Eleonora CAGLI
Cécile DUMAS
Emmanuel PROUFF

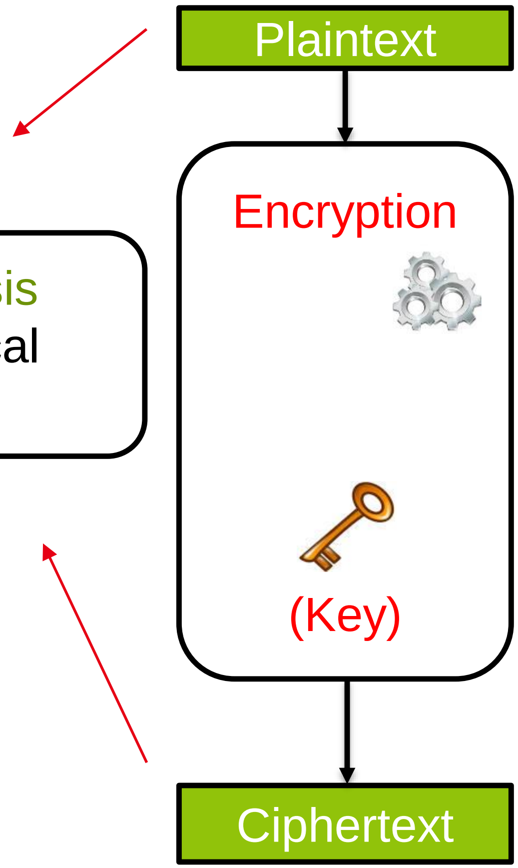
Journées C2 2017
24.04.2017 La Bresse

CONVOLUTIONAL NEURAL NETWORKS WITH DATA AUGMENTATION

A COMPREHENSIVE PROFILING SIDE-CHANNEL ATTACK STRATEGY, ROBUST TO TRACES MISALIGNMENT

CONTENTS

SIDE-CHANNEL ATTACKS: INTRODUCTION



Algebraic Cryptanalysis
Based on mathematical weaknesses

SIDE-CHANNEL ATTACKS: INTRODUCTION

Plaintext

Encryption

(Key)

Ciphertext

Power Consumption

Electromagnetic Emanation

...

Unintended Signals

Algebraic Cryptanalysis
Based on mathematical weaknesses

Side-Channel Cryptanalysis
Based on physical weaknesses

SIDE-CHANNEL ATTACKS: TARGET

Divide and Conquer Strategy:

Retrieve a *small enough* part of the secret key (subkey) at time
(*small enough* = possible to enumerate all hypotheses)

Example [AES-128]: key of 16 bytes, retrieve each byte of the key independently

Target: *Sensitive Variable*

A variable $Z \in \mathcal{Z} = \{z^1, z^2, \ldots z^{|\mathcal{Z}|}\}$ handled during the encryption algorithm, and which depends on a subkey

Example [AES-128]: $Z = S(P \oplus K)$,
being *P* a byte of plaintext, *K* a byte of key and *S* the SubByte operation (first round)

SIDE-CHANNEL TRACES AND ANALYSIS

Encryption

…+*%…

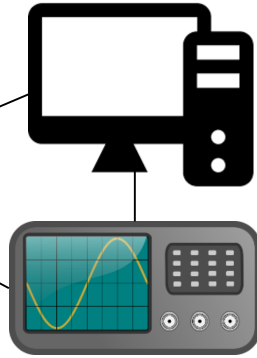
…+-/*%…

$$Z = S(P \oplus K)$$

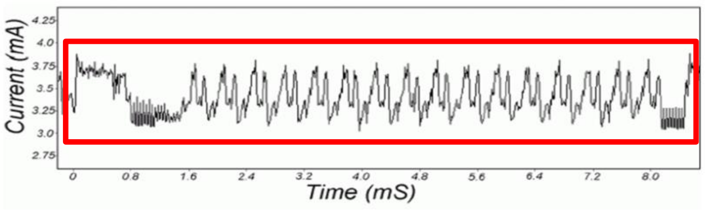
…%+*…

…+-/*%…

(Key)



$X \in \mathbb{R}^D$ Noisy signal



SIDE-CHANNEL TRACES AND ANALYSIS

Encryption

…+*%…

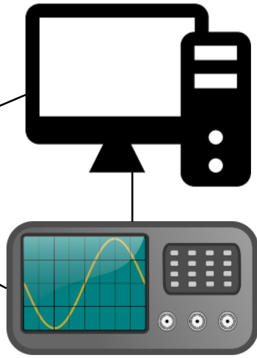
…+-/*%…

$Z = S(P \oplus K)$

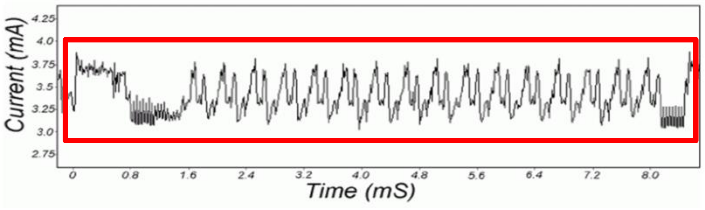
…%+*…

…+-/*%…

(Key)



$X \in \mathbb{R}^D$ Noisy signal



Make inference about Z

Estimate $\Pr[Z|X]$

SIDE-CHANNEL TRACES AND ANALYSIS

Encryption

…+*%…

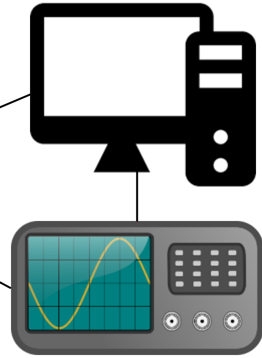
…+-/*%…

$Z = S(P \oplus K)$

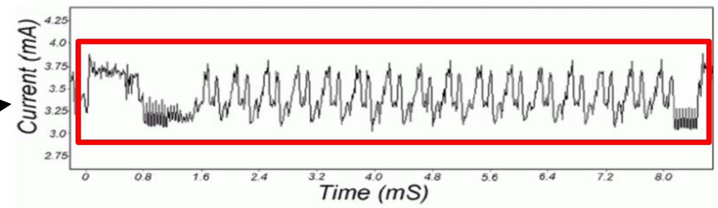
…%+*…

…+-/*%…

(Key)



$X \in \mathbb{R}^D$ Noisy signal



Make inference about Z

Estimate $\Pr[Z|X]$

Template Attack

Profiling phase

- Collect a *profiling set* $\{x_i, z_i\}_{i=1,..,N}$
- Estimate the *templates* $Pr[X|Z]$ $(\Pr[X], \Pr[Z])$

Attack phase

- Collect *attack* traces $x_1, x_2, \dots, x_{N'}$ (plaintexts $P_1, P_2, \dots, P_{N'}$)
- Make hypotheses $k_1, k_2, \dots, k_{256}$
- Use Bayes theorem: $\Pr[Z|X] = \frac{\Pr[X|Z]\Pr[X]}{\Pr[Z]}$
- $Score_{k_i} = \Pr\left[Z = S(p_n \oplus k_i) \middle| \{x_n, p_n\}_{n=1}^{N'}, k_i\right]$

