Breaking the Blindfold:

Deep Learning-based Blind Side-channel Analysis

Azade Rezaeezade

Trevor Yap

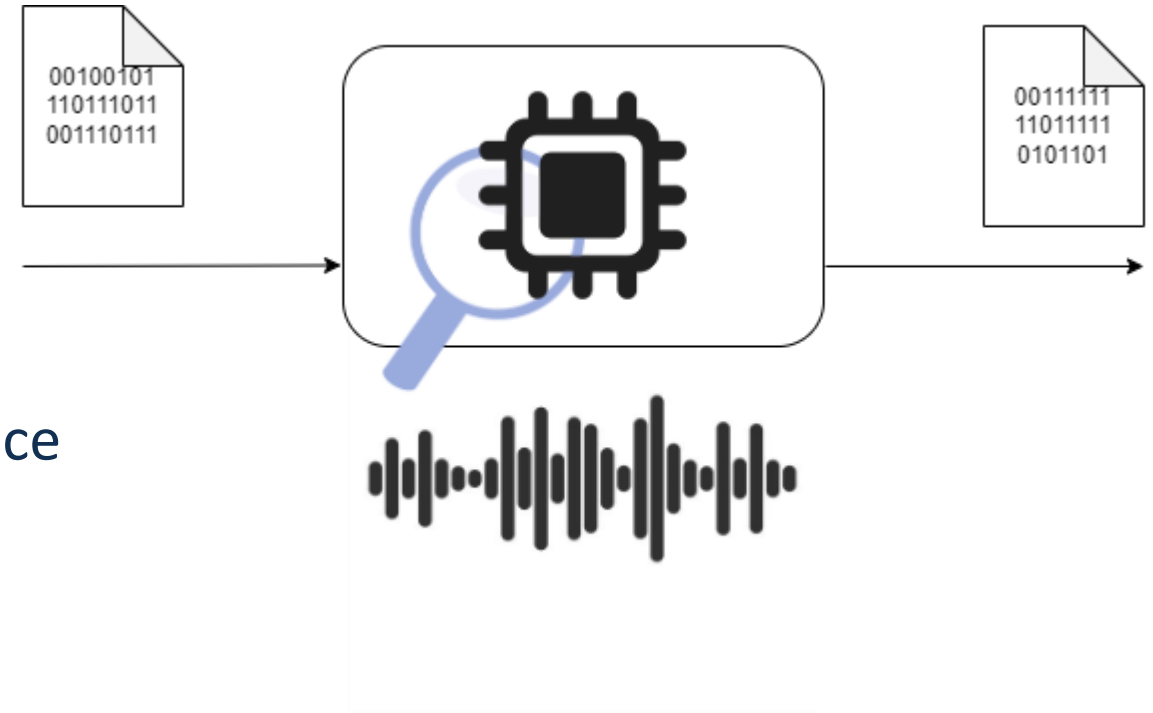
Dirmanto Jap

Shivam Bhasin

Stjepan Picek

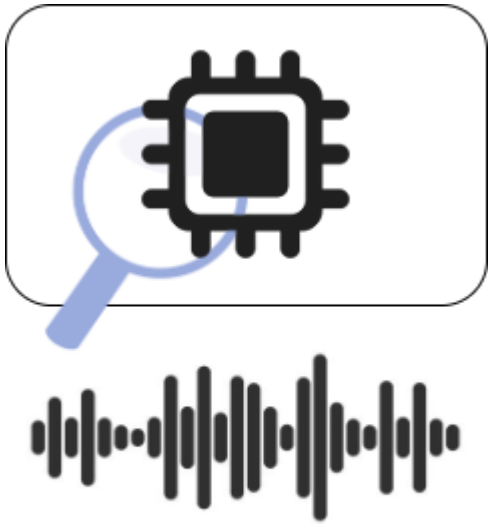
What is SCA

- Non-profiled scenario
  - Know plain/ciphertext from DUA
  - Guess about targeted variable
- Profiled scenario
  - Know plain/ciphertext from clone device
  - Profile building
  - Profile matching

