



## Deep Learning for Embedded Security Evaluation

Emmanuel Prouff<sup>1</sup>

<sup>1</sup>Laboratoire de Sécurité des Composants, ANSSI, France

April 2018, CISCO



## Contents

1. Context and Motivation
  - 1.1 Illustration
  - 1.2 Template Attacks
  - 1.3 Countermeasures
2. A machine learning approach to classification
  - 2.1 Introduction
  - 2.2 The Underlying Classification Problem
  - 2.3 Convolutional Neural Networks
  - 2.4 Training of Models
3. Building a Community Around The Subject
  - 3.1 ASCAD Open Data-Base
  - 3.2 Leakage Characterization and Training
  - 3.3 Results
4. Conclusions



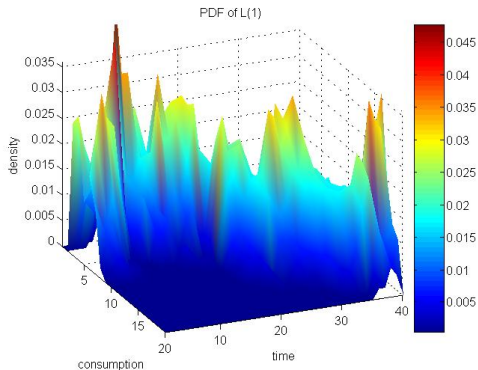
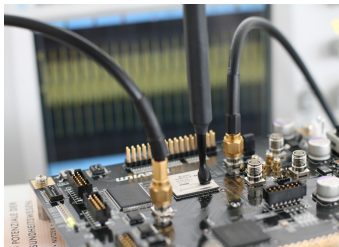
Probability distribution function (pdf) of Electromagnetic Emanations

Cryptographic Processing with a secret  $k = 1$ .



## Probability distribution function (pdf) of Electromagnetic Emanations

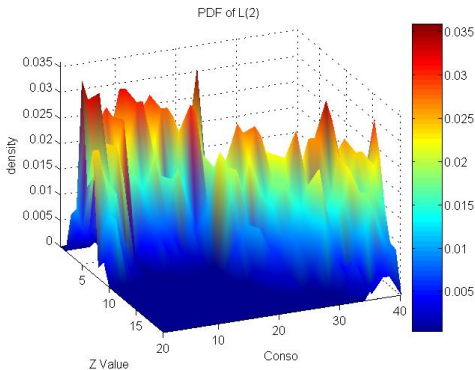
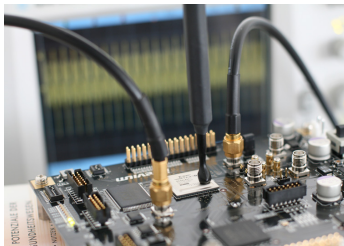
Cryptographic Processing with a secret  $k = 1$ .





## Probability distribution function (pdf) of Electromagnetic Emanations

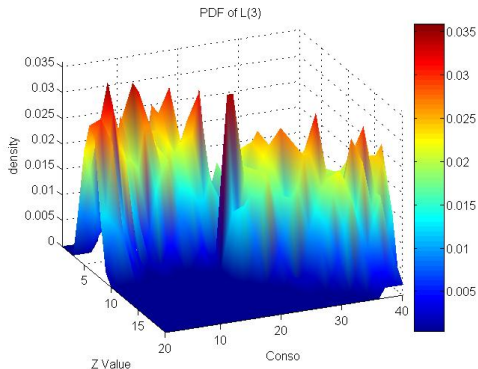
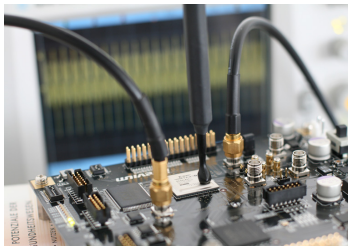
Cryptographic Processing with a secret  $k = 2$ .





## Probability distribution function (pdf) of Electromagnetic Emanations

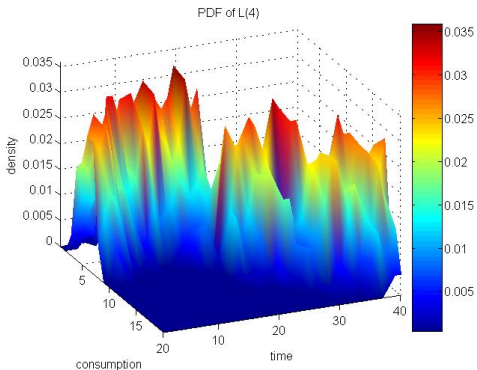
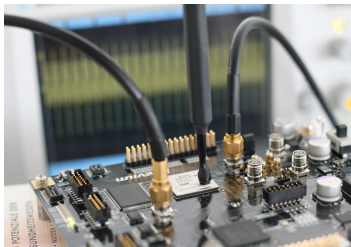
Cryptographic Processing with a secret  $k = 3$ .