SIDE-CHANNEL TRACES AND ANALYSIS

Encryption

…+*%…

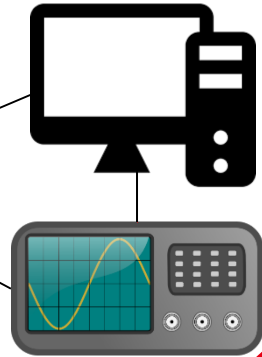
…+-/*%…

$Z = S(P \oplus K)$

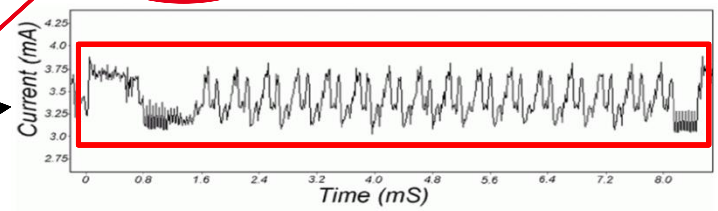
…%+*…

…+-/*%…

(Key)



$X \in \mathbb{R}^D$ Noisy signal



$X \in \mathbb{R}^D$

$Pr[X|Z]$

Make inference about Z

Estimate $\Pr[Z|X]$

Template Attack

Profiling phase

- Collect a *profiling set* $\{x_i, z_i\}_{i=1,\dots,N}$
- Estimate the *templates* $Pr[X|Z]$ $(\Pr[X], \Pr[Z])$

Attack phase

- Collect *attack* traces $x_1, x_2, \dots, x_{N'}$ (plaintexts $P_1, P_2, \dots, P_{N'}$)
- Make hypotheses $k_1, k_2, \dots, k_{256}$
- Use Bayes theorem: $\Pr[Z|X] = \frac{\Pr[X|Z]\Pr[X]}{\Pr[Z]}$
- $Score_{k_i} = \Pr\left[Z = S(p_n \oplus k_i) \middle| \{x_n, p_n\}_{n=1}^{N'}, k_i\right]$

DIMENSIONALITY REDUCTION

Problem (curse of dimensionality): *D* might be huge! *X* is strongly multivariate, estimate $\Pr[X|Z]$ is complexe

Solution:

DIMENSIONALITY REDUCTION

Problem (curse of dimensionality): *D* might be huge! *X* is strongly multivariate, estimate $\Pr[X|Z]$ is complexe

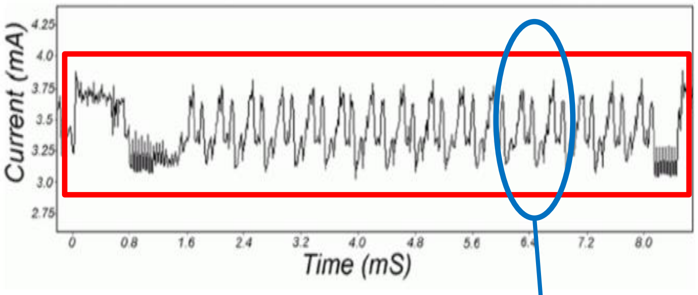
Solution:

- Gaussian Hypothesis
  (only estimate means and covariance matrices)

- Dimensionality Reduction

DIMENSIONALITY REDUCTION

Problem (curse of dimensionality): *D* might be huge! *X* is strongly multivariate, estimate $\Pr[X|Z]$ is complexe



Points of Interest: coordinates of X which depend on *Z*

Solution:

- Gaussian Hypothesis (only estimate means and covariance matrices)
- Dimensionality Reduction
  - Select a few *Points of Interest* (via some statistical tests)
  - Dimensionality Reduction techniques (e.g. PCA)

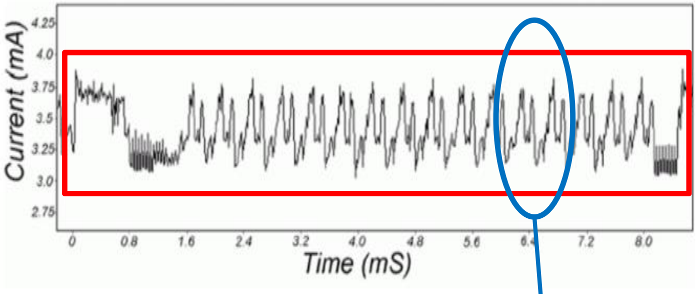
DIMENSIONALITY REDUCTION

Problem (curse of dimensionality): *D* might be huge! *X* is strongly multivariate, estimate $\Pr[X|Z]$ is complexe

Solution:

- Gaussian Hypothesis
  (only estimate means and covariance matrices)

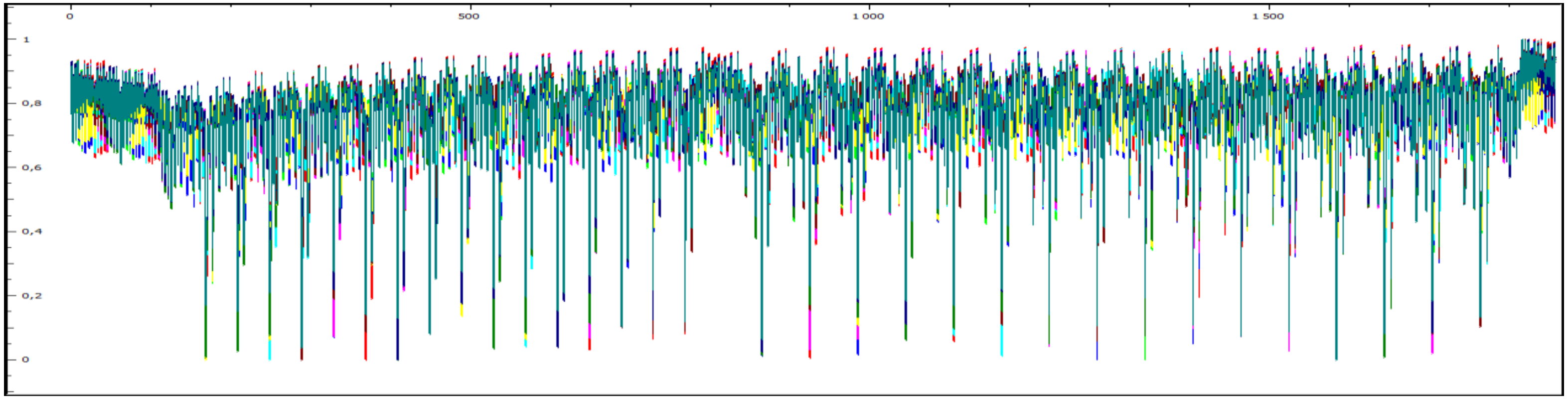
- Dimensionality Reduction
  - Select a few *Points of Interest* (via some statistical tests)
  - Dimensionality Reduction techniques (e.g. PCA)