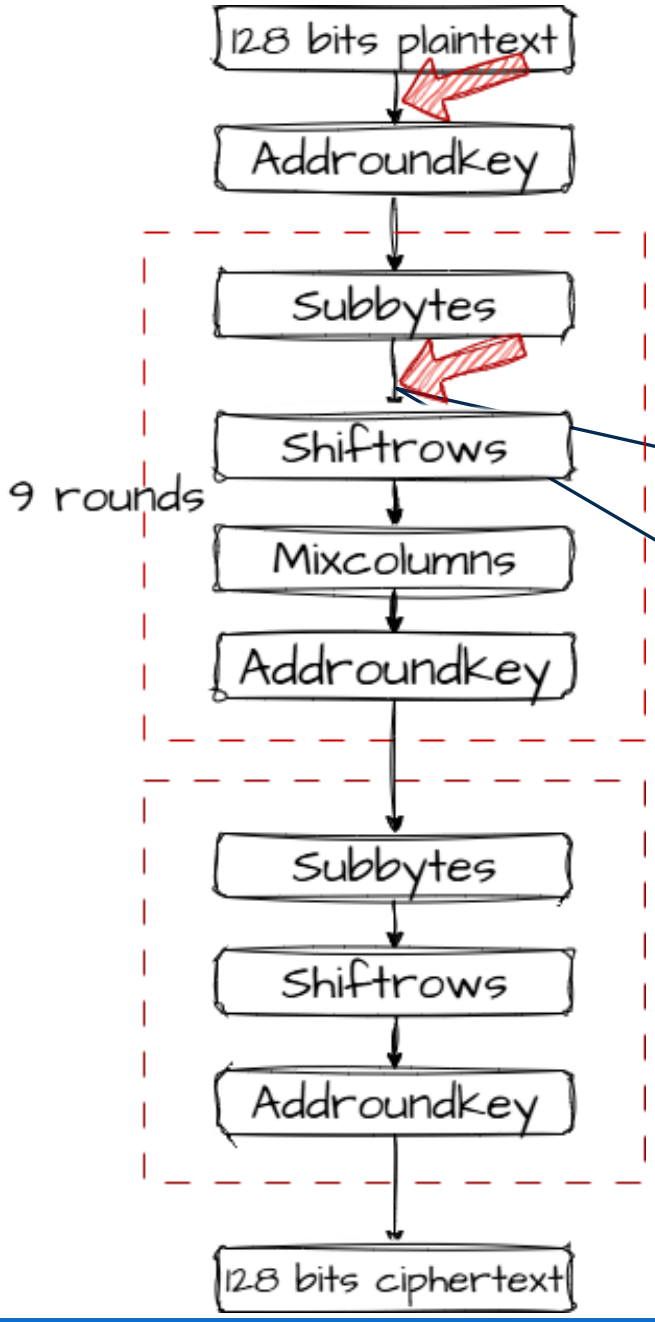


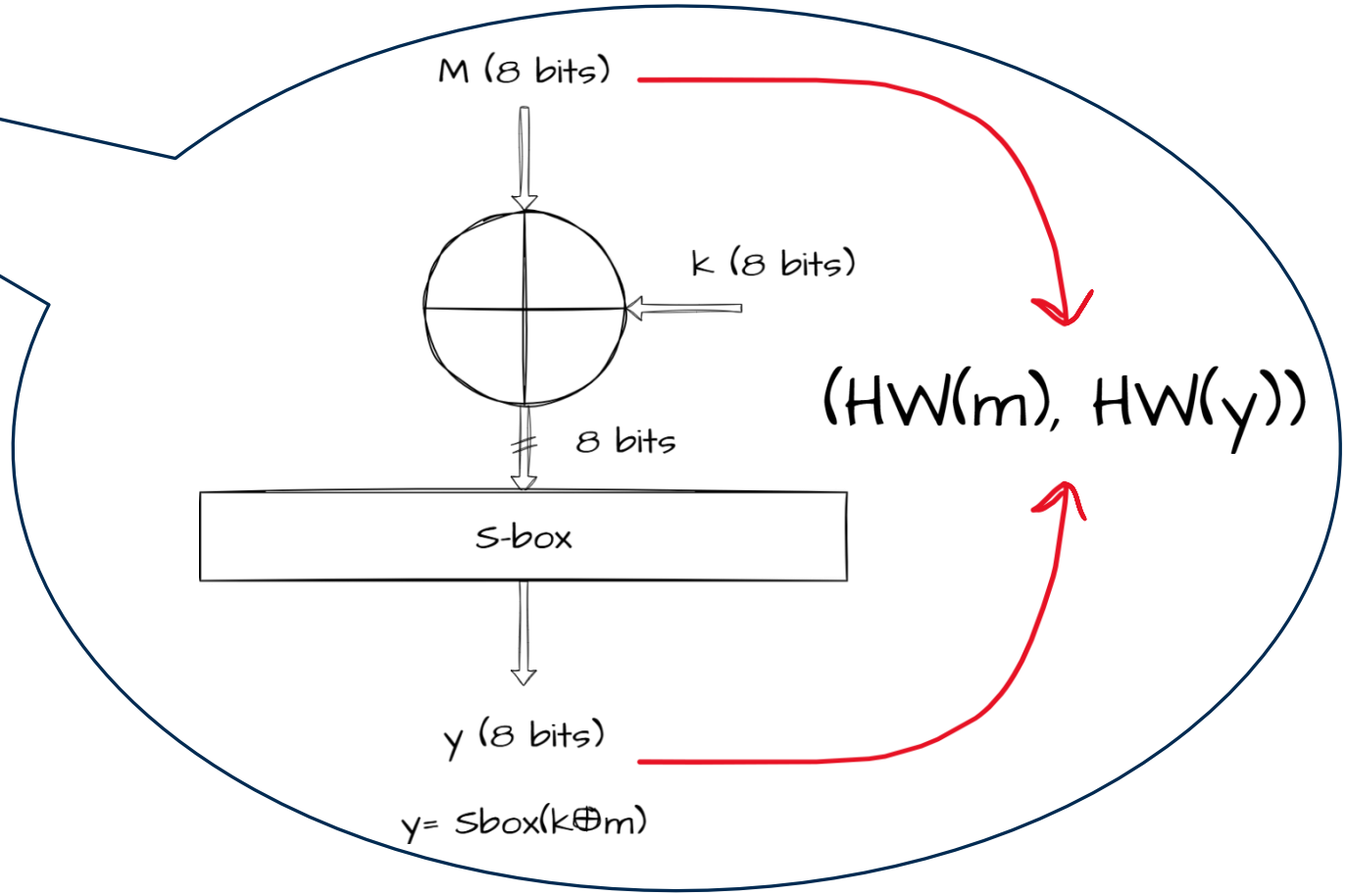
What is Blind-SCA

- Blind scenario
  - Plain/ciphertext is not accessible
  - Testing hypothesis about target variable
- The main challenges:
  - Guess possible value of intermediate data blindly



The Core Idea: Joint Distribution

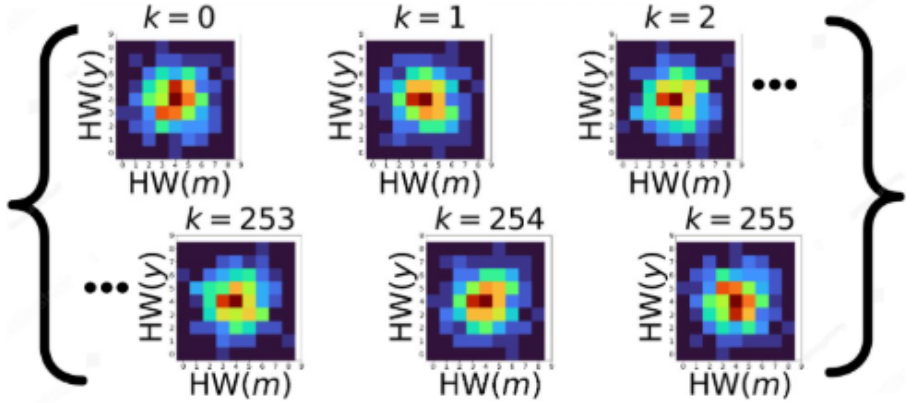




How Others Did It:

Phase I

Compute theoretical joint distribution.



Phase II

Obtain empirical distribution.

PoI Selection

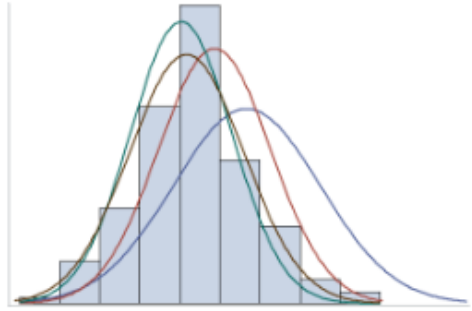
Label traces using a labeling option.