## Probability distribution function (pdf) of Electromagnetic Emanations

Cryptographic Processing with a secret  $k = 4$ .



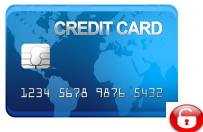


# Deep Learning for Embedded Security Evaluation

Context:



Target Device



Clone Device



Context:

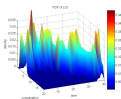


Target Device

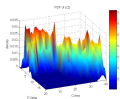


Clone Device

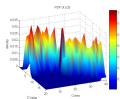
► [On Clone Device] For every  $k$  estimate the pdf of  $\vec{X} \mid K = k$ .



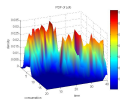
$k = 1$



$k = 2$



$k = 3$



$k = 4$

.....



Context:

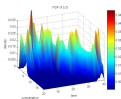


Target Device

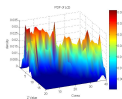


Clone Device

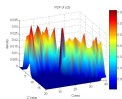
► [On Clone Device] For every  $k$  estimate the pdf of  $\vec{X} \mid K = k$ .



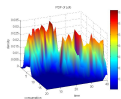
$k = 1$



$k = 2$



$k = 3$



$k = 4$

.....



Context:

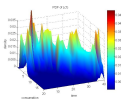


Target Device

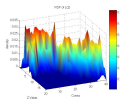


Clone Device

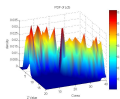
- ▶ [On Clone Device] For every  $k$  estimate the pdf of  $\vec{X} \mid K = k$ .



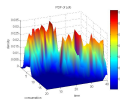
$k = 1$



$k = 2$



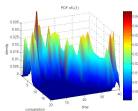
$k = 3$



$k = 4$

.....

- ▶ [On Target Device] Estimate the pdf of  $\vec{X}$ .



$k = ?$



Context:

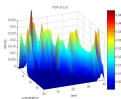


Target Device

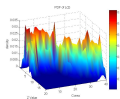


Clone Device

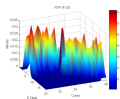
- ▶ [On Clone Device] For every  $k$  estimate the pdf of  $\vec{X} \mid K = k$ .



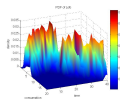
$k = 1$



$k = 2$



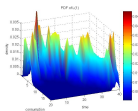
$k = 3$



$k = 4$

.....

- ▶ [On Target Device] Estimate the pdf of  $\vec{X}$ .



$k = ?$

- ▶ [Key-recovery] Compare the pdf estimations.



## Side Channel Attacks (Classical Approach)

### Notations

- ▶  $\vec{X}$  observation of the device behaviour (consumption, electro-magnetic emanation, timing, etc.)
- ▶  $P$  public input of the processing
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$



## Side Channel Attacks (Classical Approach)

### Notations

- ▶  $\vec{X}$  observation of the device behaviour (consumption, electro-magnetic emanation, timing, etc.)
- ▶  $P$  public input of the processing
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$\Pr[Z|\vec{X}]$



## Side Channel Attacks (Classical Approach)

### Notations

- ▶  $\vec{X}$  observation of the device behaviour (consumption, electro-magnetic emanation, timing, etc.)
- ▶  $P$  public input of the processing
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$\Pr[Z|\vec{X}]$

### Template Attacks

- ▶ Profiling phase (using profiling traces under known  $Z$ )
  
- ▶ Attack phase ( $N$  attack traces  $\vec{x}_i$ , e.g. with known plaintexts  $p_i$ )



## Side Channel Attacks (Classical Approach)

### Notations

- ▶  $\vec{X}$  observation of the device behaviour (consumption, electro-magnetic emanation, timing, etc.)
- ▶  $P$  public input of the processing
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$$\Pr[Z|\vec{X}]$$

### Template Attacks

- ▶ Profiling phase (using profiling traces under known  $Z$ )
  - ▶ estimate  $\Pr[\vec{X}|Z = z]$  by simple distributions for each value of  $z$
- ▶ Attack phase ( $N$  attack traces  $\vec{x}_i$ , e.g. with known plaintexts  $p_i$ )