Points of Interest: coordinates of X which depend on *Z*

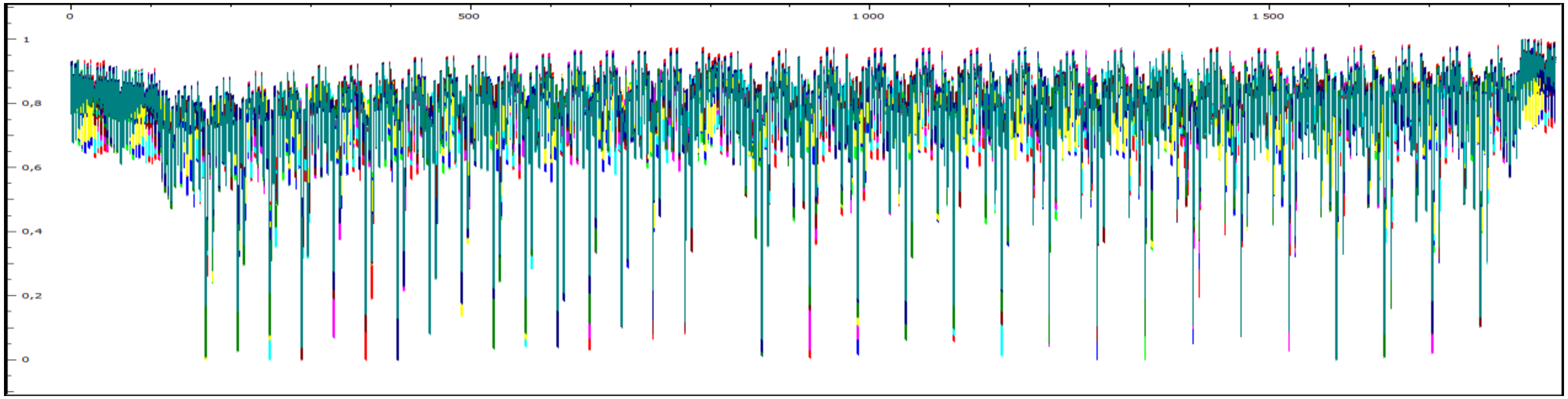
But: Approaches highly affected by geometrical deformations of the signals (e.g. delays, unstable clock frequency,…) causing trace misalignment

MISALIGNMENT

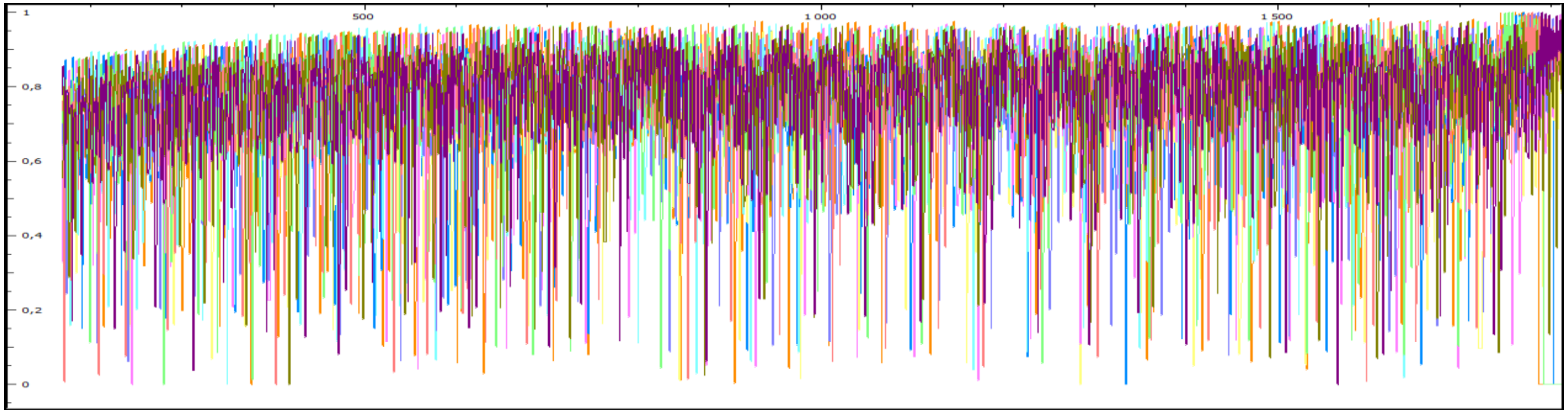


10 Aligned traces

MISALIGNMENT



10 Aligned traces



10 Misaligned traces (unstable clock frequency)

STATE OF THE ART (1)

Template attacks

- *Chari et al. "* Template attacks*." CHES, 2002*
- *Choudary and Kuhn. "* Efficient template attacks*." CARDIS, 2014*

Realignment Techniques

- *Nagashima et al. "* DPA using phase-based waveform matching against random-delay countermeasure.*" ISCAS, 2007*
- *van Woudenberg et al. "* Improving differential power analysis by elastic alignment*. " CT-RSA, 2011*

Dimensionality Reduction

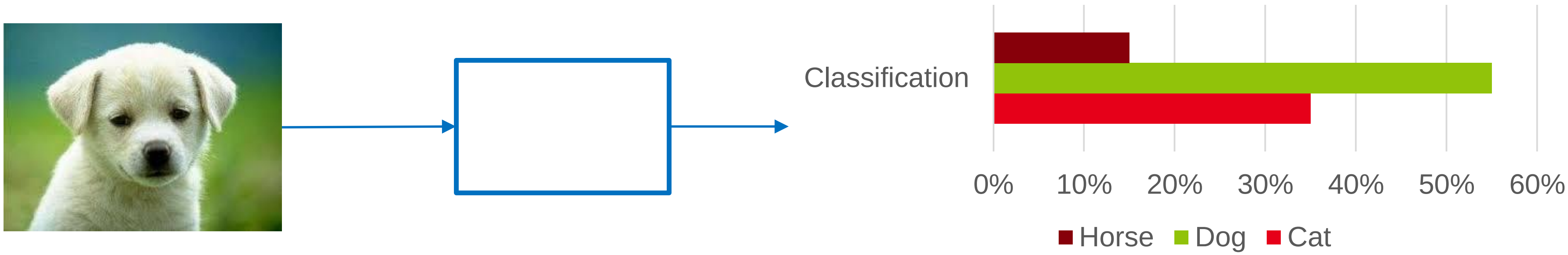
- *Standaert and Archambeau. "* Using subspace-based template attacks to compare and combine power and electromagnetic information leakages*." CHES, 2008*
- *Batina et al. "* Getting more from PCA: First results of using principal component analysis for extensive power analysis*." CT-RSA, 2012*
- *Cagli et al. "* Kernel Discriminant Analysis for information extraction in the presence of masking*." CARDIS, 2016*

STATE OF THE ART (2)

Machine Learning

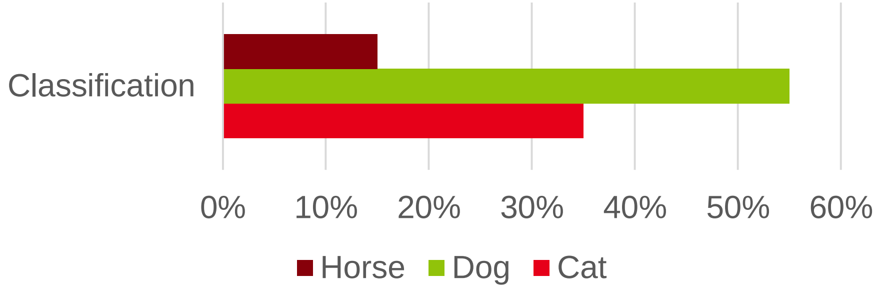
- *Bartkewitz and Lemke-Rust. "* Efficient template attacks based on probabilistic multi-class Support Vector Machines*" CARDIS, 2013*
- *Hospodar et al. "* Machine learning in side-channel analysis: a first study« *Journal of Cryptographic Engineering, 2013*
- *Lerman et al. "* Power analysis attack: an approach based on machine learning*" International Journal of Applied Cryptography, 2014*
- *Whitnall and Oswald. "* Robust Profiling for DPA-Style Attacks*" CHES, 2015*
- *Maghrebi et al. "* Breaking Cryptographic Implementations Using Deep Learning Techniques*" SPACE, 2016*

