Machine Learning-based Side-Channel Analysis of Cryptographic Chips

By

Naila Mukhtar

A thesis submitted to Macquarie University

for the degree of Doctor of Philosophy

School of Engineering

March 2021

MACQUARIE
University
SYDNEY · AUSTRALIA

Statement of Candidate

I certify that the work in this thesis "Machine Learning-based Side-Channel Analysis of Cryptographic Chips" has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree to any other university or institution other than Macquarie University.
I also certify that the thesis is an original piece of research and it has been written by me.
In addition, I certify that all information sources and literature used are indicated in the thesis.

Naila Mukhtar

Acknowledgements

My PhD has been a phenomenal and a great learning experience, both academically and personally. I firmly believe my journey would not have reached its culmination without the support and contribution of many extraordinary individuals. First of all, I would like to thank my supervisor Dr. Yinan Kong for his kind mentoring, supervision, and support throughout my PhD. I would like to acknowledge Macquarie University for awarding the Macquarie University Research Excellence Scholarship (MQRES) to pursue my PhD and Post Graduate Research Fund (PGRF) for the international research visits and conferences. I wish to thank friendly administrative staff at Macquarie University School of Engineering, who facilitated my research with utmost professionalism.

I would also like to thank Prof. Lejla Batina (Radboud University Netherlands) and Prof. Ashiq Anjum at University of Leicester as it was their kind guidance, patience and invaluable feedback which steered me through this research.

A heartfelt thanks to my collaborators as their work allowed me to establish the foundation of my research work especially Dr. Louiza Papachristodoulou, Dr. Tariq Khan, Dr. Mohamad A. Mehrabi, Dr. Apostolos P. Fournaris, and Dr. Stejpan Picek. Many thanks to my colleagues in VLSI group who were always so helpful.

My PhD journey would not have been comfortable and enjoyable without the love and support of my friends Sobia, Uzma, Amara and Zainab.

I owe my gratitude to my whole family; my loving parents, siblings and my in-laws for their support and understanding. I am so blessed to have their endless love and unceasing support.

Last but not the least, I would like to thank my husband, Naeem Akhtar, who encouraged me to live my life-long dream and pursue PhD. This PhD would not have been possible without his endless unconditional support and encouragement. A very special thanks to my little supportive son who was just one year old when I started my PhD journey.

Abstract

Advances in communication and network systems have given rise to a new era of revolutionary technology known as Internet of Things (IoT): a world where everything is connected through uniquely identifiable intercommunicating smart devices using uninterrupted connectivity and sensors. IoT brings convenience to human lives in almost all sectors, including healthcare centers, smart homes, and traffic management. The smart devices deployed in IoT systems consist of resource constraint embedded chips for processing private information. The security of the information processing on these chips has increased with the introduction of Edge Artificial Intelligence where information will be processed locally on the edge devices instead of the cloud.

Generally, on cryptographic chips, private information is secured using the standard mathematically strong cryptographic algorithms but the weak implementations of these algorithms can lead to side-channel leakages, which can be exploited to retrieve the secret information, hence endangering the user's privacy and data. Moreover, the efficiency of these attacks has greatly improved due to the introduction of machine learning techniques which has even weaken the countermeasure-protected implementations. Despite being effective, such machine learning-based attacks have their own challenges like over-fitting due to redundant features , requirement of huge datasets for learning the leakage patterns, or being prone to produce inaccurate analysis because of low instance-feature ratio or imbalance classes. Moreover, these attacks are generally analyzed for symmetric ciphers, while asymmetric ciphers remain unaddressed.

The aim of this dissertation is to propose the improved machine learning-based side-channel attacks using evolving machine learning technologies which can aid in recovering the sensitive information efficiently. To achieve this, we have designed and developed two novel hand-crafted feature engineering techniques to eliminate the redundant features, hybrid deep learning-based scheme for data with low instance-feature ratio, neural network-based model for various side-channel attack models, and a Generative Adversarial Network-based model to increase the dataset size by generating fake leakages. Moreover, a generalized few-shot learning-based leakage assessment model is proposed which combines the leakages from the multiple sources and channels to detect and differentiate the secret information in leakages. Furthermore hyperparameter tuning is performed to select the best models.

The comparison of the proposed attack models/schemes is performed with the state-of-the-art side-channel attacks and with the other machine learning techniques. To evaluate the efficiency of the developed models, experiments are conducted on the protected and unprotected algorithms' implementations leakages of both symmetric (AES) and asymmetric (ECC) ciphers on various cryptographic chips. The results demonstrate that the proposed methods outperform the state-of-the-art side-channel template attacks and can recover the sensitive information with better efficiency. It is concluded that machine learning-based side-channel attacks pose a significant threat to the security of the cryptographic chips. Based on the findings, we suggest that new countermeasures should be designed which are effective against these advance attacks to secure user information.

Contents

List of Symbols

AES	Advanced Encryption Standard
AI	Artificial Intelligence
AUC	Area Under Curve
BEC	Binary Edwards curves
Chi-Sq	Chi-Square
CGAN	Conditional Generative Adversarial Network
CT	Cipher Text
CCN	Convolutional Neural Network
DEMA	Differential Electromagnetic Analysis
DES	Data Encryption Standard
DNN	Deep Neural Network
DLP	Discrete Logarithm Problem
DPA	Differential Power Analysis
ECC	Elliptic-curve Cryptography
EM	Electromagnetic Emanations
EMA	Electromagnetic Analysis
ECSM	Elliptic-Curve Scalar Multiplication
FA	Fault-Injection Attacks
FPGA	Field-Programmable Gate Arrays
FN	False Negatives
FPR	False Positive Rate
GB	Giga Bytes
GHz	GigaHertz
IoT	Internet of Things

LDA	Linear Discriminative Analysis
LSTM	Long Short-Term Memory
ML	Machine Learning
NB	Naive Bayes
MLP	Multilayer Perceptron
NIST	National Institute of Standards and Technology
GAN	Generative Adversarial Network
TP	True Positives Rate
RF	Random Forest
ROC	Receiver Operating Characteristic
RNN	Recurrent Neural Network
RNS	Residue Number System
RSA	Rivest–Shamir–Adleman PKC Algorithm
PA	Power Analysis
PCA	Principal-Component Analysis
PT	Plain Text
PKC	Public-Key Cryptography
SCA	Side-Channel Analysis
SM	Scalar Multiplication
SPA	Simple Power Analysis
SVM	Support Vector Machines
TPR	True Positives Rate
WSN	Wireless Sensor Network

List of Figures

List of Tables

List of Publications

Discussed in Thesis

This thesis is based on the following original publications which are either published, submitted or to be submitted to the journals or conferences. The publications are referred in text in Roman numbers.

I. **Naila Mukhtar**, Yinan Kong, "Secret key classification based on electromagnetic analysis and feature extraction using machine-learning approach", Future network systems and security: 4th International Conference, FNSS 2018 Paris, France, 2018.

II. **Naila Mukhtar**, Yinan Kong, "Hyper-parameter optimization for machine-learning based electromagnetic side-channel analysis", 2018 IEEE ICSENG proceedings. Piscataway, NJ, Institute of Electrical and Electronics Engineers (IEEE), pp. 1-7, 2018.

III. **Naila Mukhtar**, Mohamad A. Mehrabi, Yinan Kong, Ashiq Anjum, "Machine-learning-based side-channel evaluation of elliptic-curve cryptographic FPGA processor", In Applied Sciences (Switzerland), MDPI, Vol. 9, No. 1. pp. 1-20, 2019.

IV. **Naila Mukhtar**, Mohamad A. Mehrabi, Yinan Kong, Ashiq Anjum, "Electromagnetic Side-Channel Attack Analytics on Elliptic Curve Cryptography using Machine Learning in IoT Devices", IEEE Access, 2020. (Accepted)

V. **Naila Mukhtar**, Louiza Papachristodoulou, Apostolos P. Fournaris, Lejla Batina, Yinan Kong, "Machine-Learning assisted Side-Channel Attacks on RNS-based Elliptic

Curve Implementations using Hybrid Feature Engineering", IACR Eurocrypt, 2021. (Submitted)

VI. **Naila Mukhtar**, Apostolos P. Fournaris, Tariq M. Khan, Charis Dimopoulos, Yinan Kong, "Improved Hybrid Approach for Side-Channel Analysis using Efficient Convolutional Neural Network and Dimensionality Reduction", in IEEE Access, vol. 8, pp. 184298-184311, 2020, doi: 10.1109/ACCESS.2020.3029206.

VII. **Naila Mukhtar**, Lejla Batina, Yinan Kong, "Data Augmentation using Generative Adversarial Networks - Synthesizing Realistic Leakage Signals", 2020. (Under Preparation)

VIII. **Naila Mukhtar**, Stejpan Picek, Yinan Kong, "Hybrid Machine-learning based Side-channel attacks using few shot Learning", 2020. (Under Preparation)

Other Publications

Below is the list of publications which are related to this thesis but are not discussed in this thesis.

- **Naila Mukhtar**, Yinan Kong, "On features suitable for power analysis - Filtering the contributing features for symmetric key recovery", 6th International Symposium on Digital Forensic and Security, Institute of Electrical and Electronics Engineers (IEEE), pp. 265-270, 2018.

Oral and Poster Presentations

- Presented virtually at 6th International Symposium on Digital Forensic and Security (ISDFS), IEEE, 2018, "On features suitable for power analysis - Filtering the contributing features for symmetric key recovery", Antalya Turkey.

- Presented at 4th International Conference on Future network systems and security (FNSS), 2018, "Secret key classification based on electromagnetic analysis and feature extraction using machine-learning approach", Paris, France.

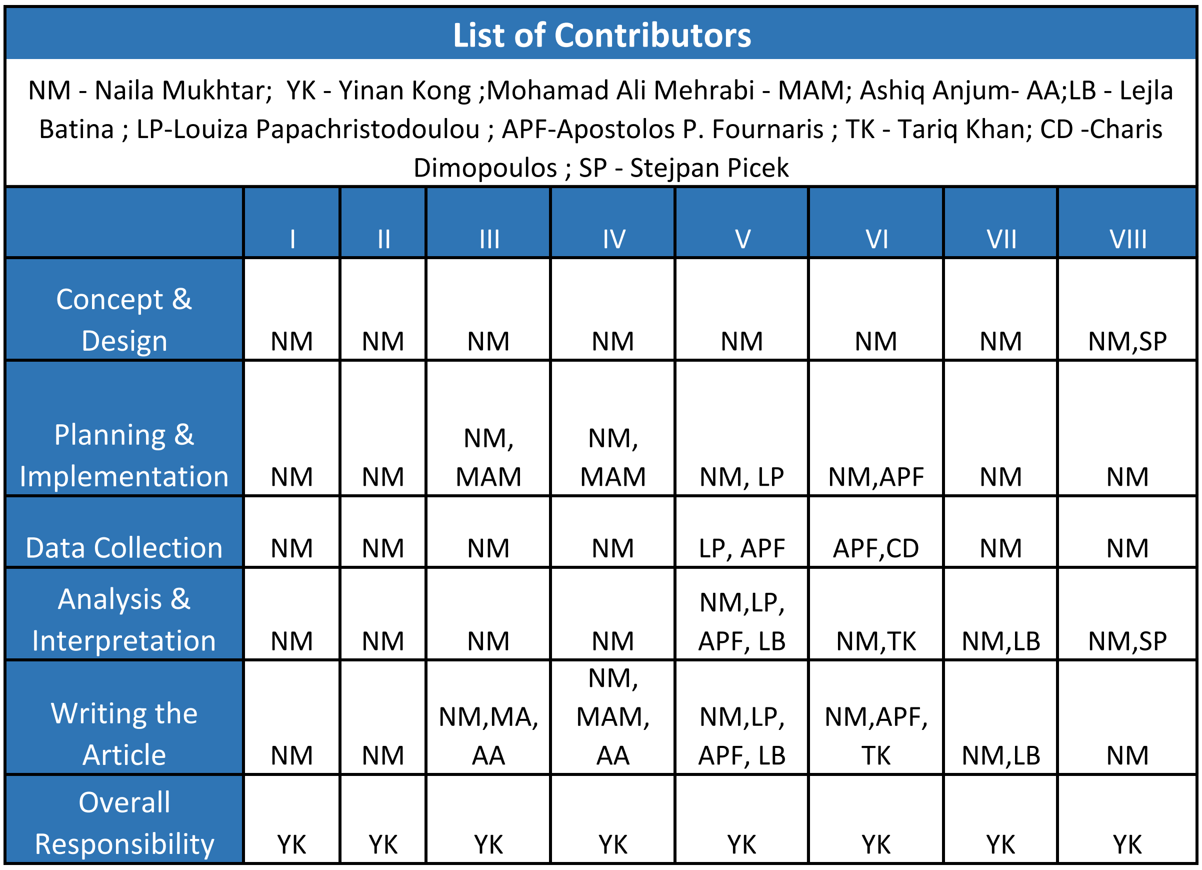
- Presented at ICSENG, IEEE, 2018, "Hyper-parameter optimization for machine-learning based electromagnetic side-channel analysis", Sydney, Australia.
- Presented at Workshop on Authentication for the Future Internet Of Things, 2018, "Side-Channel Attacks and Machine-Learning", Melbourne, Australia.
- Presented at Macquarie University School of Engineering Higher Degree Research Conference, 2018, "Side-Channel Attacks on IoT devices", Sydney, Australia.
- Presented at Macquarie University School of Engineering Higher Degree Research Conference, 2019, "Machine Learning-based Side-Channel Attacks on Cryptographic Chips", Sydney, Australia.

Awards received as a PhD student

- Macquarie University Research Excellence Scholarship (MQRES) holder, 2016-2019.
- Macquarie University Post Graduate Research Fund (PGRF), 2017 (awarded \$ 5000 for international conference travel).
- Macquarie University School of Engineering Research Fund, 2018 awarded \$ 1500 for local conference/workshop travel).
- Best Poster Presentation Award, Macquarie University School of Engineering Higher Degree Research Conference, 2019.
- Awarded IEEE WIE Inspiring Student member of the year 2019.

List of Contributors

The major machine learning-based investigation, implementation, analysis, interpretation, and paper drafting has been done by myself (Naila Mukhtar) with guidance from my supervisor Dr. Yinan Kong. I have worked on the ECC implementation by my colleague Mohamad A. Mehrabi with guidance of Dr. Yinan Kong and Prof. Ashiq Anjum. I have worked on datasets of secure algorithm implementations shared by research groups at Radboud University, Netherland and University of Patras, Greece. I have worked in collaboration with other research groups as well. The contributions from each author are explicitly listed below.