## Side Channel Attacks (Classical Approach)

### Notations

- ▶  $\vec{X}$  observation of the device behaviour (consumption, electro-magnetic emanation, timing, etc.)
- ▶  $P$  public input of the processing
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$\Pr[Z|\vec{X}]$

### Template Attacks

- ▶ Profiling phase (using profiling traces under known  $Z$ )
  - ▶ estimate  $\Pr[\vec{X}|Z = z]$  for each value of  $z$
- ▶ Attack phase ( $N$  attack traces  $\vec{x}_i$ , e.g. with known plaintexts  $p_i$ )
  - ▶ Log-likelihood score for each key hypothesis  $k$

$$d_k = \sum_{i=1}^N \log \Pr[\vec{X} = \vec{x}_i | Z = f(p_i, k)]$$



## Side Channel Attacks (Classical Approach)

### Notations

- ▶  $\vec{X}$  observation of the device behaviour (consumption, electro-magnetic emanation, timing, etc.)
- ▶  $P$  public input of the processing
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$\Pr[Z|\vec{X}]$

### Template Attacks

- ▶ Profiling phase (using profiling traces under known  $Z$ )
  - ▶ mandatory dimensionality reduction
  - ▶ estimate  $\Pr[\vec{X}|Z = z]$  for each value of  $z$
- ▶ Attack phase ( $N$  attack traces  $\vec{x}_i$ , e.g. with known plaintexts  $p_i$ )
  - ▶ Log-likelihood score for each key hypothesis  $k$

$$d_k = \sum_{i=1}^N \log \Pr[\vec{X} = \vec{x}_i | Z = f(p_i, k)]$$



## Side Channel Attacks (Classical Approach)

### Notations

- ▶  $\vec{X}$  observation of the device behaviour (consumption, electro-magnetic emanation, timing, etc.)
- ▶  $P$  public input of the processing
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$\Pr[Z|\vec{X}]$

### Template Attacks

- ▶ Profiling phase (using profiling traces under known  $Z$ )
  - ▶ manage de-synchronization problem
  - ▶ mandatory dimensionality reduction
  - ▶ estimate  $\Pr[\vec{X}|Z = z]$  for each value of  $z$
- ▶ Attack phase ( $N$  attack traces  $\vec{x}_i$ , e.g. with known plaintexts  $p_i$ )
  - ▶ Log-likelihood score for each key hypothesis  $k$

$$d_k = \sum_{i=1}^N \log \Pr[\vec{X} = \vec{x}_i | Z = f(p_i, k)]$$



## Defensive Mechanisms

### Misaligning Countermeasures

- ▶ Random Delays, Clock Jittering, ...
- ▶ In theory: assume to be insufficient to provide security
- ▶ In practice: one of the main issues for evaluators
- ▶  $\implies$  Need for efficient resynchronization techniques



## Defensive Mechanisms

### Misaligning Countermeasures

- ▶ Random Delays, Clock Jittering, ...
- ▶ In theory: assume to be insufficient to provide security
- ▶ In practice: one of the main issues for evaluators
- ▶  $\implies$  Need for efficient resynchronization techniques

### Masking Countermeasure

- ▶ Each key-dependent internal state element is randomly split into 2 shares
- ▶ The crypto algorithm is adapted to always manipulate shares at  $\neq$  times
- ▶ The adversary needs to recover information on the two shares to recover  $K$
- ▶  $\implies$  Need for efficient Methods to recover tuple of leakage samples that jointly depend on the target secret



## Contents

1. Context and Motivation
  - 1.1 Illustration
  - 1.2 Template Attacks
  - 1.3 Countermeasures
2. A machine learning approach to classification
  - 2.1 Introduction
  - 2.2 The Underlying Classification Problem
  - 2.3 Convolutional Neural Networks
  - 2.4 Training of Models
3. Building a Community Around The Subject
  - 3.1 ASCAD Open Data-Base
  - 3.2 Leakage Characterization and Training
  - 3.3 Results
4. Conclusions



### Motivating Conclusions

An evaluator can spend time to adjust realignment and/or to find points of interest...

- ▶ try realignments based over other kind of patterns
- ▶ modify parameters...



### Motivating Conclusions

An evaluator can spend time to adjust realignment and/or to find points of interest...

- ▶ try realignments based over other kind of patterns
- ▶ modify parameters...

...but

- ▶ no prior knowledge about informative patterns
- ▶ no way to evaluate realignment without launching the afterwards attack



## Motivating Conclusions

An evaluator can spend time to adjust realignment and/or to find points of interest...