CLASSIFICATION WITH NEURAL NETWORK - INTRODUCTION



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

CLASSIFICATION WITH NEURAL NETWORK - INTRODUCTION



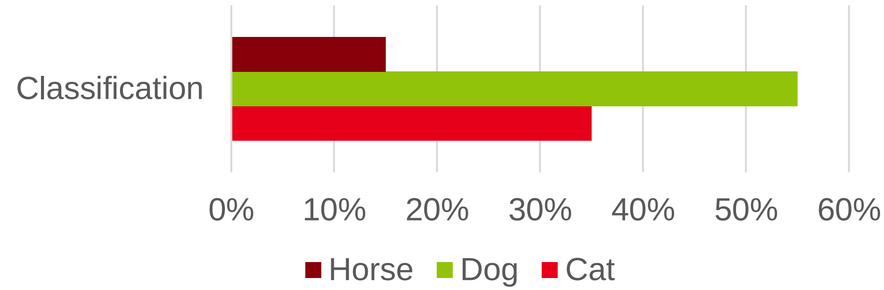
Classification: Assign to a datum X (e.g. an image) a label Z among a set of possible labels $\mathcal{Z} = \{Horse, Dog, Cat\}$

Make inference about Z

Estimate $\Pr[Z|X]$

The same as SCA!

CLASSIFICATION WITH NEURAL NETWORK - INTRODUCTION

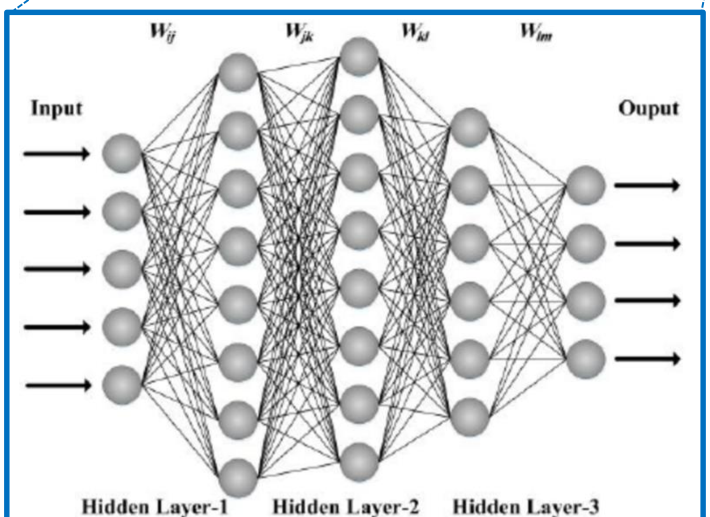


Classification: Assign to a datum X (e.g. an image) a label Z among a set of possible labels $\mathcal{Z} = \{Horse, Dog, Cat\}$

Make inference about Z

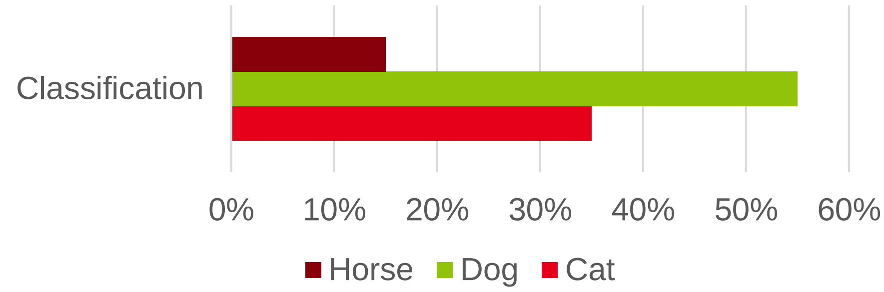
Estimate $\Pr[Z|X]$

The same as SCA!



Multilayer Perceptron – figure from www.researchgate.net/

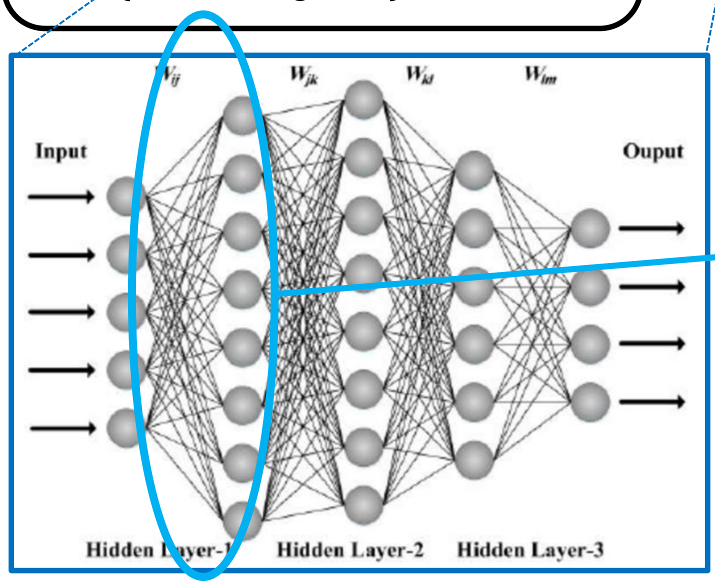
CLASSIFICATION WITH NEURAL NETWORK - INTRODUCTION



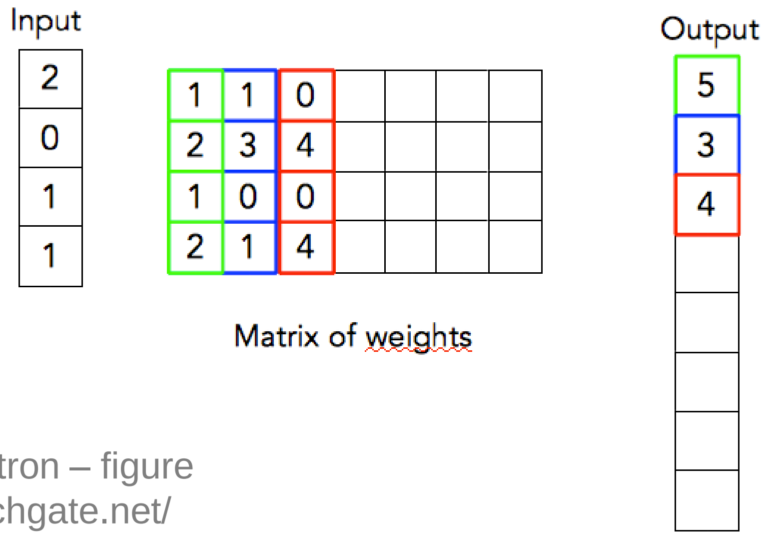
Classification: Assign to a datum X (e.g. an image) a label Z among a set of possible labels $\mathcal{Z} = \{Horse, Dog, Cat\}$

Make inference about Z
Estimate $\Pr[Z|X]$

The same as SCA!



