Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Eleonora Cagli\(^1,3\) Cécile Dumas\(^1\)
Emmanuel Prouff\(^2,3\)

\(^1\)Univ. Grenoble Alpes, F-38000, Grenoble, France
CEA, LETI, MINATEC Campus, F-38054 Grenoble, France
\{eleonora.cagli,cecile.dumas\}@cea.fr

\(^2\)Safran Identity and Security, France
emmanuel.prouff@safrangroup.com

\(^3\)Sorbonne Universités, UPMC Univ. Paris 06, CNRS, INRIA, Laboratoire d’Informatique de Paris 6 (LIP6),
Équipe PolSys, 4 place Jussieu, 75252 Paris Cedex 05, France

26/09/2017, CHES 2017, Taipei, Taiwan
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

About the authors

Who we are

Conceives a component — ITSEF
Evaluates Security Claims — ANSSI
Delivers a Security Certification — Developer
Commercialises the certified product — Developer

French Certification Scheme

26/09/2017, CHES 2017, Taipei, Taiwan | E. Cagli, C. Dumas, E. Prouff | 2/26
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

About the authors

Who we are

Conceives a component

Evaluates Security Claims

Delivers a Security Certification

Commercialises the certified product

Developer

ITSEF

ANSSI

Developer

Cécile

French Certification Scheme
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

About the authors

Who we are

Conceives a component
Evaluates Security Claims
Delivers a Security Certification
Commercialises the certified product

Developer
ITSEF
ANSSI
Developer

Conceives
Evaluates
Delivers
Commercialises

Security Claims
Security Certification
certified product

French Certification Scheme

Cécile
Eleonora
About the authors

Who we are

- **Conceives a component**: Emmanuel
  (in the past)
- **Evaluates Security Claims**: ITSEF
- **Delivers a Security Certification**: ANSSI
- **Commercialises the certified product**: Developer

*French Certification Scheme*

---

Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures
About the authors

Who we are

- Conceives a component
- Evaluates Security Claims
- Delivers a Security Certification
- Commercialises the certified product

Developer → ITSEF → ANSSI → Developer

Cécile
Eleonora
Emmanuel (today)

French Certification Scheme
About the authors

Who we are

- Conceives a component
- Evaluates Security Claims
- Delivers a Security Certification
- Commercialises the certified product

Developer
ITSEF
ANSSI
Developer

Cécile
Eleonora

Emmanuel (today)

An evaluation point of view

- profiling attacks (worst-case security)
- practical aspects are concerned
Contents

1. Context and Motivation

2. A machine learning approach to classification
   2.1 Introduction
   2.2 Convolutional Neural Networks

3. Data Augmentation

4. Experimental Results

5. Conclusions
Side Channel Attacks

**Notations**

- $X$ side channel trace
- $Z$ target (a cryptographic sensitive variable $Z = f(P, K)$)

Goal: make inference over $Z$, observing $X$
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Side Channel Attacks

Notations

- $X$ side channel trace
- $Z$ target (a cryptographic sensitive variable $Z = f(P, K)$)

Goal: make inference over $Z$, observing $X$
Side Channel Attacks

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

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

### Template Attacks
- Profiling phase (using profiling traces under known $\mathbf{Z}$)
- Attack phase ($N$ attack traces, e.g. with known plaintexts $\mathbf{p}_i$)
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Side Channel Attacks

Notations

- $X$: side channel trace
- $Z$: target (a cryptographic sensitive variable $Z = f(P, K)$)

Goal: make inference over $Z$, observing $X$

Template Attacks

- Profiling phase (using profiling traces under known $Z$)
  - estimate $Pr[X|Z = z]$ for each value of $z$
- Attack phase ($N$ attack traces, e.g. with known plaintexts $p_i$)
Side Channel Attacks

Notations

- \( X \) side channel trace
- \( Z \) target (a cryptographic sensitive variable \( Z = f(P, K) \))

Goal: make inference over \( Z \), observing \( X \)

