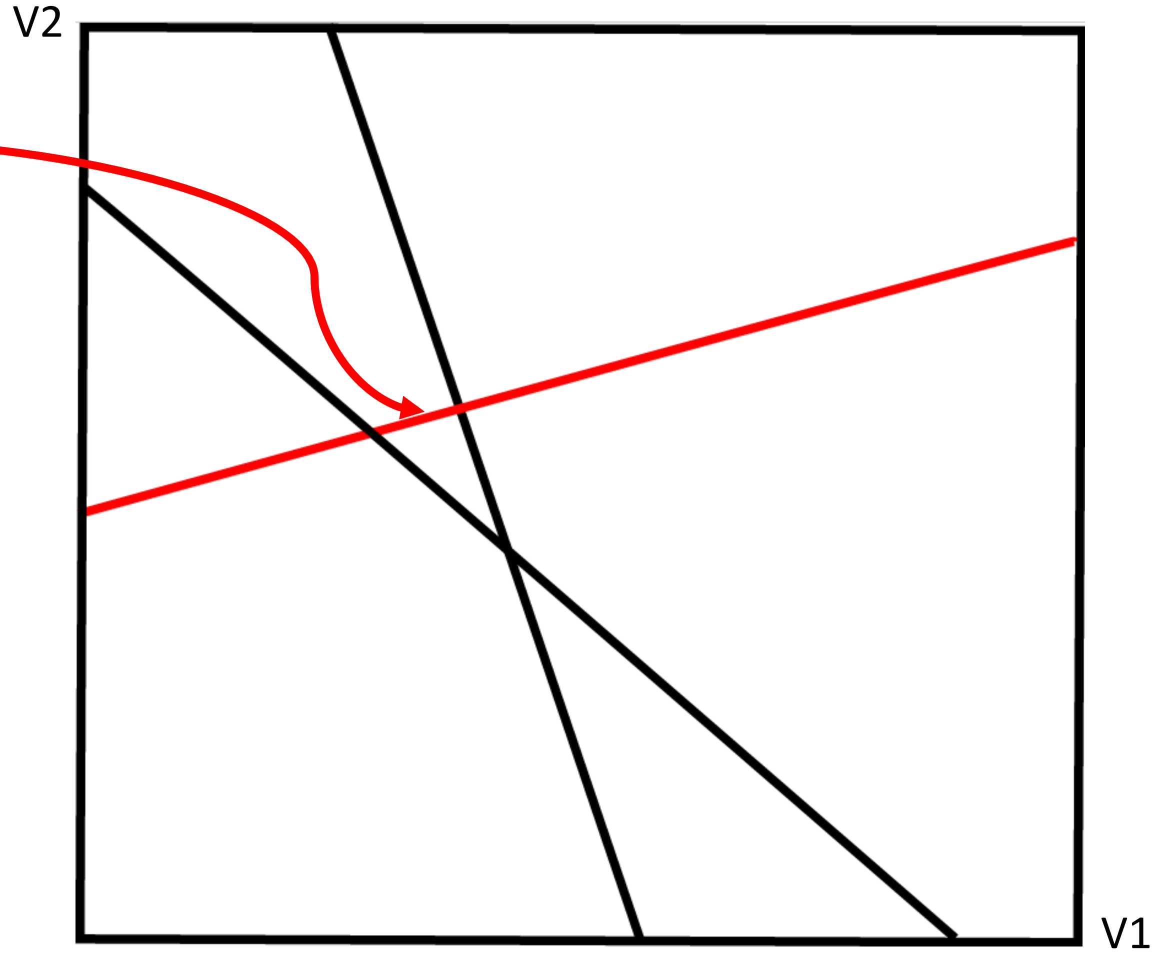
Complexity of Linear Regions in Deep Networks, ICML 2019

Model Extraction: Cryptanalysis

❖ Special case of networks using ReLU function

$$\boldsymbol{\theta}_{\boldsymbol{\eta}} \boldsymbol{V}(\boldsymbol{\eta}; \boldsymbol{X}) + \beta_{\eta} = 0$$



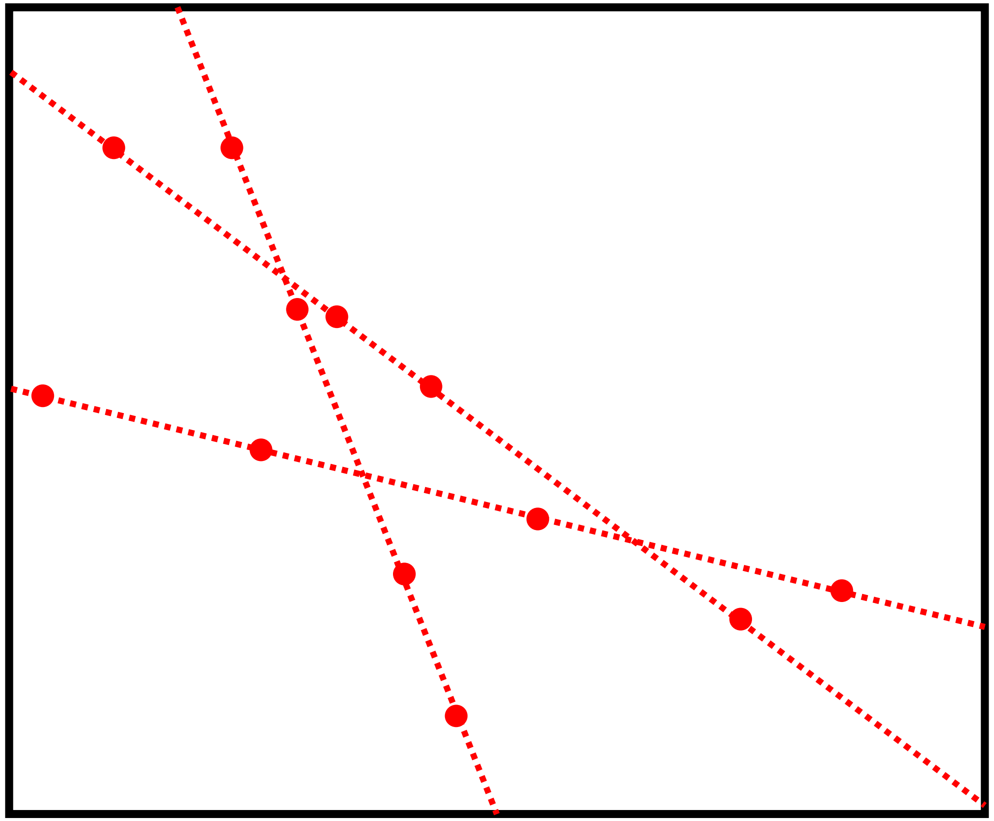


With :
- $\boldsymbol{\theta}_{\boldsymbol{\eta}}$ the weights of the targeted neuron η
- β_{η} the bias of the targeted neuron η
- $\boldsymbol{V}(\boldsymbol{\eta}; \boldsymbol{X})$ the activations values of the previous layer associated with input $\boldsymbol{X}$

Model Extraction: Cryptanalysis

❖ Global methodology

- Search for points on the hyperplanes: the critical points

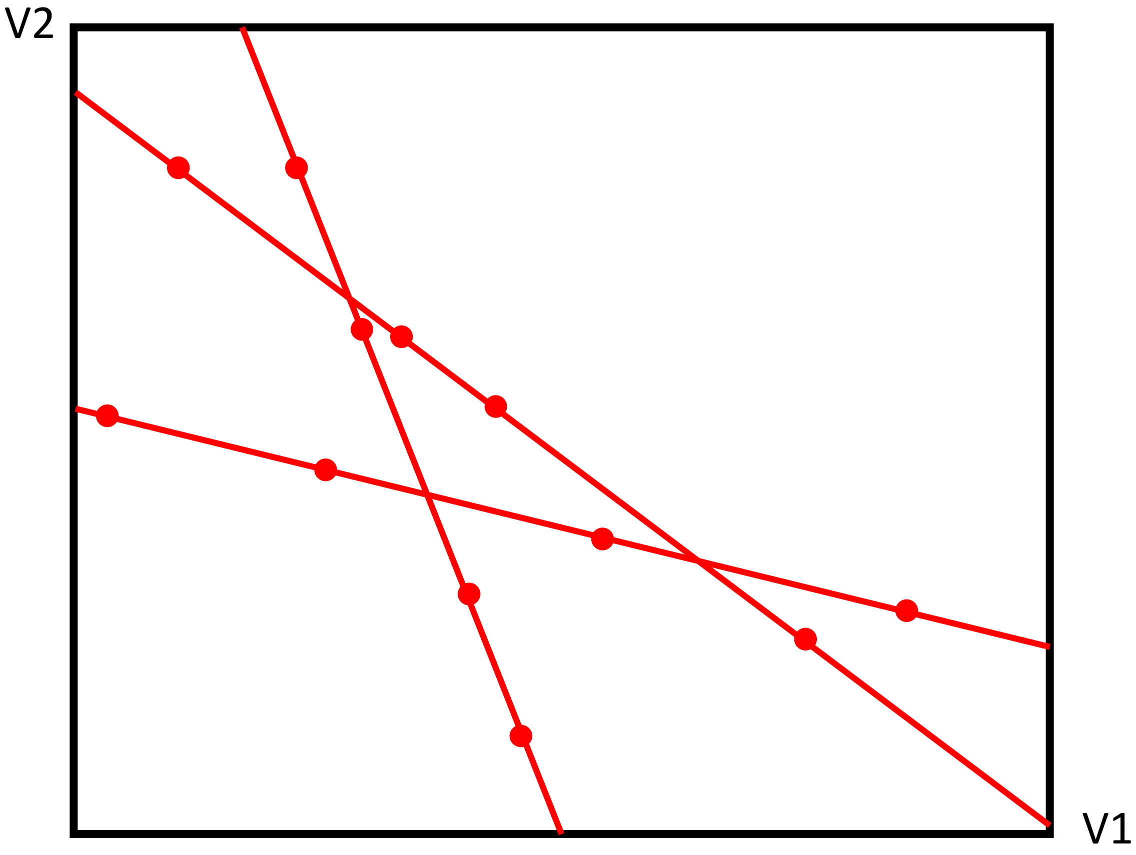


Model Extraction: Cryptanalysis

❖ Global methodology

- Search for points on the hyperplanes: the critical points
- Retrieve the equations of the hyperplane and the weights

$$\boldsymbol{\theta}_\eta \boldsymbol{V}(\boldsymbol{\eta}; \boldsymbol{X}) + \beta_\eta = 0$$



Model Extraction: Cryptanalysis

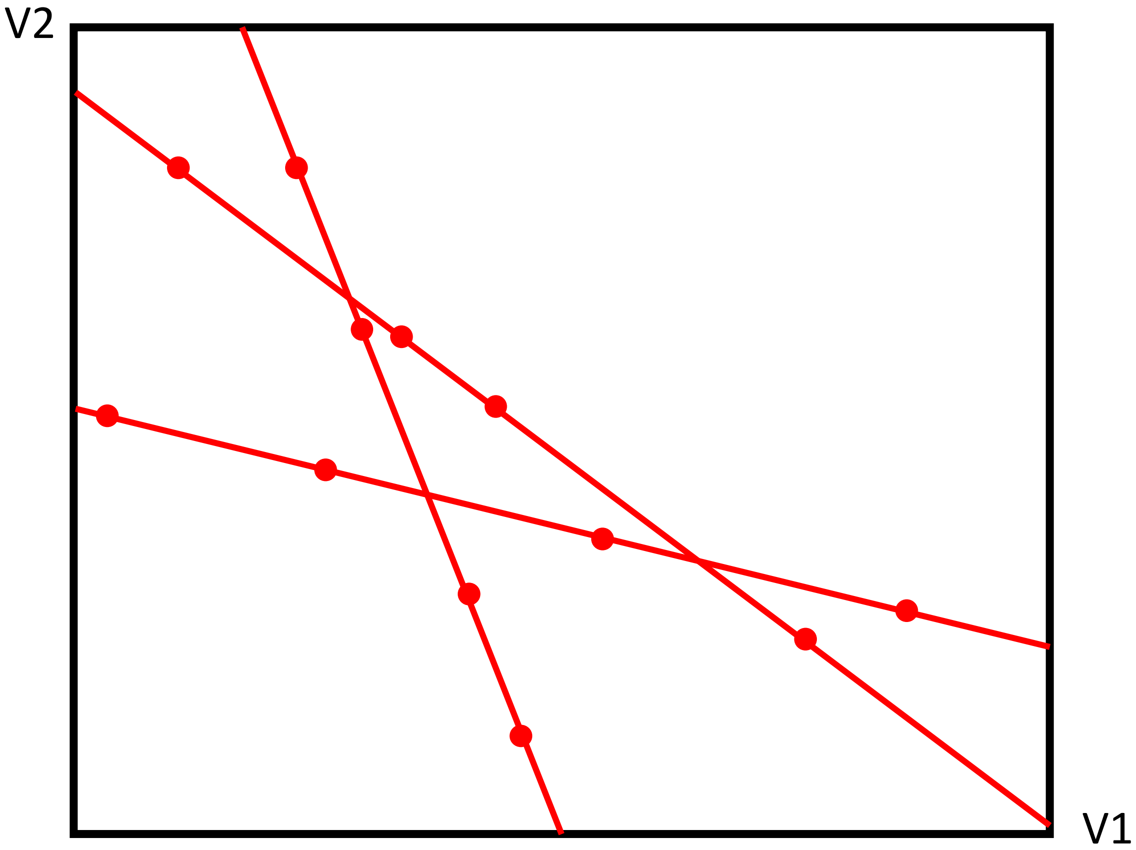
❖ Global methodology

- Search for points on the hyperplanes: the critical points

- Retrieve the equations of the hyperplane and the weights

$$\boldsymbol{\theta}_{\eta} \boldsymbol{V}(\boldsymbol{\eta}; \boldsymbol{X}) + \beta_{\eta} = 0$$