Multilayer Perceptron – figure from www.researchgate.net/



CLASSIFICATION WITH NEURAL NETWORKS

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

- λ_i affine functions ($\lambda_i(x) = Ax + b$) depending on some parameters W (*weights*)
- σ_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 $I(z^i) = (0,0,\ldots 0,1,0,\ldots 0) = f_{Z|Z=z^i}$ (probability distribution of $Z|Z=z^i$)
and $D(f_X, f_Y) = -\sum_z \mathrm{f_Y(z)log}(f_{X(z)})$ is the *cross-entropy* between two probability distributions f_X, f_Y (over the same probability space)

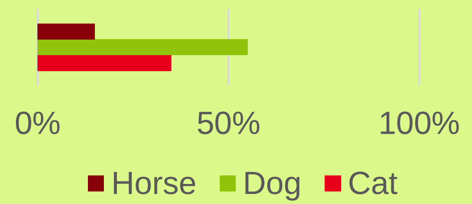
CLASSIFICATION WITH NEURAL NETWORKS

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

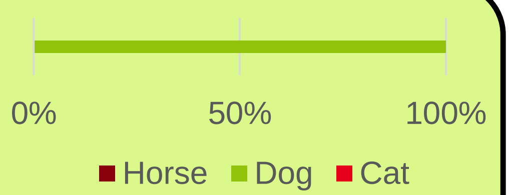
- λ_i affine functions ($\lambda_i(x) = Ax + b$) depending on some parameters W (*weights*)
- σ_i non-linear functions *(activation functions)*
- s *softmax function* $s(x)[i] = \frac{e^{x[i]}}{\sum_j e^{x[j]}}$
- The weights W are *trained* on the basis of a *training set* $\{x_i, z_i\}_{i=1,..,N}$, 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 $I(z^i) = (0,0,\dots 0,1,0,\dots 0) = f_{Z|Z=z^i}$ (probability distribution of $Z|Z=z^i$)
and $D(f_X, f_Y) = -\sum_z \mathrm{f_Y(z)}\log(f_{X(z)})$ is the *cross-entropy* between two probability distributions f_X, f_Y (over the same probability space)



$F(x_i, W) \approx Pr[Z|X = x_i]$



$I(z_i) = Pr[Z|Z = z_i]$

Cross-entropy measures dissimilarity between two probability distributions

CLASSIFICATION WITH NEURAL NETWORKS

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

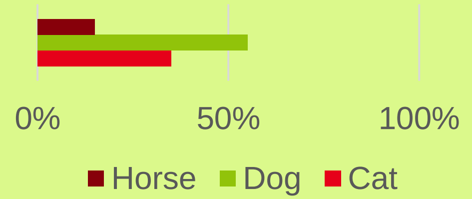
- λ_i affine functions ($\lambda_i(x) = Ax + b$) depending on some parameters W (*weights*)
- σ_i non-linear functions *(activation functions)*
- s *softmax function* $s(x)[i] = \frac{e^{x[i]}}{\sum_j e^{x[j]}}$
- The weights W are *trained* on the basis of a *traini* the *loss function*

$$L(W, x_i, z_i) = \frac{1}{N}\sum_{i=1}^{N} D(F(x_i, W), I(z_i))$$

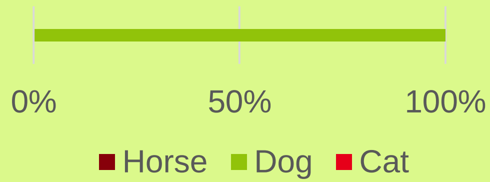
Minimization is performed iteratively via (Stochastic) Gradient Descent

$$W_{t+1}[w] = W_t[w] - \gamma \frac{\partial L(W, x_i, z_i)}{\partial W[w]}$$

where $I(z^i) = (0,0,\dots 0,1,0,\dots 0) = f_{Z|Z=z^i}$ (probability distribution of $Z|Z=z^i$)
and $D(f_X, f_Y) = -\sum_z \mathrm{f}_Y(\mathrm{z})\log(f_{X(z)})$ is the *cross-entropy* between two probability distributions f_X, f_Y (over the same probability space)