Template Attacks

- Profiling phase (using profiling traces under known \( Z \))
  - estimate \( \Pr[X|Z = z] \) for each value of \( z \)
  - Attack phase (\( N \) attack traces, e.g. with known plaintexts \( p_i \))
    - Log-likelihood score for each key hypothesis \( k \)
      \[
      d_k = \sum_{i=1}^{N} \log \Pr[X = x_i|Z = f(p_i, k)]
      \]
Side Channel Attacks

**Notations**

- \( X \) side channel trace
- \( Z \) target (a cryptographic sensitive variable \( Z = f(P, K) \))

Goal: make inference over \( Z \), observing \( X \)

\[ \Pr[Z|X] \]

**Template Attacks**

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

\[
d_k = \sum_{i=1}^{N} \log \Pr[X = x_i|Z = f(p_i, k)]
\]
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Side Channel Attacks

Notations
- X side channel trace
- Z target (a cryptographic sensitive variable Z = f(P, K))

Goal: make inference over Z, observing X

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

\[
d_k = \sum_{i=1}^{N} \log \Pr[X = x_i|Z = f(p_i, k)]
\]
Misalignment

Misaligning Countermeasures

- Random Delays, Clock Jittering, ...
- In theory: insufficient to provide security, since information still leak (somewhere)
- In practice: one of the main issues for evaluators
Misalignment

Misaligning Countermeasures

- Random Delays, Clock Jittering, ...
- In theory: insufficient to provide security, since information still leak (somewhere)
- In practice: one of the main issues for evaluators

Realignment

Mandatory realignment preprocessing

- not a wide literature
- in practice: evaluation labs home-made realignment techniques
Risks of realignment

An example
Risks of realignment

An example
Risks of realignment

An example
Risks of realignment

An example

Aligned extracted regions
Risks of realignment

An example
Risks of realignment

An example

Aligned extracted regions

Informative region not extracted!
Risks of realignment

An example

Aligned extracted regions
Risks of realignment

An example
Risks of realignment
An example

Aligned extracted regions

Informative region not extracted!
Motivating Conclusions

An evaluator can spend time to adjust realignment...

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

An evaluator can spend time to adjust realignment...
  ▶ 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...
  ▶ 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
  ▶ characterization to extract information

Our paper perspective:
  ▶ preprocessing to prepare data
  ▶ direct information extraction
Motivating Conclusions

An evaluator can spend time to adjust realignment...
  ▶ 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
  ▶ characterization to extract information

No preprocessing ⇒ No risks of information loss

Our paper perspective:
  ▶ preprocessing to prepare data
  ▶ direct information extraction
**Side Channel Attacks**

### Notations

- **X** side channel trace
- **Z** target (a cryptographic sensitive variable \( Z = f(P, K) \))

Goal: make inference over \( Z \), observing \( X \)

### Template Attacks

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

\[
d_k = \sum_{i=1}^{N} \log \Pr[X = x_i|Z = f(p_i, k)]
\]
Side Channel Attacks with a Classifier

Notations

- $X$ side channel trace
- $Z$ target (a cryptographic sensitive variable $Z = f(P, K)$)

Goal: make inference over $Z$, observing $X$  \(\Pr(Z|X)\)

Template Attacks

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

$$d_k = \sum_{i=1}^{N} \log \Pr(X = x_i|Z = f(p_i, k))$$
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Side Channel Attacks with a Classifier

Notations
- $X$ side channel trace
- $Z$ target (a cryptographic sensitive variable $Z = f(P, K)$)

Goal: make inference over $Z$, observing $X$

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

$$d_k = \sum_{i=1}^{N} \log \Pr[X = x_i|Z = f(p_i, k)]$$
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Side Channel Attacks with a Classifier

Notations

- $X$ side channel trace
- $Z$ target (a cryptographic sensitive variable $Z = f(P, K)$)

Goal: make inference over $Z$, observing $X$

Template Attacks

- **Training phase (using training traces under known $Z$)**
  - manage de-synchronization problem
  - mandatory dimensionality reduction
  - **construct a classifier** $F(x) = y \approx \Pr[Z|X = x]$  
- **Attack phase** ($N$ attack traces, e.g. with known plaintexts $p_i$)
  - Log-likelihood score for each key hypothesis $k$
    