Labeling Options

Slicing VA

Phase III

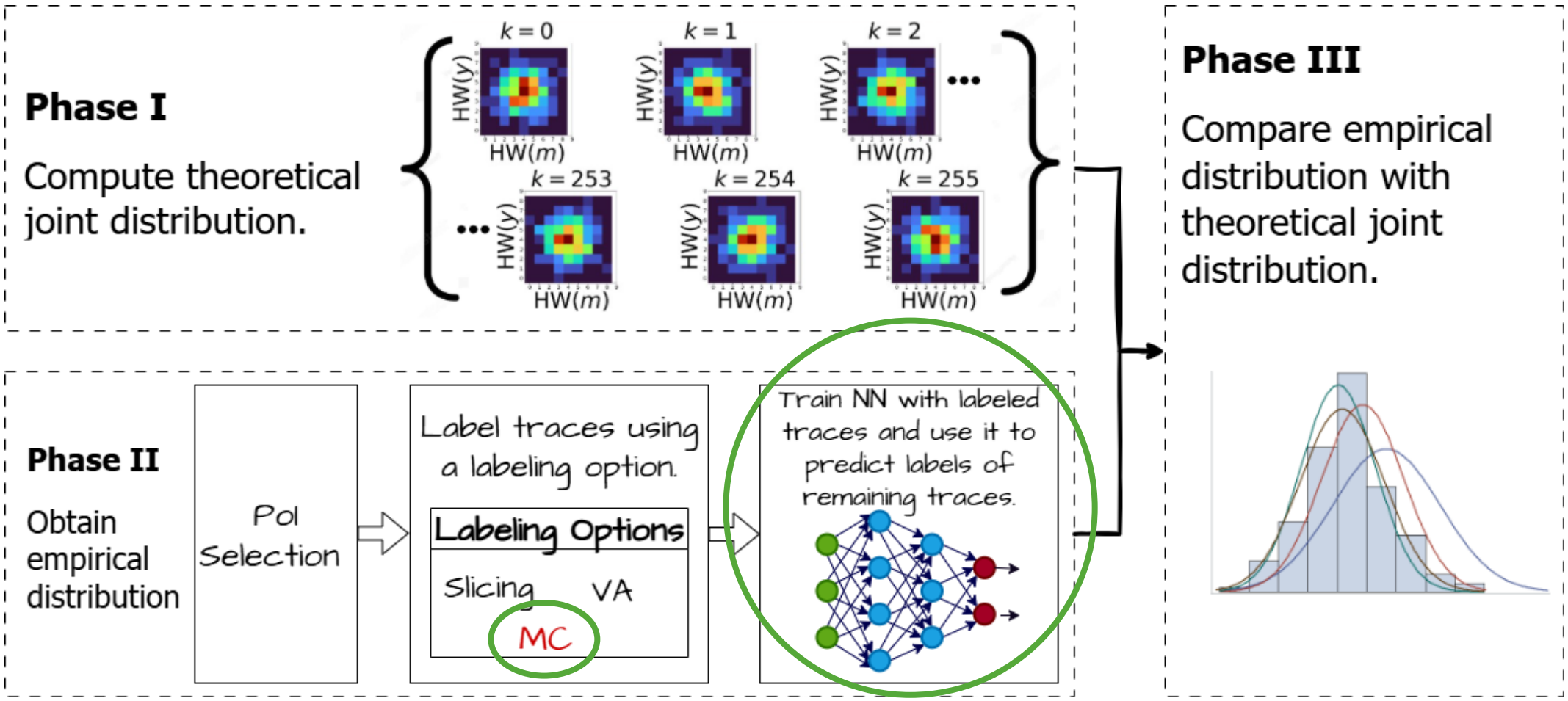
Compare empirical distribution with theoretical joint distribution.



What Are The Limitations?

- In practice it does not work
  - More complex microcontroller: ARM Cortex-M4
  - The real measurements are much more noisy
  - The targeted variables are much bigger and their HW can go to 16
- It requires precise knowledge of a single PoI

How We Do It:


Phase I
Compute theoretical joint distribution.
k = 0
k = 1
k = 2
k = 253
k = 254
k = 255
HW(y)
HW(m)
Phase II
Obtain empirical distribution
PoI Selection
Label traces using a labeling option.
Labeling Options
Slicing
VA
MC
Train NN with labeled traces and use it to predict labels of remaining traces.
Phase III
Compare empirical distribution with theoretical joint distribution.

Our contributions are:

- We formulate blind problem as Noisy label problem in Deep Learning (DL)
- We propose an alternative labeling technique: multipoint clustering
- We practically validated on 3 different algorithms, 4 different platforms and for first time, on protected implementation

How We Do It

Phase I

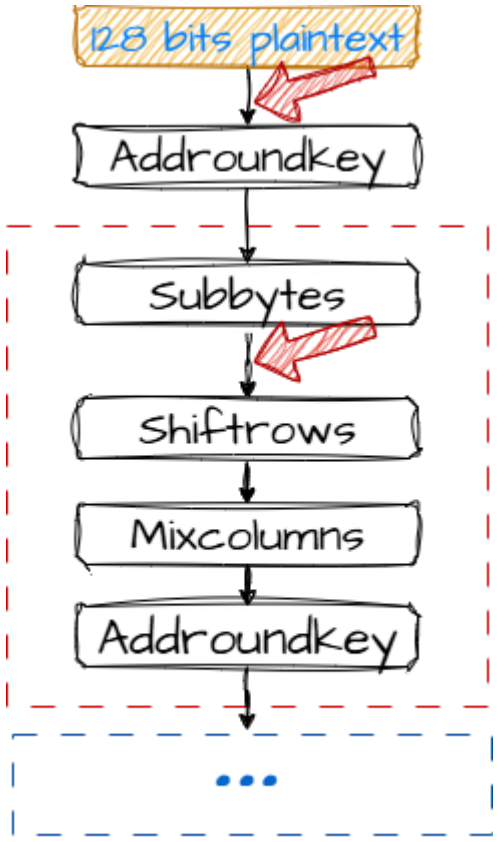
- Specify the random variables
- Calculate joint distribution of (HW(m), HW(y)) for all keys