- ▶ try realignments based over other kind of patterns
- ▶ modify parameters...

...but

- ▶ no prior knowledge about informative patterns
- ▶ no way to evaluate realignment without launching the afterwards attack

Now:

- ▶ preprocessing to prepare data
- ▶ make strong hypotheses on the statistical dependency
- ▶ characterization to extract information

The proposed perspective:

- ▶ ~~preprocessing to prepare data~~
- ▶ ~~make strong hypotheses on the statistical dependency~~
- ▶ Train algorithms to directly extract information



## Side Channel Attacks

### Notations

- ▶  $\vec{X}$  side channel trace
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$$\Pr[Z|\vec{X}]$$

### Template Attacks Machine Learning Side Channel Attacks

- ▶ Profiling phase (using profiling traces under known  $Z$ )
  - ▶ manage de-synchronization problem
  - ▶ mandatory dimensionality reduction
  - ▶ estimate  $\Pr[\vec{X}|Z = z]$  for each value of  $z$
- ▶ Attack phase ( $N$  attack traces, e.g. with known plaintexts  $p_i$ )
  - ▶ Log-likelihood score for each key hypothesis  $k$

$$d_k = \sum_{i=1}^N \log \Pr[\vec{X} = \vec{x}_i | Z = f(p_i, k)]$$



## Side Channel Attacks **with a Classifier**

### Notations

- ▶  $\vec{X}$  side channel trace
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$\Pr[Z|\vec{X}]$

### ~~Template Attacks~~ Machine Learning Side Channel Attacks

- ▶ Profiling phase (using profiling traces under known  $Z$ )
  - ▶ manage de-synchronization problem
  - ▶ mandatory dimensionality reduction
  - ▶ estimate  $\Pr[\vec{X}|Z = z]$  for each value of  $z$
- ▶ Attack phase ( $N$  attack traces, e.g. with known plaintexts  $p_i$ )
  - ▶ Log-likelihood score for each key hypothesis  $k$

$$d_k = \sum_{i=1}^N \log \Pr[\vec{X} = \vec{x}_i | Z = f(p_i, k)]$$



## Side Channel Attacks with a Classifier

### Notations

- ▶  $\vec{X}$  side channel trace
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$$\Pr[Z|\vec{X}]$$

### Template Attacks Machine Learning Side Channel Attacks

- ▶ Training phase (using training traces under known  $Z$ )
  - ▶ manage de-synchronization problem
  - ▶ mandatory dimensionality reduction
  - ▶ estimate  $\Pr[\vec{X}|Z = z]$  for each value of  $z$
- ▶ Attack phase ( $N$  attack traces, e.g. with known plaintexts  $p_i$ )
  - ▶ Log-likelihood score for each key hypothesis  $k$

$$d_k = \sum_{i=1}^N \log \Pr[\vec{X} = \vec{x}_i | Z = f(p_i, k)]$$



## Side Channel Attacks with a Classifier

### Notations

- ▶  $\vec{X}$  side channel trace
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$\Pr[Z|\vec{X}]$

### Template Attacks Machine Learning Side Channel Attacks

- ▶ Training phase (using training traces under known  $Z$ )
  - ▶ manage de-synchronization problem
  - ▶ mandatory dimensionality reduction
  - ▶ construct a classifier  $F(\vec{x}) = y \approx \Pr[Z|\vec{X} = \vec{x}]$
- ▶ Attack phase ( $N$  attack traces, e.g. with known plaintexts  $p_i$ )
  - ▶ Log-likelihood score for each key hypothesis  $k$

$$d_k = \sum_{i=1}^N \log \Pr[\vec{X} = \vec{x}_i | Z = f(p_i, k)]$$



## Side Channel Attacks with a Classifier

### Notations

- ▶  $\vec{X}$  side channel trace
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$\Pr[Z|\vec{X}]$

### Template Attacks Machine Learning Side Channel Attacks

- ▶ Training phase (using training traces under known  $Z$ )
  - ▶ manage de-synchronization problem
  - ▶ mandatory dimensionality reduction
  - ▶ construct a classifier  $F(\vec{x}) = y \approx \Pr[Z|\vec{X} = \vec{x}]$
- ▶ Attack phase ( $N$  attack traces, e.g. with known plaintexts  $p_i$ )
  - ▶ Log-likelihood score for each key hypothesis  $k$

$$d_k = \sum_{i=1}^N \log F(\vec{x}_i)[f(p_i, k)]$$



## Side Channel Attacks with a Classifier

### Notations

- ▶  $\vec{X}$  side channel trace
- ▶  $Z$  target (a cryptographic sensitive variable  $Z = f(P, K)$ )