    $d_k = \sum_{i=1}^{N} \log \Pr[X = x_i|Z = f(p_i, k)]$
Side Channel Attacks with a Classifier

Notations

- $X$ side channel trace
- $Z$ target (a cryptographic sensitive variable $Z = f(P, K)$)

Goal: make inference over $Z$, observing $X$

Template Attacks

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

$$d_k = \sum_{i=1}^{N} \log F(x_i)[f(p_i, k)]$$
Side Channel Attacks with a Classifier

**Notations**

- **X** side channel trace
- **Z** target (a cryptographic sensitive variable \( Z = f(P, K) \))

Goal: make inference over **Z**, observing **X**

\[ \Pr[Z|X] \]

**Template Attacks**

- **Training phase** (using training traces under known **Z**)
  - manage de-synchronization problem
  - mandatory dimensionality reduction
  - construct a classifier \( F(x) = y \approx \Pr[Z|X = x] \)

- **Attack phase** (\( N \) attack traces, e.g. with known plaintexts \( p_i \))
  - Log-likelihood score for each key hypothesis \( k \)

\[ d_k = \sum_{i=1}^{N} \log F(x_i)[f(p_i, k)] \]
Contents

1. Context and Motivation

2. A machine learning approach to classification
   2.1 Introduction
   2.2 Convolutional Neural Networks

3. Data Augmentation

4. Experimental Results

5. Conclusions
Classification problem

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

Classification problem

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

![Diagram of a classifier system with a puppy image and bar chart representing classification probabilities.](image)

$$\Pr[\mathbf{Z}|\mathbf{X}]$$
Classification problem

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

SCA as a Classification Problem

$P(Z|X=x)$

Pr$[Z|X]$
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

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*
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*

**Multi-Layer Perceptron (MLP)**

\[
F(x, W) = s \circ \lambda_n \circ \sigma_{n-1} \circ \lambda_{n-1} \circ \cdots \circ \lambda_1(x) = y \approx \Pr[Z|X = x]
\]
Machine Learning Approach

Overview of Machine Learning Methodology

Human effort:
- choose a class of algorithms
  \textit{Neural Networks}
- choose a model to fit +
tune \textbf{hyper-parameters}
  \textit{MLP, ConvNet}

Automatic training:
- automatic tuning of
  \textbf{trainable parameters}
to fit data
  \textit{Stochastic Gradient Descent}

Multi-Layer Perceptron (MLP)

\begin{align*}
F(x, W) &= s \circ \lambda_n \circ \sigma_{n-1} \circ \lambda_{n-1} \circ \cdots \circ \lambda_1(x) = y \approx \Pr[Z|X = x] \\
\lambda_i & \text{ linear functions (linear combinations of time samples) depending on} \\
& \text{some \textbf{trainable weights} } W
\end{align*}
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*

Multi-Layer Perceptron (MLP)

\[ F(x, W) = s \circ \lambda_n \circ \sigma_{n-1} \circ \lambda_{n-1} \circ \cdots \circ \lambda_1(x) = y \approx \Pr[Z|X = x] \]

- \( \lambda_i \) linear functions (linear combinations of time samples) depending on some **trainable weights** \( W \)
- \( \sigma_i \) non-linear **activation** functions
Machine Learning Approach

Overview of Machine Learning Methodology

Human effort:
- choose a class of algorithms
  \textit{Neural Networks}
- choose a model to fit +
tune \textbf{hyper-parameters}
  \textit{MLP, ConvNet}

Automatic training:
- automatic tuning of
  \textbf{trainable parameters}
to fit data
  \textit{Stochastic Gradient Descent}

Multi-Layer Perceptron (MLP)

\[ F(x, W) = s \circ \lambda_n \circ \sigma_{n-1} \circ \lambda_{n-1} \circ \cdots \circ \lambda_1(x) = y \approx \Pr[Z|X = x] \]

- \( \lambda_i \) linear functions (linear combinations of time samples) depending on some \textbf{trainable weights} \( W \)
- \( \sigma_i \) non-linear \textit{activation} functions
- \( s \) normalizing \textit{softmax} function
Convolutional Neural Networks

An answer to translation-invariance

Convolutional Neural Networks

Classification

Horse
Dog
Cat

Classifier

0% 20% 40% 60%

Horse Dog Cat
Convolutional Neural Networks

An answer to translation-invariance

It is important to explicitly state the data translation-invariance.