Phase II

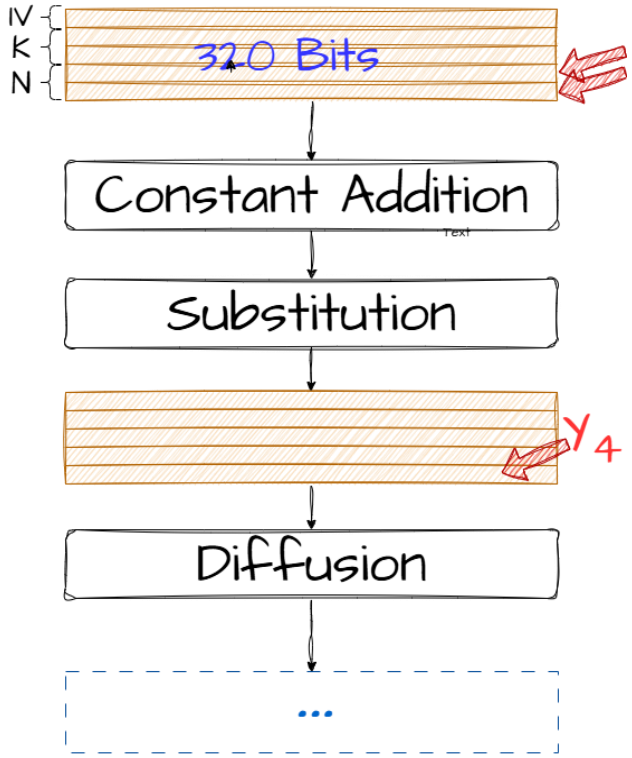
Phase III

Specify Attack Points

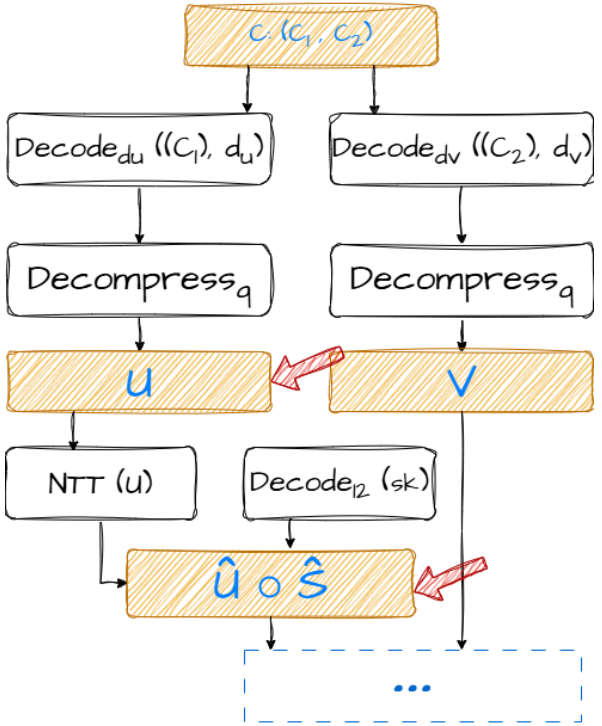
AES



Ascon

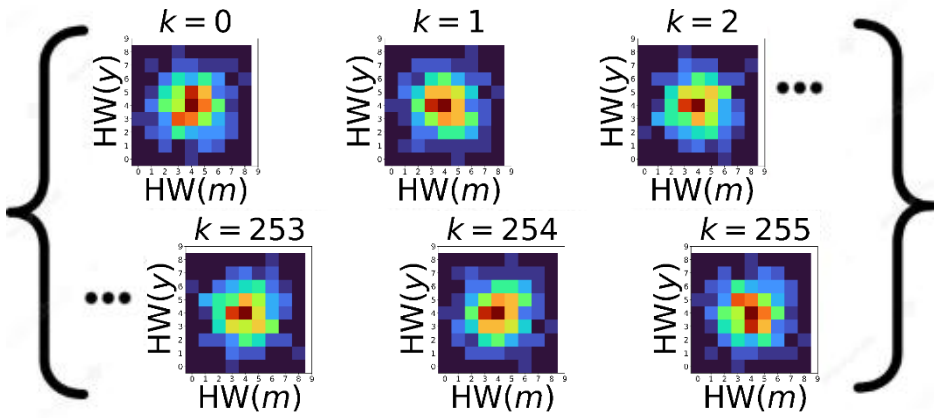


Kyber



Theoretical Joint Distributions

2^8 theoretical distributions = 256 Models



Algorithm 2 $(HW(m), HW(y))$ Joint Distribution Calculation for AES

1: **for** fixed k, $k \in \{0, \ldots, 255\}$ **do**
2: **for** each m, $m \in \{0, \ldots, 255\}$ **do**
3: Calculate $y = Sbox(k \oplus m)$
4: Calculate $HW(m)$ and $HW(y)$
5: Record occurrence of $(HW(m), HW(y))$ tuple
6: **end for**
7: Count the frequency of each tuple $(HW(m), HW(y))$
8: Divide by the total number of observations
9: Save values obtained in line 8 as expected theoretical joint distribution while using key k to be used later
10: **end for**