List of Contributors

NM - Naila Mukhtar; YK - Yinan Kong ;Mohamad Ali Mehrabi - MAM; Ashiq Anjum- AA;LB - Lejla Batina ; LP-Louiza Papachristodoulou ; APF-Apostolos P. Fournaris ; TK - Tariq Khan; CD -Charis Dimopoulos ; SP - Stejpan Picek

	I	II	III	IV	V	VI	VII	VIII
Concept & Design	NM	NM	NM	NM	NM	NM	NM	NM,SP
Planning & Implementation	NM	NM	NM, MAM	NM, MAM	NM, LP	NM,APF	NM	NM
Data Collection	NM	NM	NM	NM	LP, APF	APF,CD	NM	NM
Analysis & Interpretation	NM	NM	NM	NM	NM,LP, APF, LB	NM,TK	NM,LB	NM,SP
Writing the Article	NM	NM	NM,MA, AA	NM, MAM, AA	NM,LP, APF, LB	NM,APF, TK	NM,LB	NM
Overall Responsibility	YK	YK	YK	YK	YK	YK	YK	YK

Chapter 1

Introduction

1.1 Overview

Developments in smart devices and the advent of ever-ready uninterrupted connectivity have changed our world drastically. The new era, known as the Internet of Things (IoT) has revolutionized our communities. Our communities, small or big, communicating locally or globally, are interconnected in the form of a giant cluster through millions of devices over the internet. In this ever-growing community connectivity through IoT, the need for security has increased significantly. Every component in IoT requires security and protection at some layers, including networks, hardware chips, data/information, processes, protocols, and physical infrastructure. The new concept of Edge Artificial Intelligence (AI) introduced into the chip technology industry has further heightened the security requirements [1, 2]. In Edge AI, the AI algorithm based processing is carried out on the end device itself instead of sending data to the cloud for processing. Recently, Field Programmable Gate Arrays (FPGA) and ARM manufacturing companies have joined the league and started designing chips like Cortex-M55 equipped with the Ethos-U55, 'microNPU'. This is a new class of machine learning (ML) processor specifically designed to accelerate ML inference in area-constrained embedded and IoT devices.

A secure set of algorithms, as recommended by the National Institute of Standards and Technology (NIST) and other local standardizing authorities, serves to protect information processing through the hardware chips, including FPGA, and ARM processors. These mathematically and theoretically secure algorithms promise a reliable infrastructure for an IoT environment, but during the development phase, several factors, including hardware systems constraints, can lead to implementation compromises. These weak implementations may produce unintended side-channel leakages that an adversary can exploit to analyze the patterns and determine the sensitive information or key. These side channels include power consumption, electromagnetic emissions, timing information, vibrations, or sound produced by the system [3–5].

Traditionally, various classical side-channel attacks have proven to be successful in compromising mathematically secure algorithm implementations [6–8]. Some of the most powerful side-channel attacks like profiled-based side-channel attacks, in which an adversary is assumed to have access to the open copy of the target device, were able to break standard industry grade encrypted systems. In the past, various countermeasures have been proposed including masking and hiding to safeguard the sensitive parameters [9–16]. The profiled-based attacks have recently been extended to the machine learning-based side-channel attacks (ML-SCA) due to the overlapping nature of both the attacks [17–22].

With the advances being made in machine learning and upcoming Quantum Computing on the horizon, these machine learning-based side-channel attacks will become more accurate, speedy, and effective, even defeating the existing countermeasures. However, there are few factors which may hinder successful ML-SCA in recovering the sensitive information. These factors include; the requirement of huge datasets for the efficient performance of deep learning algorithms, noisy traces with redundant features, poorly trained models, high-dimensional data due to high sampling frequency during the acquisition process, and small datasets having low instance-feature ratio. With evolving machine learning technologies, these challenging factors can be addressed to improve machine learning-based side-channel attacks' performance and efficiency. This thesis focuses on improving the efficiency of side-channel attacks by investigating, analyzing, and proposing novel models and methods based on the emerging machine learning techniques.

1.2 Challenges and Objectives

It should be noted that while modeling the side-channel problem with a machine learning-based algorithm, redundant irrelevant features can be a limiting factor in terms of accurate analysis. This problem can be solved by selecting the most contributing features through hand-crafted or state-of-the-art feature engineering techniques. Basic machine learning algorithms, along with feature engineering techniques, can offer simple efficient key recovery methods.

With the recent advances made in machine learning, data scientists and cryptographers have developed better side-channel attack methodologies that can successfully recover the secret key information without requiring any pre-processing or selection of point of interest [23]. However, for the success of such deep-learning attacks, huge datasets are required to learn the leakage pattern from the noisy high-dimensional data traces, which results in high processing time, more memory requirement, and increased time needed to acquire more data traces to form a huge dataset. Moreover, noisy data may give rise to over-fitting or curse of dimensionality, where the model trains itself so well on the training data that it fails to generalize on the unseen data.

All the above-mentioned factors reveal the limitations to the practicality of the attacks. Hence, the efficiency of these (ML-SCA) attacks is still debatable. This discussion points to the following research question; "Machine learning analysis has remarkably improved performance and efficiency in other research domains due to the proposed advanced techniques. Is it possible that an adversary can exploit the existing machine learning techniques to design an efficient attack that can learn the patterns from the noisy leakages and help in retrieving the secret information in less time by consuming fewer resources?"

This thesis aims to explore this research question further and investigate methods based on machine learning and feature engineering techniques to propose efficient attacks on the secure implementations of cryptographic primitives on various embedded chips. The proposed improved attacks will help the research community to devise effective countermeasures.

According to the No-Free-Lunch theorem, a machine learning algorithm will provide varied results for the different datasets. Keeping this in view, this thesis also aims to provide an evaluation of the proposed attacks on a variety of standard protected (secured) and unprotected cryptographic algorithm implementations; Advanced Encryption Standard (AES) and Elliptic Curve Cryptography (ECC). In this research, two side channels, power consumption and electromagnetic emanations, are exploited.

One of the most significant challenges in SCA is benchmarking. Due to the lack of availability of the leakage datasets, it becomes hard to compare and reproduce analysis results. At the start of this research work, there were no standard repositories for the leakage traces datasets of the standard algorithms except AES (DPAContest data [24]). For this research, we have also implemented an automated setup for leakage data collection. Moreover, over past two years, few datasets have been presented by the research community, so we have also evaluated our proposed methodologies for those datasets.

Based on the aforementioned challenges, the aims and objectives are explicitly listed below.

Aim 1

To investigate, propose and formulate generalized machine-learning based methodologies and feature engineering schemes, in order to improve the efficiency and performance of the machine learning-based side-channel analysis, tailored according to the dynamics and characteristics of the particular algorithms' leakage signals acquired from the secure implementations on Field Programmable Gate Array (FPGAs) and ARM processors.

Objective

- First objective of this research was to formulate generalized machine-learning based methodologies and to explore feature engineering methods that can reduce the redundant features, hence enabling machine learning models to perform better. As mentioned before, it is common for the side-channel leakages to have noisy high-dimensional data due to the high sampling frequency during the acquisition process. This high sampling frequency is set to ensure that the minor details of the sensitive entity are captured. Moreover, noise might be added as a countermeasure, e.g. masking and RNS, which increases the feature complexity. Furthermore, the noise might be induced due to the neighboring components on the device. Traces with redundant, irrelevant, and noisy features may introduce over-fitting, which can lead to inefficient poor-performing models. To achieve this objective, we have proposed the following:

  - A novel method was developed in which the leakage signal properties were utilized as features instead of the raw samples (signal amplitudes) (Publication I and III).
  - A hybrid feature engineering technique was proposed by exploiting the state-of-the-art feature engineering characteristics that can handle multi-dimensional data features for Public-Key cryptosystems. Moreover, a generalized machine learning-based evaluation methodology for recovering the sensitive information from the RNS-ECC implementation leakages was proposed. (Publication V).

- Another objective was to analyze the impact of the proposed feature engineering approaches on implementations of various standard algorithms. For this, the proposed approaches have been evaluated on protected and unprotected symmetric and asymmetric ciphers' implementations including AES, ECC, and RNS (Publication I, III and V).

- In general, machine learning algorithms comprise a number of hyperparameters that required tuning to achieve optimum performance. Our objective was to tune the parameters to obtain the best generalized trained model (Publication II - V).

- The objective was to analyze the practical side-channel leakages acquired from the hardware system instead of evaluating the presented models on the simulated leakages. To achieve this an automated system was setup that can collect the side-channel leakages from the secure implementations of the algorithms. This was done through the development and integration of the hardware and software setup using an oscilloscope and a proprietary desktop application developed using C# and MATLAB libraries. This setup is explained in Publication I.

Aim 2

To analyze the effect of the machine learning techniques on different side-channel attack models by formulating suitable methods, and to evaluate the designed methods on the challenging noisy leakages having low signal-noise ratio which contain significantly less information about the sensitive entity.

Objective

- Based on the nature of the electromagnetic analysis (EMA) and power analysis (PA) leakage models, ML-SCA attack can be segregated into the three models. Our objective was to formulate and investigate these models from machine learning perspective on noisy leakages (Publication IV).
- The formulated models have been evaluated according to the algorithm under analysis, which is ECC in this case (Publication IV).

Aim 3

To propose and develop efficient methods and models to improve machine learning-based side-channel attacks by exploiting the deep learning algorithms' pattern recognition capability especially for limited datasets.

Objective

- Deep learning algorithms are data-hungry, and existing deep learning-based side-channel analysis techniques have been evaluated on a huge dataset containing more than 60,000 data instances. Our main aim was to investigate the possibility of recovering the key from the small number of leakage traces. Two scenarios were explored in this regard, as given below.
  - The objective was to design and develop the deep learning-based attack model by utilizing the data balancing and dimensionality reduction techniques. The proposed model was assessed on a small dataset having low instance-feature ratio, meaning there were very few leakage traces and each trace had a huge number of features (Publication VI).

  - Another objective was to explore a possibility of increasing the leakage dataset size without collecting more data. This is especially required in scenarios where data collection is constrained by the implemented countermeasure in the cryptographic device. This was achieved by designing a model based on Conditional Generative Adversarial Network (GAN). The proposed model was evaluated on both symmetric and asymmetric datasets (Publication VII).

Aim 4

To design and develop a standalone generalized efficient leakage assessment system for detecting the leakages in black-box scenario by combining and processing the input leakages from multiple channels and multiple sources with only a few data instances.

Objective

- The objective was to design a generic system which can process information from multiple sources and channels to identify the leakage based on just few traces. This was achieved by using emerging few-shot learning concept. For evaluating the model, both AES and ECC implementation leakages were given as input to this developed system (Publication VIII).

1.3 Main Contributions

In this dissertation, various machine learning algorithms and techniques are explored. This study presents various novel attack models that can efficiently recover the secret information from the cryptographic implementation of widely used standard algorithms (AES and ECC). We have also explored the scenarios where the existing datasets are very small in size, particularly an adversary is constrained in collecting the limited data traces perhaps due to the implemented countermeasure. We have proposed an efficient deep learning-based model to process the datasets having low instance-feature ratio. We have also proposed a scheme based on standard Conditional Generative Adversarial Network (CGAN) and Siamese networks, which can generate artificial data similar to the original leakage traces and hence can address the small d ataset issue. Moreover, the key scientific contributions are explained below.

- An efficient feature engineering technique is proposed as a pre-processing step before applying machine learning-based side-channel attack, which exploits the leakage signal trace characteristics. A comparative analysis is provided using various standard machine learning algorithms on the new featured-engineered dataset for protected and unprotected implementations of the standard cryptographic algorithms; AES and ECC (Publication I and III).

- We have implemented an independent automated setup for leakage trace collection from FPGA chips mounted over Sakura-X. Sakura-X is a standard side-channel leakage evaluation, and trace collection board mounted with FPGA. This setup can be utilized to obtain the data from the hardware implementation of any cryptographic algorithm (Publication I).

- Various machine learning-based attack models for ECC algorithm implementations, based on the hamming weight, hamming distance, and identity values, are designed. A quantitative analysis is presented by performing analysis using six machine learning and deep learning algorithms. Moreover, the impact of previously proposed hand-crafted feature engineering scheme is also analyzed on all the presented attack models (Publication IV).

- A machine learning-based evaluation methodology is proposed along with a novel hybrid feature engineering scheme for the Residue Number System (RNS)-based ECC cryptosystems. RNS-ECC side-channel leakages consist of high dimensional complex feature dataset due to the inherent parallel processing characteristic of the Residue Number System, and retrieving information about sensitive entity from these side-channel leakages is a challenging task. The proposed feature engineering scheme exploits the best characteristics of dimensional reduction (Principal Component Analysis (PCA) and Linear Discriminative Analysis (LDA)) and feature selection techniques to achieve an optimal number of features which contribute most to the efficient analysis. The proposed methodology successfully retrieves the sensitive entity from the leakages, obtained from the protected and unprotected RNS-ECC Montgomery Power Ladder implementations on ARM processor, using two attack (location-dependent and data-dependent) scenarios (Publication V).

- An efficient deep learning-based attack model is designed for a successful attack on datasets having a smaller instance-feature ratio. Dimensionality reduction and class imbalance techniques are utilized with the proposed Convolutional Neural Network (CNN) architecture to create a successful attack model. The presented model, with optimum layered architecture, is evaluated on the protected and unprotected implementation of Montgomery Powering Ladder (MPL) Elliptic Curve Cryptographic (ECC) implementations. For a protected version, a secure BEC algorithm is used [25]. Various hyper-parameters are tuned to achieve a stable, efficient, and reliable model. Our designed method is computationally less expensive and outperforms the existing complex deep learning SCA models (Publication VI).

- A layered approach is proposed which is based on Conditional Generative Adversarial Network and Siamese network along with a stopping criteria to generate artificial/fake leakage data, which possess similar characteristics to that of the real leakages. A quantitative comparative analysis is provided between original and generated datasets using simple and deep learning-based models. The best performing model is further tested on the generated dataset of both AES and ECC implementations. (Publication VII).

- A multi-source leakage assessment evaluation system is designed, which is based on the emerging few-shot learning concept. It can combine leakages from the different sources of various algorithm implementations and can successfully distinguish the leakage traces with high probability. The system is validated with AES and ECC leakages. The prominent feature of the developed system is easy integration of new leakage datasets from multiple sources (symmetric and asymmetric), without retraining

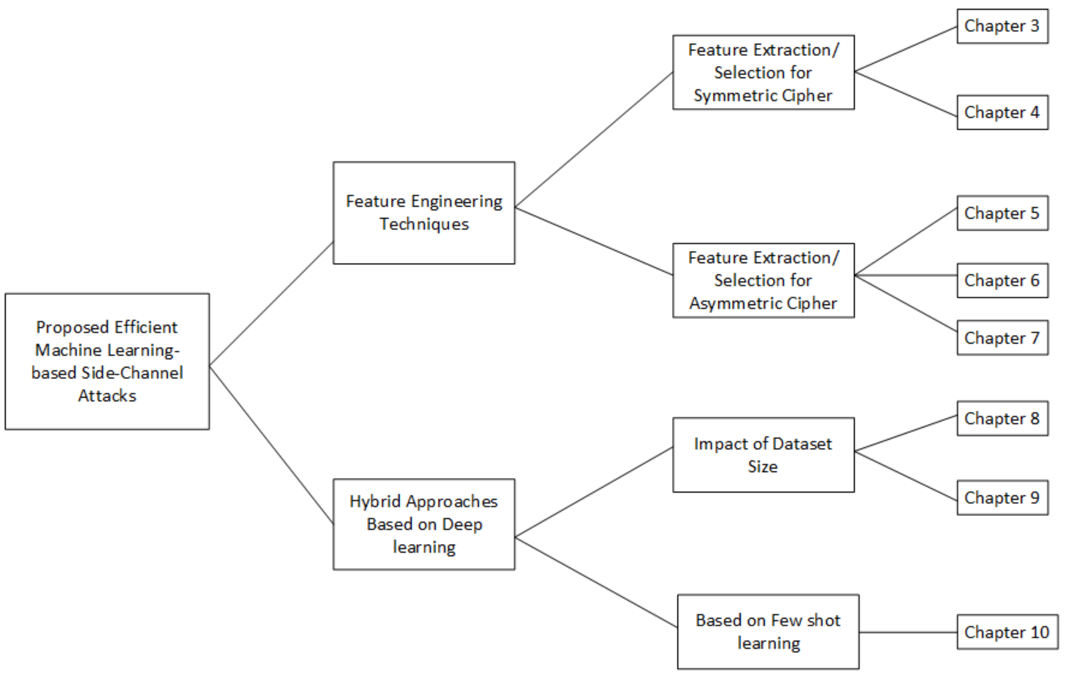

Proposed Efficient Machine Learning-based Side-Channel Attacks
Feature Engineering Techniques
Hybrid Approaches Based on Deep learning
Feature Extraction/ Selection for Symmetric Cipher
Feature Extraction/ Selection for Asymmetric Cipher
Impact of Dataset Size
Based on Few shot learning
Chapter 3
Chapter 4
Chapter 5
Chapter 6
Chapter 7
Chapter 8
Chapter 9
Chapter 10

FIGURE 1.1: Thesis Outline

the system from scratch (Publication VIII).