Convolutional Neural Networks: share weights across space.
Convolutional Neural Networks

An answer to translation-invariance

Convolutional Neural Networks

- Share weights across space

Classification

0% 20% 40% 60%

Horse Dog Cat

Classifier
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
Convolutional Neural Networks

An answer to translation-invariance

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

Linear layer in an MLP (Fully Connected Layer)
Convolutional Neural Networks

An answer to translation-invariance

It is important to explicit the data translation-invariance

Convolutional Neural Networks: share weights across space

Linear layer in a ConvNet (Convolutional Layer)

Linear layer in an MLP (Fully Connected Layer)
Convolutional Neural Networks

An answer to translation-invariance

It is important to explicit the data translation-invariance

Convolutional Neural Networks: share weights across space

Linear layer in a ConvNet (Convolutional Layer)

Linear layer in an MLP (Fully Connected Layer)
Convolutional Neural Networks

Convolutional Neural Networks: share weights across space

It is important to explicit the data translation-invariance

Convolutional Neural Networks

An answer to translation-invariance

Classifier

\[
P(Z | X=x)
\]

\begin{align*}
Z &= 1 \\
Z &= 0
\end{align*}

Linear layer in a ConvNet (Convolutional Layer)

Matrix of weights
9x11 parameters

Linear layer in an MLP (Fully Connected Layer)

\[
\begin{array}{cccccccc}
2 & 4 & 2 & 6 & 2 & 6 & 1 & 2 & 6 & 4 \\
3 & 1 & 0 & 1 & 0 & 1 & 0 & 7 & 0 & 1 & 0 & 0 \\
1 & 5 & 4 & -1 & 1 & 1 & 0 & 1 & 8 & 1 & 0 & 4 \\
1 & 0 & 1 & 0 & 5 & 4 & 2 & 6 & 5 & 4 & 0 \\
2 & 1 & 0 & 1 & 0 & 7 & 0 & -1 & 1 & 0 \\
1 & 7 & 0 & 5 & 4 & -1 & 1 & 7 & 0 & -1 & 1 & 0 \\
1 & 1 & 8 & 1 & 0 & 1 & 8 & 1 & 0 & 8 \\
1 & 8 & 2 & 6 & 7 & 0 & 5 & 4 & 2 & 6 & 5 & 4 \\
9 & 1 & 0 & 1 & 8 & 1 & 0 & 7 & 0 & 1 & 0 & 0 \\
9 & -1 & 1 & 2 & 6 & 1 & 0 & 1 & 8 & 1 & 0 & 1 \\
\end{array}
\]

\[
\begin{array}{cccccccc}
28 & 61 & 53 & 66 & 132 & 66 & 66 & 87 & 79 & 53 \\
61 & 87 & 53 & 66 & 132 & 66 & 66 & 87 & 79 & 53 \\
66 & 132 & 66 & 87 & 79 & 53 & 66 & 66 & 87 & 79 \\
66 & 132 & 66 & 87 & 79 & 53 & 66 & 66 & 87 & 79 \\
122 & 66 & 66 & 87 & 79 & 53 & 66 & 66 & 87 & 79 \\
122 & 66 & 66 & 87 & 79 & 53 & 66 & 66 & 87 & 79 \\
66 & 132 & 66 & 87 & 79 & 53 & 66 & 66 & 87 & 79 \\
66 & 132 & 66 & 87 & 79 & 53 & 66 & 66 & 87 & 79 \\
53 & 66 & 66 & 87 & 79 & 53 & 66 & 66 & 87 & 79 \\
53 & 66 & 66 & 87 & 79 & 53 & 66 & 66 & 87 & 79 \\
\end{array}
\]
Convolutional Neural Networks