How We Do It

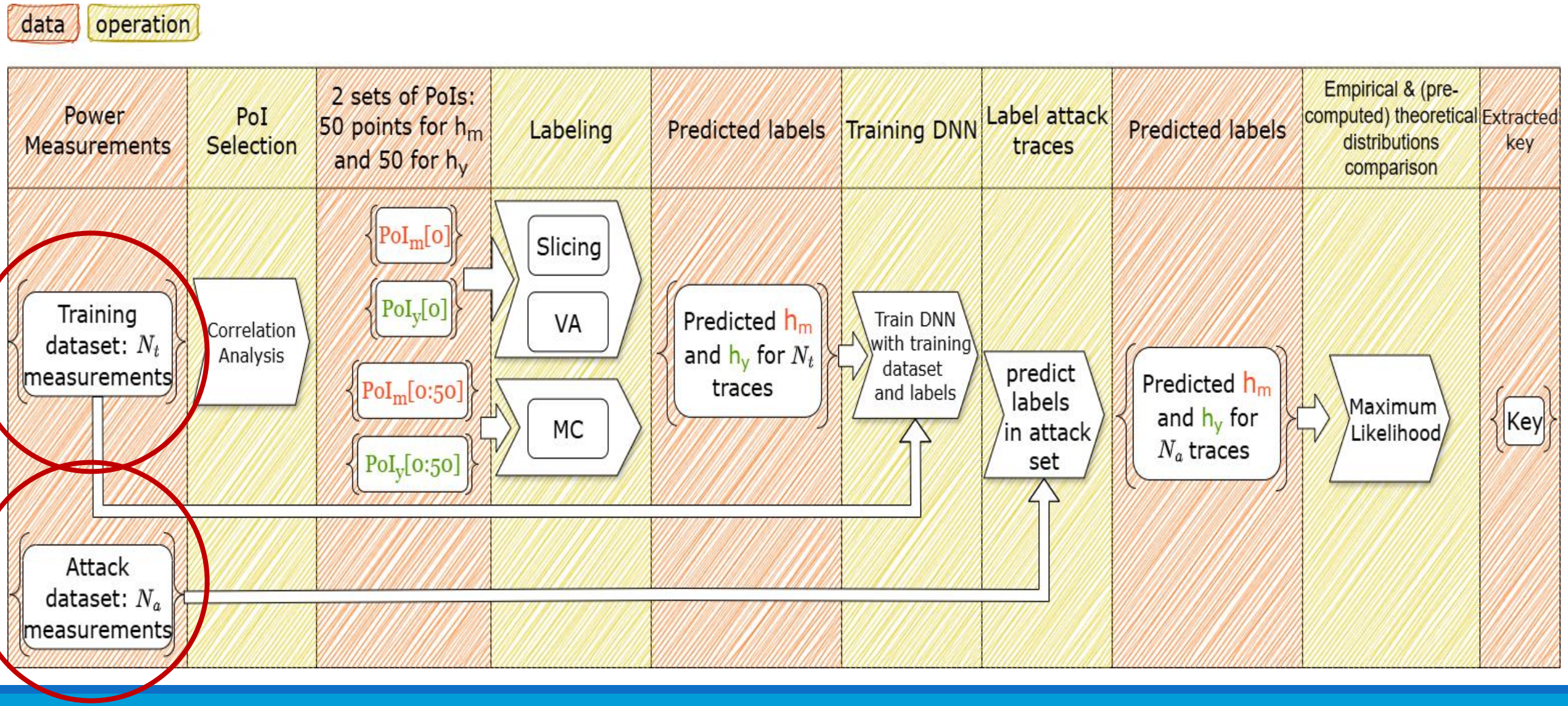
Phase I

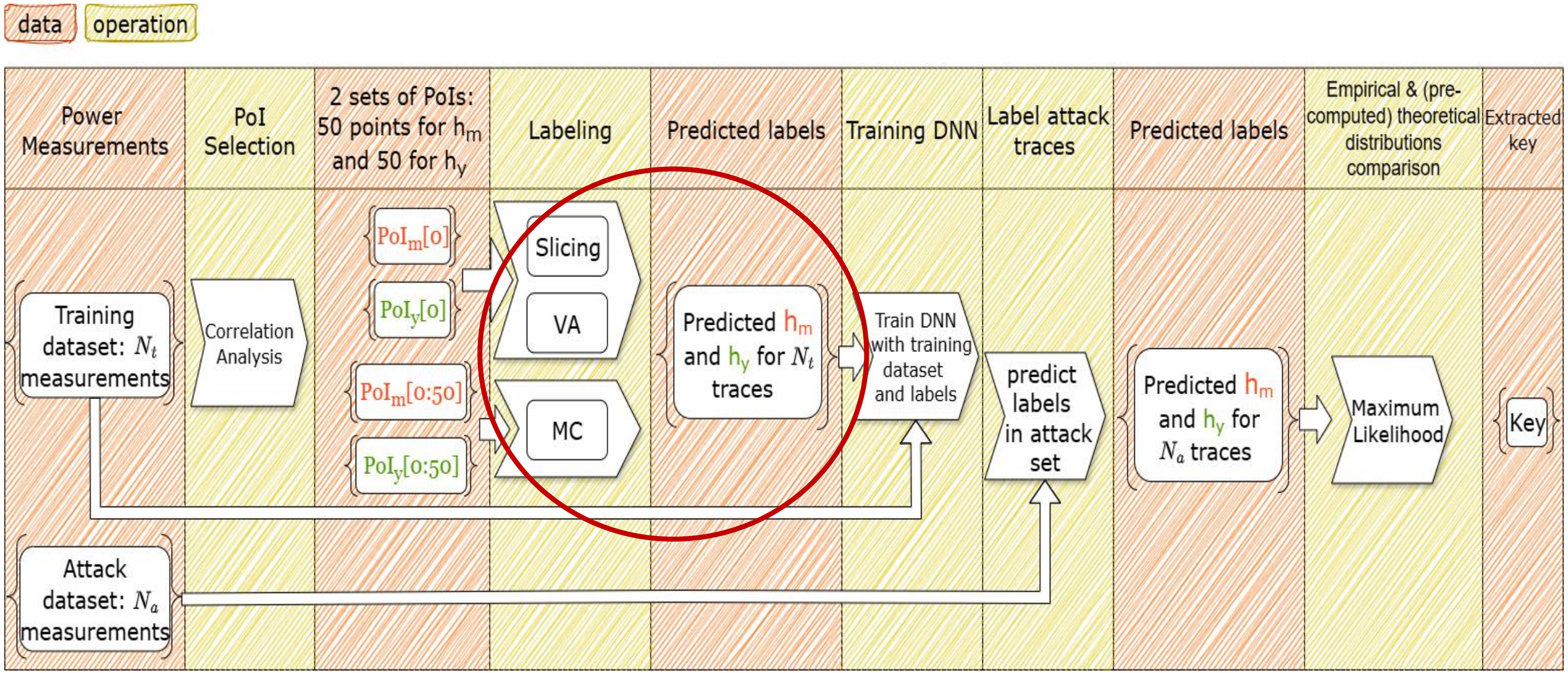
- Specify the attack point
- Calculate joint distribution of $(HW(u_0), HW(w_0))$ for all keys

Phase II

- Labeling a subset of the traces and train DNN with it
- Predict label of remaining traces using DNN to obtain empirical distribution

Phase III


data
operation
Power Measurements
PoI Selection
2 sets of PoIs: 50 points for h_m and 50 for h_y
Labeling
Predicted labels
Training DNN
Label attack traces
Predicted labels
Empirical & (pre-computed) theoretical distributions comparison
Extracted key
Training dataset: N_t measurements
Correlation Analysis
$PoI_m[0]$
$PoI_y[0]$
$PoI_m[0:50]$
$PoI_y[0:50]$
Slicing
VA
MC
Predicted h_m and h_y for N_t traces
Train DNN with training dataset and labels
predict labels in attack set
Predicted h_m and h_y for N_a traces
Maximum Likelihood
Key
Attack dataset: N_a measurements