- Get the sign of the neuron

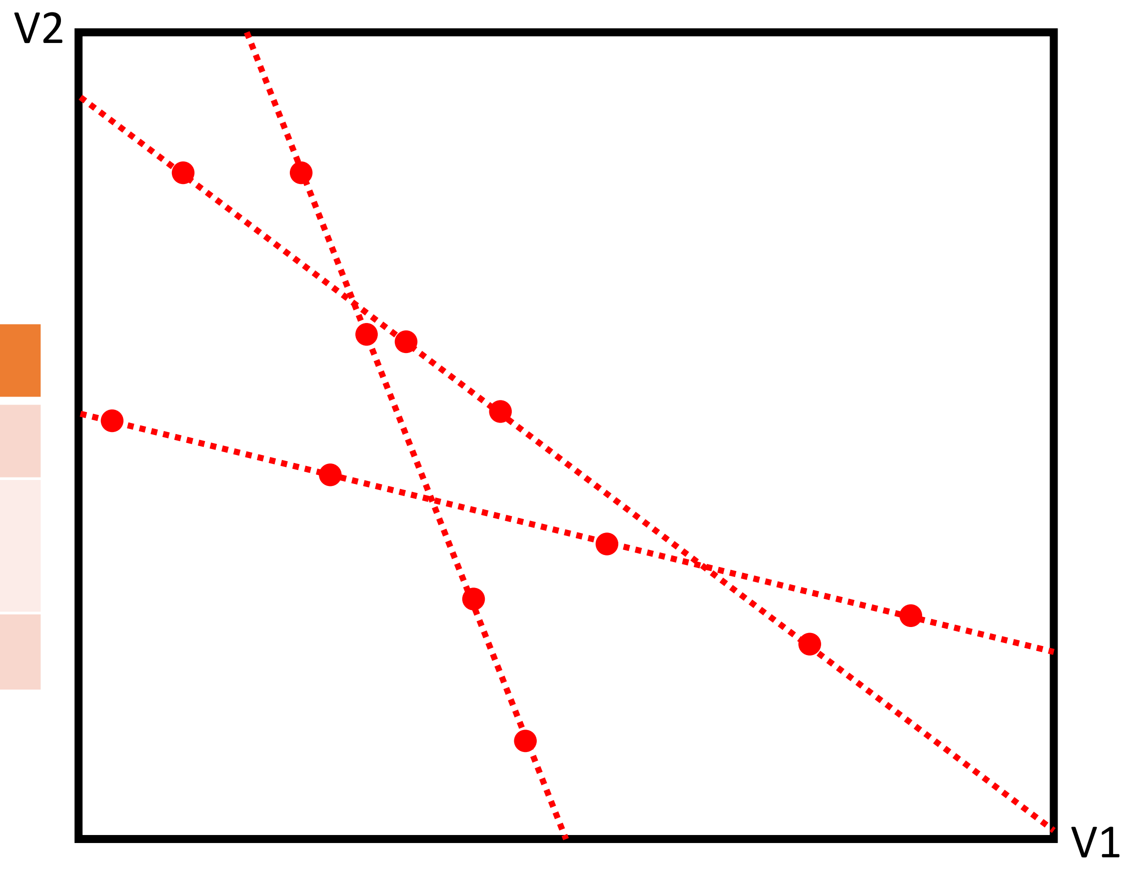


Model Extraction: Cryptanalysis

- Search for the critical points is the crucial step
  - Highly dependent on the gradient

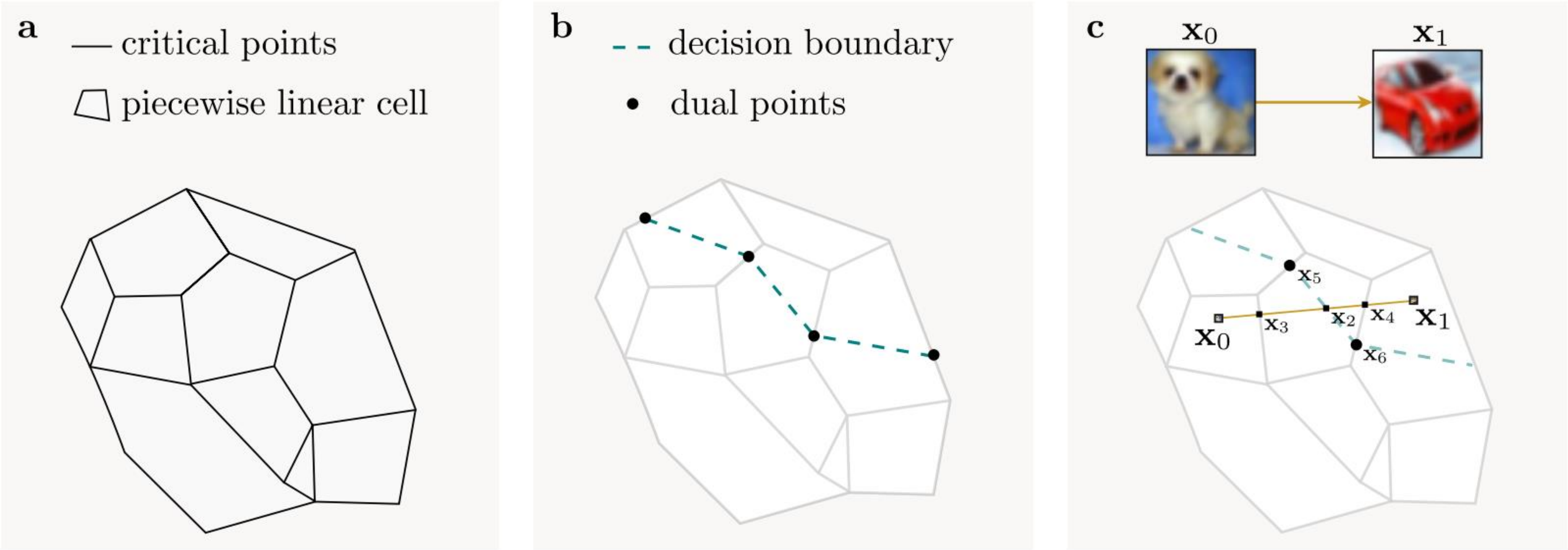
- Current limitations

Issue	Solution
Hard-label settings	Adaptation with dual points
Restriction to fully connected layers	None
Special cases of neurons	None



Model Extraction: Limitations

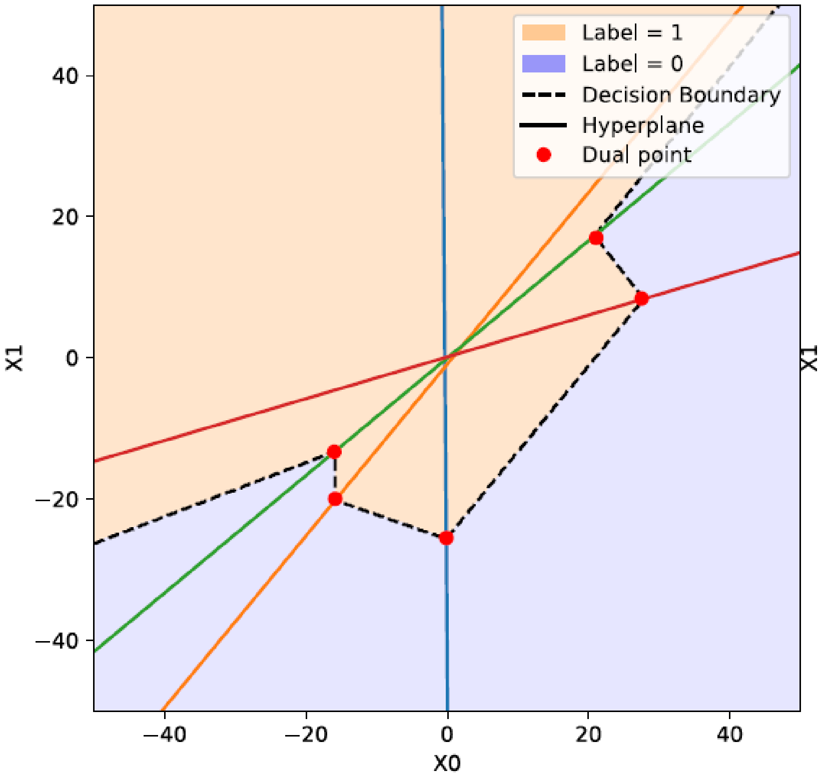
❖ Hard-label settings



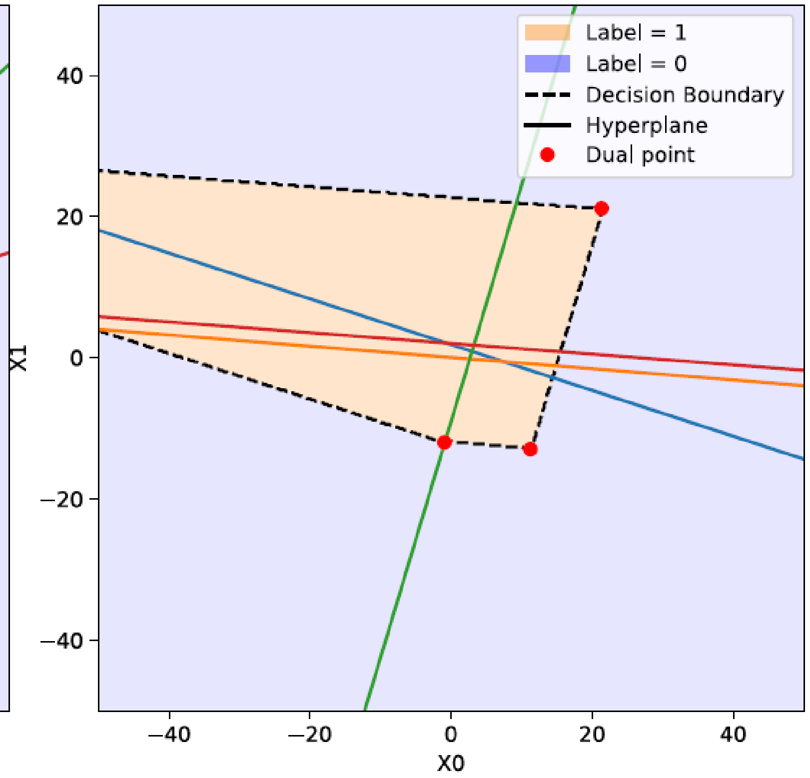
Polynomial Time Cryptanalytic Extraction of Deep Neural Networks in the Hard-Label Setting, EuroCrypt 2025

Model Extraction: Limitations

- ❖ Restriction to fully-connected layers



(a) Fully connected network

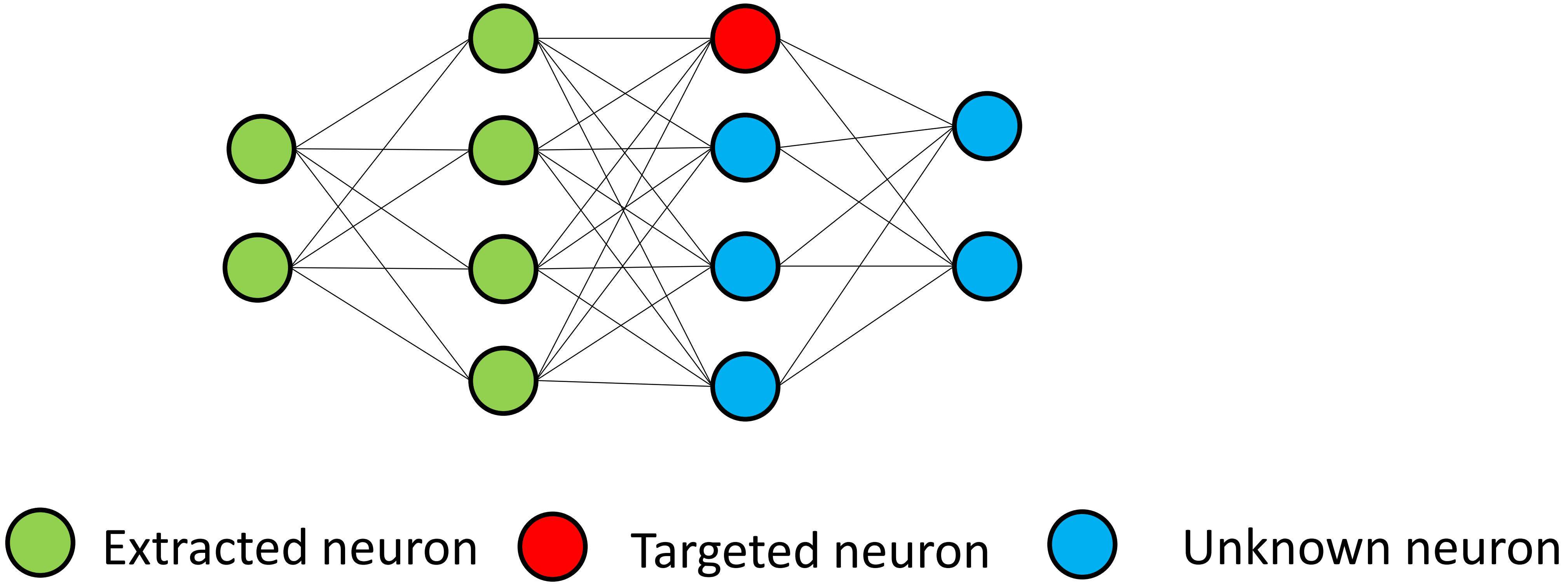


(b) DNN with a max pooling layer

- ❖ Wrong estimation of the dual points
- ❖ Pooling layer change the geometry of the decision boundary