An answer to translation-invariance

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

Temporal Features

Side-Channel Trace

Convolution + Pooling

Convolution + Pooling

Abstract Features

Convolution + Pooling

Fully Connected Layer + Softmax

Scores
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Training and overfitting

Training

Profiling set → \{ Training set \\
\quad Validation set
\}

Randomly partition training set into batches

Iterative algorithm over batches (one weights update per processed batch)

\textit{Epoch:} = one pass over the entire training set
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Training and overfitting

Training

Profiling set → \{ \text{Training set} \quad \text{Validation set} \}

Randomly partition training set into batches

Iterative algorithm over batches (one weights update per processed batch)

*Epoch*: one pass over the entire training set

Evaluate and compare training and validation accuracy

Understand significant features

Learn by heart (OVERFITTING)

Accuracy vs. Epoch for Training and Validation sets

Accuracy vs. Epoch for Training and Validation sets
Training and overfitting

Training

Profiling set $\rightarrow \begin{cases} \text{Training set} \\ \text{Validation set} \end{cases}$

Randomly partition training set into batches

Iterative algorithm over batches (one weights update per processed batch)

$Epoch$:= one pass over the entire training set

Evaluate and compare training and validation accuracy

Learn by heart (OVERFITTING)

Why?

Too complex model

Not enough training data

Solution?

Data augmentation

Accuracy

Training

Validation

Epoch

26/09/2017, CHES 2017, Taipei, Taiwan | E. Cagli, C. Dumas, E. Prouff | 14/26
Contents

1. Context and Motivation

2. A machine learning approach to classification
   2.1 Introduction
   2.2 Convolutional Neural Networks

3. Data Augmentation

4. Experimental Results

5. Conclusions
Data Augmentation

Artificially generate new training data by deforming those previously acquired, applying transformations that preserve the label $Z$.

Countermeasure Emulation Idea

Emulate the effects of misaligning countermeasures to generate new traces.

**SHIFTING**

Parameter $T$: \# of possible positions

Parameter $R$: \# of added and removed points

Data Augmentation techniques are applied online during training phase.
Data Augmentation

Artificially generate new training data by deforming those previously acquired, applying transformations that preserve the label $Z$.

Countermeasure Emulation Idea

Emulate the effects of misaligning countermeasures to generate new traces.

**SHIFTING**

Parameter $T$: # of possible positions
Parameter $R$: # of added and removed points

Data Augmentation techniques are applied online during training phase.
Data Augmentation (2)

Quantitative example

A training set of 10,000 traces of 1,000 time samples

$\downarrow SH_{10}AR_5$

A training set of $\approx 10^{31}$ traces of 9,990 time samples
Contents

1. Context and Motivation

2. A machine learning approach to classification
   2.1 Introduction
   2.2 Convolutional Neural Networks

3. Data Augmentation

4. Experimental Results

5. Conclusions
Experimental Results

- Random delays
- Artificial Jitter
- Real Jitter
Experimental Results

- Random delays
- Artificial Jitter
- Real Jitter

- Network architecture: \( s \circ [\lambda]^1 \circ [\delta \circ [\sigma \circ \gamma]^1]^4 \)
- Keras library with Tensorflow backend [Ker] (open source)
Experimental Results

- Random delays
- Artificial Jitter
- Real Jitter

- Network architecture: $s \circ [\lambda]^1 \circ [\delta \circ [\sigma \circ \gamma]^1]^4$
- Keras library with Tensorflow backend [Ker] (open source)
Random delays

**Setup**

- Target Chip: Atmega328P
- Target Variable: $Z = \text{HW}(\text{Sbox}(P \oplus K))$
- Acquisition: through *ChipWhisperer*® platform, $\approx 4,000$ time samples
- Countermeasure: Random Delays - insertion of $r$ *nop* operations, $r \in [0, 127]$ uniform random
- 1,000 training traces
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Random delays
Data augmentation vs overfitting

Metrics

- Test accuracy: classification accuracy over the attack traces
- $N^*$: minimum number of attack traces to make guessing entropy of the right key permanently equal to one ($N^*$ estimated over 10 independent attacks)
Random delays
Data augmentation vs overfitting