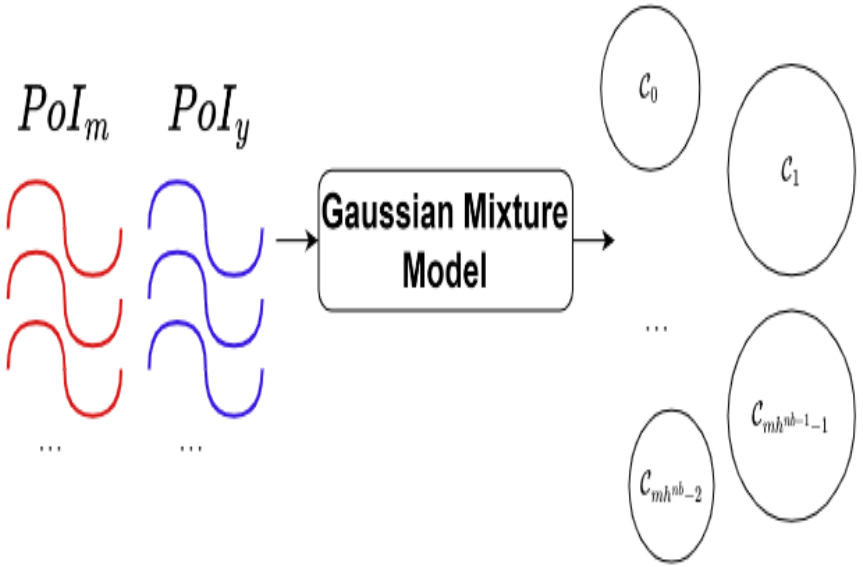

data
operation
Power Measurements
PoI Selection
2 sets of PoIs: 50 points for h_m and 50 for h_y
Labeling
Predicted labels
Training DNN
Label attack traces
Predicted labels
Empirical & (pre-computed) theoretical distributions comparison
Extracted key
Training dataset: N_t measurements
Correlation Analysis
$PoI_m[0]$
$PoI_y[0]$
$PoI_m[0:50]$
$PoI_y[0:50]$
Slicing
VA
MC
Predicted h_m and h_y for N_t traces
Train DNN with training dataset and labels
predict labels in attack set
Predicted h_m and h_y for N_a traces
Maximum Likelihood
Key
Attack dataset: N_a measurements

Why Clustering?

- Slicing and VA use a single PoI
- Clustering offers the possibility to use multiple PoIs
  - More information from the leakage
- Decide about the all the random variables at the same time
  - Capture the interaction

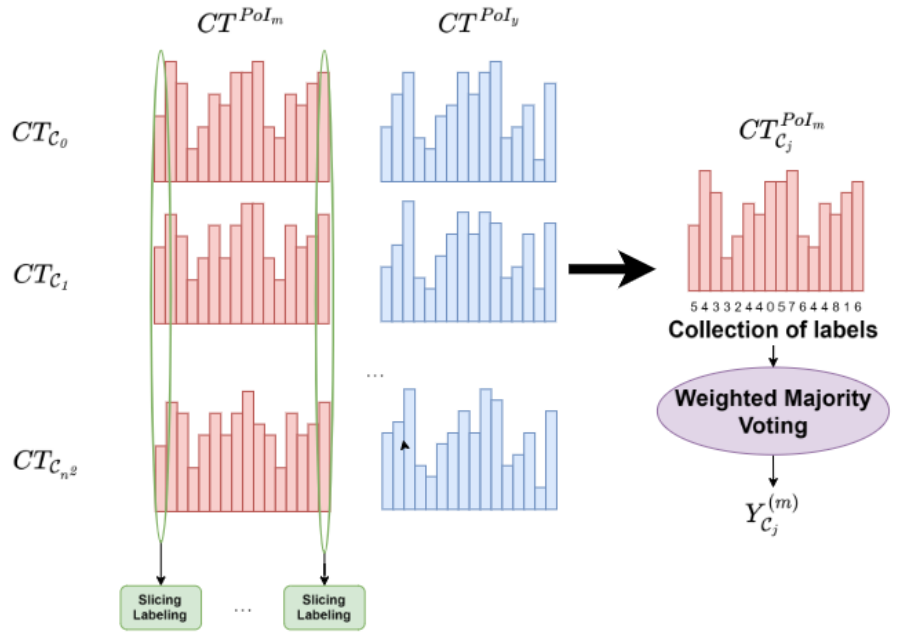
Clustering-Produce Clusters

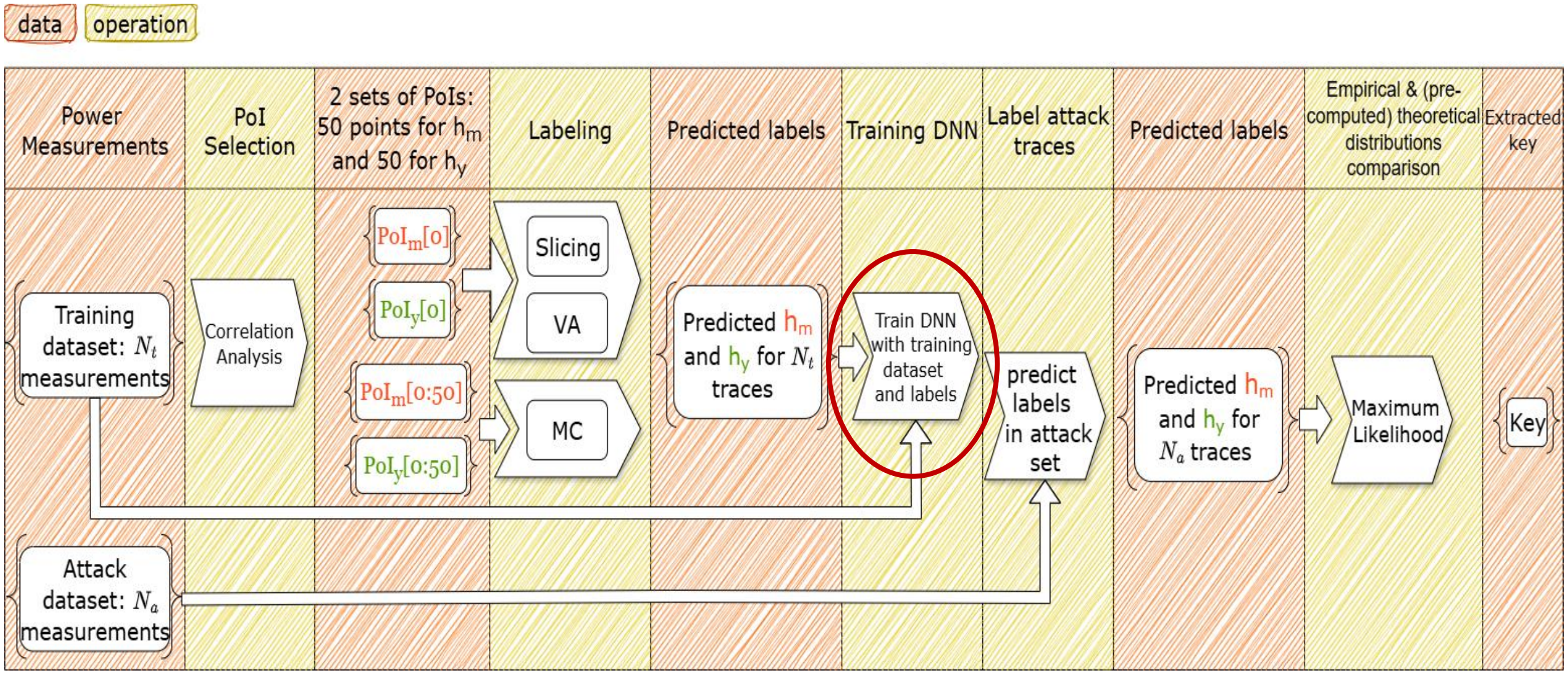
- 50 points of interest for each variable
- Apply Gaussian Mixture Model (GMM)
  - Assumes data is generated from mixture of Gaussian distributions
  - Yields better assumption about the underlaying distribution
  - Number of output clusters: $|h_m| * |h_y|$