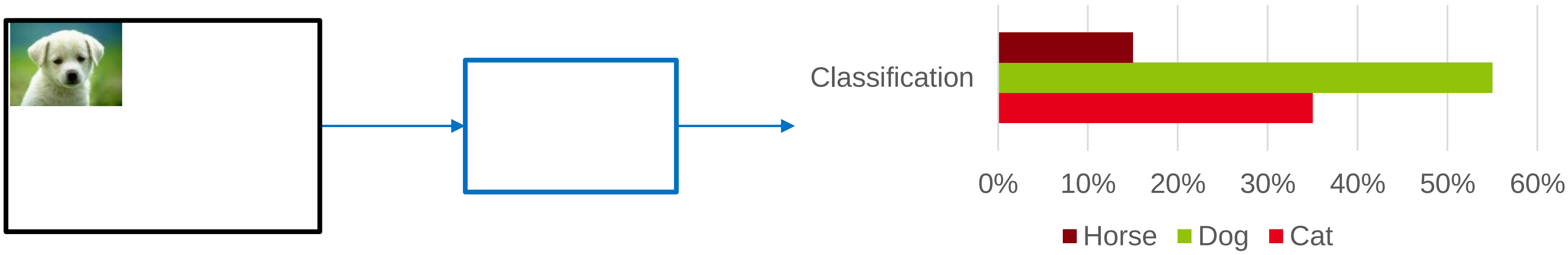
$F(x_i, W) \approx Pr[Z|X = x_i]$



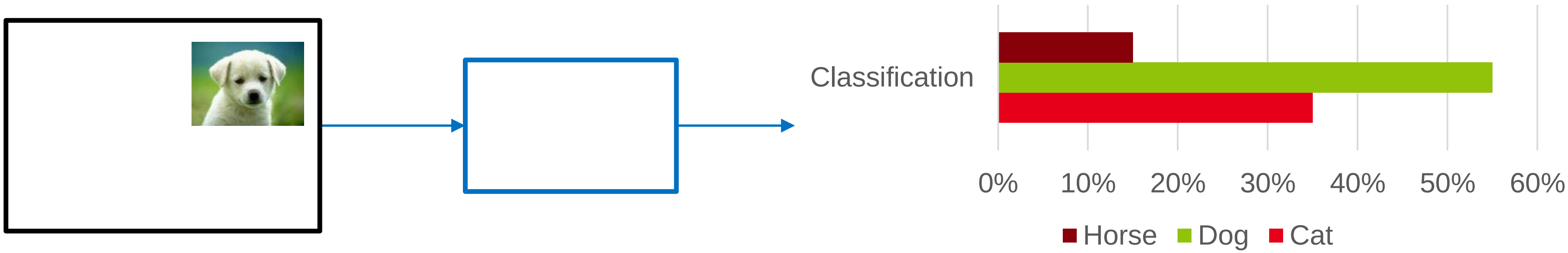
$I(z_i) = Pr[Z|Z = z_i]$

Cross-entropy measures dissimilarity between two probability distributions

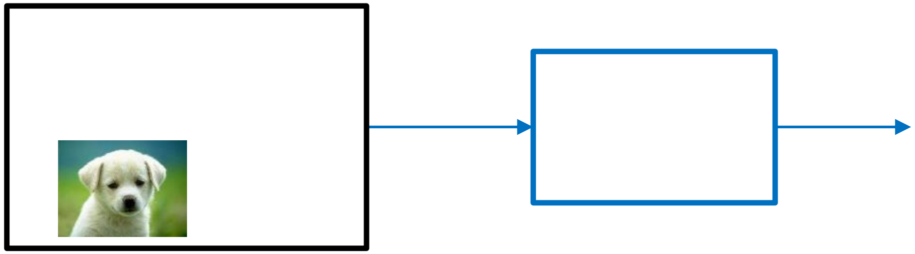
CONVOLUTIONAL NEURAL NETWORKS

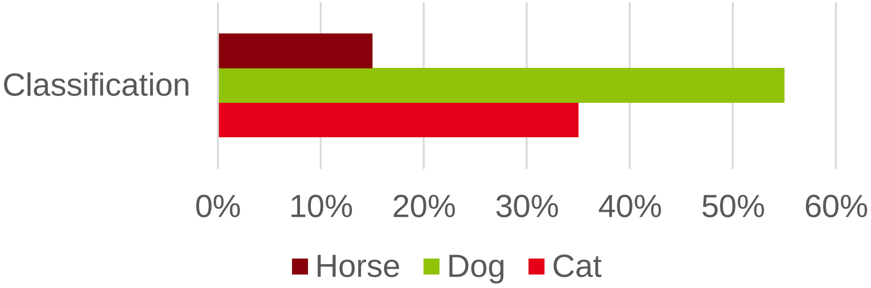


CONVOLUTIONAL NEURAL NETWORKS

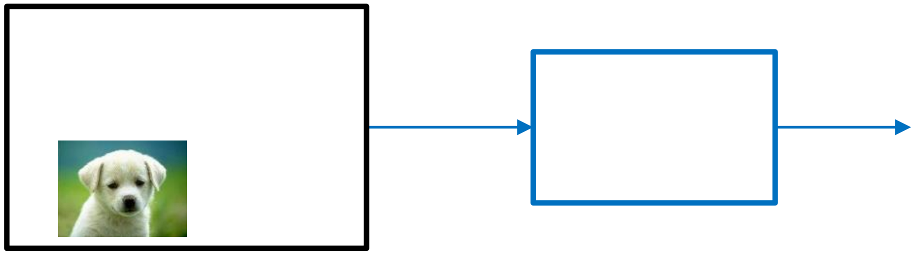


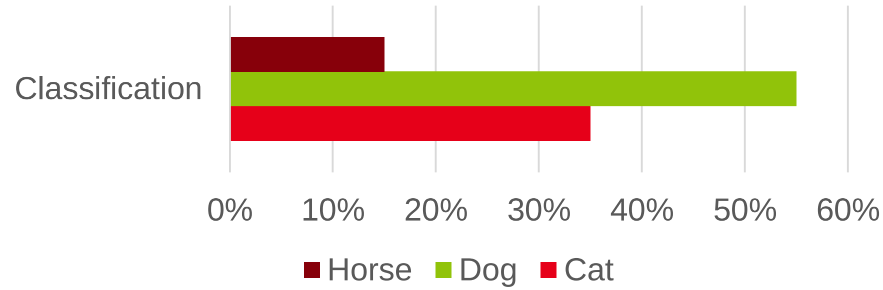
CONVOLUTIONAL NEURAL NETWORKS





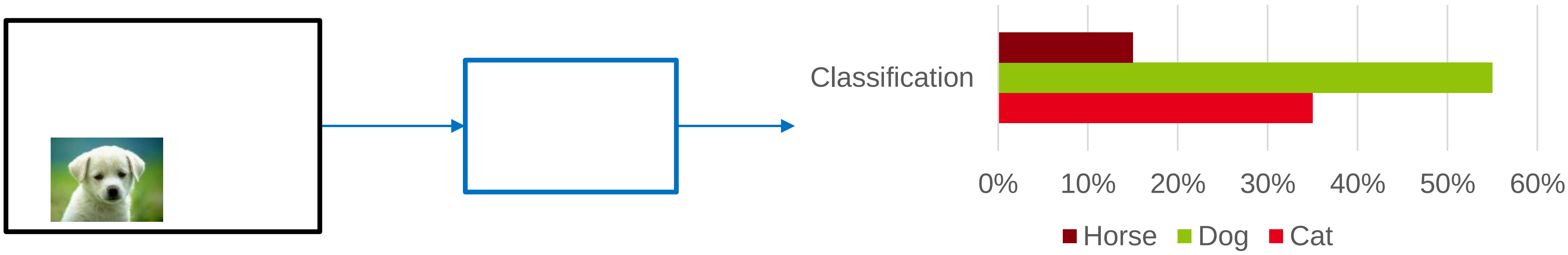
CONVOLUTIONAL NEURAL NETWORKS



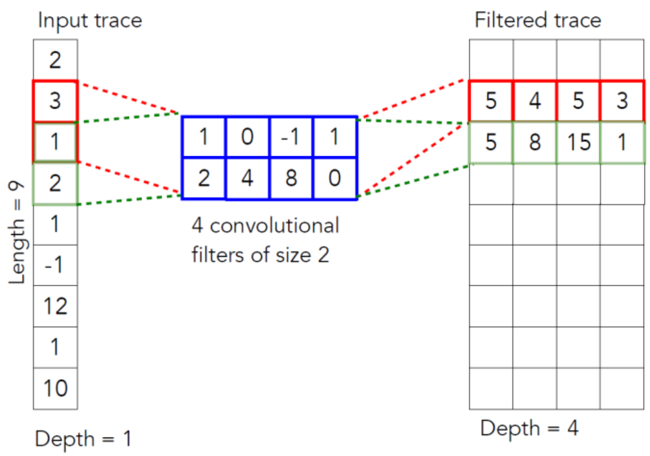


How to take shift-invariance into account? → Sharing weights

CONVOLUTIONAL NEURAL NETWORKS

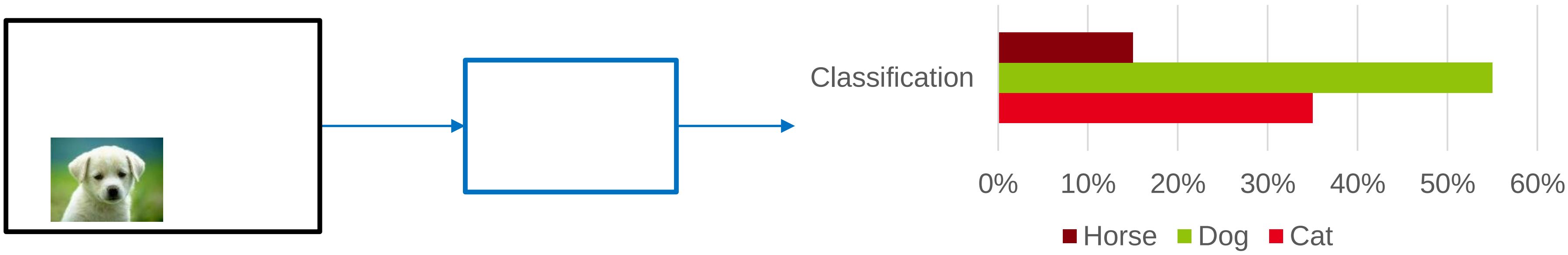


How to take shift-invariance into account? → Sharing weights