Goal: make inference over  $Z$ , observing  $\vec{X}$

$$\Pr[Z|\vec{X}]$$

### ~~Template Attacks~~ Machine Learning Side Channel Attacks

- ▶ Training phase (using training traces under known  $Z$ )
  - ▶ manage de-synchronization problem
  - ▶ mandatory dimensionality reduction
  - ▶ construct a classifier  $F(\vec{x}) = y \approx \Pr[Z|\vec{X} = \vec{x}]$
- ▶ Attack phase ( $N$  attack traces, e.g. with known plaintexts  $p_i$ )
  - ▶ Log-likelihood score for each key hypothesis  $k$

} Integrated approach

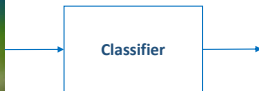
$$d_k = \sum_{i=1}^N \log F(\vec{x}_i)[f(p_i, k)]$$



## Classification

### Classification problem

Assign to a datum  $\vec{X}$  (e.g. an image) a label  $Z$  among a set of possible labels  $Z = \{\text{Cat, Dog, Horse}\}$

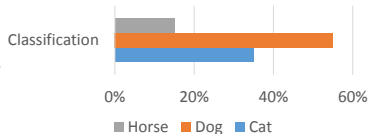
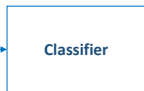




## Classification

### Classification problem

Assign to a datum  $\vec{X}$  (e.g. an image) a label  $Z$  among a set of possible labels  $Z = \{\text{Cat}, \text{Dog}, \text{Horse}\}$

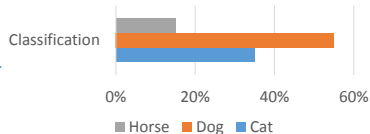




## Classification

### Classification problem

Assign to a datum  $\vec{X}$  (e.g. an image) a label  $Z$  among a set of possible labels  $Z = \{\text{Cat}, \text{Dog}, \text{Horse}\}$



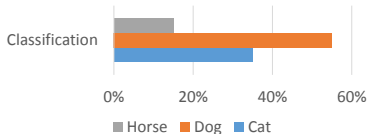
$$\Pr[Z|\vec{X}]$$



## Classification

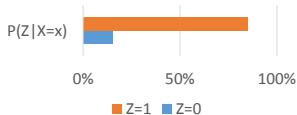
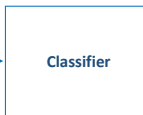
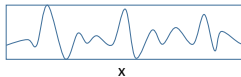
### Classification problem

Assign to a datum  $\vec{X}$  (e.g. an image) a label  $Z$  among a set of possible labels  $Z = \{\text{Cat}, \text{Dog}, \text{Horse}\}$



$$\Pr[Z|\vec{X}]$$

### SCA as a Classification Problem





## Machine Learning Approach

### Overview of Machine Learning Methodology

#### Human effort:

- ▶ choose a class of algorithms
- ▶ choose a model to fit + tune **hyper-parameters**

#### Automatic training:

- ▶ automatic tuning of **trainable parameters** to fit data



## Machine Learning Approach

### Overview of Machine Learning Methodology

Human effort:

- ▶ choose a class of algorithms  
*Neural Networks*
- ▶ choose a model to fit +  
tune **hyper-parameters**

Automatic training:

- ▶ automatic tuning of  
**trainable parameters**  
to fit data  
*Stochastic Gradient Descent*



## Machine Learning Approach

### Overview of Machine Learning Methodology

#### Human effort:

- ▶ choose a class of algorithms  
*Neural Networks*
- ▶ choose a model to fit +  
tune **hyper-parameters**  
*MLP, ConvNet*

#### Automatic training:

- ▶ automatic tuning of  
**trainable parameters**  
to fit data  
*Stochastic Gradient Descent*



## Machine Learning Approach

### Overview of Machine Learning Methodology

Human effort:

- ▶ choose a class of algorithms  
*Neural Networks*
- ▶ choose a model to fit +  
tune **hyper-parameters**  
*MLP, ConvNet*

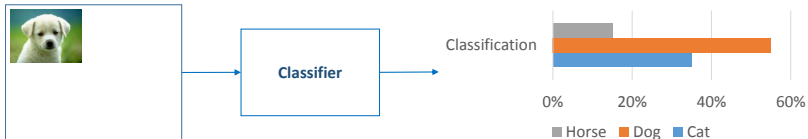
Automatic training:

- ▶ automatic tuning of  
**trainable parameters**  
to fit data  
*Stochastic Gradient Descent*



## Convolutional Neural Networks

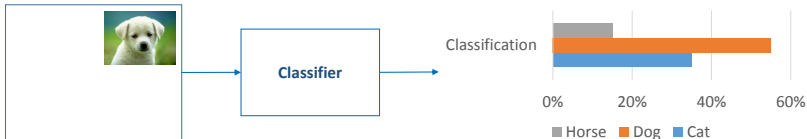
### An answer to translation-invariance





## Convolutional Neural Networks

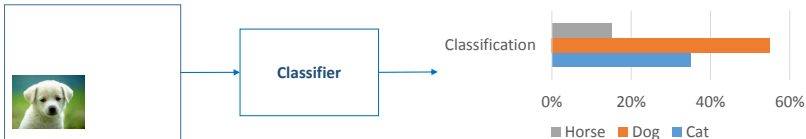
### An answer to translation-invariance





## Convolutional Neural Networks

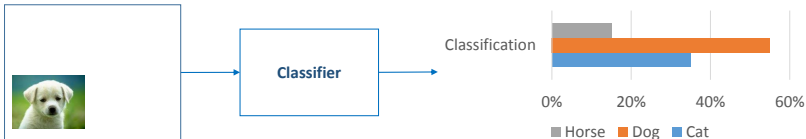
### An answer to translation-invariance





## Convolutional Neural Networks

### An answer to translation-invariance

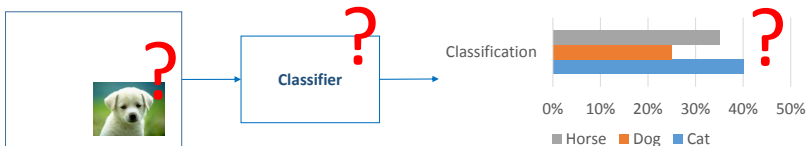


It is important to explicit the data translation-invariance



## Convolutional Neural Networks

### An answer to translation-invariance



It is important to explicit the data translation-invariance



## Convolutional Neural Networks

### An answer to translation-invariance

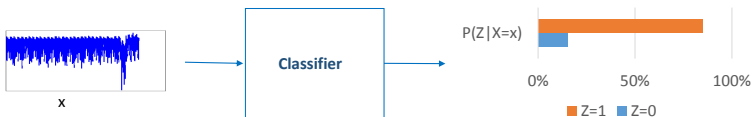


It is important to explicit the data translation-invariance



## Convolutional Neural Networks

### An answer to translation-invariance



It is important to explicit the data translation-invariance



## Convolutional Neural Networks

An answer to translation-invariance



It is important to explicit the data translation-invariance



## Convolutional Neural Networks

An answer to translation-invariance



It is important to explicit the data translation-invariance  
Convolutional Neural Networks: share weights across space



## Training of Neural Networks

Trading Side-Channel Expertise for Deep Learning Expertise .... or huge computational power!

### Training

Aims at finding the **parameters** of the model (aka approximation) for the the statistical dependency between the target value and the leakage.

The approximation is done by solving a minimization problem with respect to some metric (aka cost function)

The training algorithm has itself some **training hyper-parameters**:

- the number of iterations (aka **epochs**) of the minimization procedure,
- the number of input traces (aka **batch**) treated during a single iteration.

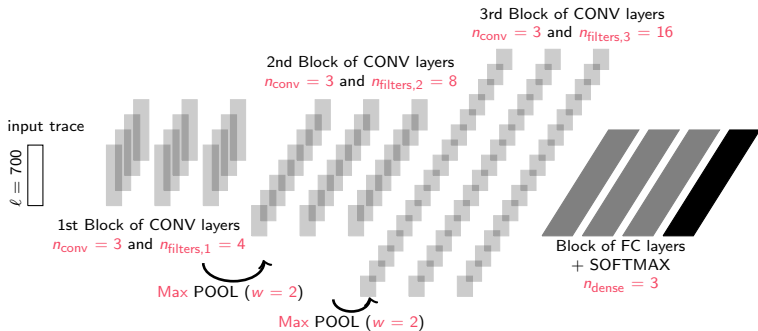
The architecture of the trained model has **architecture hyper-parameters**:

- the size of the layers,
- the nature of the layers,
- the number of layers,
- etc.