Clustering-Provide Labels

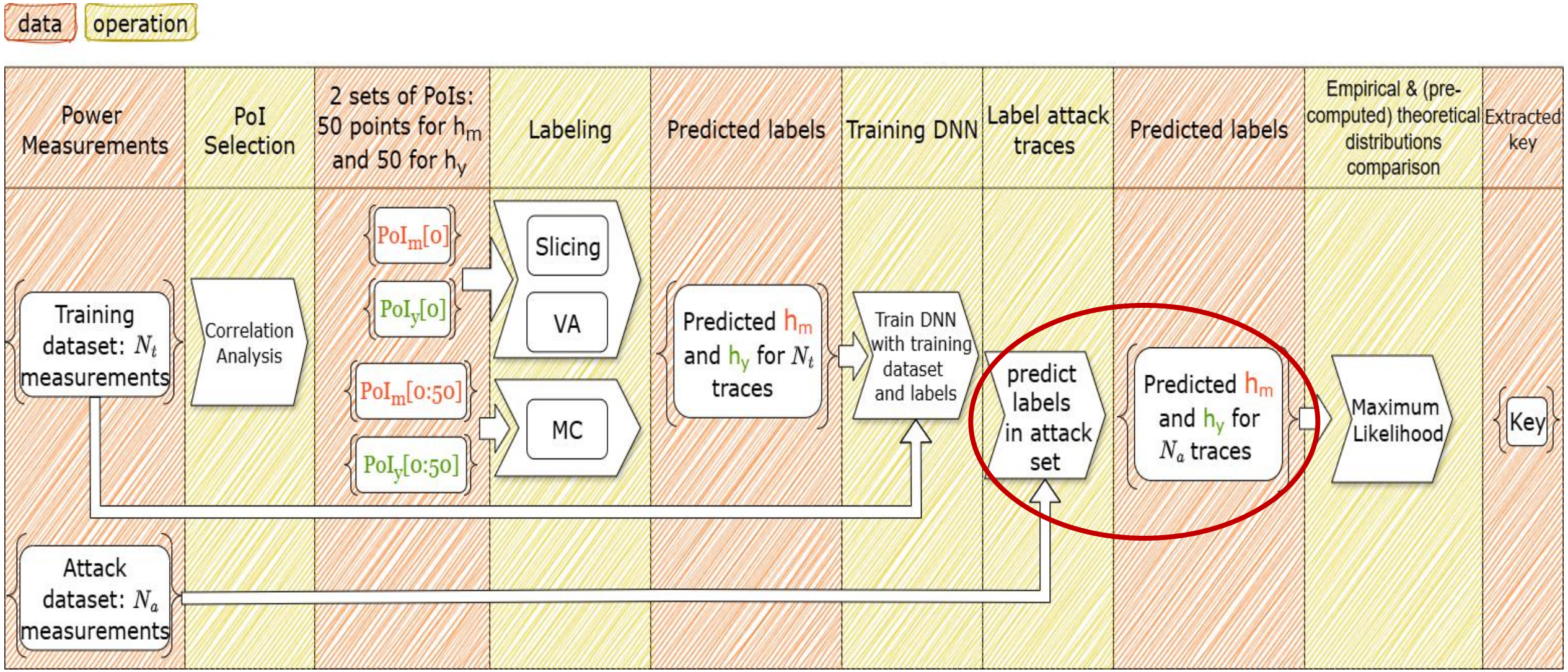
- Clustering puts similar example in a bucket
- It does not give us the labels
- We adopt slicing method to label center of clusters
- We leverage Weighted Majority Voting since we have multiple PoIs




data
operation
Power Measurements
PoI Selection
2 sets of PoIs: 50 points for h_m and 50 for h_y
Labeling
Predicted labels
Training DNN
Label attack traces
Predicted labels
Empirical & (pre-computed) theoretical distributions comparison
Extracted key
Training dataset: N_t measurements
Correlation Analysis
$PoI_m[0]$
$PoI_y[0]$
$PoI_m[0:50]$
$PoI_y[0:50]$
Slicing
VA
MC
Predicted h_m and h_y for N_t traces
Train DNN with training dataset and labels
predict labels in attack set
Predicted h_m and h_y for N_a traces
Maximum Likelihood
Key
Attack dataset: N_a measurements

Why DNN?

- Using DNN we can formulate the blind-SCA as a noisy labels problem for deep neural networks
- Advantages of using DL:
  - Allows profiling scenarios using estimated label from clustering
  - Captures the input-output relation for HW pairs better


data
operation
Power Measurements
PoI Selection
2 sets of PoIs: 50 points for h_m and 50 for h_y
Labeling
Predicted labels
Training DNN
Label attack traces
Predicted labels
Empirical & (pre-computed) theoretical distributions comparison
Extracted key
Training dataset: N_t measurements
Correlation Analysis
$PoI_m[0]$
$PoI_y[0]$
$PoI_m[0:50]$
$PoI_y[0:50]$
Slicing
VA
MC
Predicted h_m and h_y for N_t traces
Train DNN with training dataset and labels
predict labels in attack set
Predicted h_m and h_y for N_a traces
Maximum Likelihood
Key
Attack dataset: N_a measurements