1.4 Dissertation Outline

This dissertation follows the non-traditional "Thesis-by-Publication" format which has been approved by the Macquarie University Higher Degree Research Office (HDRO). It consists of an introduction, background information, findings of this research (Chapters 1, 2, and 11), and the list of author's major scientific research contributions. Except for Chapters 1, 2 and 11, all the text and graphics derive from author's research articles, prepared, published, or under review in the conference proceedings or a journal. The published articles, in Chapters 3-10, are reformatted to improve their readability. The outline of the thesis is summarised in Fig. 1.1. This dissertation mainly proposes efficient machine learning-based side-channel attacks for symmetric and asymmetric ciphers and is divided into two main sets. First, a set of proposed attacks offers improvement by exploiting the hand-crafted and state-of-the-art feature engineering techniques. Conversely, the second set of the proposed attacks improves the performance with the help of proposed architectures based on deep neural networks (DNN). Each chapter's contribution to this thesis is summarized below.

Chapter 2 gives the necessary background and preliminary studies. A comprehensive review of the literature available on the approaches and methodologies related to side-channel attacks and machine learning-based side-channel attacks on cryptographic chips is provided, followed by the details of symmetric and asymmetric ciphers under attack, state-of-the-art machine learning algorithms, and background description of the feature engineering techniques.

Chapter 3 describes the proposed hand-crafted feature engineering techniques for selecting/extracting contributing features, which is a challenging task in applying machine learning techniques to side-channel attacks. To reduce the trace length and thus the attack complexity, time- and frequency-domain signal properties are used as features instead of the raw leakage signals' amplitude. The crafted features are further processed using state-of-the-art feature selection/extraction techniques to improve attack efficiency. Electromagnetic leakages are analyzed to recover the secret key from the symmetric cipher, using various machine learning algorithms. A standalone data collection setup is is also presented for acquiring the data (Publication I).

Chapter 4 advances the work performed in the previous chapter. In this chapter the impact of hyperparameter tuning is analyzed for the selected machine learning algorithms and best performing hyperparameters are selected for an improved attack (Publication II).

Then in Chapter 5, we have proposed a hamming weight based attack methodology which utilizes the previously proposed feature generation methodology to asymmetric ciphers along with the concept of using feature engineering techniques in hybrid mode. This approach is used to evaluate Elliptic Curve Cryptography (ECC) always-double-and-add algorithm. A quantitative comparative analysis is performed, for the proposed attack, to recover the secret key from the raw unprocessed leakage traces and from the processed feature-engineered traces (Publication III).

Side-channel attacks can be categorized into divide-and-conquer and analytic attacks [26, 27]. A complete sub-key is retrieved in analytic attacks, whereas only partial key is retrieved in the divide-and-conquer attacks. Based on this, various machine learning attack models can be formulated to approach the problem of the secret key recovery focusing on key byte or key bit recovery. In Chapter 6, three machine learning-based attack models are presented along with the selection strategy for the best performing model. These presented attacks are evaluated on electromagnetic leakages from asymmetric key ECC algorithm for raw and feature-engineered datasets as proposed in Chapters 3-5. Generally, noise is introduced as a countermeasure to hide the relationship between the leakage and the secret key. Hence in this study, another challenge is introduced by performing analysis on the heavily noised leakage traces, consisting of relevant noise-free information in the 3% portion of the trace only. The presented attack models still outperform by recovering the secret key from the noisy traces (Publication IV).

Based on the side-channel vulnerability findings of the classical side-channel analysis, various countermeasures are proposed to secure the algorithms against such implementation attacks, including masked key and key randomization. In previous chapters, in addition to unprotected algorithm versions, masked key countermeasure is also considered. For asymmetric ciphers, particularly ECC, the Residue Number System (RNS) is considered one of the strongest countermeasures due to its parallel processing on the moduli. In Chapter 7, we have proposed a systematic machine learning-based evaluation methodology of RNS-ECC cryptosystem by analyzing the electromagnetic leakages from an ARM processor implementation. Moreover, in this chapter, we have offered a hybrid feature processing approach, based on the state-of-the-art feature selection/extraction techniques. This has been done to select the most contributing features efficiently. For the purpose of evaluation, two simple machine learning and two deep learning algorithms are employed. For recovering the secret key from the asymmetric algorithm, two attack scenarios are investigated, i.e. location dependent and data dependent attacks. (Publication V).

Successful efficient deep learning based side-channel attacks (DL-SCA) require huge datasets consisting of a large number of instances. The number of instances in target classes directly affects the system's capability to determine the pattern between the secret key and the leakage data. With more instances for all the target classes, a better pattern can be learnt about the sensitive information and leakage data relationship. The small dataset problem might arise due to the system limitation of the adversary, where the adversary is not allowed to collect more leakages in the provided time frame. The situation is aggravated when the instance-feature ratio is low for the high-dimensional data traces. Ideally, the more features there are per instance, more instances are required for the efficient analysis. Chapter 8 addresses this concern. A deep learning CNN-based architecture is presented which utilizes the dimensionality reduction and class imbalance techniques for improving the efficiency and performance. The presented model is evaluated on ECC Montgomery Power Ladder (MPL) protected (BEC) and unprotected implementation leakage datasets which have low instance-feature ratio. A comparative analysis is provided to assess the presented model by analyzing the impact of dimensionality reduction and class imbalance alone and by comparing the performance results with the existing complex models (Publication VI).

Chapter 9 discusses the problem scenario where fewer instances are available for one class, which is also referred to as the curse of class imbalance [28], and might fail to generalize on the unseen data and thus leading to an over-fitted poorly trained machine learning model. The solution to this problem is also discussed in the chapter. To handle the issue of small dataset and the dataset experiencing a class imbalance issue, I have proposed a data augmentation scheme based on Conditional Generative Adversarial Network (CGAN) and Siamese networks for generating artificial data traces. The generated data samples are similar in characteristics to the original leakages and help to improve the attack model's efficiency. The evaluation results on both symmetric and asymmetric ciphers implementation leakages are also demonstrated. This chapter also investigates non-convergent networks' effect on the generation of fake leakage signals using two CGAN based deep learning models (Publication VII).

Chapter 10 presents a few-shot learning-based leakage assessment model which combines side-channel leakage datasets from different algorithm implementations. The chapter also presents results of integration of two cryptographic algorithm leakages, i.e. AES and ECC, acquired from FPGA and STM32F4 microcontroller with an ARM-based architecture (Publication VIII).

Concluding remarks are given in Chapter 11 along with the future directions.

Chapter 2

Background and Related Work

This chapter provides background knowledge of basic concepts and preliminaries related to Side-channel analysis, cryptographic algorithms under attack, state-of-the-art feature engineering and dimensionality reduction techniques. It also presents the overview of existing machine learning-based side-channel attacks along with the limitations.

2.1 Introduction to Cryptology

Cryptology, which is the science of secrets, refers to secure data communication and storage. It consists of two branches, cryptography and cryptanalysis. Cryptography, from the Greek kryptós (hidden, secret) and graphein (to write), refers to the practices and study of techniques required for securing data in transit or in an inactive state. It generally involves transforming and hiding data so that a third party, adversary, cannot intercept and reveal the secret information over an insecure channel. However, cryptanalysis refers to the study of analyzing the strength of the proposed cryptographic techniques by designing and evaluating the attack scenarios to break the security claims. With the rapid advances being made in technology, modern cryptology brings together various technologies from different fields including applied mathematics, electrical engineering and computer science, to form a cryptosystem which ensures the secure transmission and storage of the secret data. A cryptosystem consists of the mathematically secure algorithms which are implemented in software or on hardware using electronic circuits exploiting the resources of the processors or controllers (like Field Programmable Gate Array (FPGA), ARM, MSP, AVR, and MSP430) deployed in the IoT based embedded systems. These secure cryptosystems ensure the following; confidentiality, data integrity, authenticity, or non-repudiation.

2.1.1 Cryptographic Algorithms Under Analysis

Cryptography is further categorized into symmetric and asymmetric cryptography. In symmetric cryptography both the parties (sender and receiver) share the same secret key k to encrypt and decrypt the data. However, in asymmetric cryptography (also known as pubic-key cryptography), a (private-public) key pair is generated through a mathematical function, user encrypts data using one key whereas receiver decrypts using the other.

Symmetric key cryptographic primitives have the advantage of fast processing, However, the practical deployment of these symmetric primitives is limited by the distribution of unique shared key among all the involved users. For a symmetric cipher, each user needs to share a unique key with the other participating user for the secure communication. Assuming there are n users then n^2 keys are required to be shared among the users, which is practically not feasible for the large networks. On the contrary, asymmetric ciphers are computationally expensive and require a lot of processing resources, which limits their usage in practical systems; for example if used for voice encryption, the asymmetric cipher will present a considerable amount of delay. Hence, a hybrid approach is used where sessional symmetric keys are encrypted using asymmetric algorithms. The symmetric keys are generally smaller in size; varying between 128-bits to 256-bits.

Modern cryptography does not follow 'security by obscurity' and relies on the 'Kerckhoff's principle'. According to this principle, the security of a cryptosystem solely lies in the secrecy of the cryptographic keys. This means that the cryptosystem should be secure by design, provided that all the details of the cryptographic primitives are known to the adversary, yet the adversary cannot break the system. The primitives include low-level cryptograhic routines which act as a building block in any secure protocol. These primitives are standardized by the global institutions, with the publicly available design and implementations. Hence, rigorous evaluation has been carried out on these standard algorithms by the community. With reference to the found weaknesses, countermeasures are proposed to increase the level of security. For this research, two standard primitives are selected as target algorithm; Advance Encryption Standard (AES) and Elliptic Curve Cryptography (ECC), where former belongs to the symmetric category while the later belongs to the asymmetric category. Below is a brief introduction to both standard primitives, whose protected and unprotected implementations have been evaluated using proposed machine learning-based attacks.

Advanced Encryption Standard(AES)

The National Institute of Standards and Technology (NIST) is a United States governmental agency and it standardized Advanced Encryption Standard (AES) in 2001 through the Federal Information Processing Standards Publication 197 (FIPS PUB 197) [29]. AES is a symmetric block cipher which requires a same shared secret key for both encryption and decryption. Encryption and decryption engines take input (plaintext and ciphertext respectively) of a fixed length (128-bit). Greater size inputs are divided into chunks and are then processed by respective blocks. There are three variations of AES depending on the key size being used, that is 128, 192 and 256. Information in processed in rounds for encryption and the number of rounds depend on the length of encryption key (10 rounds for 128 bits, 12 for 192 and 14 for 256). There are five functions involved and for processing of data through the functions, the input bytes are arranged in the form of two-dimensional array called state, s, consisting of 4 rows and 4 columns, hence total 16 bytes (represented in Fig. 2.2). Each byte can be accessed using $s_{i,j}$ (for $i, j \epsilon 1, 2, 3$), where i and j represent i-th row, j-th column of s, respectively. Each state element $s_{i,j}$ is mathematically an element from the Rjindael finite field, defined as GF(28) = Z/2Z[X]/P(X) where $P(X) = X^8 + X^4 + X^3 + X + 1$. The five functions performed during the AES encyrption/decryption are KeySchedule, AddRoundKey, Sub-Bytes, ShiftRows and MixColumns. All functions are performed during all rounds except for the first and last round. Each function is briefly explained below.

- KeySchedule - Key expansion is performed and round keys are derived from the secret key using AES Key Schedule.
- AddRoundKey - Each byte of state s is XORed with the corresponding byte of the round key over the Rjindael field $GF(2^8)$.

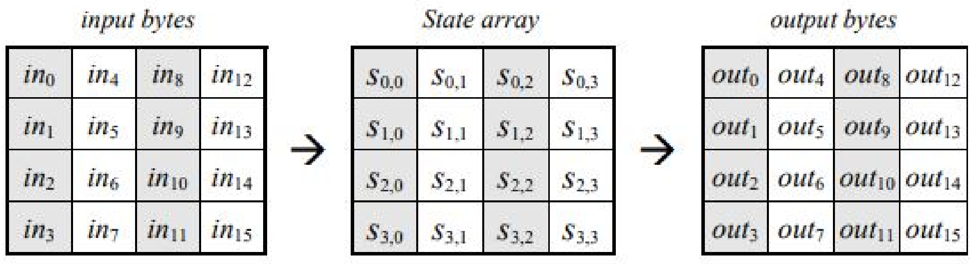

input bytes
State array
output bytes
in_0 in_4 in_8 in_{12}
in_1 in_5 in_9 in_{13}
in_2 in_6 in_{10} in_{14}
in_3 in_7 in_{11} in_{15}
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{1,1}$ $s_{1,2}$ $s_{1,3}$
$s_{2,0}$ $s_{2,1}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,1}$ $s_{3,2}$ $s_{3,3}$
out_0 out_4 out_8 out_{12}
out_1 out_5 out_9 out_{13}
out_2 out_6 out_{10} out_{14}
out_3 out_7 out_{11} out_{15}

FIGURE 2.1: AES State Array. Source [29]

- Sub-Bytes - It is an affine non-linear byte invertible transformation performed on each state byte using substitution boxes (Sbox).

- ShiftRows - It cyclically shifts the rows of the state towards the left. The first row is not changed in this case.

- MixColumns - Each column is considered a polynomial and is multiplied modulo $X^4 + 1$ with a fixed polynomial $a(X) = 3X^3 + X^2 + X + 2$.