Finding sound hyper-parameters is the main issue in Deep Learning: **this can be done thanks to a good understanding of the underlying structure of the data and/or access to important computational power.**



## ConvNet typical architecture





## Contents

1. Context and Motivation
  - 1.1 Illustration
  - 1.2 Template Attacks
  - 1.3 Countermeasures
2. A machine learning approach to classification
  - 2.1 Introduction
  - 2.2 The Underlying Classification Problem
  - 2.3 Convolutional Neural Networks
  - 2.4 Training of Models
3. Building a Community Around The Subject
  - 3.1 ASCAD Open Data-Base
  - 3.2 Leakage Characterization and Training
  - 3.3 Results
4. Conclusions



## Creation of an open database for Training and Testing

### ANSSI recently publishes

- ▶ source codes of secure cryptographic implementations for public 8-bit architectures (<https://github.com/ANSSI-FR/secAES-ATmega8515>)
- ▶ data-bases of electromagnetic leakages (<https://github.com/ANSSI-FR/ASCAD>)
- ▶ example scripts for the training and testing of Deep Learning models in the context of security evaluations

### Goal

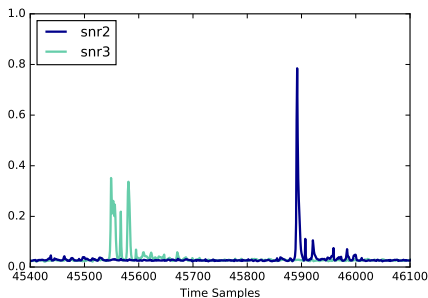
- ▶ Enable easy fair and easy benchmarking
- ▶ Initiate discussions and exchanges on the application of DL to SCA
- ▶ Create a community of contributors on this subject



## Nature of the Observations/Traces

Side-channel observations in ASCAD correspond to the masked processing of a simple cryptographic primitive

Information leakage validated thanks to SNR characterization

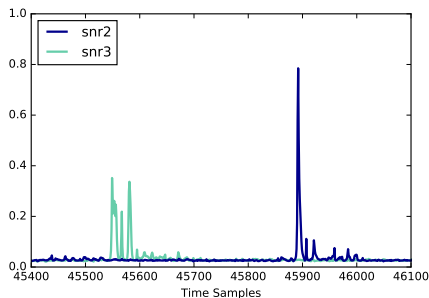




## Nature of the Observations/Traces

Side-channel observations in ASCAD correspond to the masked processing of a simple cryptographic primitive

Information leakage validated thanks to SNR characterization



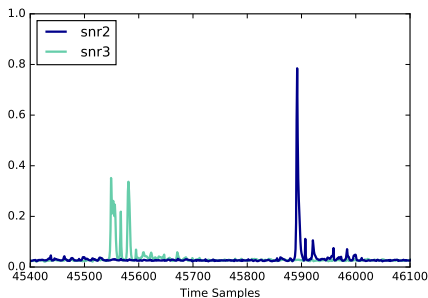
Validate that shares are **manipulated at different times**



## Nature of the Observations/Traces

Side-channel observations in ASCAD correspond to the masked processing of a simple cryptographic primitive

Information leakage validated thanks to SNR characterization



Validate that shares are **manipulated at different times**  
Scripts are also proposed to add **artificial signal jittering**



## Our Training Strategy

Find a **base model architecture** and find training hyper-parameters for which a convergence towards the good key hypothesis is visible

Fine-tune all the hyper-parameters one after another to get the best efficiency/effectiveness trade-off

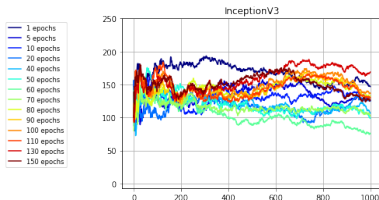
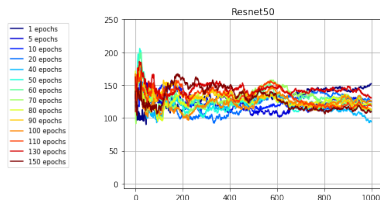
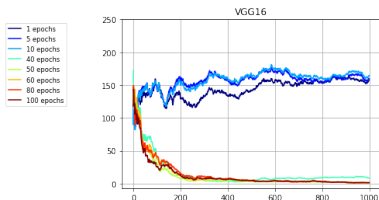
**Table:** Benchmarks Summary