How We Do It

Phase I

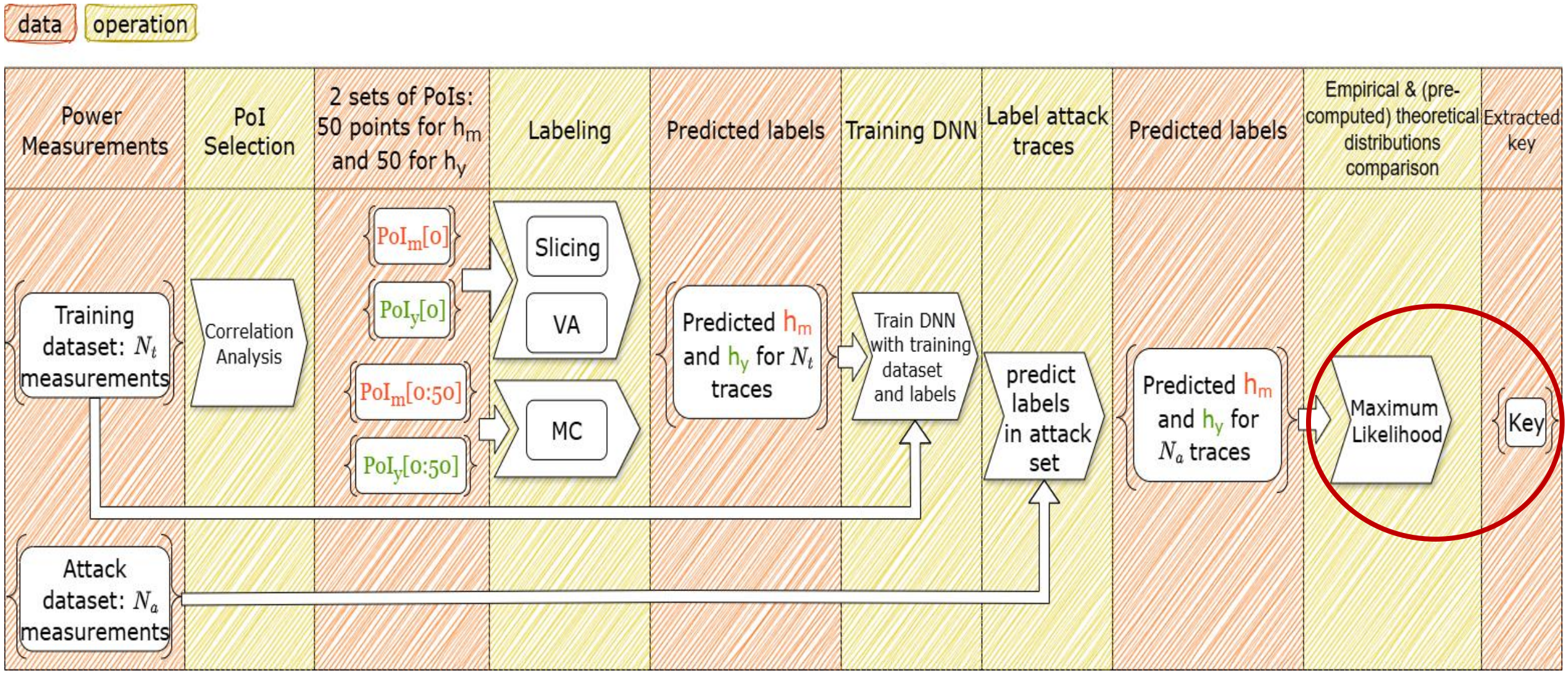
- Specify the attack point
- Calculate joint distribution of $(HW(u_0), HW(w_0))$ for all keys

Phase II

- Labeling a subset of the traces and train DNN with it
- Predict label of remaining traces using DNN to obtain empirical distribution

Phase III

- Using Maximum Likelihood

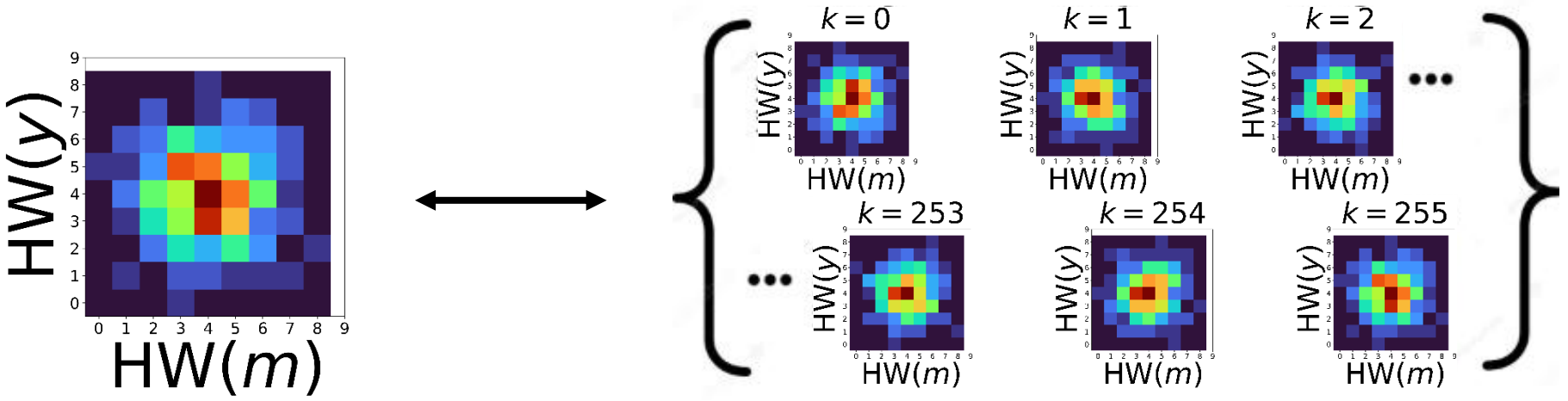

data
operation
Power Measurements
PoI Selection
2 sets of PoIs: 50 points for h_m and 50 for h_y
Labeling
Predicted labels
Training DNN
Label attack traces
Predicted labels
Empirical & (pre-computed) theoretical distributions comparison
Extracted key
Training dataset: N_t measurements
Correlation Analysis
$PoI_m[0]$
$PoI_y[0]$
$PoI_m[0:50]$
$PoI_y[0:50]$
Slicing
VA
MC
Predicted h_m and h_y for N_t traces
Train DNN with training dataset and labels
predict labels in attack set
Predicted h_m and h_y for N_a traces
Maximum Likelihood
Key
Attack dataset: N_a measurements

Using Maximum Likelihood

- Maximum Likelihood criterion is then used to compare empirical distribution with theoretical distribution (histograms)

K =?

$$\left(\frac{1}{\sigma_u\sqrt{2\pi}}e^{\frac{-1}{2}(\frac{p_u-h_u^*}{\sigma_u})^2}\right).\left(\frac{1}{\sigma_w\sqrt{2\pi}}e^{\frac{-1}{2}(\frac{p_w-h_w^*}{\sigma_w})^2}\right)$$



Experimental Results

Target	CW	CW (desync)	Kyber	ASCON
Device	8-bit AVR XMEGA	8-bit AVR XMEGA	32-bit STM32F3	32-bit STM32F3
Linge et al. [2]	✗	✗	✗	✗
Clavier & Reynaud [1]	✗	✗	✗	✗
MLP + MC	✗	✓	✓	✓
MLP + MC + Dropout	✓	✓	✗	✓
CNN + MC	✓	✓	✓	✓
CNN + MC + Dropout	✓	✓	✓	✓

Tab. 1: Comparison of proposed results on different datasets/devices with prior works. We highlight successful attacks in ✓ and failed attacks in ✗.

Thank you!