Elliptic Curve Cryptography (ECC)

ECC was introduced by Koblitz and Miller in early 1980's. Due to its fast processing it is preferred public-key cryptosystem for resource-constraint environments like cryptographic chips in IoT-based systems. The most expensive operation in ECC is the scalar multiplication (SM). In other words the addition of point

$$P(x, y, z)$$

on curve to itself. If P is added to itself k times then multiplication $Q(x, y, z)$ will be a new point on the same curve and is given by eq. 5.1. This multiplication is also known as Elliptic-Curve Scalar Multiplication (ECSM).

$$Q = k * P \tag{2.1}$$

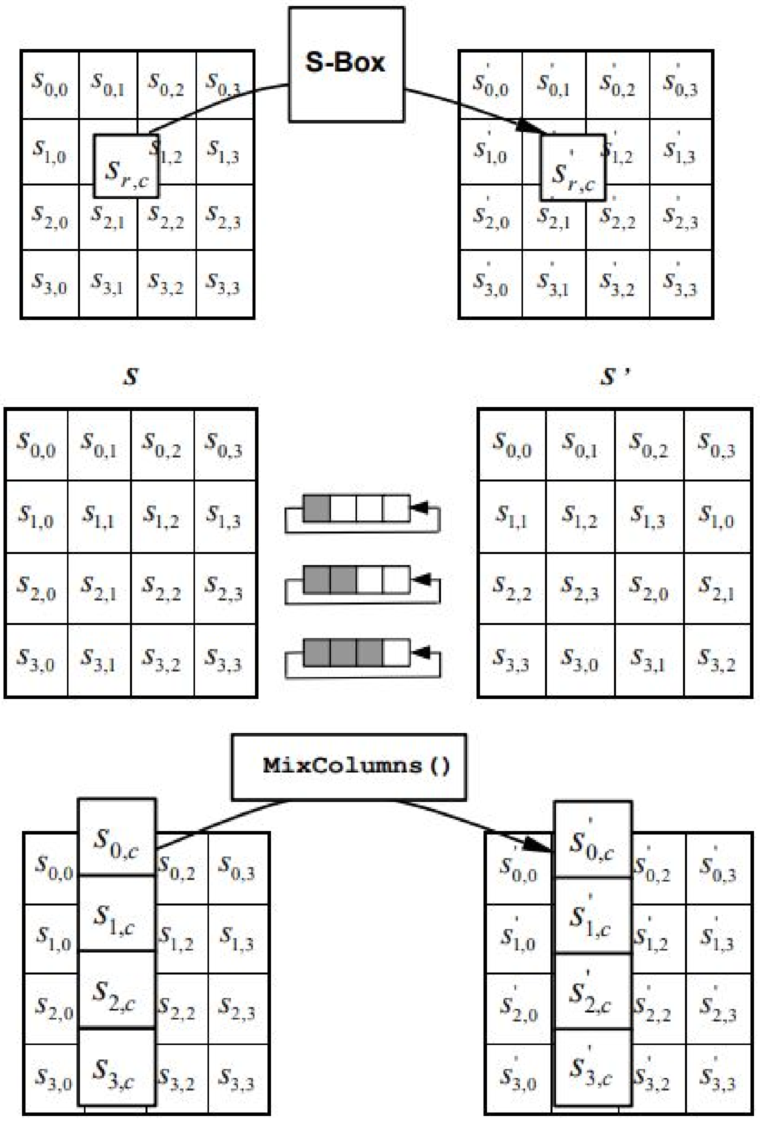


FIGURE 2.2: AES Sbox, Shiftrows, MixColumns operations. Source [29]

2.1.2 Elliptic Curve Domain Parameters

To use ECC, both participating parties should agree on the domain parameters (p,a,b,G,n,h) for the field p defined over prime field F_p, where a and b are constants used in Weierstrass equation, and G is a point o curve. These parameters are defined by the standardization authority and are categorized as separate curves.

NIST Standard for 256-Bit Koblitz Curve

The NIST curve (SECP256K1), used in this analysis, over prime fields F_p, is defined as E: $y^2 = x^3 + ax + b$ mod p, where $a = 0$ and $b = 7$ and $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$.

The two main field operations in the double-and-add-always algorithm are point doubling and point addition in Jacobian coordinates over curve E [30]. Jacobian coordinates are preferred over affine coordinates because the inversion operation can be avoided while performing the addition or doubling operations.

Point Doubling in Jacobian Coordinates

This section gives the formulas used for implementing point doubling. Suppose: P(X_1, Y_1, Z_1) and

$$\alpha = 3X_1^2 + \alpha Z_1^4, \beta = 4X_1Y_1^2, \tag{2.2}$$

Point Q on curve E is defined as: $Q(X_2, Y_2, Z_2) = 2 * P(X_1, Y_1, Z_1)$

$$X_2 = \alpha^2 - 2\beta, \tag{2.3}$$

$$Y_2 = \alpha(\beta - X_2) - 8Y_1^4, \tag{2.4}$$

$$Z_2 = 2Y_1Z_1, \tag{2.5}$$

Point Addition in Jacobian Coordinates

This section describes the formulas used for implementing point addition. Suppose: $P_1(X_1, Y_1, Z_1)$ and $P_2(X_2, Y_2, Z_2)$ are two points on curve (E) and

$$\gamma = Y_1Z_2^3, \lambda = X_1Z_2^2, \ \mu = Y_2Z_1^3 - Y_1Z_2^3, \xi = X_2Z_1^2 - X_1Z_2^2, \tag{2.6}$$

The new point P_3 on curve (E) such that: $P_3(X_3, Y_3, Z_3) = P_1(X_1, Y_1, Z_1) + P_2(X_2, Y_2, Z_2)$ is:

$$X_3 = \mu^2 - \xi^3 - 2\lambda\xi^2 \tag{2.7}$$

$$Y_3 = \mu(\lambda\xi^2 - X_3) - \gamma\xi^3, \tag{2.8}$$

$$Z_3 = Z_1Z_2\xi, \tag{2.9}$$

All calculations are to be done in finite field F_p, meaning that *mod p* reduction is applied to Formulas (6.1)–(6.8).

2.1.3 Elliptic Curve Point Multiplication Algorithms

Binary double-and-add Algorithm

Double-and-add algorithm is the most simple algorithm to computer ECSM operation. In double-and-add, operations are performed based on branching of key value k; if k is 1 then both point addition and point double operations are performed. However, only point double is performed if k is 0. The simple double-and-add algorithm is susceptible to a simple power-analysis (SPA) attack, in which key can be recovered by merely looking at the power consumption signals without using any advanced techniques. It is possible due to the fact that both the operations consume different amount of power while performing computations. A simple countermeasure to this attack is to use double-and-add-always algorithm. In this algorithm, both double and add operations are performed always irrespective of the key bit value. Hence, uniform power is consumed. This countermeasure-protected algorithm is resistant against SPA but is not secure against safe-error attacks. However, double-and-add-always is still a preferred choice in certain scenarios. Further details of the algorithm 6.3.3 can be found in [31].

Algorithm 2.1**. double-and-add-always**

```
Input: $P$, $k_n$
Output: $Q$ , $kP$
1. $R_0 = P$, $R_1 = 0$
2. For $i = 1$ to $n - 2$
     $R_0 = 2R_0$
     $R_1 = R_0 + P$
     If ($k_i = 1$) then
       $R_0 = R_1$
     end if
3. Return $Q = R_0$
```

Montgomery Powering Ladder Algorithm

Another algorithm to compute ECSM is known as Montgomery Powering Ladder (MPL). In the MPL scalar multiplication algorithm, ECC point multiplication core requires ECPA and

ECPD to run independently. Therefore, the arithmetic hardware cannot have shared hardware resources. The MPL scalar multiplication algorithm [32] computes the point multiplication in a fixed amount of time. This is particularly useful when timing or power consumption is exposed to side-channel attacks. The algorithm uses a similar representation to the double-and-add. An initial point-doubling is required to compute the value of $2P$. Then, P and $2P$ are saved as default values of $R0$ and $R1$ registers, respectively. One point-doubling and one point-addition is performed for every bit of the key k. Then, the value of the $R0$ and $R1$ registers are updated. Montgomery Power Ladder has been used in different settings. Detailed analysis and further explanations are given in the respective chapters. . Generic MPL algorithm is given in Algorithm 2.2.

Algorithm 2.2. **Montgomery Powering Ladder algorithm**

```
Input: $P$ : EC base point $\in EC(F)$,
$k = (k_{t-1}, k_{t-2}, ...k_0) \in GF(2^K)$
Output: $e \cdot P$
1. $R_0 = O, R_1 = P$
2. For $i = t - 1$ to 0
    If $(k_i = 0)$ then
      (a) $R_1 = R_0 + R_1, R_0 = 2 \cdot R_0$
    else
      (b) $R_0 = R_0 + R_1, R_1 = 2 \cdot R_1$
    end if
3. Return $R_0$
```

2.1.4 ECC Security

The security of ECC relies on the elliptic curve discrete logarithm problem (ECDLP). Let E be an elliptic curve defined over the prime field $\mathbb{F}_p$, and $E(\mathbb{F}_p)$ denote the group of rational points on the curve E. Given a point $P \in E(\mathbb{F}_p)$ of order n, the cyclic subgroup of $E(\mathbb{F}_p)$ generated by point P, i.e., $\langle P\rangle = \{O, P, 2P, \ldots, (n-1)P\}$, a random integer $k \in [1, n-1]$, and $Q = k \cdot P$. Then Q is defined by adding point P to itself $k - 1$ times as shown in eq. 6.5.

$$Q = k \cdot P = \underbrace{P + P + \cdots + P}_{k \text{ times}}. \tag{2.10}$$

Given all the domain parameters and Q, the integer k cannot be easily determined computationally. This problem is known as ECDLP [30].

2.2 Security Vulnerabilities in IoT Systems

Today's world is more about connected devices through the internet. IoT-based systems are flourishing and improving the standard of living in every sector of life including personal, industry, home life, healthcare, to name a few; improved life-style through smart cities, better real-time intelligent health-care monitoring system saving lives, and controlling traffic conditions using sensor-based system, smart grids, intelligent vision, and real-time power state estimation. Moreover, cognitive Industrial Internet of Things (IIoT) is revolutionizing the industrial manufacturing infrastructure. Cognitive Industrial Internet of Things (IIoT) applications include cloud-connected smart factories that automate the manufacturing process, smart grids, intelligent vision, intelligent medical systems, and real-time power state estimation. Cognitive IoT is breeding smart factories built on top of artificial intelligence technology, with well-instrumented and interconnected devices, control systems, and sensors. When put to predictive modeling, the industrial automated system-generated Big Data offers numerous benefits like early defects and production failure detection, 100% product verification instead of random sampling.

There are a plethora of devices involved in the above mentioned IoT-based systems, including sensors, embedded processing chips, internet routers, cloud servers, etc. These devices communicate over the network to exchange the data and ample focus has been given to the network security in this regard. Embedded chips are the core of these IoT based systems, ranging from low-end to high-end devices. Micro-controllers and microprocessors including ARM, AVR, and MSP430, are commonly used for data processing in IoT devices. Various manufacturers, including Intel, are proposing Industry 4.0 autonomous solutions based on Field Programmable Gate Array (FPGA). FPGA chip-based designs offer less production time and quick integration of new features in the existing systems due to their re-configurable nature. Identification of side-channel leakages from security algorithm implementations on any of these processor chips will potentially introduce vulnerability to all the IoT devices mounted with the particular chip. In this research one low-end (ARM) and one high-end (FPGA) processor is selected for experiments.

One of the limitations of these embedded devices is the availability of on-chip resources. For the efficient implementations on these chips, security is often compromised. Breach of security in control systems can cause fatal financial and reputational damages and, above all, can jeopardize people's lives. This leads to the fact that in all IoT based systems, security measures must be integrated at the design level [33]. The standard AES and ECC algorithms are mathematically secure, but the weak implementation of these algorithm can introduce side-channel attack vulnerability.

IoT devices, processing sensitive information, face the potential risk of being exposed to the physical threats. They can be physically captured, disassembled and analyzed forensically to recover the secret information using side-channel leakages. Meulenaer et al. have presented successful power analysis attack on the commonly used nodes (MICAz and the TelosB) in wireless sensor network (WSN), to recover the secret information in a stealthy way [34]. The threat and damage are aggravated through the help of machine learning paradigm. Machine learning can be utilized to exploit the side-channel leakages to investigate the internal software activities in the IoT devices. This can introduce security vulnerability as well as can help in malicious activity or anomaly detection, proactively. Wang et al. presented results for the anomaly and software activity detection based on the changes in EM leakages from IoT devices, and evaluated them using MLP, LSTM and Autoencoders [35]. Sayakkara et al. examined a case of using ML-SCA for forensic analysis to detect a wide variety of changes to target IoT devices so that secret information can be exploited [36].

2.3 Side-Channel Attacks

In classical cryptanalysis, the cryptosystems were considered secure if the underlying cryptographic algorithm was mathematically secure, that is no mathematical relationship can be found between the key, the plain-text, and the ciphertext. To evaluate the strength mathematically, an attack model is designed in which the cryptosystem is deemed to be a black-box providing no information about the internal operation executions or internal sensitive variables. The attacker/adversary has knowledge of the encryption algorithm E and can control the input (plaintext) P and/or the output (ciphertext) C. She aims to retrieve the unknown secret key K by exploiting the relationship (if any) of K with P and/or C. This security assumption of a blackbox was shattered in 1996 when Paul Kocher showed in his seminal paper [3], that the weak implementations of the cryptographic algorithm on a physical component can help in deducing the information about the internal variables, operations and the key components. This finding led to whole new era of cryptanalysis. Researchers started exploring other possible physical leakages that can be exploited to retrieve the secret key.

Side-channel attacks are the class of cryptanalytic attacks which exploits the physical information leaked from the embedded system. These attacks are segregated on the basis of a parameter used to retrieve the information e.g. power, electromagnetic radiations, timing, sound, etc. [4–6]. Initially, it was assumed that the leakages from the low-end devices, having microprocessors, can be exploited only. However, recently it is shown that the leakages like acoustic and electromagnetic emanations from the high-end devices can be exploited as well [8, 37]. An attacker can take advantage of the sound produced by the hardware while cryptographic software is running over them to extract the secret information from the system. Another class of attacks is timing attacks in which timing measurement information is exploited on the basis of key/data dependent branch instructions, cache hits or processor instruction execution time. Moreover, the signal displayed on CRT can be reconstructed due to diffuse reflection.

In SCA, there are three main phases/stages as explained below.