Parameter	Reference	Metric	Range	Choice
Training Parameters				
Epochs	-	rank vs time	10, 25, 50, 60, . . . , 100, 150	up to 100
Batch Size	-	rank vs time	50, 100, 200	200
Architecture Parameters				
Blocks	$n_{\text{blocks}}$	rank, accuracy	[2..5]	5
CONV layers	$n_{\text{conv}}$	rank, accuracy	[0..3]	1
Filters	$n_{\text{filters},1}$	rank vs time	$\{2^i; i \in [4..7]\}$	64
Kernel Size	-	rank	{3, 6, 11}	11
FC Layers	$n_{\text{dense}}$	rank, accuracy vs time	[0..3]	2
ACT Function	$\alpha$	rank	ReLU, Sigmoid, Tanh	ReLU
Pooling Layer	-	rank	Max, Average, Stride	Average
Padding	-	rank	Same, Valid	Same



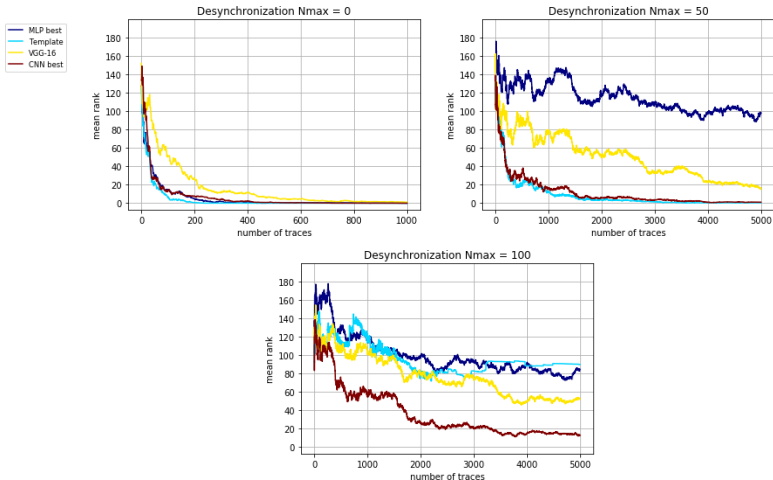
## The Base Architecture

Mean rank of the good-key hypothesis obtained with **VGG-16**, **ResNet-50** and **Inception-v3** w.r.t. different epochs:





## Comparisons with State-Of-the-Art Methods





## Contents

1. Context and Motivation
  - 1.1 Illustration
  - 1.2 Template Attacks
  - 1.3 Countermeasures
2. A machine learning approach to classification
  - 2.1 Introduction
  - 2.2 The Underlying Classification Problem
  - 2.3 Convolutional Neural Networks
  - 2.4 Training of Models
3. Building a Community Around The Subject
  - 3.1 ASCAD Open Data-Base
  - 3.2 Leakage Characterization and Training
  - 3.3 Results
4. Conclusions



### Conclusions

- ▶ State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization



### Conclusions

- ▶ State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ▶ Deep Learning provides an integrated approach to directly extract information from rough data (no preprocessing)



### Conclusions

- ▶ State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ▶ Deep Learning provides an integrated approach to directly extract information from rough data (no preprocessing)
- ▶ Many recent results validate the practical interest of the Machine Learning approach



### Conclusions

- ▶ State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ▶ Deep Learning provides an integrated approach to directly extract information from rough data (no preprocessing)
- ▶ Many recent results validate the practical interest of the Machine Learning approach
- ▶ We are in the very beginning and we are still discovering how much Deep Learning is efficient



### Conclusions

- ▶ State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ▶ Deep Learning provides an integrated approach to directly extract information from rough data (no preprocessing)
- ▶ Many recent results validate the practical interest of the Machine Learning approach
- ▶ We are in the very beginning and we are still discovering how much Deep Learning is efficient
- ▶ New needs:
  - ▶ big data-bases for the training,
  - ▶ platforms to enable comparisons and benchmarking,
  - ▶ create an open community "ML for Embedded Security Analysis",
  - ▶ encourage exchanges with the Machine Learning community,
  - ▶ understand the efficiency of the current countermeasures



### Conclusions

- ▶ State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ▶ Deep Learning provides an integrated approach to directly extract information from rough data (no preprocessing)
- ▶ Many recent results validate the practical interest of the Machine Learning approach
- ▶ We are in the very beginning and we are still discovering how much Deep Learning is efficient
- ▶ New needs:
  - ▶ big data-bases for the training,
  - ▶ platforms to enable comparisons and benchmarking,
  - ▶ create an open community "ML for Embedded Security Analysis",
  - ▶ encourage exchanges with the Machine Learning community,
  - ▶ understand the efficiency of the current countermeasures

Thank You!

Questions?