**Metrics**

- Test accuracy: classification accuracy over the attack traces
- $N^*$: minimum number of attack traces to make *guessing entropy* of the right key permanently equal to one ($N^*$ estimated over 10 independent attacks)

<table>
<thead>
<tr>
<th></th>
<th>$SH_0$</th>
<th>$SH_{100}$</th>
<th>$SH_{500}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc</td>
<td>27.0%</td>
<td>31.8%</td>
<td>78%</td>
</tr>
<tr>
<td>$N^*$</td>
<td>&gt; 1,000</td>
<td>101</td>
<td>7</td>
</tr>
</tbody>
</table>
Real Jitter (1)

Target

- AES hardware implementation
- strong jitter effect
- Target Variable: $Z = \text{Sbox}(P \oplus K)$
- 2,500 selected time samples
- 99,000 training traces

SNR first Sbox
Real Jitter (1)

**Target**

- AES hardware implementation
- strong jitter effect
- Target Variable: \( Z = \text{Sbox}(P \oplus K) \)
- 2,500 selected time samples
- 99,000 training traces

**SNR second Sbox without realignment**

Entry region for CNN (2,500 pts) 26/09/2017, CHES 2017, Taipei, Taiwan | E. Cagli, C. Dumas, E. Prouff | 22/26
### Real Jitter (2)

<table>
<thead>
<tr>
<th></th>
<th>$SH_0AR_0$</th>
<th>$SH_{10}AR_{100}$</th>
<th>$SH_{20}AR_{200}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc</td>
<td>$N^*$</td>
<td>1.2%</td>
<td>1.3%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>137</td>
<td>89</td>
</tr>
</tbody>
</table>

26/09/2017, CHES 2017, Taipei, Taiwan | E. Cagli, C. Dumas, E. Prouff | 23/26
Real Jitter (2)

<table>
<thead>
<tr>
<th></th>
<th>$SH_0 \text{AR}_0$</th>
<th>$SH_{10}\text{AR}_{100}$</th>
<th>$SH_{20}\text{AR}_{200}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc</td>
<td>1.2%</td>
<td>1.3%</td>
<td>1.8%</td>
</tr>
<tr>
<td>$N^*$</td>
<td>137</td>
<td>89</td>
<td>54</td>
</tr>
</tbody>
</table>

SNR second Sbox with realignment

Number of attack traces vs Guessing Entropy
Contents

1. Context and Motivation

2. A machine learning approach to classification
   2.1 Introduction
   2.2 Convolutional Neural Networks

3. Data Augmentation

4. Experimental Results

5. 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.
- **ConvNets** provide an integrated approach to directly extract information from rough data (no preprocessing).

Thank You! Questions?
Conclusions

- State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ConvNets provide an integrated approach to directly extract information from rough data (no preprocessing)
- ConvNet models can require plenty of data to be trained
Conclusions

- State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ConvNets provide an integrated approach to directly extract information from rough data (no preprocessing)
- ConvNet models can require plenty of data to be trained
- Data Augmentation provides an answer to the lack of data
Conclusions

- State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ConvNets provide an integrated approach to directly extract information from rough data (no preprocessing)
- ConvNet models can require plenty of data to be trained
- Data Augmentation provides an answer to the lack of data
- We proposed two Side-Channel-adapted Data Augmentation techniques
Conclusions

- State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ConvNets provide an integrated approach to directly extract information from rough data (no preprocessing)
- ConvNet models can require plenty of data to be trained
- Data Augmentation provides an answer to the lack of data
- We proposed two Side-Channel-adapted Data Augmentation techniques
- We verified the effectiveness/efficiency of the ConvNet+Data Augmentation approach over different sets of misaligned data
Conclusions

- State-of-the-Art Template Attack separates resynchronization/dimensionality reduction from characterization
- ConvNets provide an integrated approach to directly extract information from rough data (no preprocessing)
- ConvNet models can require plenty of data to be trained
- Data Augmentation provides an answer to the lack of data
- We proposed two Side-Channel-adapted Data Augmentation techniques
- We verified the effectiveness/efficiency of the ConvNet+Data Augmentation approach over different sets of misaligned data

Thank You! Questions?