- Phase I is the side-channel data acquisition phase, in which a combination of the specialized hardware and software collects the traces from the cryptographic device. For instance, to launch power analysis attack, a shunt resistor should be connected in series with VCC to FPGA and leakages are collected using Oscilloscopes or openADC. For electromagnetic attacks, special probes are required to collect EM radiations, by

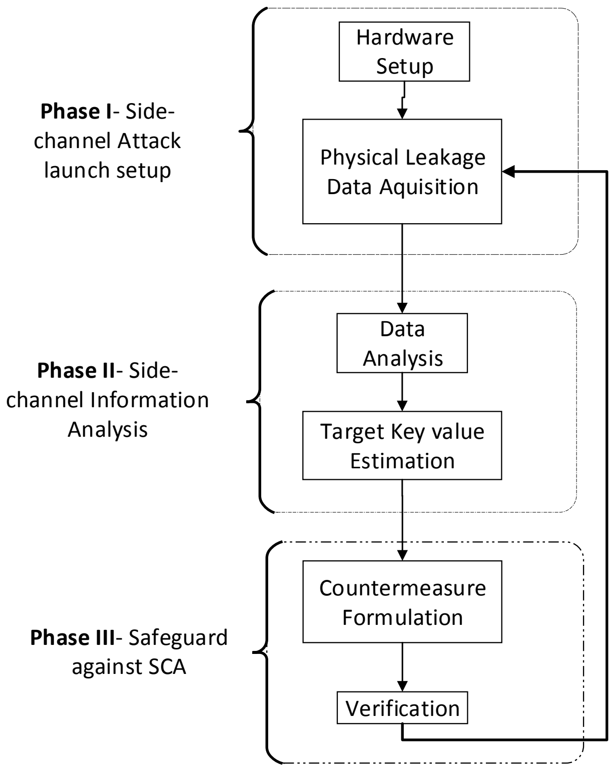

Phase I- Side-channel Attack launch setup
Hardware Setup
Physical Leakage Data Aquisition
Phase II- Side-channel Information Analysis
Data Analysis
Target Key value Estimation
Phase III- Safeguard against SCA
Countermeasure Formulation
Verification

FIGURE 2.3: Security Cycle for SCA

placing the probe on a suitable point generating the best signal. Amplifiers or noise reduction techniques may be used to improve the signal quality and to attenuate the noise introduced due to the interference by the other components. Similarly, for acoustic attacks, special microphones are required to be placed near the target device which receives the sound produced when a particular algorithm is performed and transmit it to the analysis device.

- Phase II is actual side-channel Information Analysis. In this phase, an attack model is defined and the target sensitive entity is recovered using the specialized statistical or machine learning techniques. Details of each are given in the next sections.

- Phase III is designing countermeasures to safeguard the secret information from the potential threats. The proposed countermeasures are then evaluated again using the existing SCA techniques.

Fig. 2.3 shows the security cycle with regards to SCA.

2.3.1 Attacks Categories

The literature documents varying views on categorizing SCAs. Generally they are divided into two wide classes, hardware and software attacks. In the physical hardware attacks, device is accessed through physical interfaces to gain access to the secret information or to gain access to the information which can be exploited to retrieve the secret information. In contrast to the physical hardware attacks on the cryptographic devices, the software attacks also offer a great vulnerability to the data security. In these attacks, the attacker can induce a malware in the device and monitors or retrieves the secret information from the memory.

Side-channel attacks belong to a wider family of physical hardware attacks, also known as Implementation Attacks. Physical hardware attacks can be mainly categorized into Fault-Injection Attacks (FAs) and Side-Channel Attacks (SCAs). However, overall all these physical attacks can be classified depending on the nature of invasion/interaction with the device under attack, as shown in the Fig. 2.4 ([38]) and explained briefly below:

- Active Attacks - In active attacks, attacker tampers with the device affecting the normal functionality of the device. For example, in FAs, errors/faults (in voltage, clock, temperature, light etc.) are induced to change the behavior of the device. Fault attacks can either be computational fault attacks caused by voltage manipulation or can be one in which faults are introduced using corrupt input.

- Passive Attacks - In passive attacks, attacker monitors the traffic and device behavior without altering the regular functionality.

- Invasive Attacks - In invasive attacks, the device is de-packaged to gain access to the internal components of the device. This is one of the strongest and expensive types of attack. An attacker can conduct micro-probing to remove the passivation layer of the integrated circuit and subsequently analyze the signals while the cryptographic algorithm is executed.

- Non-Invasive Attacks - In non-invasive attacks, the externally accessible interfaces are exploited without modifying the device. For example, timing information, power

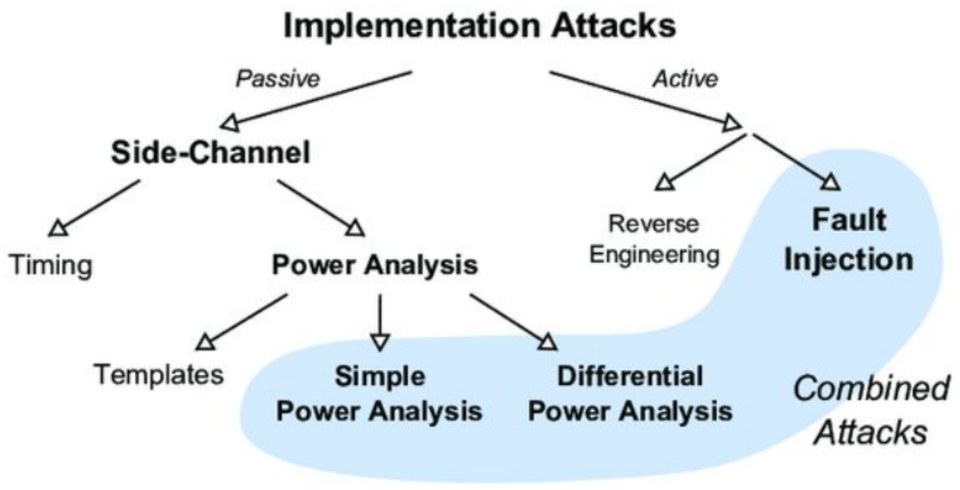


FIGURE 2.4: Classification of Implementation Attacks [38]

consumption by the cryptographic chip, electromagnetic emanations from the system, can be obtained from the system and analyzed further to recover the secret key.

- Semi-Invasive Attack - In semi-invasive attacks, the device is unpacked but the passivation layer is not damaged and there is no direct electrical contact with the device.

SCAs commonly categorized as passive non-invasive attacks as the side-channel leakages are obtained without unpacking the device, for example in timing attack or electromagnetic attacks. However, in certain scenarios the device might be unpacked to amplify the signal. In this case, SCAs can belong to the semi-invasive category.

2.3.2 Target Sensitive Entity

Physical leakage signals, named as traces, are acquired from the cryptographic device, during the execution of a primitive operation, using appropriate probes and instruments. Let the complete leakage trace dataset is denoted by X, then x_i represents the instance containing time samples. The traces consist of noise due to the neighbouring components on the board or it might be intentionally induced as a countermeasure. Side-channel analysis deals with distinguishing or classifying the traces by exploiting the relationship between the trace and the target sensitive entity. This target sensitive entity can represent:

- an encryption key k or chunk of key bits. If a constant key is used for encryption using unprotected device primitive then a 'simple attack' can aid in recovering the target value. However, for noisy protected traces, advance machine learning techniques can be applied to label and classify the target key bit. For this type of target entity (i.e. key itself) ML-based SCA for ECC is presented in Chapter 6.

- information based on the relationship between a secret key and input variables. To recover target sensitive entity in this case, 'advanced attacks' like DPA or DEMA can be launched;

- an operation, whose execution is dependent on the secret key ($f(k,E)$). The most common example is branching, where different operations are performed based on different key values. For example in RSA and always-double-and-add algorithms, square and multiply operations and double and addition operations are differentiated to recover the secret key, respectively;

- a register whose value are routed depending on the value of the secret key. For example in Montgomery Ladder, registers $R0$ and $R1$ receive the result of the operations based on the key and represent either the key bit was '0' or '1'. For this type of target entity ML-based SCA for ECC is presented in Chapter 8.

- a function that carries some information about the secret key. In certain scenarios for SCA and especially machine learning-based SCA, targeting a non-injective function,like hamming weight or hamming distance of the key value, can provide better information to recover the sensitive information (as analysis is shown in Chapters 3,5, and 7. The hamming weight operation is represented as HW(.).

For ML-SCA, this sensitive entity plays a more important role in shaping the attacks. Further details on the ML-SCA attack details are given in sub-section 2.4.2.

2.3.3 Types of Attacks

Due to the ever-evolving taxonomy of the SCAs and a variety of the side channels, it is becoming harder to divide SCAs into proper types. However, there is a common consensus on the two types as describe in this section. In this research two main side channels are focused on, i.e. power consumption and electromagnetic emanations, hence the types are explained with respect to these two specifically. However, the nature of attacks are generic.

Simple Attack

Simple attack exploits the relationship between instruction and secret information by visually analyzing the leakage trace to extract the data. As the name implies, in simple attacks no complex statistics or computations are applied. The target entity can be recovered from a single trace by observing it with the naked eye. These attacks are referred to as Simple Power Analysis (SPA) or Simple Electromagnetic Analysis (SEMA). Such attacks can be launched on the cryptographic devices in which target sensitive entity is associated with the execution timing or execution flow of the particular operation. For example, Fig. 2.5 shows the power consumption pattern of the operations performed during processing of RSA and ECC crypto primitives. For RSA, multiplication operation consumes more power when compared to square operations and can easily be distinguished from the acquired leakage trace. Similarly, for ECC doubling need more power as compared to the addition.

One of the characteristics of simple attack is that generally they do not need a variation in signals to break the systems, if observed with the naked eye. One single trace can be used to recover the sensitive secret information. For this reason they are also referenced as one-trace attacks. In the literature, one-trace attack and simple attack are equivalent [39]. A class of simple attacks can also process more than one trace observations to recover the secret information, with a variable or fixed target sensitive entity. For the fixed target entity, the attacker can exploit several acquisitions either by computing their average or by reducing the impact of noise or by attacking each acquisition [40, 41]. These analyses shows that these simple attacks can be extended to an investigation of several observations, collected for a fixed or variable entity, using machine learning-based SCA. Further details are discussed in

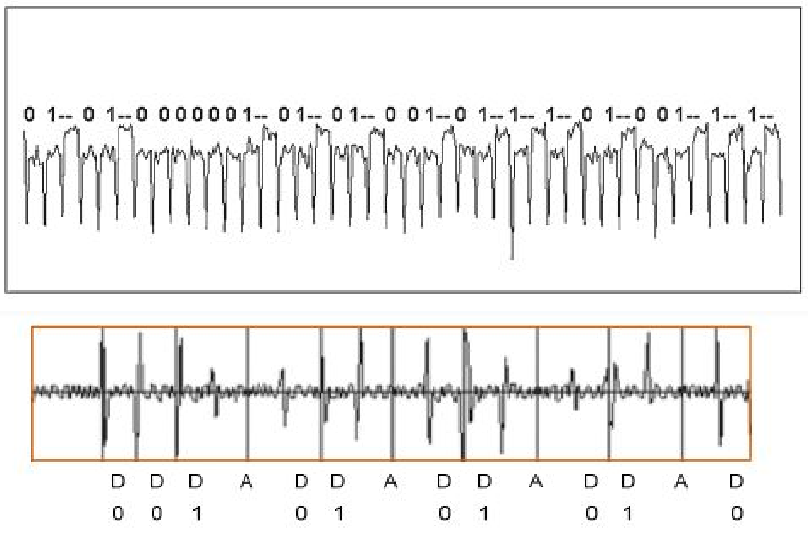


FIGURE 2.5: Simple Power Analysis of RSA. Source [42] and [43]

Sec. 2.4.2.

Advanced Attack

In the SPA, attacker analyzes the variations in signals from 0 to 1, having high signal-to-noise (SNR) ratio with the detail information about the underlying cryptographic device. However, as explained before, leakage signals are not noise-free and can have inherent builtin noise, probably due to the proposed countermeasures. To recover the secret information from these noisy leakages, in classical SCA advance statistical methods are utilized, known as advanced attacks or differential attacks. Differential attacks (Differential Power Analysis (DPA) and Differential Electromagnetic Analysis (DEMA)), are data dependent; the relationship between data and power consumption or EM emanations is exploited with the help of a hypothetical model and requires less information about the implementation of the cryptographic primitive. Term differential refers to fact that the method exploits the small differences in the behaviour of the device. The first DPA tool performing advance attack (referred to as Difference of Means (DoM)) actually took differences using subtraction [6]. In contrast to the simple attacks, these attacks require more than one trace collected using the variable target sensitive entity instead of a fixed target entity. Interestingly, the small differences increase by means of averaging over an considerable amount of acquisitions, meaning with more noise hiding the trace-target relationship, more acquisitions are required to recover the secret information.

However, template attacks might remove the need for a large number of samples [44].

2.3.4 SCA Leakage Model

The CMOS technology is widely used as standard in the applications. Static CMOS generally has three dissipation sources [45]. First one is the leakage current in transistors, which is prominent due to scale-down technology. Second is the direct-path current, which exists during the duration when the switching of a gate while the NMOS (pull-down transistor) and the PMOS (pull-up transistor) are conducting simultaneously, and it accounts for around 20% of total power consumption. Third is the most important dissipation from SCA perspective, and this is due to the charge and discharge of the capacitor. The SCA is based on the fact that in the CMOS technology, the peaks of the power consumption peaks are observable and distinguishable whenever there is a bit transition that is a gate transitions from '0' to '1' or '1' to '0'. In 1998, Kocher et al. presented a hamming weight based attack model which exploited the power consumed by the microchip to recover the secret information [6]. The idea was later utilized on a variety of processing chips including FPGAs [46–51]. Moreover, when the current flows through the transistors, the electromagnetic (EM) field is generated and the radiations can be acquired with specialized equipment.

Various studies showed practical results for recovering the secret information from the cryptographic device by using electromagnetic radiations [52, 53]. The power consumption and the emitted EM radiations rely on the underlying transitions of 1's and 0's, which are infact dependent on the underlying cryptographic operation. Based on the leaked information various leakage models can be formed, leading to power analysis or electromagnetic analysis attacks. Let Z represent the sensitive target entity then leakage model $L(Z)$ can be the deterministic function of Z in one of the following settings; mono-bit model (one bit of Z), identity model (Z itself), Hamming weight model (HW(Z)), Hamming distance model(HW between Z and another intermediate value), and linear model (linear combination of Z bits). Out of these leakage models, the first four are considered in this research for designing a ML-SCA model for AES and ECC implementations. In any of the above mentioned leakage model, the leakages are the noised observation of $L(Z)$, assumed to be a combination of the sensitive entity information and noise. Hence, in literature, researchers have tried to specify the form of these noise leakage models to better understand the relationship of the leakages and sensitive entity, and to retrieve the secret information [54, 55]. These leakage models either consider noise to follow a Gaussian distribution, as an addend of the deterministic function L(Z) and quantified by standard deviation, or is quantified as a statistical distance between the distribution of sensitive target entity (Z) and the conditional distribution of Z given noise variable. For the machine learning analysis employed in this thesis, we do not focus on determining the noise leakage model. We assume that noise impacts the quality of the acquisitions only and hence the noisy patterns can aid in learning more better about Z.

2.3.5 SCA Countermeasures

There are two main strategies to counteract SCAs, with the aim of either hiding the data or for masking the data so that secret information cannot be retrieved from the side-channel leakages, leading to hiding and masking countermeasures. Each countermeasure is explained briefly below.

2.3.6 Hiding

In the hiding based approach, the processing of the underlying algorithm is attempted to be hidden and it does not change the intermediate data values. This is usually achieved using temporal misalignment or de-synchronization by randomizing the power consumption, which is achieved by changing the time at which sensitive entity is processed. Dual-rail precharge logic cells is one of the popular hiding techniques [56]. Hiding techniques include shuffling the operations at the software level or insertion of dummy instructions [9–11]. Misalignment or hiding countermeasure can be defeated using re-alignment techniques, signal processing techniques or pattern matching [57–59]. Misalignment is one of main motivations that leads to applying deep learning techniques like CNN to side-channel attacks.