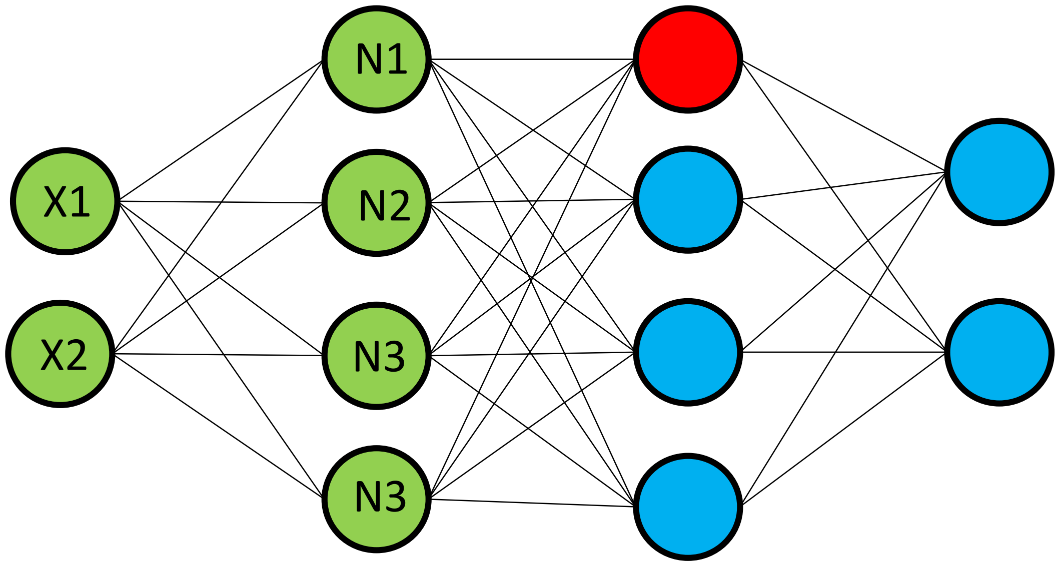
Model Extraction: Limitations

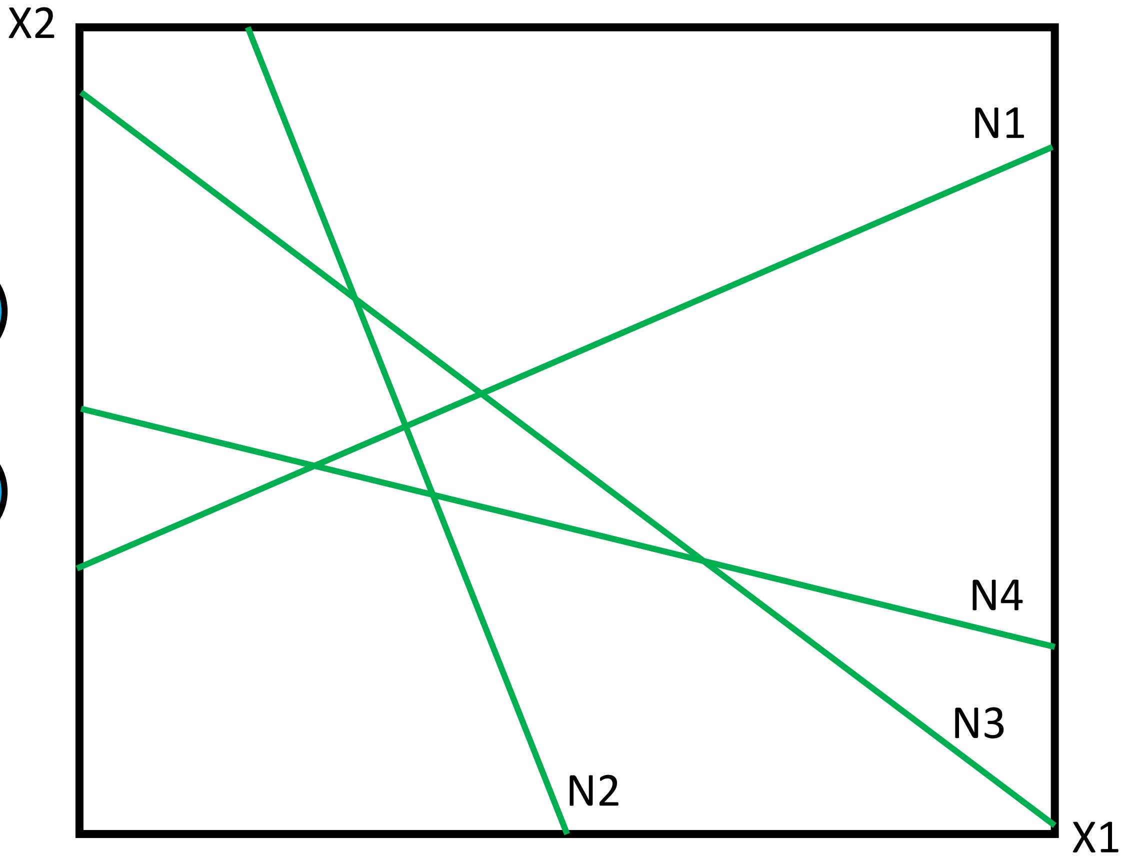
❖ Impact of special cases of neurons



Model Extraction: Limitations

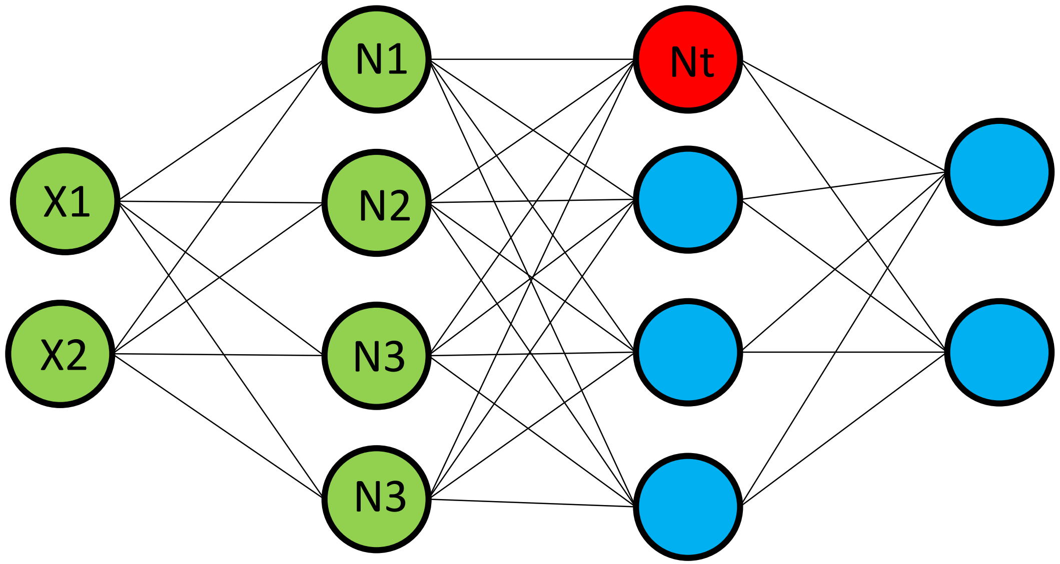
❖ Impact of special cases of neurons

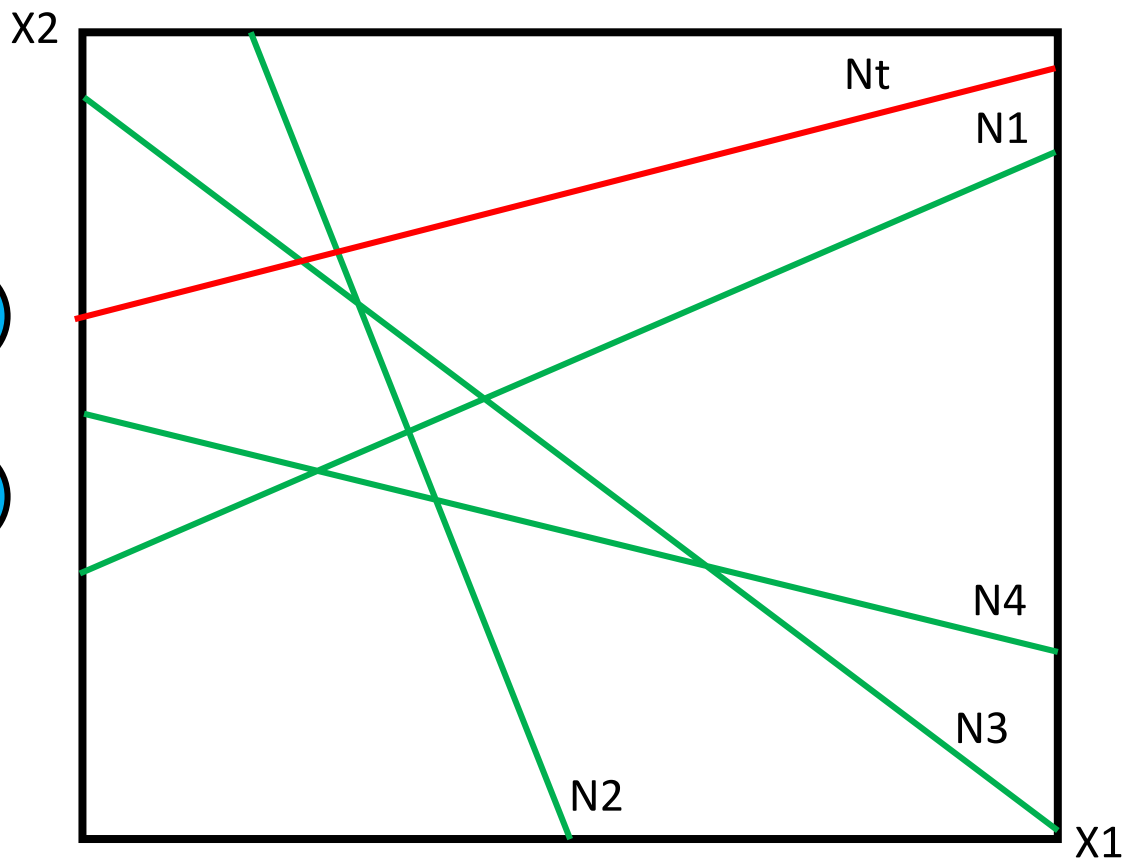




Model Extraction: Limitations

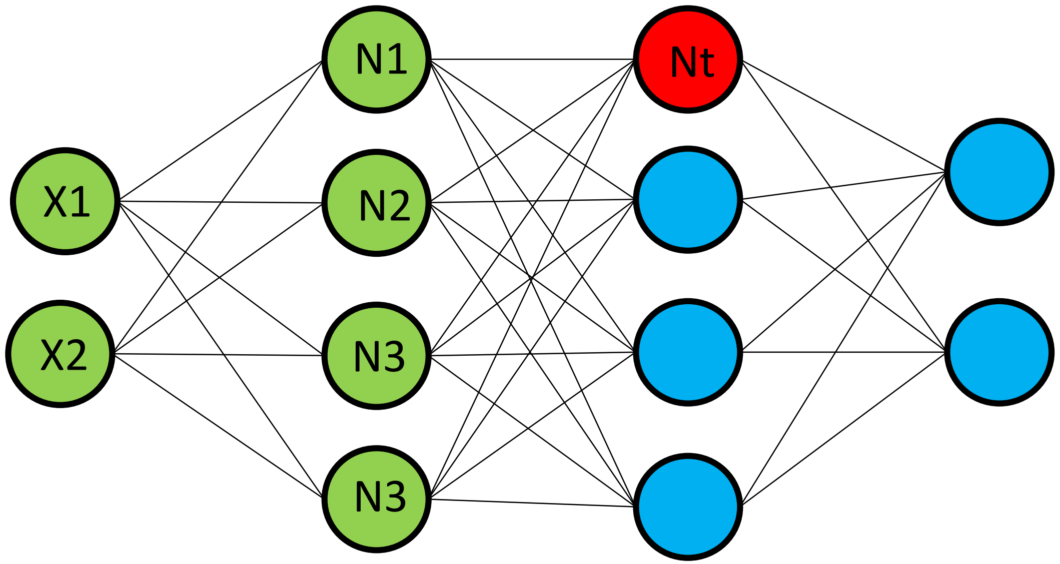
- Impact of special cases of neurons

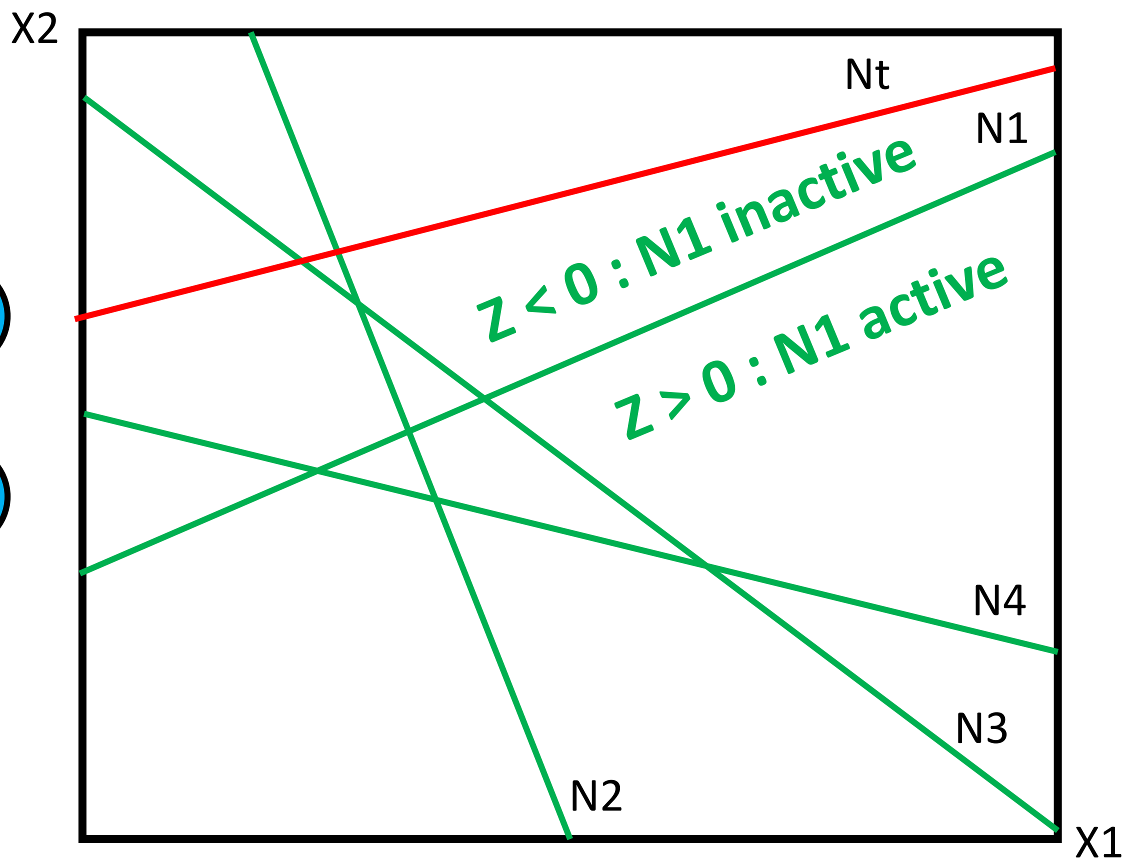




Model Extraction: Limitations

- Impact of special cases of neurons

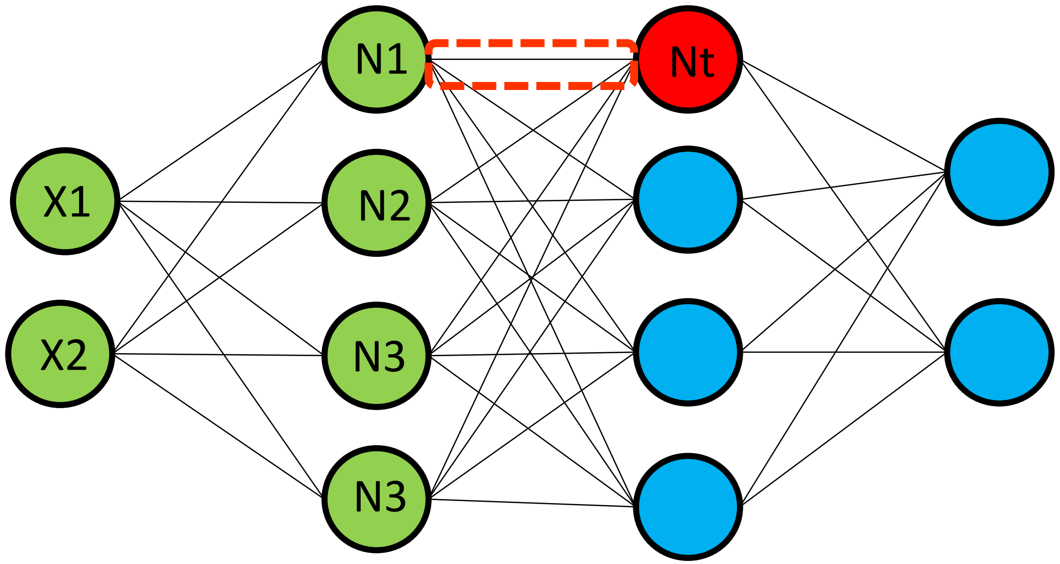


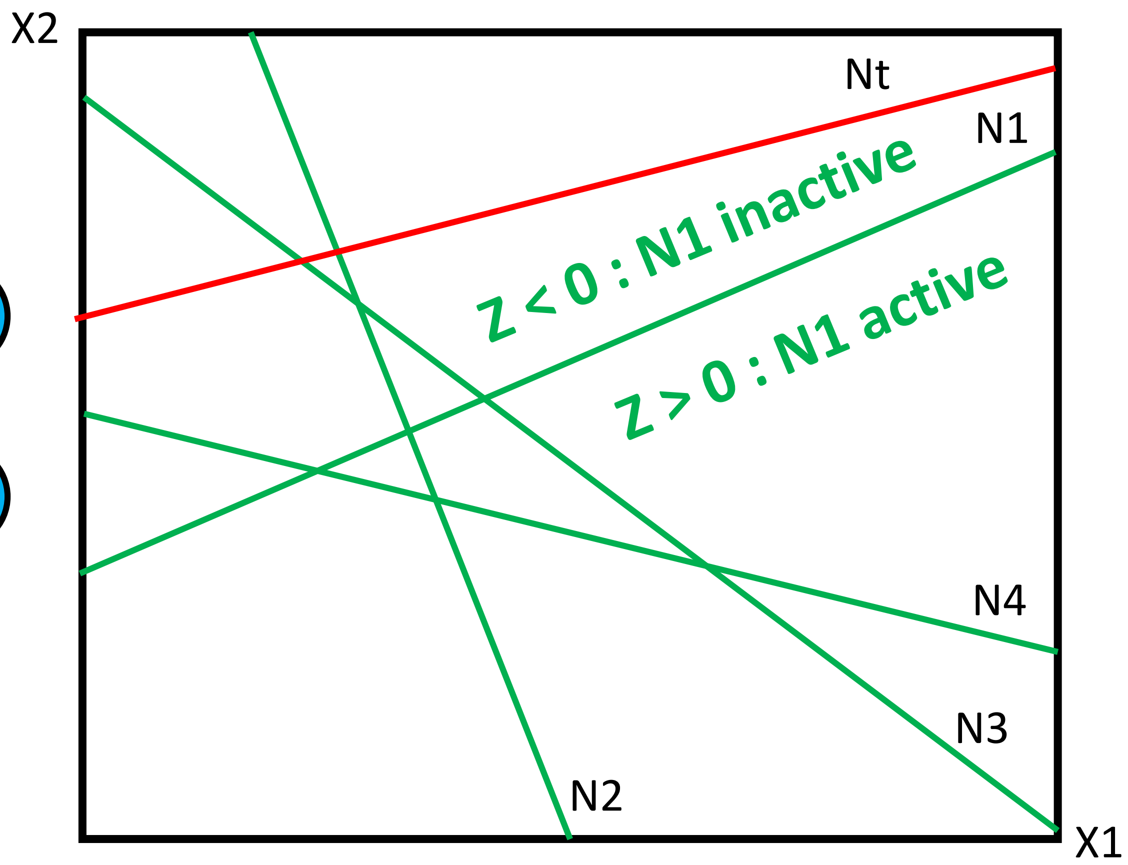


Model Extraction: Limitations

❖ Impact of special cases of neurons

Weight associated with this connection can never be estimated





Model Extraction: Cryptanalysis