Cross-entropy Loss Function

\[ F(x, W) = s \circ \lambda_n \circ \sigma_{n-1} \circ \lambda_{n-1} \circ \cdots \circ \lambda_1(x) = y \approx \Pr[Z|X = x] \]

- \( \lambda_i \) affine functions \( (\lambda_i(x) = Ax + b) \) depending on some parameters \( W \) (weights)
- \( \sigma_i \) non-linear functions (activation functions)
- \( s \) softmax function \( s(x)[i] = \frac{e^{x[i]}}{\sum_j e^{x[j]}} \)
- The weights \( W \) are trained on the basis of a training set \( \{x_i, z_i\}_{i=1,\ldots,N} \), by minimizing the loss function
  \[ L(W, x_i, z_i) = \frac{1}{N} \sum_{i=1}^{N} D(F(x_i, W), I(z_i)) \]

where \( I(z^i) = (0,0,\ldots,0,1,0,\ldots,0) \) (probability distribution of \( Z|Z = z^i \))
and \( D(f_X, f_Y) = -\sum_z f_Y(z) \log(f_X(z)) \) is the cross-entropy between two probability distributions \( f_X, f_Y \) (over the same probability space)

Cross-entropy measures dissimilarity between two probability distributions
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Nowadays challenging image classification problems

Imagenet Large Scale Visual Recognition Challenge 2012 (ILSVRC 2012)

Classification and other tasks. 1000 categories.

Convolutional Neural Networks (AlexNet [KSH12]) ≈ 84% of accuracy
Best of other submissions ≈ 74% of accuracy
Nowadays challenging image classification problems

Imagenet Large Scale Visual Recognition Challenge 2012 (ILSVRC 2012)

Classification and other tasks. 1000 categories.

Convolutional Neural Networks (AlexNet [KSH12]) \( \approx 84\% \) of accuracy
Best of other submissions \( \approx 74\% \) of accuracy
At ILSVRC 2014 classification accuracy of winner GoogLeNet was \( \approx 93\% \)
(close to human rate)
Nowadays challenging image classification problems

Imagenet Large Scale Visual Recognition Challenge 2012 (ILSVRC 2012)

Classification and other tasks. 1000 categories.

Convolutional Neural Networks (*AlexNet* [KSH12]) \(\approx 84\%\) of accuracy
Best of other submissions \(\approx 74\%\) of accuracy
At ILSVRC 2014 classification accuracy of winner *GoogLeNet* was \(\approx 93\%\)
(close to human rate)

Today, CNNs are the most diffused and best performing technique in many domains (image recognition, video analysis, natural language processing,...)
Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasures

Nowadays challenging image classification problems

Imagenet Large Scale Visual Recognition Challenge 2012 (ILSVRC 2012)

Classification and other tasks. 1000 categories.

Convolutional Neural Networks \((\text{AlexNet [KSH12]})) \approx 84\% \text{ of accuracy}
Best of other submissions \approx 74\% \text{ of accuracy}
At ILSVRC 2014 classification accuracy of winner GoogLeNet was \approx 93\% (close to human rate)

Today, CNNs are the most diffused and best performing technique in many domains (image recognition, video analysis, natural language processing,...)
It deals geometric deformation and de-synchronization (spacial or temporal)
Artificial Jitter

Target Variable: \( Z = \text{HW}(\text{Sbox}(P \oplus K)) \)

\( \approx 2000 \) time samples

Countermeasure: artificial signal treatment simulating clock jitter

10000 training traces
## Artificial Jitter (2)

### Low-jitter

<table>
<thead>
<tr>
<th>Acc</th>
<th>(N^*)</th>
<th>(\text{SH}_0)</th>
<th>(\text{SH}_{20})</th>
<th>(\text{SH}_{40})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{AR}_0)</td>
<td>57.4%</td>
<td>14</td>
<td>82.5%</td>
<td>6</td>
</tr>
<tr>
<td>(\text{AR}_{100})</td>
<td>86.0%</td>
<td>6</td>
<td>87.0%</td>
<td>5</td>
</tr>
<tr>
<td>(\text{AR}_{200})</td>
<td>86.6%</td>
<td>6</td>
<td>85.7%</td>
<td>6</td>
</tr>
</tbody>
</table>