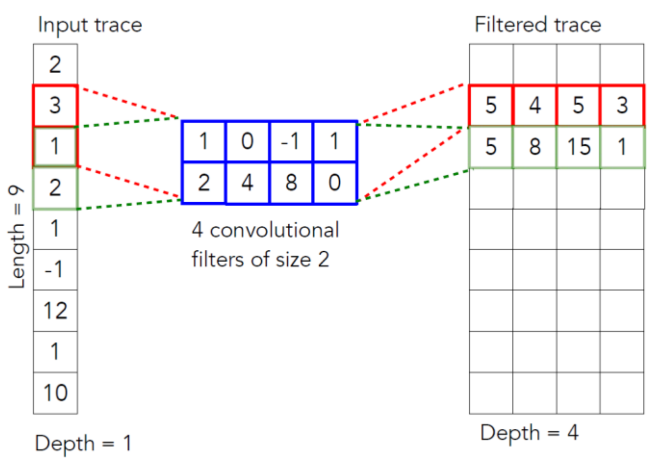


Convolutional Layer

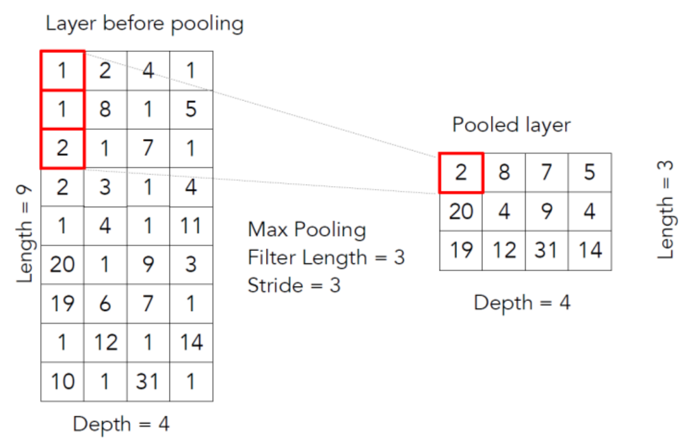
CONVOLUTIONAL NEURAL NETWORKS



How to take shift-invariance into account? → Sharing weights

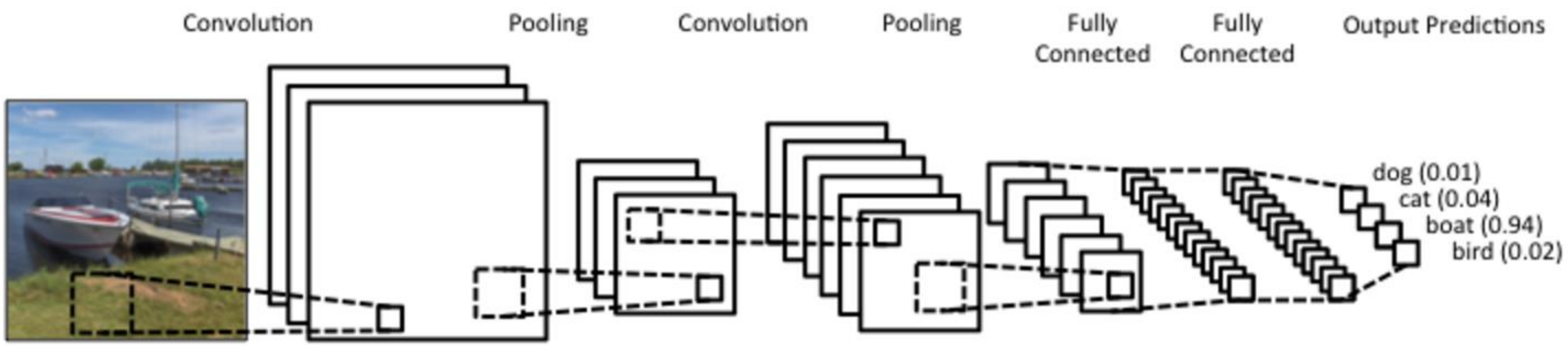


Convolutional Layer

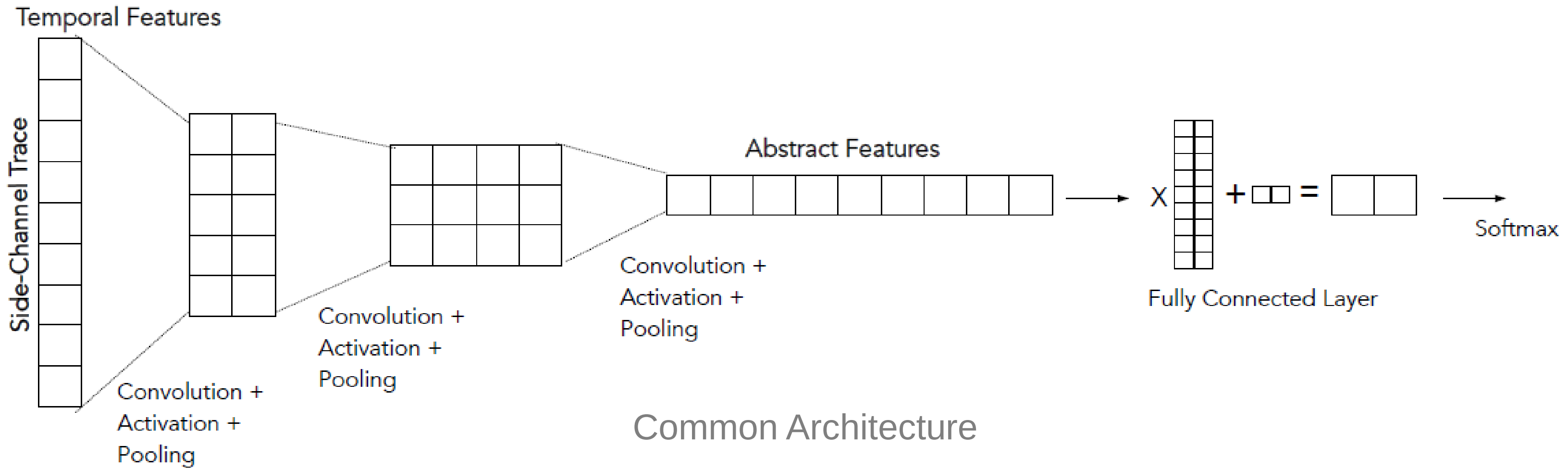


Max-Pooling Layer

CONVOLUTIONAL NEURAL NETWORKS – LAYERS AND ARCHITECTURE



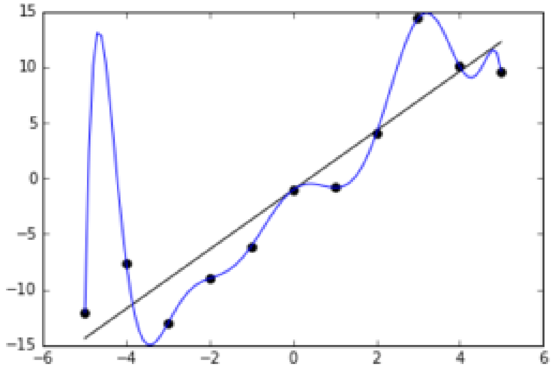
ConvNet scheme from *www.wildml.com*



Common Architecture

OVERFITTING

- The more weights, the more flexible the network is (able to better fit the training data)
- BUT: overfitting phenomenon



Example of overfitting for a regression problem

Solutions:

1) More data
2) A more constrained network (less parameters)
3) Regularization or Data Augmentation