❖ Current limitations

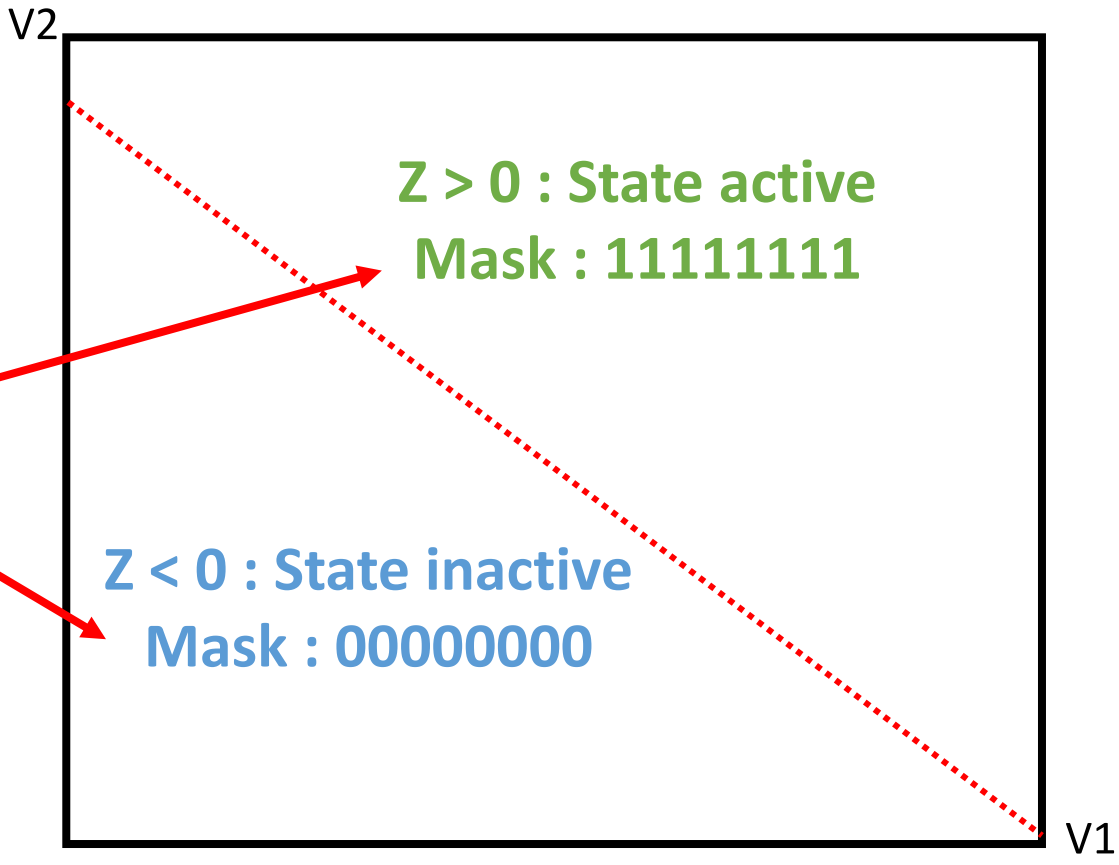
Issue	Solution
Hard-label settings	Adaptation with dual points
Restriction to fully connected layers	None
Special cases of neurons	None

Can we use side-channel to propose a robust framework for cryptanalytical extraction of complex DNN in hard-label settings ?

Model Extraction: Side-Channel

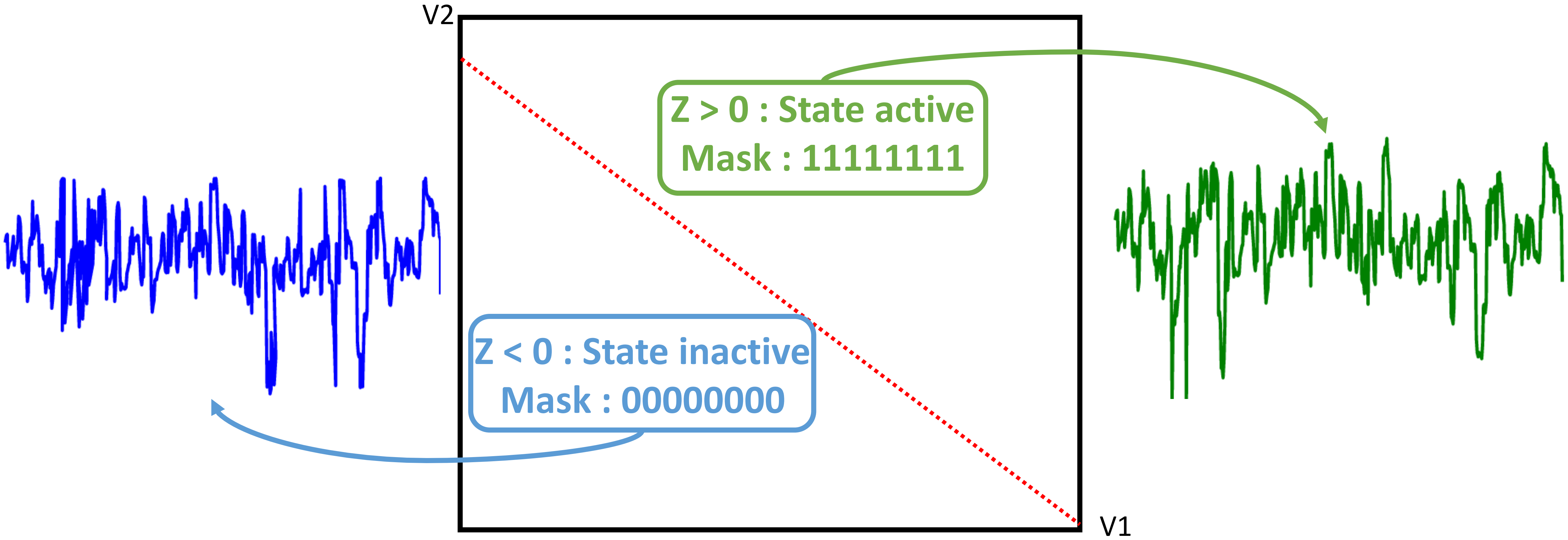
- ReLU implementation
  - ARM CMSIS-NN, open source

```
while (i)
{
    in = arm_nn_read_s8x4_ia((const int8_t **)&input);

    /* extract the first bit */
    buf = (int32_t)ROR((uint32_t)in & 0x80808080, 7);

    /* if MSB=1, mask will be 0xFF, 0x0 otherwise */
    mask = QSUB8(0x00000000, buf);

    arm_nn_write_s8x4_ia(&output, in & (~mask));

    i--;
}
```