### High-jitter

<table>
<thead>
<tr>
<th>Acc</th>
<th>(N^*)</th>
<th>(\text{SH}_0)</th>
<th>(\text{SH}_{20})</th>
<th>(\text{SH}_{40})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{AR}_0)</td>
<td>40.6%</td>
<td>35</td>
<td>51.1%</td>
<td>9</td>
</tr>
<tr>
<td>(\text{AR}_{100})</td>
<td>50.2%</td>
<td>15</td>
<td>72.4%</td>
<td>11</td>
</tr>
<tr>
<td>(\text{AR}_{200})</td>
<td>64.0%</td>
<td>11</td>
<td><strong>75.5%</strong></td>
<td>8</td>
</tr>
</tbody>
</table>

![Graph](image1)

![Graph](image2)

**Low Jitter**

**High Jitter**
Random Delays - Two Leaking Operations

Two leaking operations

First operation - Test acc: 76.8%, $N^* = 7$
Second operation - Test acc: 82.5%, $N^* = 6$
## Artificial Jitter

<table>
<thead>
<tr>
<th>DS_{low_jitter}</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>(b)</td>
<td>(SH_0)</td>
<td>(SH_{20})</td>
<td>(SH_{40})</td>
</tr>
<tr>
<td>(c)</td>
<td>(d)</td>
<td></td>
<td>(%)</td>
<td></td>
</tr>
<tr>
<td>(AR_0)</td>
<td>100.0%</td>
<td>68.7%</td>
<td>99.8%</td>
<td>86.1%</td>
</tr>
<tr>
<td>57.4%</td>
<td>14</td>
<td>82.5%</td>
<td>6</td>
<td>83.6%</td>
</tr>
<tr>
<td>(AR_{100})</td>
<td>87.7%</td>
<td>88.2%</td>
<td>82.4%</td>
<td>88.4%</td>
</tr>
<tr>
<td>86.0%</td>
<td>6</td>
<td>87.0%</td>
<td>5</td>
<td>87.5%</td>
</tr>
<tr>
<td>(AR_{200})</td>
<td>83.2%</td>
<td>88.6%</td>
<td>81.4%</td>
<td>86.9%</td>
</tr>
<tr>
<td>86.6%</td>
<td>6</td>
<td>85.7%</td>
<td>6</td>
<td>87.7%</td>
</tr>
<tr>
<td>(AR_{500})</td>
<td></td>
<td></td>
<td></td>
<td>85.0%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>86.2%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DS_{high_jitter}</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>(b)</td>
<td>(SH_0)</td>
<td>(SH_{20})</td>
<td>(SH_{40})</td>
</tr>
<tr>
<td>(c)</td>
<td>(d)</td>
<td></td>
<td>(%)</td>
<td></td>
</tr>
<tr>
<td>(AR_0)</td>
<td>100%</td>
<td>45.0%</td>
<td>100%</td>
<td>60.0%</td>
</tr>
<tr>
<td>40.6%</td>
<td>35</td>
<td>51.1%</td>
<td>9</td>
<td>62.4%</td>
</tr>
<tr>
<td>(AR_{100})</td>
<td>90.4%</td>
<td>57.3%</td>
<td>76.6%</td>
<td>73.6%</td>
</tr>
<tr>
<td>50.2%</td>
<td>15</td>
<td>72.4%</td>
<td>11</td>
<td>73.5%</td>
</tr>
<tr>
<td>(AR_{200})</td>
<td>83.1%</td>
<td>67.7%</td>
<td>82.0%</td>
<td>77.1%</td>
</tr>
<tr>
<td>64.0%</td>
<td>11</td>
<td>75.5%</td>
<td>8</td>
<td>74.4%</td>
</tr>
<tr>
<td>(AR_{500})</td>
<td></td>
<td></td>
<td></td>
<td>83.6%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>68.2%</td>
</tr>
</tbody>
</table>