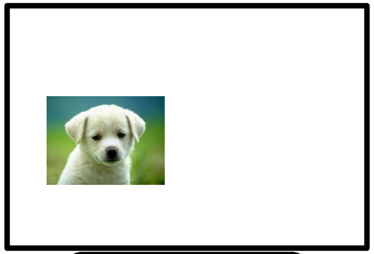
DATA AUGMENTATION

Generate artificially new training data by deforming those previously acquired,
Applying transformations that preserve the label Z

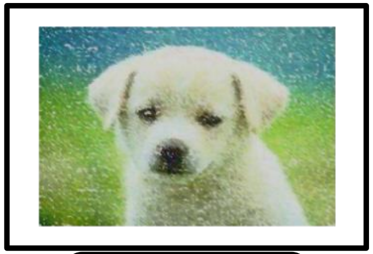
$Z = Dog$

Original Datum

$Z = Dog$



$Z = Dog$



$Z = Dog$

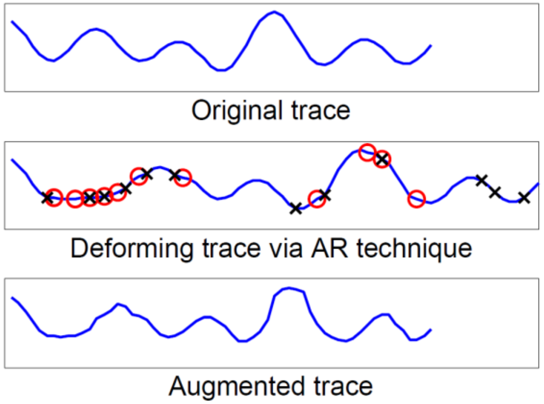
Augmented Data

Deformation techniques for images

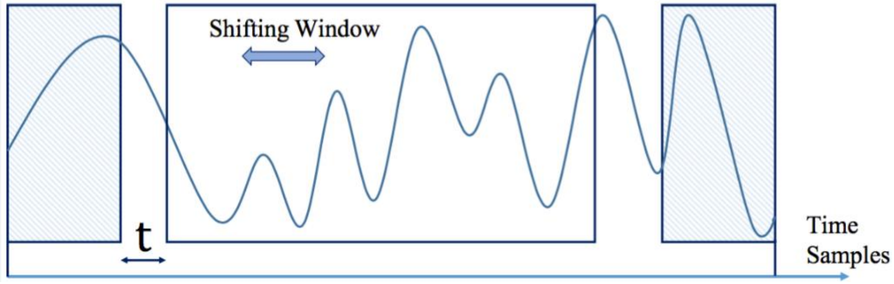
DATA AUGMENTATION

Generate artificially new training data by deforming those previously acquired,
Applying transformations that preserve the label Z

Deformation techniques for side-channel traces

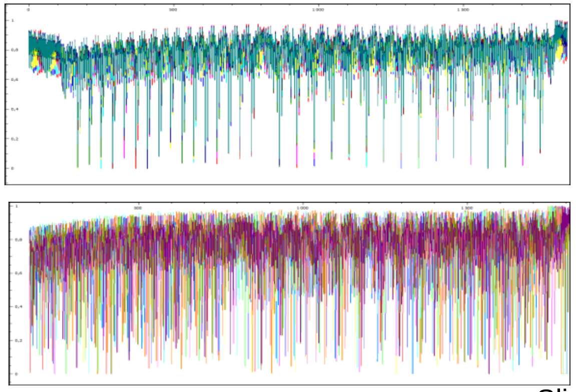


Add-Remove Deformation



Shifting Deformation

EXPERIMENTAL RESULTS

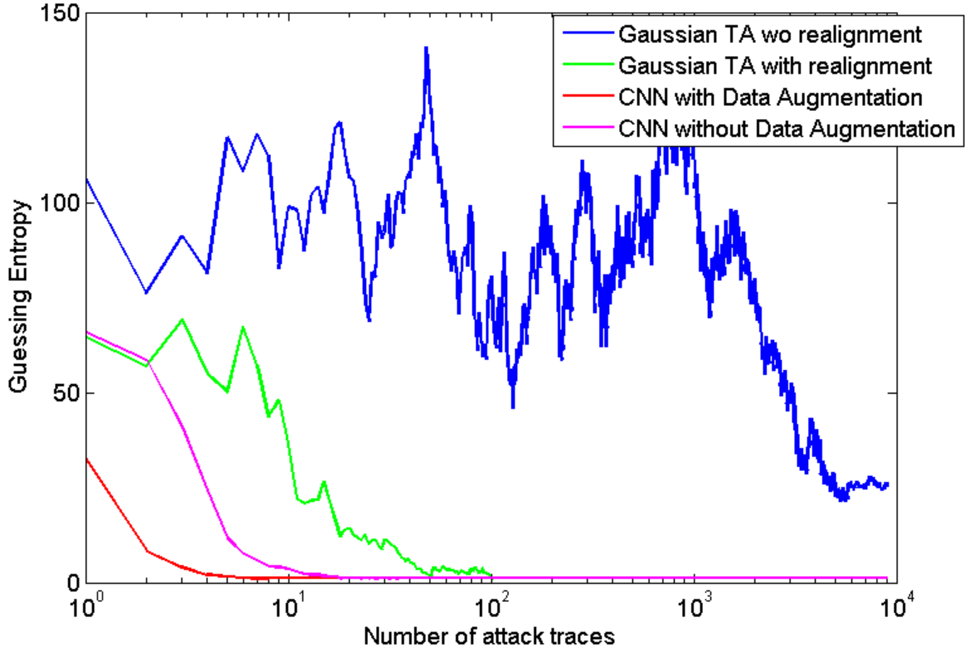


Slide 7

- $Z = S(P \oplus K)$
- Aligned case: Gaussian and CNN approach same performances (~5 traces for success)
- Misaligned case:
  - For Gaussian approach a wide range of PoI selections has been tried
  - CNN architecture:

$$F(x) = s \circ \lambda \circ \delta_4 \circ \sigma_4 \circ \gamma_4 \circ \delta_3 \circ \sigma_3 \circ \gamma_3 \circ \delta_2 \circ \sigma_2 \circ \gamma_2 \circ \delta_1 \circ \sigma_1 \circ \gamma_1$$

    where γ are convolutional layers and δ are poolings
  - Data Augmentation is done composing AR and SH deformations



CONCLUSIONS

Template attacks issues

CONCLUSIONS

Template attacks issues

Dimensionality

Misalignment

Bounded number of acquisitions

CONCLUSIONS

Template attacks issues

Dimensionality

Misalignment

Bounded number of acquisitions

Analogy between SCA and classification
Analogy between geometrical deformation and misalignment

CONCLUSIONS

Template attacks issues

Dimensionality

Misalignment

Bounded number of acquisitions

Convolutional Neural Networks

- Robust to geometrical deformations
- Not affected by curse of dimensionality

Data Augmentation

Analogy between SCA and classification
Analogy between geometrical deformation and misalignment

CONCLUSIONS

Template attacks issues

Dimensionality

Misalignment

Bounded number of acquisitions

Convolutional Neural Networks

- Robust to geometrical deformations
- Not affected by curse of dimensionality

Data Augmentation

Analogy between SCA and classification
Analogy between geometrical deformation and misalignment

Will CNN take the place of Gaussian TA?

THANK YOU! ANY QUESTIONS?