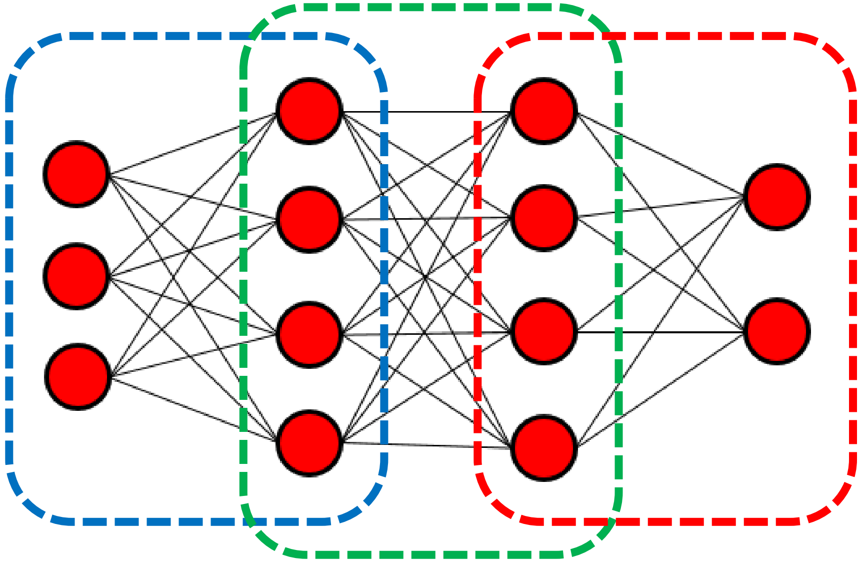


Model Extraction: Side-Channel

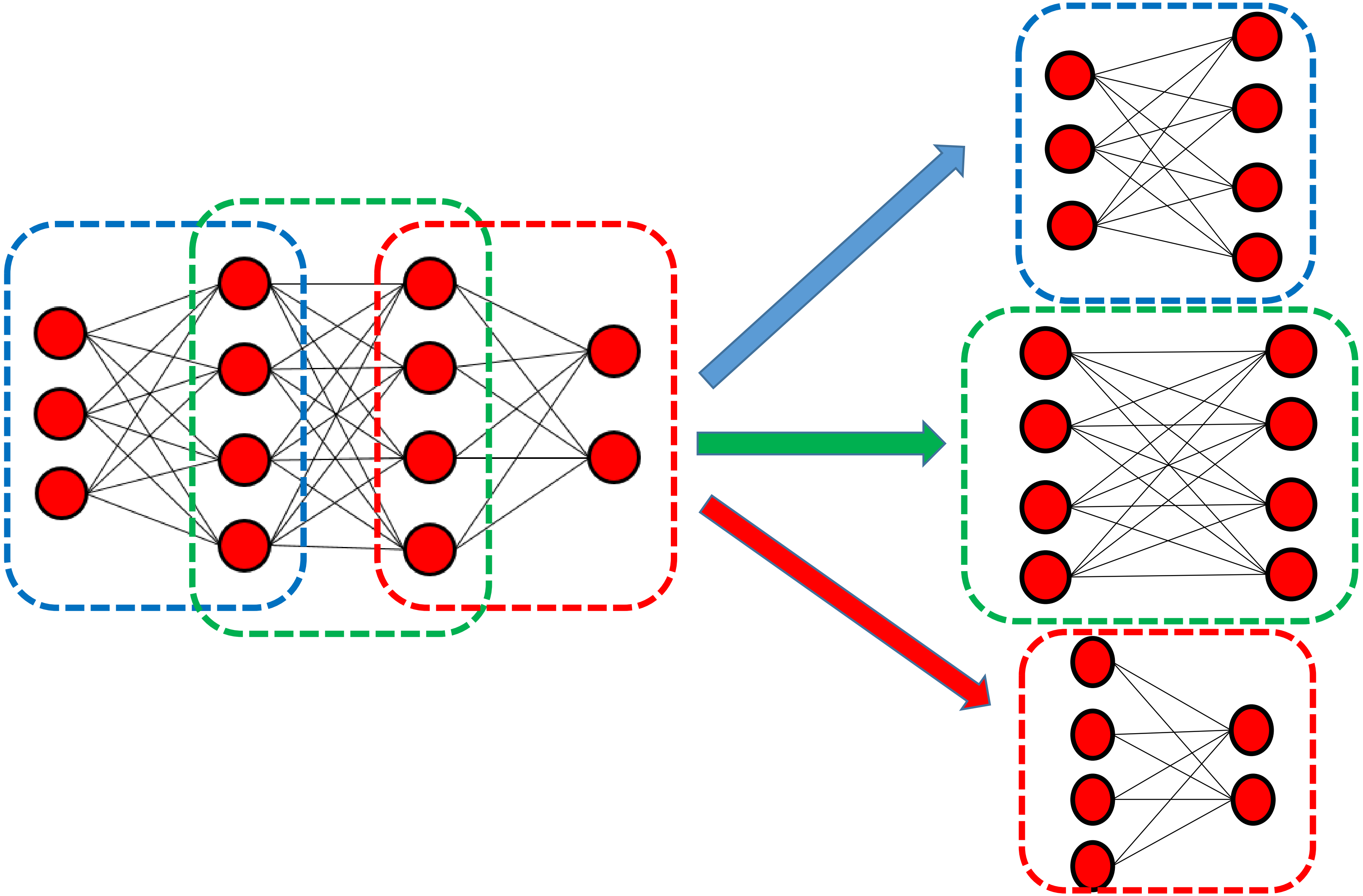
❖ Different states have different electromagnetic traces



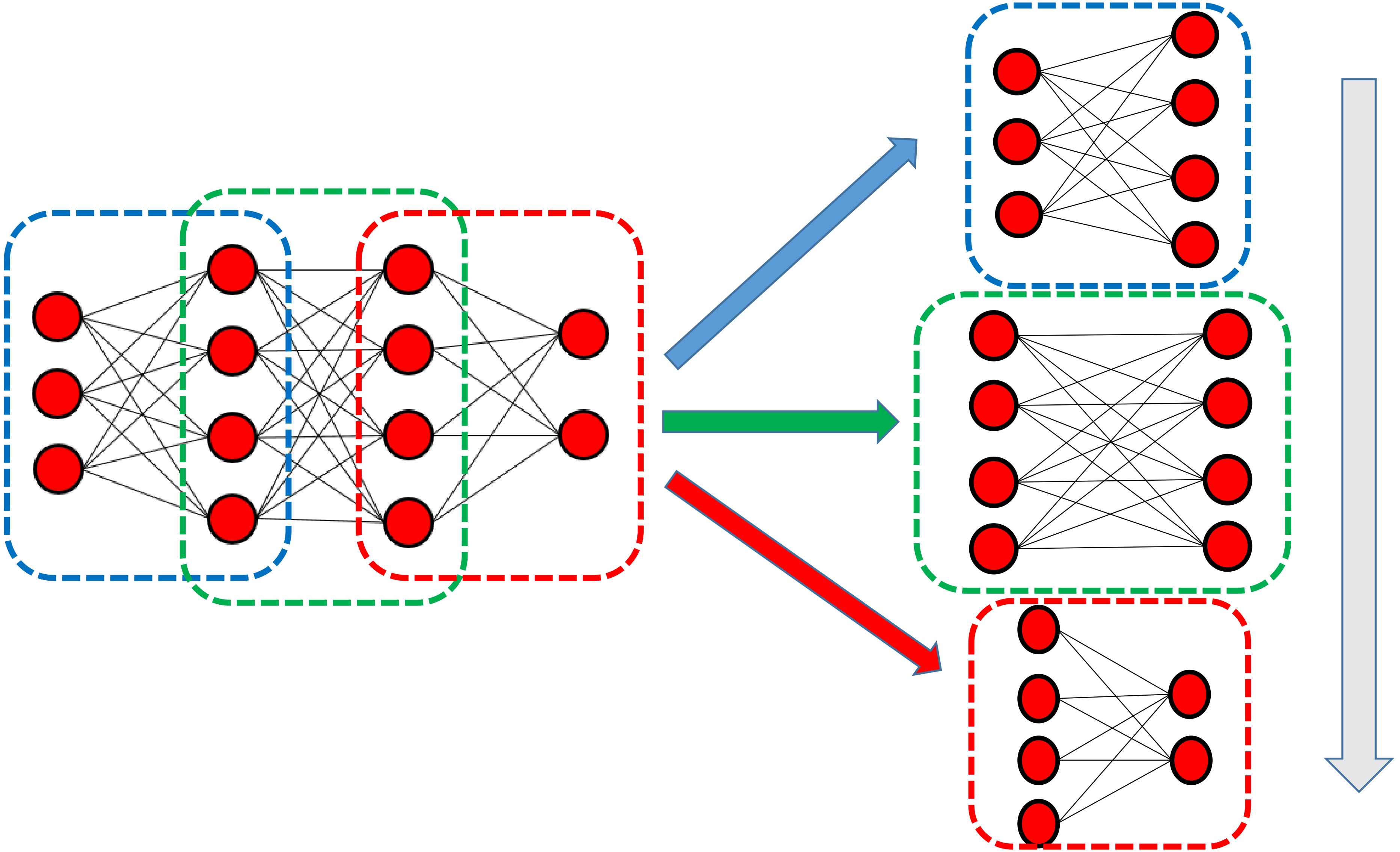
Model Extraction: Divide-And-Conquer



Model Extraction: Divide-And-Conquer



Model Extraction: Divide-And-Conquer



Sequential extraction of the subparts

Model Extraction: Divide-And-Conquer

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

- Takes advantage of the fact that the order is Conv – BN - Activation
- Allows to split the model at each layer

MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, 2017

Model Extraction: Divide-And-Conquer

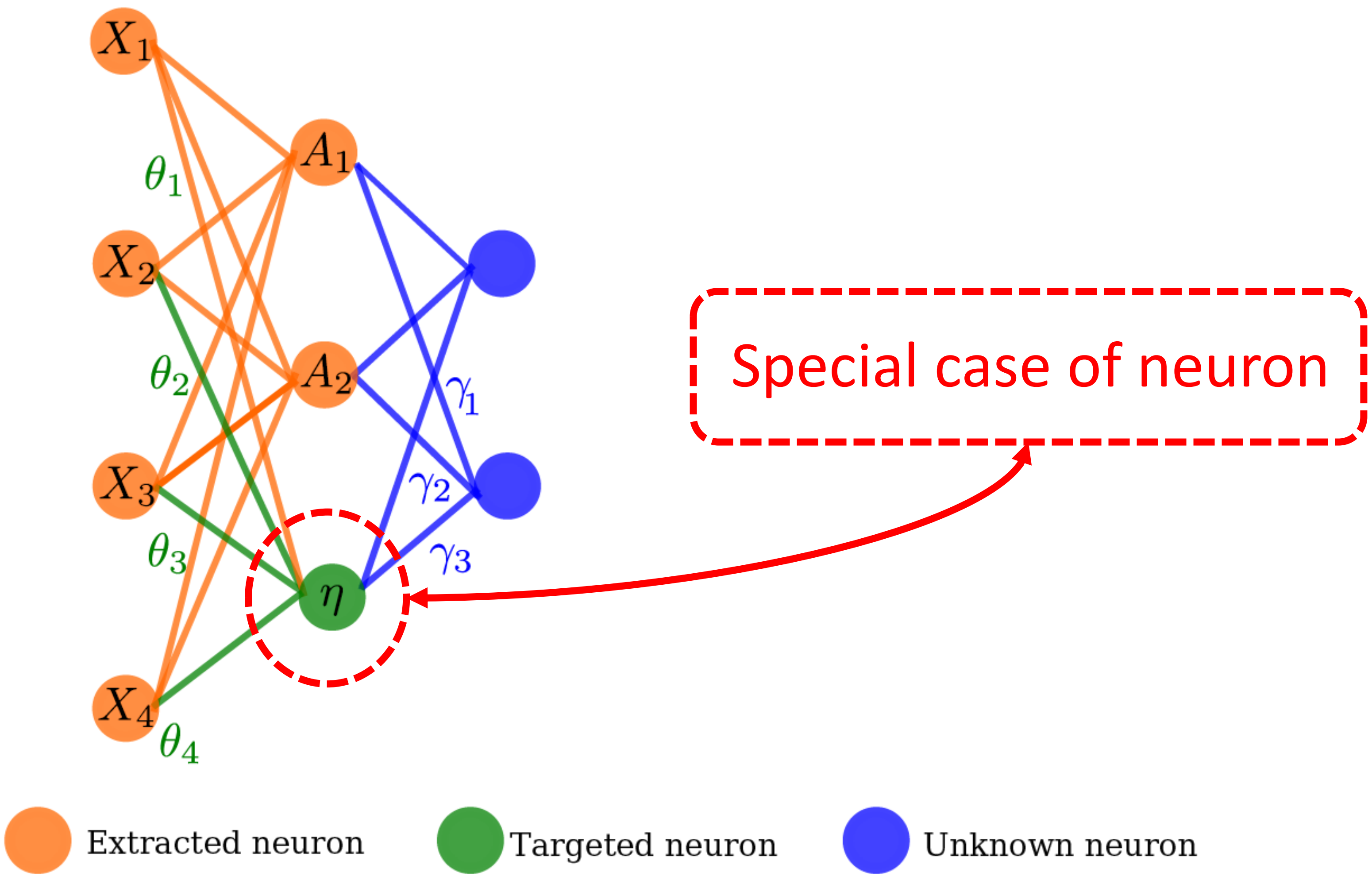
Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