Masking

Masking countermeasure is based on the idea of the secret-sharing method, in which a secret is distributed among participants and can only be reconstructed if a sufficient number of participants are willing to collaborate to share their part of the secret. The idea of using the secret-sharing approach to side-channel leakage information was introduced by Chari et al. and Goubin and Patarin [54, 60], in 1999. Since then many different masking techniques have been proposed for various cryptograhic algorithms [12–16, 61, 62]. In masking the countermeasure, the secret variable Z is randomly split across multiple shares $M_1, M_2, ...M_d$, such that $Z = M_1 \circ M_2 \circ ...M_d$, where $\circ$ represents the group operation on $d-1$ random variable, resulting in $(d-1)th$-order masking, masked with one masked variable. Higher-order side-channel Attacks (HOSCA) have been proposed to attack this kind of masking countermeasure in which attack needs d time samples to retrieve the seret information from the countermeasure protected information [48, 49, 63]. In this research, we have also performed ML-SCA on the protected versions (using the masking approach) of cryptographic algorithms.

Residue Number System

Svoboda and Valach proposed Residue number systems (RNS) in 1955 [64] while working on number systems for implementation of fast arithmetic and fault-tolerant computing.

The use of residue number systems (RNS) has greatly increased over past few years due to their ability of processing high-speed operations on long integers. Modern PKC systems rely on the long integer arithmetic, which makes RNS suitable for computations in these systems. RNS is based on the fact that a large integer can be represented by the smaller residue integers, a concept introduced by Chinese mathematician Sun Tsu around 1500 years ago [65]. A brief introduction to RNS is given below.

RNS is a non-positional number system and is defined by a N co-prime psoitive integers, called a moduli set.

$$m = m_0, m_1, ..., m_{N-1} \tag{2.11}$$

In this moduli set, size of each modulus m_i represents the channel width of the RNS. Any positive integer, lets say A, can be represented as $A = a_0, a_1, .., a_{N1}$, where $a_i = (A mod m_i)$, and is in the range $(0, D - 1)$, where D is represented by eq. 2.12.

$$D = \prod_{i=0}^{n-1} m_i \tag{2.12}$$

The arithmetic operations in RNS can be categorized into simple and complex operations. Simple operations refer to addition, subtraction, and multiplication. However, complex operations represent division, modulus, sign detection, and magnitude comparison. Let $A = a_0, a_1, .., a_{N1}$ and $B = b_0, b_1, .., b_{N1}$ represent two integers then the arithmetic simple operation <.> can be performed on A and B by processing all channels concurrently, without carry propagation, using following:

$$C = < a_0 b_0 >_{m_0}, < a_1 b_1 >_{m_1}, ..., < a_{N-1} b_{N-1} >_{m_{N-1}} \tag{2.13}$$

The computation are fast due to the carry-free propagation and concurrent nature of the executions [66]. The concurrent execution of the operations (addition and multiplication) on all the channels makes it an ideal candidate to counteract side-channel attacks in PKC-based systems (RSA and ECC both). The parallel processing hides the sensitive information related to the particular operation in the leakages. However, the leakages can be exploited if the implemented algorithm for RNS operations has inherent branching based on the sensitive information (e.g. if key is '1', perform operation x, otherwise perform operation y). Papachristodoulou et al. have presented practical evaluations of countermeasure-protected RNS based systems using Test Vector Leakage Assessment (TVLA) and template attacks [67].

Chinese remainder theorem (CRT) plays an important role in RNS. It helps in conversion from RNS back to binary representation. [68]. Let D be the dynamic range of RNS on moduli set $m = m_0, m_1, ..., m_{N-1}$, $D_i = D/m_i$ and $D_i^(- 1)$ represents the multiplicative inverse of D_i i.e. $D_I.D_i^(- 1) mod m_i = 1$, and x_i represent the ith value of X in RNS, then CRT is defined as:

$$X = \left\langle \sum_{i=0}^{N-1} \left\langle x_i . D_i^{-1} \right\rangle D_i \right\rangle_D \tag{2.14}$$

2.4 Machine Learning-based Side-Channel Analysis (ML-SCA)

2.4.1 Profiled Attacks

One of the strongest side-channel attacks is the profiling attack. Here, the adversary has access to the cloned open copy of the device under target, called a profiling device. There are two phases; profiling (characterization or training) phase and attack (classification or matching) phase. In the profiling phase, a profile leakage model is estimated based on the obtained information from the device under attack and in the attack phase the unknown secret key is recovered by using the estimated profiled model. Traces during the attack phase are different from the traces in the profiling phase. Template attacks [44] and stochastic model [69] constitute the subset of the profiling attacks.

Suppose an adversary has a cloned copy of the IoT device and is capable of capturing the leakage traces L_i, while encryption E is performed on the device with the key k. Let k (where $k \in K$) represents the fixed cryptographic key and t (where $t \in P$) represents the plaintext or ciphertext of the cryptographic algorithm. Then y can be represented as the mapping of the plaintext or ciphertext t and key $k \in K$ to a value that is assumed to be related to the deterministic part of the measure the leakage L, based on the model being used. Based on this, the measured leakage L can be represented (eq. 2.15) as a function of independent additive noise r and device-specific function ϕ. This multivariate leakages $L = L_1, ...L_N$ (where N represents total number of leakage traces) is exploited in profiling attacks. So, in profiling phase, attacker has N leakage traces ($L = L_{P1}, L_{P2}, ...L_{PN}$), collected by computing encryption using plaintext ($P = t_{P_1}, t_{P_2}, ...t_{P_N}$) and secret key k. In attacking phase, Q traces ($L = L_{Q_1}, L_{Q_2}, ...L_{Q_N}$) collected by using different plaintext and key are used.

$$L = \phi(y(t,k)) + r \tag{2.15}$$

Template attack (TA) is theoretically the most powerful and commonly used side-channel attack. It relies on the Bayes theorem and the assumption that the Leakage estimation function $L|(P,K)$ has multivariate Gaussian distribution which is parameterized by its mean ($\mu_{p,k}$) and covariance ($\sum_{p,k}$), for each template (p,k). TA uses the generative model strategy which is used for classification in machine learning as well.

2.4.2 Application of Machine Learning to Side-Channel Leakages

Machine learning (ML) is a sub-field of Artificial Intelligence (AI), which deals with the study of computer algorithms that aid in automatic learning through experience by utilizing the concepts of statistics. Through statistics, ML techniques can learn the complex patterns in the data. ML-based systems train themselves based on the training data so that they can generalize and predict the class of the unseen data. Machine learning based systems have outperformed in various domains by significantly improving performance. Over the past few years, applicability of machine learning has been accepted as able to efficiently recover the secret information by exploiting the side-channel leakage information from various cryptographic algorithms including ECC, AES RSA, 3DES and lightweight cryptographic algorithms [19, 20, 70, 71]. With the improved efficiency, the ML-SCA seem to surpass the traditional side-channel analysis methods.

ML-SCA is an extension of profiling template attacks in which the adversary first trains a model (profiling phase) and then launches the attack on the actual unknown test traces to recover the secret key (classification phase). Machine learning can be used in supervised, unsupervised or semi-supervised modes. In supervised, unsupervised and semi-supervised learning modes, the model learns from the dataset that consists of labeled leakage traces, unlabelled leakage traces, and leakage traces with and without labels, respectively. ML-SCA based on the power consumption leakages was first introduced by Hospodar et al. [22], followed by a comparative study by Lerman et al. [21].

ML-SCA can be further divided into a 6-step methodology. A generic overview of the ML-SCA methodology is shown in Fig 7.1 and details of each step are given below.

- Data Collection - In the first step, leakage traces L are collected from the target IoT profiling device and a dataset consisting of raw traces is formed.
- Pre-Processing and labelling - In the second step, raw traces are further processed to handle misalignment using filtering and windowing techniques. Pre-processed leakage traces are then labeled for supervised and semi-supervised learning. Each sample point in the leakage dataset represents a feature ($L_{i,f}$) and the label (C) represents the target key class. Each data trace is also referenced as data instance in machine learning.
- Data Splitting - The dataset consisting of the aligned traces is then divided into three datasets; training, validation and testing. Training dataset is a subset of leakage traces dataset L, which are used to train the model. Validation and testing datasets consist of leakage traces which are used for validating the model during training and testing the model after training, respectively. Testing dataset is never shown to the model during training for an accurate efficient fitted model to avoid over-fitting. Over-fitting is a modeling error that occurs when the model closely fits on confined data instances, and fails to generalize on an unseen dataset.
- Feature Engineering - The datasets can be further processed to reduce noise using traditional machine learning feature engineering techniques. Not all sample points in a leakage trace or instance contribute tothe classification of the target class key. The redundant, insignificant features in the dataset might compromise the model's performance.
- Machine Learning based Classification and Hyperparameter tuning- The dataset from previous step is given as an input to the classification model with a machine learning algorithm of choice. Various machine learning algorithms including Support Vector Machine (SVM), Random Forest (RF), Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), Long-Short Term Memory (LSTM), Autoencoders, have been

evaluated against side-channel leakages. There are few hyperparameters for each algorithm which can be further tuned to improve the performance of the trained model.

- Secret Key Prediction - This last stage of the ML-SCA methodology is the actual attack phase. The unseen traces are used to test the model's performance. The best performing model is then used to predict the keys from unseen data traces.

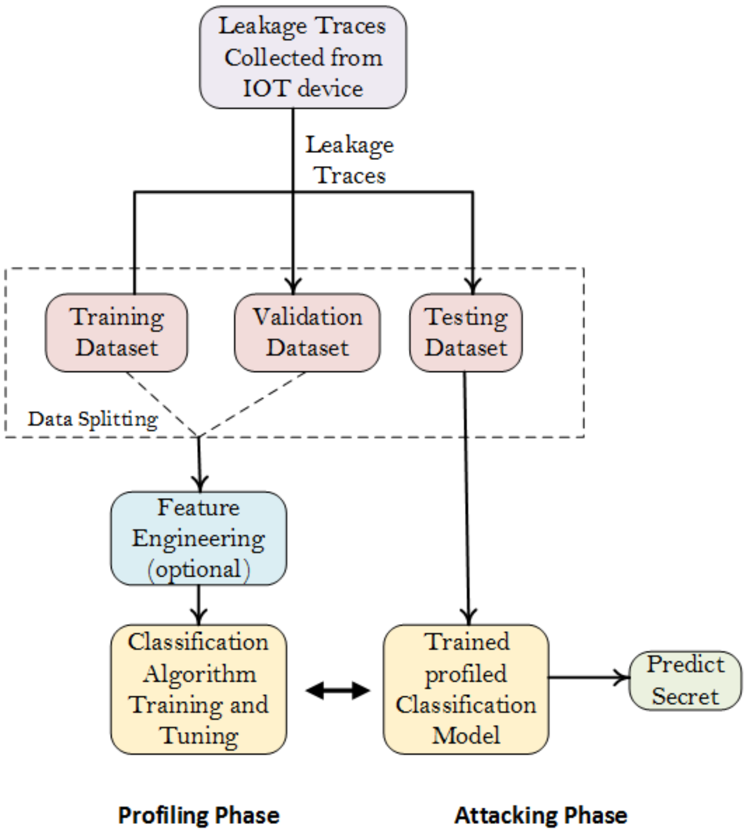

Leakage Traces Collected from IOT device
Leakage Traces
Training Dataset
Validation Dataset
Testing Dataset
Data Splitting
Feature Engineering (optional)
Classification Algorithm Training and Tuning
Trained profiled Classification Model
Predict Secret
Profiling Phase
Attacking Phase

FIGURE 2.6: Machine Learning based Evaluation Methodology

Simple machine learning algorithms including Random Forest (RF), Support Vector Machine (SVM), and Naive Bayes (NB) can successfully recover the secret key information from the implementations. ML-SCA based on the power consumption leakages was introduced by Hospodar et al. [22], followed by a comparative study undertaken by Lerman et al. [21].

2.4.3 Deep Learning and Side-Channel Analysis

Deep learning is a subset of machine learning that is inspired by the biological neural network and is popular due to its ability to self-learn from the data patterns using artificial neural networks without the requirement of pre-processing or feature engineering, which makes it practical from the attacker's perspective. In deep learning neural networks, computational models are built which consist of multiple processing layers including an input layer, a series of hidden layers and an output layer. Most popular neural network include Multi-Layer Perceptrons, Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Gated Recurrent Units (GRUs), Hopfield Network. In ML-SCA context, most commonly used network is CNN [72]. Details of the machine learning algorithms utilized in this research are given in the respective Chapter.

Success of traditional side-channel attacks greatly depends on the selection of the significant features or point of interest (POI), and alignment. However, deep learning-based SCA (DL-SCA) has removed the need for this pre-processing and alignment. The lucrative effortless secret key recovery analysis using deep learning neural networks opened up a new avenue of side-channel research. Maghrebi et al. have documented results for the recovery of secret information using neural networks [19]. Cagli et al. have demonstrated that deep learning-based SCA (based on CNN) can help in neutralizing jitter-based hiding countermeasure without requiring any pre-processing, alignment or feature engineering [23]. Kim et al. proposed a method to add noise to the traces for a robust deep learning-based SCA [73]. The visualization of the features and the hyper parameters can help in building an efficient neural network model [74].

2.4.4 Performance Metrics in ML-SCA

After training a machine learning model, the next step is to assess how effective the model is. There exist various metrics which can be used to evaluate a machine learning-based trained model for binary and multi-class classification problems. These metrics include accuracy, precision, recall, F1-score, area under curve (AUC), and receiver operating characteristic (ROC). Not every metric is suitable for all the datasets under study. For example accuracy might be the best candidate for the balanced dataset. However, it might be misleading for imbalanced datasets (having more instances for one class as compared to the other classes). Below is a brief description of each metric.

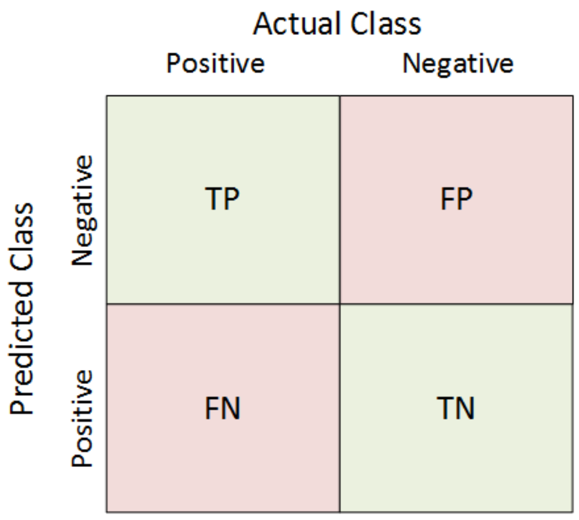


FIGURE 2.7: Confusion Matrix

- Confusion Matrix - Confusion Matrix is not a metric itself but it helps in finding the correctness and other parameters, based on which model can be evaluated. It is intuitive and gives an easy visual description of the experiments. It is a 2-dimensional matrix with each column representing the actual class and each row representing the predicted values. Fig. 2.7 is the graphical representation of a confusion matrix. There are few terms associated with the confusion matrix as described below.