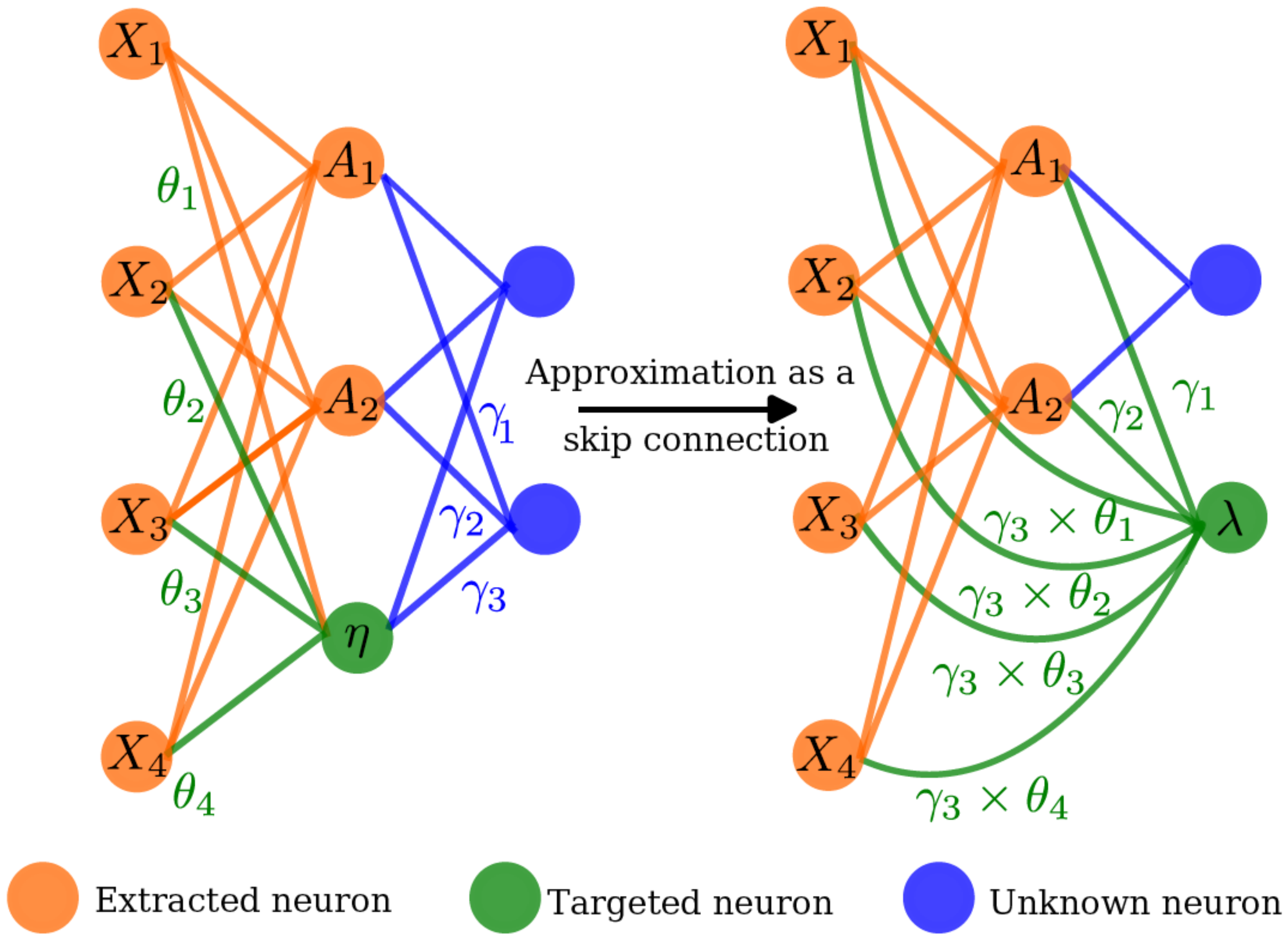
MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, 2017

- Subdivision only impacts the extraction of the last layer
- In most architecture the pooling is directly after the activation layer
  - Equivalent to a transformation on known inputs

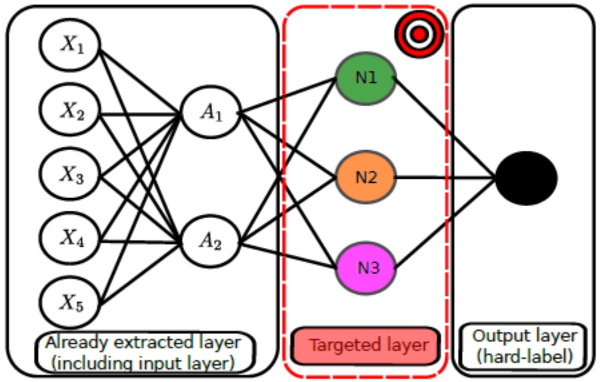
Model Extraction: Special Cases Of Neurons



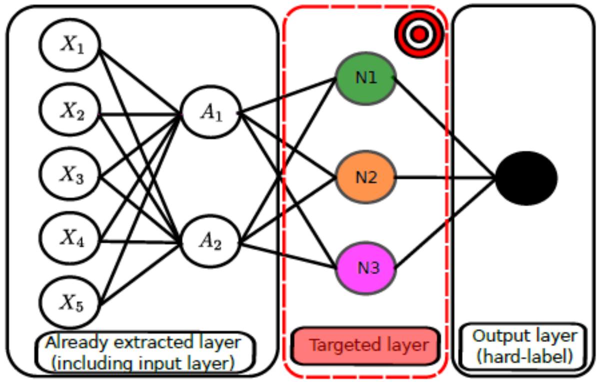
Model Extraction: Special Cases Of Neurons

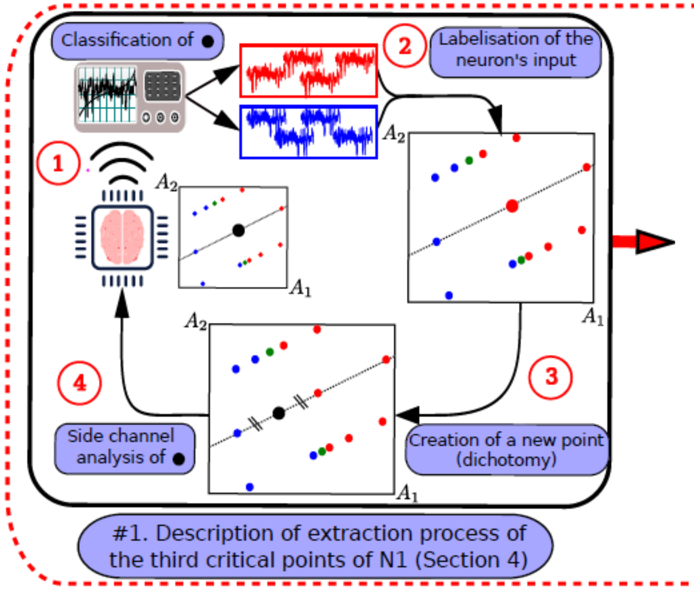


Model Extraction: Our Method

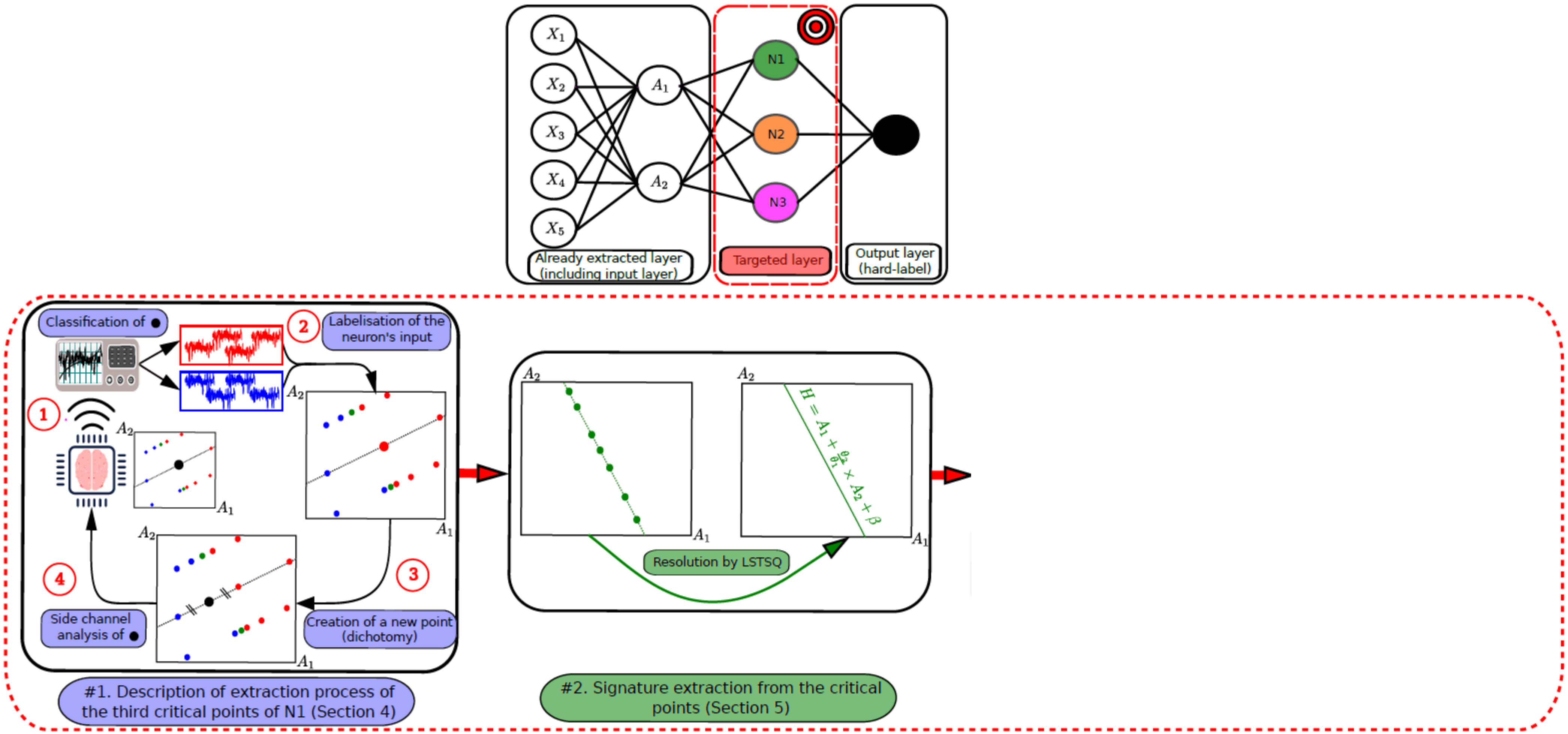


Model Extraction: Our Method

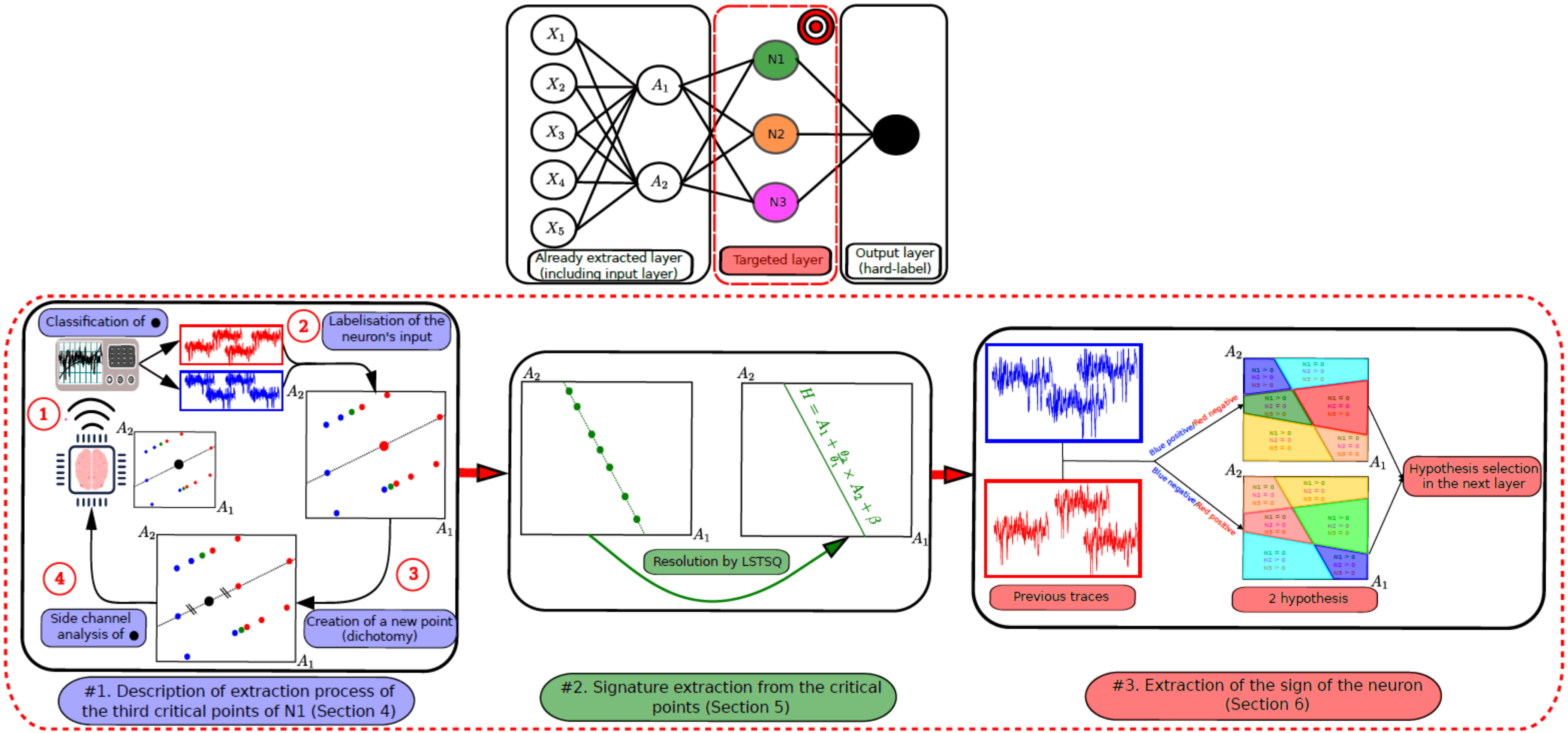




Model Extraction: Our Method



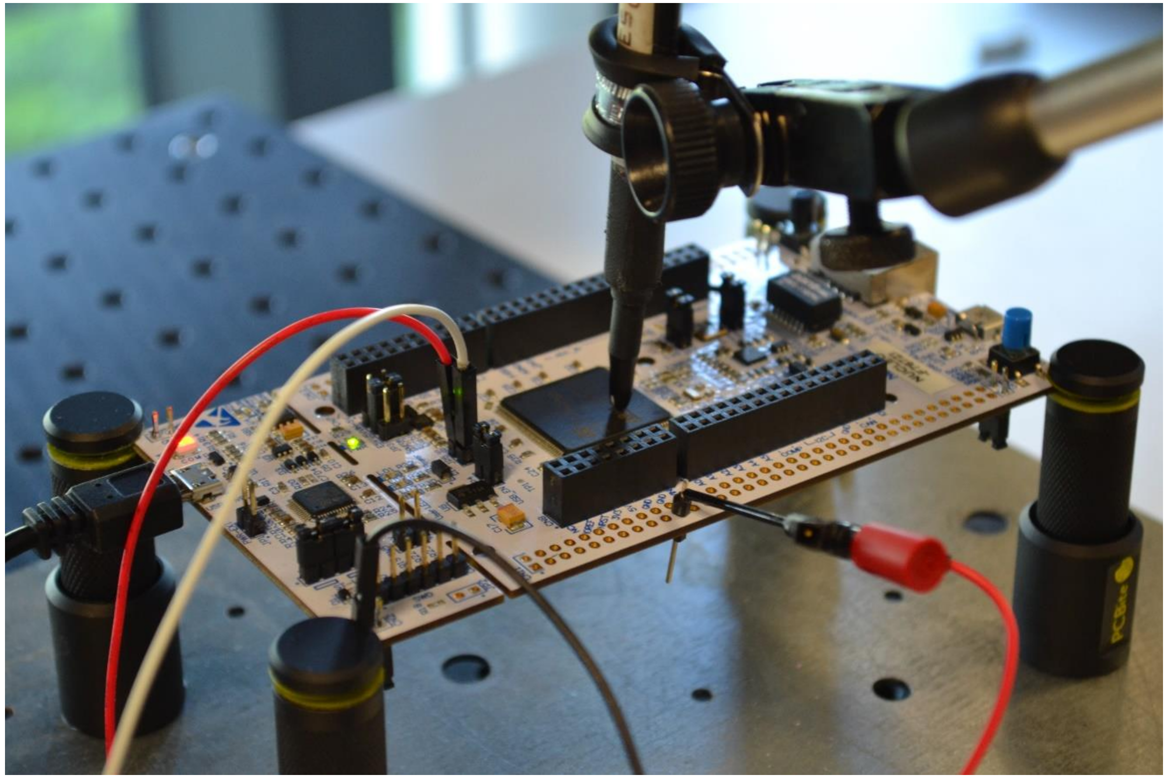
Model Extraction: Our Method



Model Extraction: Target

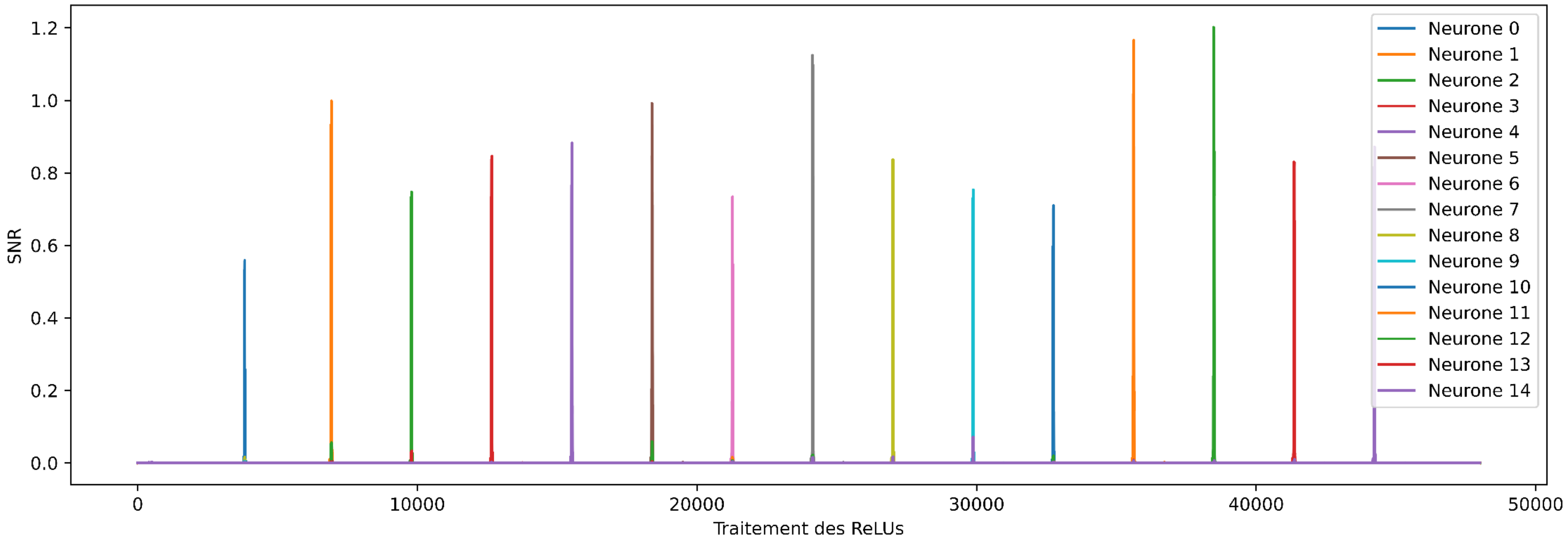
- Targeted DNN
  - Truncated version of MobileNetv1
  - 11 layers (Depthwise Separable convolutions + batchnorm + ReLU)

- Hardware
  - STM32F767ZI
  - X-Cube-AI



Model Extraction: Results

- State extraction for 15 neurons in a layer
  - Signal to noise ratio on the state of the neuron



- Success rate in one EM trace: 86.3% (k-means algorithm)

Model Extraction: Results

- Metrics used for classifier: Fidelity, Accuracy Under Attack and Number of queries

  - Fidelity: percentage of label agreement between the stolen and the targeted model (different from accuracy)

  - Accuracy Under Attack: transfer rate of adversarial examples generated on the stolen model to the target

  - Number of queries: number of random queries made to the targeted model (results are given under the assumption that the state of the neuron is obtained in one trace)

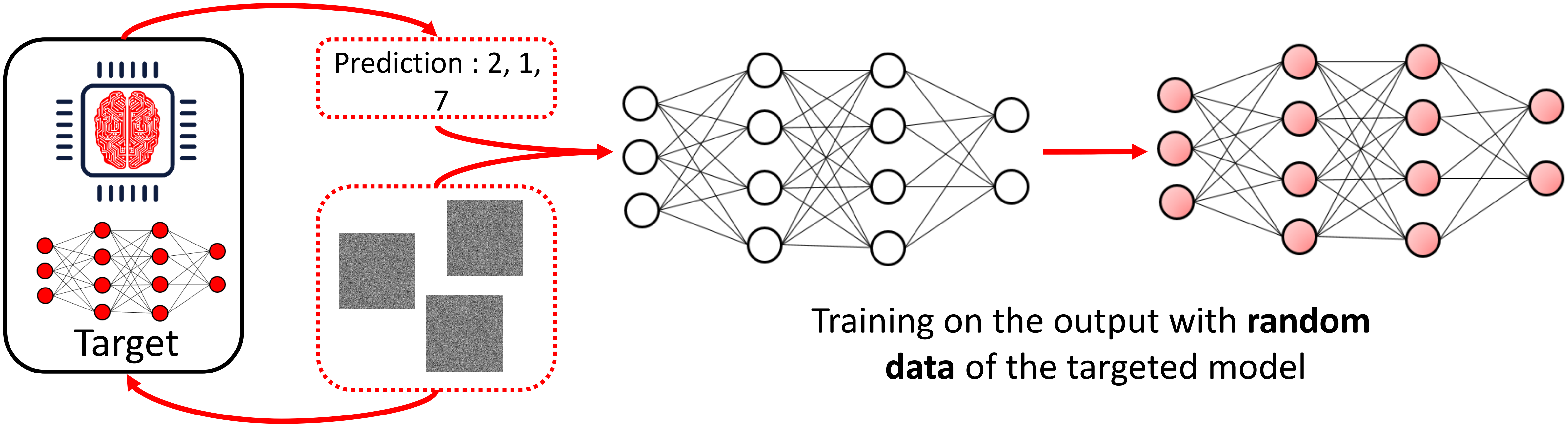
Model Extraction: Results

- Metrics used for classifier: Fidelity, Accuracy Under Attack and Number of queries

Architecture	Parameters	Number of queries	Fidelity	Accuracy Under Attack
3072-256-256-256-64-10	935 370	$2^{26.2}$	97.2%	98.6%
3072-512-256-64-10	1 721 802	$2^{26.0}$	93.2%	96.7%
Truncated MobileNetv1	5 234	$2^{18.8}$	88.4%	95.7%

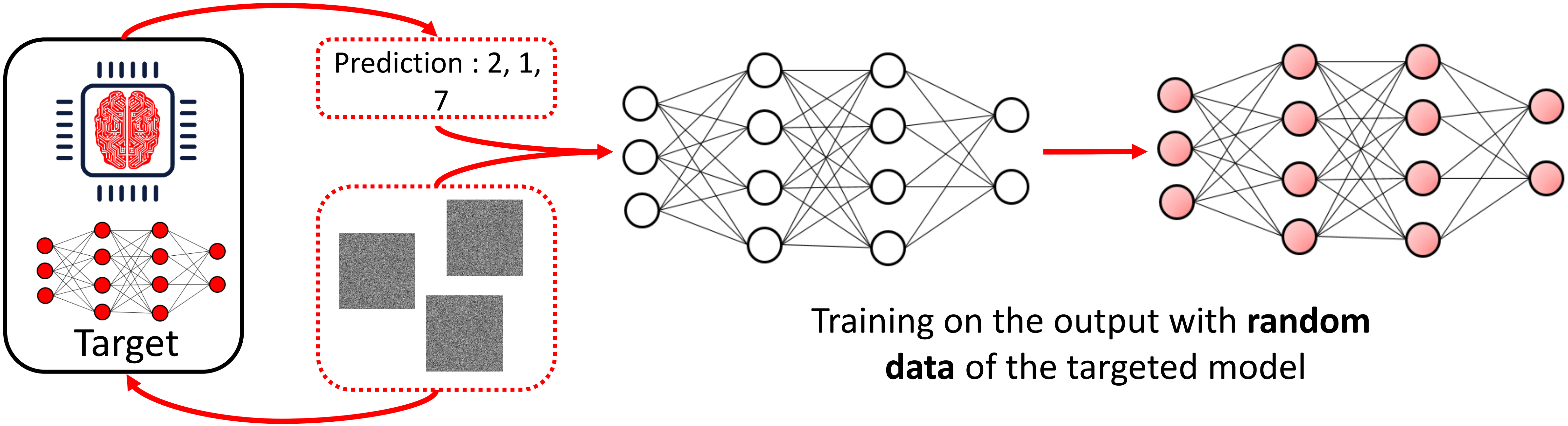
- One query corresponds to a prediction made by the model on random data ($2^{20} \sim 1\ 000\ 000$)

Model Extraction vs Active Learning



- Comparison with Simple Active Learning on the truncated MobileNetv1

Model Extraction vs Active Learning



Training on the output with **random data** of the targeted model

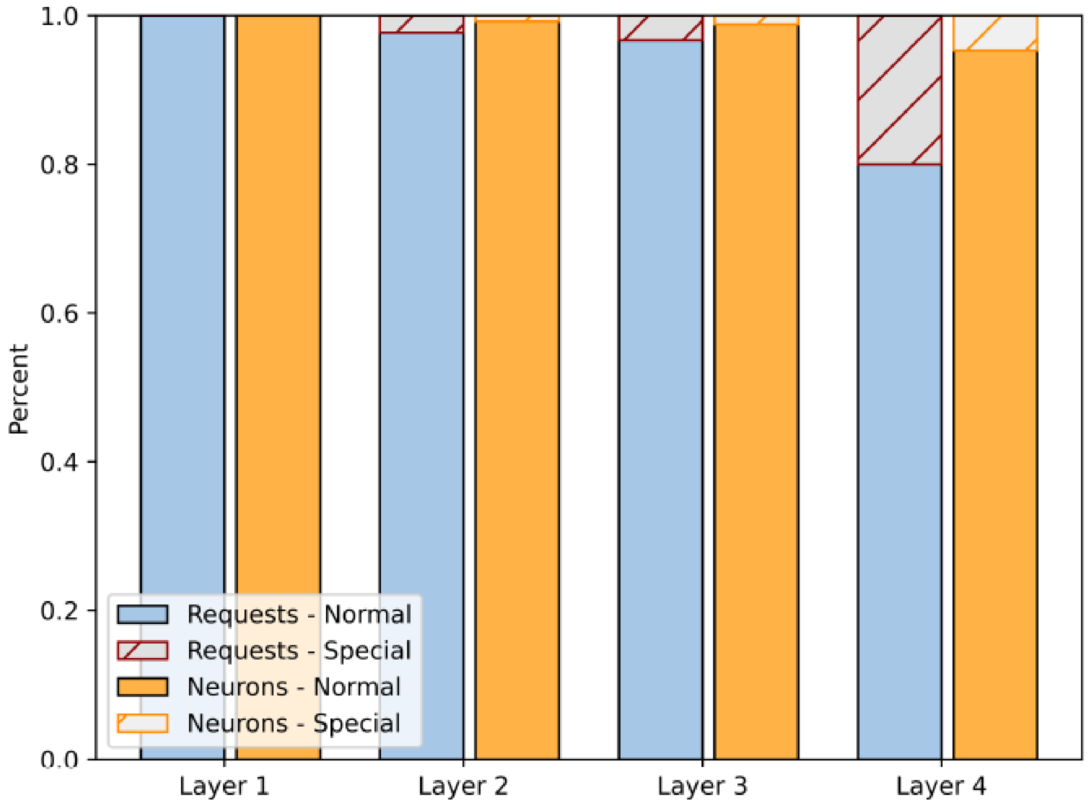
- Comparison with Simple Active Learning on the truncated MobileNetv1
  - Training with the same hyperparameters and a balanced dataset
  - Achieve 56% of accuracy on the random dataset
  - Accuracy of 19.6% and Fidelity of 21.1% on the CIFAR-10 dataset

Model Extraction: Special Case Neurons

- Number of special neurons
  - Increases with the depth of the layer
  - Most of them correspond to input-off
  - Framework improves the efficiency on their extraction
  - Trade off between requests and precision