  - True Positives (TP) - TP represent the cases when the actual class of the instance was true and the predicted class is also true.
  - True Negatives (TN) - TN represent the cases when the actual class of the instance was false and the predicted class is also false.
  - False Positives (FP) - FP represent the cases when the actual class of the instance was false and the predicted class is true.
  - False Negative (FN) - FN represent the cases when the actual class of the instance was true and the predicted class is false.

- Accuracy - Accuracy represents the ratio of correct predictions to all the predictions. It is a good performance metric when the target classes in the dataset are balanced. If one class is in the majority then it can lead to inaccurate classification.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.16}$$

- Precision- Precision is a measure which tells how many cases represented as true were actually true. It is the ratio of TP to TP+FP.

$$Precision = \frac{TP}{TP + FP} \tag{2.17}$$

- Recall or Sensitivity- Recalls represents how many cases actually true were predicted by the system as true. It is the ratio of TP to TP+FN.

$$Recall = \frac{TP}{TP + FN} \tag{2.18}$$

- F1-score (F1) - F1-score represents both precision and recall.

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{2.19}$$

- Area Under Curve - Receiver Operating Characteristic (AUC-ROC) - AUC-ROC represents the performance measurement at various thresholds. ROC is a probability curve between true positive rate (TPR) and false positive rate (FPR) while AUC represents degree or measure of separability. The higher the AUC, then the better the model is at classifying and predicting the classes.

Traditionally, to evaluate a side-channel attacks, guessing entropy and success rate are used. Success rate (SR) of order o refers to the probability for the right key candidate to be ranked among the first o candidates. Guessing entropy (GE) represents the rank of the candidate from the guessing vectors, which are acquired as a result of applying a side-channel attack.

2.4.5 Curse of Dimensionality

The high dimensions of the data might affect the estimation of the probability densities of the leakage data samples, known as the curse of dimensionality. It was first introduced in Slepian1961. Side-channel leakage traces have a high number of time samples and are considered to be in D-dimensional space (D represents the number of time samples and usually is greater than 3 for side-channel leakages). Applying machine learning to multi-dimensional data might be resource consuming and might lead to inaccurate classification. To handle this problem, we have proposed to use state-of-the-art dimensionality reduction techniques to reduce the feature sub-space before modelling with machine and deep learning models.

2.4.6 Under-fitting and Over-fitting

The expectation of training with neural network is that the trained model should be able to predict or classify on the unseen data instances. The model is trained with a set of labeled examples and is validated on the unknown examples which are not shown before. Ideally, the trained model should be able to return good accuracy results on the unseen data. However, on some datasets the model might not train successfully and give rise to following two scenarios.

- Under-fitting - It is a situation where the model architecture is so simple that it fails to learn from the training examples, hence results in poor accuracy on the training and test sets both.
- Over-fitting - This is a common phenomena which occurs if the instances (traces in SCA context) are too noisy or if the model is too complex with a lot of parameters. The model learns from the noisy patterns so well that it fails to generalize on the unknown data samples. In case of over-fitting, the training accuracy is very high as compared to validation/test accuracy which is very low.

Various techniques can be used to avoid over-fitting including dropout layers, L1 or L2 regularization, early stopping or data augmentation.

Chapter 3

Secret Key Classification based on Electromagnetic Analysis and Feature Extraction using Machine-learning Approach

This chapter is an adapted version of a published article (Publication I). This chapter presents the proposed feature engineering technique which is used to select the most contributing features. Side-channel leakages consist of high-dimensional complex features, most of which might not be contributing to the classification predication. Selection of the most important contributing features is a challenging task. This chapter presents a solution by reducing the trace length and thus the attack complexity, by using time and frequency-domain signal properties as features instead of using raw leakage signals. Electromagnetic leakages are analyzed to recover the secret key from the symmetric cipher, using various machine learning algorithms. A standalone data collection setup is also presented for acquiring the data.

3.1 Abstract

Despite having a secure algorithm running on a cryptographic chip, in an embedded system device on the network, secret private data is still vulnerable due to Side-Channel leakage information. In this paper, we have focused on retrieving secret-key information obtained from one of the Side Channels, namely Electromagnetic radiation signals. We have captured

leaked Electromagnetic signals from a Kintex-7 FPGA, while AES is running over it, and analyzed them using machine and deep-learning based algorithms to classify each bit of the key. Moreover, we aim to analyze the effect of having different signal properties as features in these classification algorithms. The results will help in defining which features give maximum information about the captured signal, hence leading to key recovery.

3.2 Introduction

Using Side-Channel analysis to recover key, goes back to early 90s when a group of researchers proposed a method of using the Side-Channel leakage to recover secret information [17] [3]. Following the trend, Mulder et al. presented analysis of Electromagnetic radiations emitting out of FPGA to get information about the secret used for encryption [37].

Electromagnetic radiations from circuits due to magnetic fields produced by electric currents. The captured EM radiations are then analyzed to look for secret-information retrieval. Power signals and electromagnetic signal leakage can cause a great risk to secret information, however the later is a variant of the former. Over the last decade, the research focus was on the power-analysis attack as it is convenient to launch, though not practical in all scenarios [75]. On the other hand, Electromagnetic attacks are non-invasive and more practical, with the right probes for signal capturing and the correct analysis for key recovery. De Mulder has shown a way of capturing the EM Radiations from FPGAs, processing and analyzing them using mathematical and statistical models to recover a secret key [37]. The defined process works offline and can be time consuming for key retrieval. Similar work is shown by the authors in [7]. Machine learning can help in fast information extraction, based on the classification models being used. Different classification models have been tried and tested for Power analysis signals but not much analysis exists for Electromagnetic signal analysis [76]. Lerman et al. have worked on the key recovery from AES using machine learning by capturing the power signals emitted out of the device [77] [78]. In addition to embedded systems, Genkin et al. found a way of attacking mobile phones using EM analysis [79]. Moreover, neural network based classifiers have been tested for Side-Channel leakage from hardware systems [19]. Our first contribution is to set up a system which is used to capture the EM radiations from a Kintex-7 FPGA while AES is running on the chip (as AES data for Kintex-7 does not exist), secondly we have used the signal properties as features to be fed to classification algorithms, and finally we have analyzed the signals using machine learning and neural network classification techniques (with different signal properties) to classify and recover each secret key bit. Our aim is to find which features (based on EM signal properties) or combinations of features can help in better key bit classification.

The rest of the paper is organized as follows, Section 2 explains our methodology for key recovery using classification and outlines the properties of a signal used for feature formation along with feature selection and extraction methods; this section also explains the classification techniques used, Section 3 explains the experimental setup and Section 4 gives the results of analysis while Section 5 concludes the paper.

3.3 Methodology

The purpose of this research is to capture and analyze the EM radiations out of the FPGA, while AES is encrypting data with a secret key. The analysis is carried out using Machine learning and deep-learning based classifiers rather than traditional statistical methods. To use the classification algorithms, the machine needs to be trained with a set of data. This set of data (EM radiations) is obtained from Kintex-7, mounted over a Sakura-X board. Sakura-X is a series of specialised boards designed to evaluate the algorithm implementation on FPGA, against Power Analysis, EM Analysis and Fault Injection Attacks [80]. The captured raw datasets are then processed to form feature datasets, which are created by using signal properties. Nine signal properties are measured and different combinations of these properties are used as features for training the learning machine, after filtering and evaluating the feature sets using nine standard evaluators/selectors. Reason for using the evaluators is to screen the features to overcome the problem of over-fitting, hence reducing the training time and diminishing the chances of miss-classification. The larger the dataset, the greater are the chances of inaccurate classification. For testing the trained system, another featureset

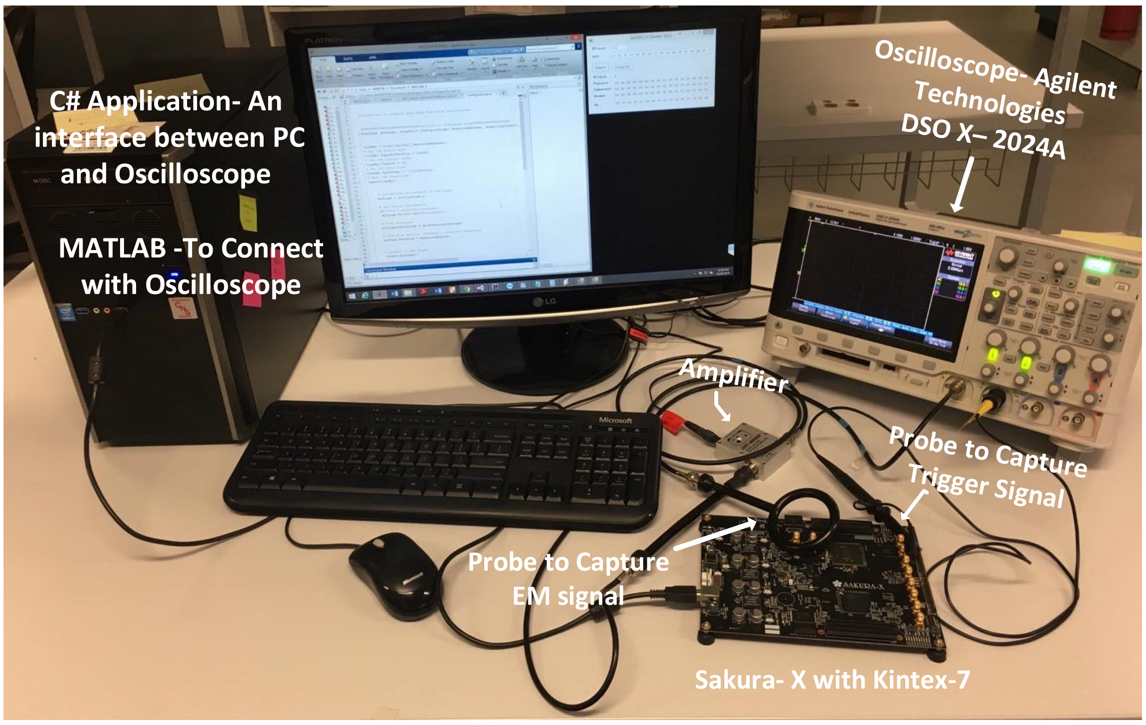

C# Application- An interface between PC and Oscilloscope
MATLAB -To Connect with Oscilloscope
Oscilloscope- Agilent Technologies DSO X– 2024A
Amplifier
Probe to Capture Trigger Signal
Probe to Capture EM signal
Sakura- X with Kintex-7

FIGURE 3.1: Setup for Electromagnetic Analysis

is formed based on the same methodology with a different secret key. The hardware setup used for acquisition of raw datasets is shown in Fig. 3.1.

3.3.1 Advance Encryption Algorithm

The algorithm under test is the Advance Encryption Standard (AES) which is NIST standard for secure communications [81]. It consists of four blocks (Sboxes, ShiftRows, MixColumns and AddRoundKey) for encrypting data, using three different key sizes, i.e. 128, 192 and 256-bit keys having 10, 12 and 14 rounds respectively. All rounds are the same except for the last one which lacks addroundkey block, which makes it vulnerable to Side-Channel attack. Side-Channel attacks can be categorized into Divide-and-Conquer and Analytic attacks [26], [27]. In Analytic attacks, a complete sub-key is recovered using mathematical equations, while in Divide-and-Conquer attacks, only a part of the key is recovered. We will be following the latter approach [78]. To recover the key, one byte is selected at a time and in each byte a single bit is targeted as shown in Fig. 4.1. For a single byte 256 combinations exist. To mark and classify the samples as ’1’ or ’0’, relevant bit location samples are segregated in

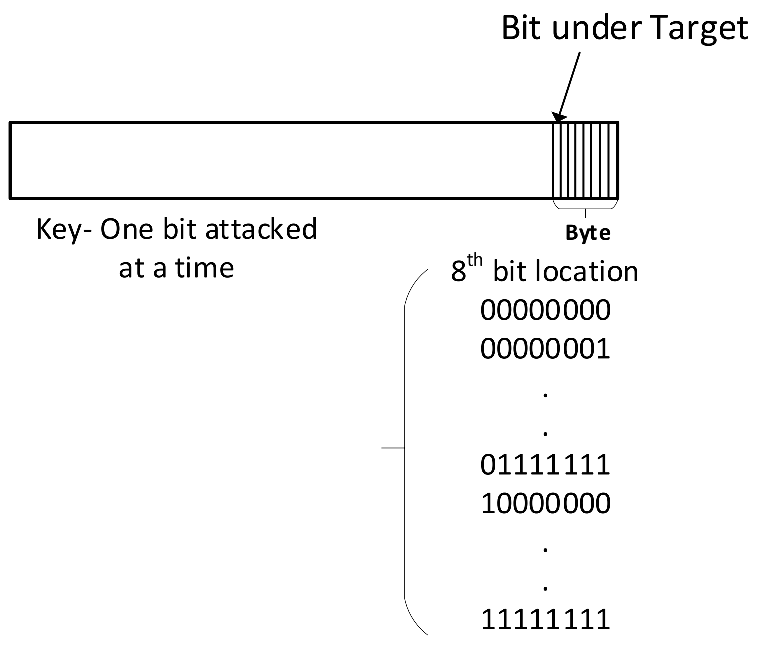


FIGURE 3.2: 256 possible combinations for each bit location

the form of a group, e.g. to target MSB of the last byte, collected samples having the MSB as 1 are classified as '1' while others are marked as '0'. We have collected 100 samples (each sample encrypted with random plaintext) for each possible combination of the fixed key (i.e. total 256 combinations), leading to a total of 100*256 samples as shown in Fig. 3.5. 51200 samples are then processed using MATLAB to calculate the signal properties (explained in section 2.3). The resulting output datasets are then processed to form feature sets (using Java code) for input to the classification algorithm. Fig. 3.4 shows the preparation process of the input datasets to be fed to classification algorithms (for the training phase), for the most significant bit of the byte under examination, after forming feature sets based on the signal properties. As mentioned before, one aim of this research is to find the features or combination of features which can produce better results for the secret key bit classification (used for this study as mentioned in Section 2.2), which has not been analyzed before in the literature for leaked Electromagnetic Radiation.

3.3.2 Classification Algorithms

Three main classification algorithms have been used for analysis, two machine-learning and one deep-learning based algorithm.

Random Forest

Random Forest is a type of supervised machine-learning classification algorithm in which a number of trees are built during the training process and the classification mode is determined during the testing phase. Random trees use the feature-bagging scheme to build the trees. Details of the algorithm can be found in [82].

Naive Bayes

Naive Bayes is a class of supervised learning algorithm, based on Bayes' theorem. It works on a probability model built on the probabilities of outcomes and reveals the uncertainty of the model [83].

MultiLayer Perceptron (MLP)

A multilayer Perceptron is a supervised artificial neural network consisting of three or more layers - one input layer, one output layer and two or more hidden layers. Each node in a hidden layer acts as a neuron which works on a nonlinear activation function. MLP is different from a linear perception because of its multiple layers and non-linear activation functions. MLP is best for solving complex problems stochastically.