- Number of request for these neurons

Metrics associated with the special neurons for the 3072-512-256-64-10 MLP



Model Extraction: Limitations Of The Method

Evolution of the error on the truncated MobileNetv1

Datatype	Metrics	L0	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11
32 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-18.9}$	$2^{-17.6}$	$2^{-7.9}$	$2^{-18.2}$	$2^{-7.6}$	$2^{-13.9}$	$2^{-9.9}$	$2^{-11.4}$	$2^{-8.8}$	$2^{-6.7}$	$2^{-4.3}$	$2^{3.7}$
64 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-46.6}$	$2^{-43.8}$	$2^{-37.4}$	$2^{-34.2}$	$2^{-29.0}$	$2^{-27.1}$	$2^{-26.0}$	$2^{-26.8}$	$2^{-23.1}$	$2^{-22.7}$	$2^{-15.3}$	$2^{3.8}$

- Propagation of error between the layers
  - Small error on the estimation of the weights
  - Dependent on the data format
  - Accumulate from one layer to another

- Maximum number of layers that can be extracted (dependent on the data format)

Model Extraction: Limitations Of The Method

Evolution of the error on the truncated MobileNetv1

Datatype	**Metrics**	**L0**	**L1**	**L2**	**L3**	**L4**	**L5**	**L6**	**L7**	**L8**	**L9**	**L10**	**L11**
32 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-18.9}$	$2^{-17.6}$	$2^{-7.9}$	$2^{-18.2}$	$2^{-7.6}$	$2^{-13.9}$	$2^{-9.9}$	$2^{-11.4}$	$2^{-8.8}$	$2^{-6.7}$	$2^{-4.3}$	$2^{3.7}$
64 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-46.6}$	$2^{-43.8}$	$2^{-37.4}$	$2^{-34.2}$	$2^{-29.0}$	$2^{-27.1}$	$2^{-26.0}$	$2^{-26.8}$	$2^{-23.1}$	$2^{-22.7}$	$2^{-15.3}$	$2^{3.8}$

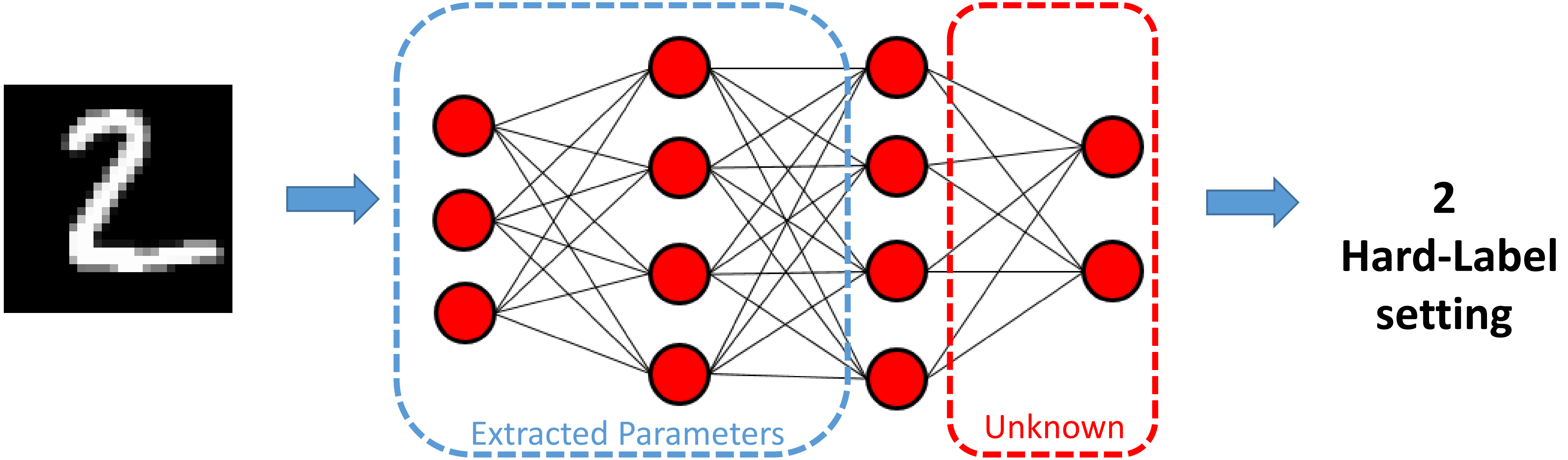
- Impact of the last layer
  - Error increases by a factor of 256 for 32-bit data and by a factor of nearly 600 000 for 64-bit data
  - Fidelity remains at 88.4% between the targeted model and the stolen one

Model Extraction: Limitations Of The Method

Evolution of the error on the truncated MobileNetv1

Datatype	**Metrics**	**L0**	**L1**	**L2**	**L3**	**L4**	**L5**	**L6**	**L7**	**L8**	**L9**	**L10**	**L11**
32 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-18.9}$	$2^{-17.6}$	$2^{-7.9}$	$2^{-18.2}$	$2^{-7.6}$	$2^{-13.9}$	$2^{-9.9}$	$2^{-11.4}$	$2^{-8.8}$	$2^{-6.7}$	$2^{-4.3}$	$2^{3.7}$
64 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-46.6}$	$2^{-43.8}$	$2^{-37.4}$	$2^{-34.2}$	$2^{-29.0}$	$2^{-27.1}$	$2^{-26.0}$	$2^{-26.8}$	$2^{-23.1}$	$2^{-22.7}$	$2^{-15.3}$	$2^{3.8}$

- Impact of the last layer

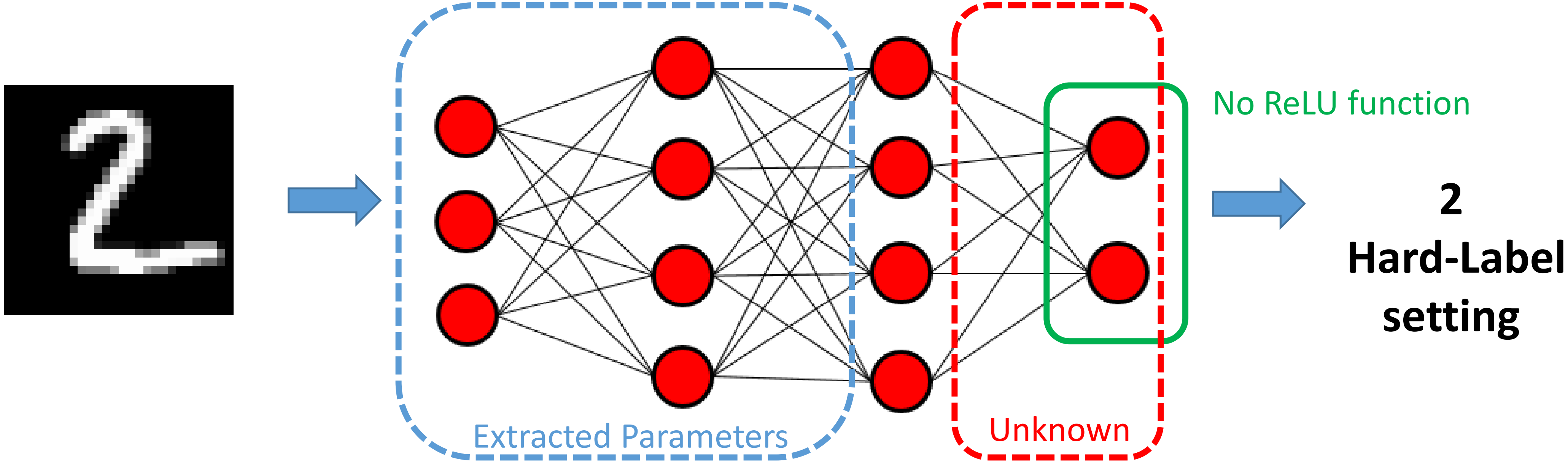


Model Extraction: Limitations Of The Method

Evolution of the error on the truncated MobileNetv1

Datatype	Metrics	L0	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11
32 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-18.9}$	$2^{-17.6}$	$2^{-7.9}$	$2^{-18.2}$	$2^{-7.6}$	$2^{-13.9}$	$2^{-9.9}$	$2^{-11.4}$	$2^{-8.8}$	$2^{-6.7}$	$2^{-4.3}$	$2^{3.7}$
64 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-46.6}$	$2^{-43.8}$	$2^{-37.4}$	$2^{-34.2}$	$2^{-29.0}$	$2^{-27.1}$	$2^{-26.0}$	$2^{-26.8}$	$2^{-23.1}$	$2^{-22.7}$	$2^{-15.3}$	$2^{3.8}$

- Impact of the last layer



Model Extraction: Limitations Of The Method

Evolution of the error on the truncated MobileNetv1

Datatype	Metrics	L0	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11
32 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-18.9}$	$2^{-17.6}$	$2^{-7.9}$	$2^{-18.2}$	$2^{-7.6}$	$2^{-13.9}$	$2^{-9.9}$	$2^{-11.4}$	$2^{-8.8}$	$2^{-6.7}$	$2^{-4.3}$	$2^{3.7}$
64 bits	$max\|\theta - \hat{\theta}\|^L$	$2^{-46.6}$	$2^{-43.8}$	$2^{-37.4}$	$2^{-34.2}$	$2^{-29.0}$	$2^{-27.1}$	$2^{-26.0}$	$2^{-26.8}$	$2^{-23.1}$	$2^{-22.7}$	$2^{-15.3}$	$2^{3.8}$

- Impact of the last layer
  - Extraction via supervised learning
  - Dataset composed of the activation of the previous layer and the hard-label
  - Cause major drop in fidelity
    - Hybrid model composed of the first eleventh extracted layer and the true last layer
    - Achieve 99.6% of fidelity

Model Extraction: Results

- Results from simulation with 64-bits data for regression tasks

Architecture (Regression task)	Parameters	Number of queries	$max\lvert\theta - \hat{\theta}\rvert$
784-128-1	100 480	x2 $2^{22.6}$ $\mathbf{2^{21.5}}$ [5]	x2 700 $\mathbf{2^{-40.8}}$ $2^{-29.4}$ [5]
10-20-20-1	620	x4 $\mathbf{2^{15.6}}$ $2^{17.1}$ [5]	x700 $\mathbf{2^{-46.5}}$ 2^{-37} [5]
40-20-10-10-1	1 110	x2 $\mathbf{2^{16.8}}$ $2^{17.8}$ [5]	x32 000 $\mathbf{2^{-42.0}}$ $2^{-27.1}$ [5]

- One query corresponds to a prediction made by the model on random data ($2^{20} \sim 1\ 000\ 000$; $2^{-41} \sim 4 \times 10^{-13}$)

Conclusion

- Conclusion
  - Fidelity-based model extraction of a complex DNN in hard-label settings
  - Complementarity between hardware and software attacks
  - Paper under review
  - Extend this work on more complex architecture
  - Evaluate the impact of the data representation on the attack

- ST was noticed in September 2024

References

[1] Tramèr, Florian et al. “Stealing Machine Learning Models via Prediction APIs.” *USENIX Security Symposium* (2016).
[2] Rakin, Adnan Siraj et al. “DeepSteal: Advanced Model Extractions Leveraging Efficient Weight Stealing in Memories.” *2022 IEEE Symposium on Security and Privacy (SP)* (2021): 1157-1174.
[3] Batina, Lejla et al. “CSI NN: Reverse Engineering of Neural Network Architectures Through Electromagnetic Side Channel.” *USENIX Security Symposium* (2019).
[4] Jagielski, Matthew et al. “High Accuracy and High Fidelity Extraction of Neural Networks.” *USENIX Security Symposium* (2019).
[5] Carlini, Nicholas et al. “Cryptanalytic Extraction of Neural Network Models.” *Annual International Cryptology Conference* (2020).
[6] Shamir, Adi et al. “Polynomial Time Cryptanalytic Extraction of Neural Network Models.” *IACR Cryptol. ePrint Arch.* 2023 (2023): 1526.
[7] Rolnick, David and Konrad Paul Kording. “Reverse-engineering deep ReLU networks.” *International Conference on Machine Learning* (2019).
[8] Carlini, Nicholas et al. “Polynomial Time Cryptanalytic Extraction of Deep Neural Networks in the Hard-Label Setting.” *IACR Cryptology ePrint Archive* (2024).

Appendix

- Complete results with 32-bit data

Architecture	Parameters	Queries	$\max\lvert\Delta_\theta\rvert^L$
784-32-1	25, 120	$2^{19.8}$	$2^{-17.7}$
784-128-1	100, 480	$2^{21.7}$	$2^{-17.4}$
10-10-10-1	210	$2^{13.0}$	$2^{-18.2}$
10-20-20-1	620	$2^{14.5}$	$2^{-17.8}$
40-20-10-10-1	1, 110	$2^{16.4}$	$2^{-12.1}$
80-40-20-1	4, 020	$2^{19.1}$	$2^{-14.8}$

Appendix

❖ Complete results with 64-bit data

Architecture	Parameters	Approach	Queries	$\max\|\Delta_\theta\|^L$
10-10-10-1	210	[5] [7] **This work**	$2^{16.0}$ $2^{22.0}$ $\mathbf{2^{15.6}}$	$2^{-36.0}$ $2^{-12.0}$ $\mathbf{2^{-46.2}}$
10-20-20-1	620	[5] **This work**	$2^{17.1}$ $\mathbf{2^{15.6}}$	$2^{-37.0}$ $\mathbf{2^{-46.5}}$
40-20-10-10-1	$1,110$	[5] **This work**	$2^{17.8}$ $\mathbf{2^{16.8}}$	$2^{-27.1}$ $\mathbf{2^{-42.0}}$
80-40-20-1	$4,020$	[5] **This work**	$2^{18.5}$ $\mathbf{2^{18.3}}$	$2^{-39.7}$ $\mathbf{2^{-44.2}}$
784-32-1	$25,120$	[5] [4] **This work**	$2^{19.2}$ $\mathbf{2^{18.2}}$ $2^{20.6}$	$2^{-30.2}$ $2^{-1.7}$ $\mathbf{2^{-43.5}}$
784-128-1	$100,480$	[5] **This work**	$\mathbf{2^{21.5}}$ $2^{22.6}$	$2^{-24.7}$ $\mathbf{2^{-40.8}}$