3.3.3 Properties used as Features

Captured EM signals are subjected to analysis by the above mentioned machine-learning algorithms, based on signal properties (frequency and time domain) as given below.

- Mean of Absolute Value (MAV) - For MAV, the mean of all signals is calculated.
- Slope Sign Change (SSC) - In a signal, slope sign changes are recorded against a pre-determined threshold.

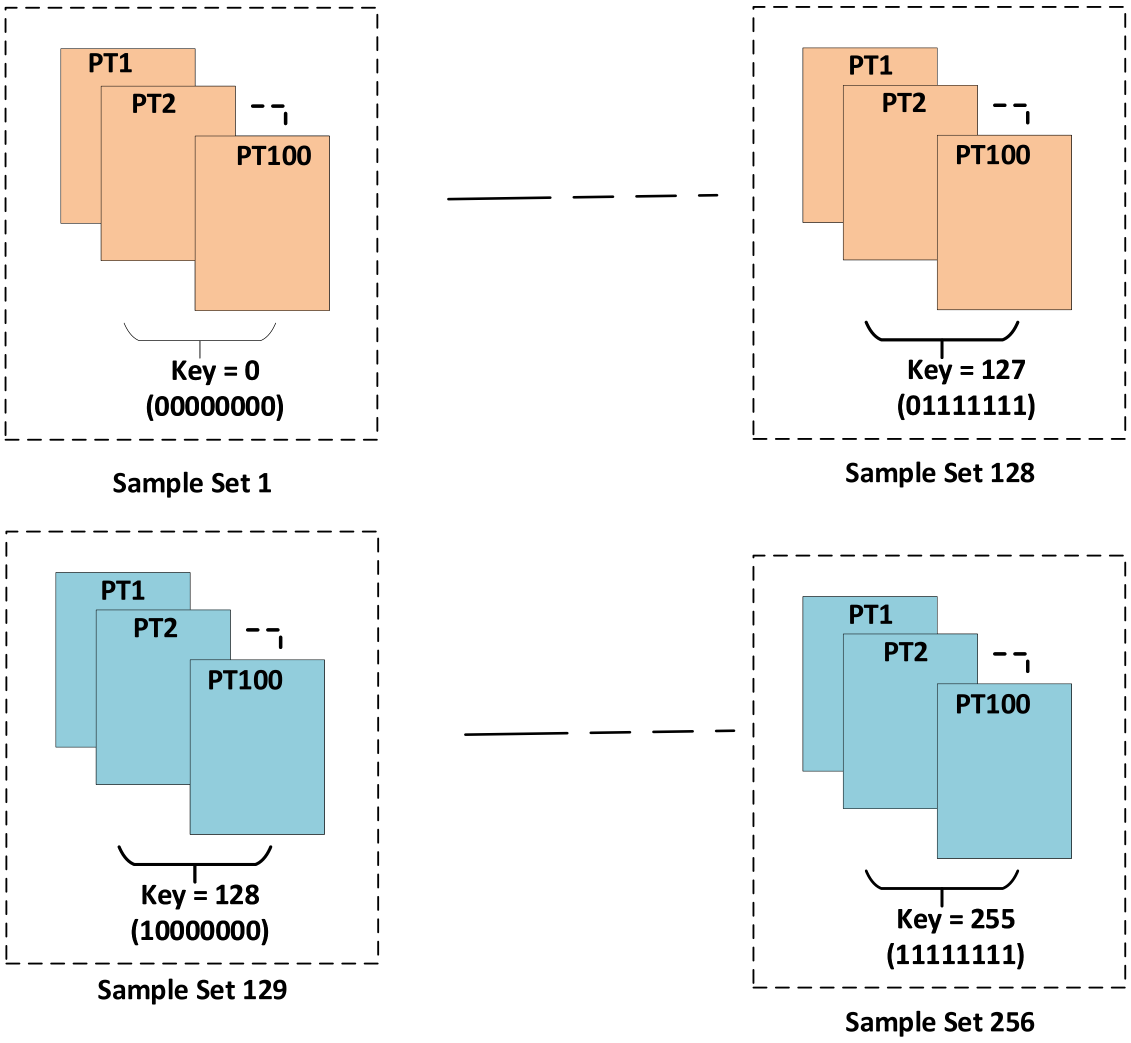


FIGURE 3.3: Sample Set Formation for a Single bit location in a Byte

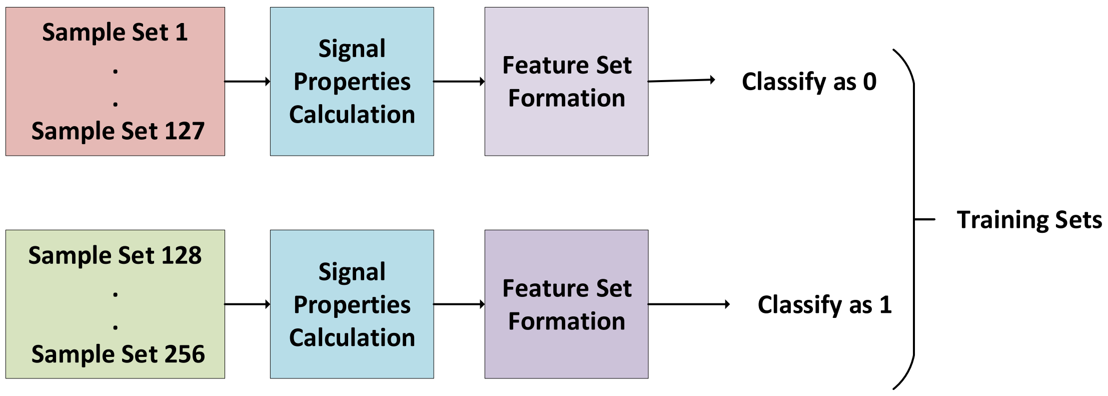


FIGURE 3.4: Process of preparing Training and Testing Data for Classification

- Sum of Squares (SSI) - In a signal, the sum of the squares of the values is calculated.
- Zero Count (ZC) - The number of times the signal crosses zero is calculated.
- Kurtosis - The sharpness of the peak of a frequency-distribution curve is noted.
- Median PSD (FMD) - The median of a distribution, in the frequency domain, is calculated.
- Mean PSD (FMN) – The mean of a distribution, is recorded.
- Frequency Ratio (FR) – The ratio of the lowest to the highest frequency is calculated.
- Median Amplitude Spectrum (MFMD) – For signals, the median amplitude spectrum is calculated.

3.3.4 Feature/Attribute Selection and Extraction

Feature Engineering is an important task when it comes to the problem of over-fitting in machine-learning classification. It helps in reducing/rearranging, by selecting/extracting those features, which can give the best results. Having too many features can lead to miss-classification. There are two main concepts in feature engineering, used for analysis in this paper and given below.

- Feature Selection: In feature selection, a subset of features is selected from the available pool of feature data.
- Feature Extraction: In feature extraction, a new set of features is formed from existing sets of features.

For our analysis using supervised classification techniques, we need to have defined features from the raw set of data signals captured. We have defined the features based on the signal properties as mentioned in the previous section. Now, from the available set of feature data, we need to form a usable set of features in a dataset on which classification techniques can be applied for training and testing of data, to determine which features will give the best classification results. Below are the feature selection/extraction algorithms used.

Learner-based feature Selection - LBS

In this technique, a generic yet powerful algorithm is selected to analyze the performance of the algorithm under test, with subsets of datasets. The subset which performed the best is selected for further analysis. Generally, a decision tree is used as the algorithm.

Chi-Square

Chi-Square is a statistical test which measures the dependency of features on the output variable. If a dependency exists then the features are selected, otherwise they are discarded.

Correlation-based Feature Selection

The correlation between the attributes and the output variables is calculated using Pearson's correlation co-efficient function given in eq. (3.1). Attributes having a high correlation (close to -1 or 1) are selected.

$$\text{Correlation} = \frac{\mu_i(1) - \mu_i(0)}{\sigma_i(1) + \sigma_i(0)} \tag{3.1}$$

$\mu$and σ represent the mean and standard deviation of the features, with respect to class 0 and class 1.

Gain Ratio

Eq. (6.1) can be used for gain-ratio calculation of features based on class.

$$\begin{aligned} Gain(C, A) = H(C) - H(C|A)/H(A) \\ H(C) = EntropyofClass \\ H(A) = EntropyofAttribute \\ H(C|A) = EntropyofClassgivenAttribute \end{aligned} \tag{3.2}$$

Information-Gain based Feature Selection

For the output variable, information gain or entropy is calculated for each possible feature. Scores/ranks are assigned to the features based on the information contribution towards the output variable. To evaluate it with Weka, ranker is selected. Eq. (6.2) is used to calculate the entropy.

$$InformationGain(C, A) = H(C) - H(C|A) \tag{3.3}$$

One-Rule Attribute Evaluation - OneR

As the name implies, one-rule attribute evaluation means to have one rule set for all predictors. Calculate errors for each predictor, based on frequency table, and select the one which shows least error.

Principal Components

In Principal components, a subset of the data (linearly uncorrelated) is formed based on the original feature set, whose data is correlated. The newly formed variables are known as principal components. It is a kind of data extraction not selection, because a new linearly independent subset is formed.

Relief

This method is based on a feature-weighting approach [84]. A target sample is selected and then the relevance of the features in the neighborhood of that sample are measured. The samples are marked as ‘hit’ and ‘miss’, if they belong to the same category as the target sample or to a different category, respectively. After marking, the distance from all hits and misses is calculated for the target sample, and is used as the weight of a target feature.

Symmetric Uncertainty

Ideally, the information gain calculated in Eq. (6.2) should be symmetric, i.e. the information gained about Y while observing X is same as the amount of information gained while observing Y. Unfortunately, that is not the case, as it is biased towards features with higher values. Moreover, the correlation measured among different features should be normalized and comparable. To handle the information gains' biased behavior towards attributes, the symmetric uncertainty is calculated, which brings out an unbiased response with normalized values in the range of [0, 1]. Equation (6.3) gives the formula for calculating uncertainty.

$$SymmetricUncertainty = \frac{H(C) - H(C|A)}{(H(C) + H(A)} \tag{3.4}$$

3.4 Experiments

3.4.1 Step 1 - Hardware Experimental Setup

To conduct our experiment, we have captured the EM radiation (shown in Fig. 3.5) out of the FPGA (Kintex-7), mounted over SAKURA-X and operating at 200 MHz, while AES is running on it. During the encryption process after Sbox, ShiftRows, MixCoulmns and AddroundKey, samples are taken using a KeySight Agilent Oscilloscope. We have targeted one byte of the key at a time, so each bit is classified as '0' or '1'. For each bit classification, we have acquired 100 samples, each consisting of 10k points, for all possible 256 values. Samples are collected using MATLAB and C# platforms. The C# application acts as an interface between the FPGA board (Sakura-X) and the Oscilloscope, which is configured and operated using MATLAB libraries in C#. This gives an automated stand-alone application for the data collection process without frequent involvement of the user, the GUI is shown in Fig. 3.6. The application is a modified automated version of the one provided by SAKURA [80].

3.4.2 Step 2 -Datasets Formation

Once samples are obtained, then features (properties) are calculated using MATLAB customized code. After having a defined set of features, combinations of different features are formed using a Java snippet, written using Weka Libraries [85]. Combinations of features used for analysis are shown in Table 3.1.

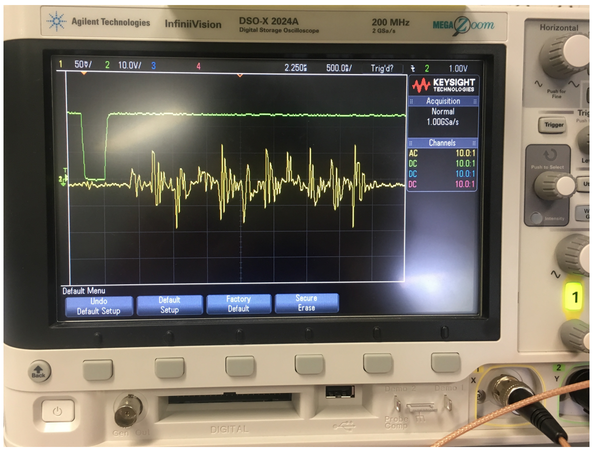
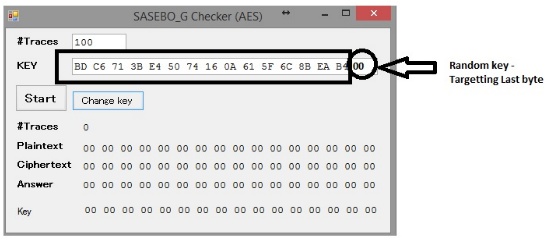


FIGURE 3.5: Captured Electromagnetic Radiation emitting out while AES is running on SAKURA-X



```
#Traces     100
KEY         BD C6 71 3B E4 50 74 16 0A 61 5F 6C 8B EA B4 00
#Traces     0
Plaintext   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Ciphertext  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Answer      00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Key         00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

FIGURE 3.6: Application GUI- Start of App

3.4.3 Step 3 - Analysis

The feature sets formed are then subjected to filtering using feature extraction and selection to reduce the number of features. As our target is to test different features and combinations of features with three classification algorithms, so these features' dataset files are used as input to the algorithms, mentioned in Section 2.2, for the training phase.

3.5 Results

At first, the classification accuracy is calculated for all feature sets combinations, as given in Table 3.1, without using any feature extractor or selectors, for three classifiers (Random Forest, Naive Bayes and MLP). After that, the classification accuracies are calculated for all the feature combination sets using the feature selectors/extractors, and then the difference

TABLE 3.1: Combinations of feature sets

Combination Of Feature	***Features***
Comb-1	MAV
Comb-2	SSC
Comb-3	SSI
Comb-4	ZC
Comb-5	KURTOSIS
Comb-6	FMD
Comb-7	FMN
Comb-8	FR
Comb-9	MFMD
Comb-10	ZC, KURTOSIS, FMD, FMN, FR,MFMD
Comb-11	MAV, SSC, MFMD
Comb-12	MAV, SSC, SSI, FR

of accuracies is calculated, to see how much improvement occurred using the newly formed feature sets. It is worth noting that the comparisons are relative and are not based on the best classification algorithm. Our target is to deduce from the analysis which features can improve the results. Fig. 3.7 show results for the analyzed data.

3.5.1 Random Forest (RF)

Varying trends are seen for feature evaluation with Random Forest, as shown in Fig. 3.7. It is observed that Principal Component Analysis, when applied to Comb-4, gives the best results by increasing the accuracy. Principal Component Analysis performs poorly for Comb-2, Comb-5, and Comb-10. The accuracy gain for Symmetrical Uncertainty and OneR remains almost the same as that of the applied classifier, without any specific features selected. It can be seen that LBS showed decreased accuracy for all feature combinations, however for Comb-4 and Comb-5 the performance is very poor. The gain ratio specifically showed good results for Comb-8. Correlation evaluation, Chi-Square and Information gain exhibits varying trends and gave good accuracy for Comb-2. Overall, it can be seen that every feature combination gave improved results for Comb-12.

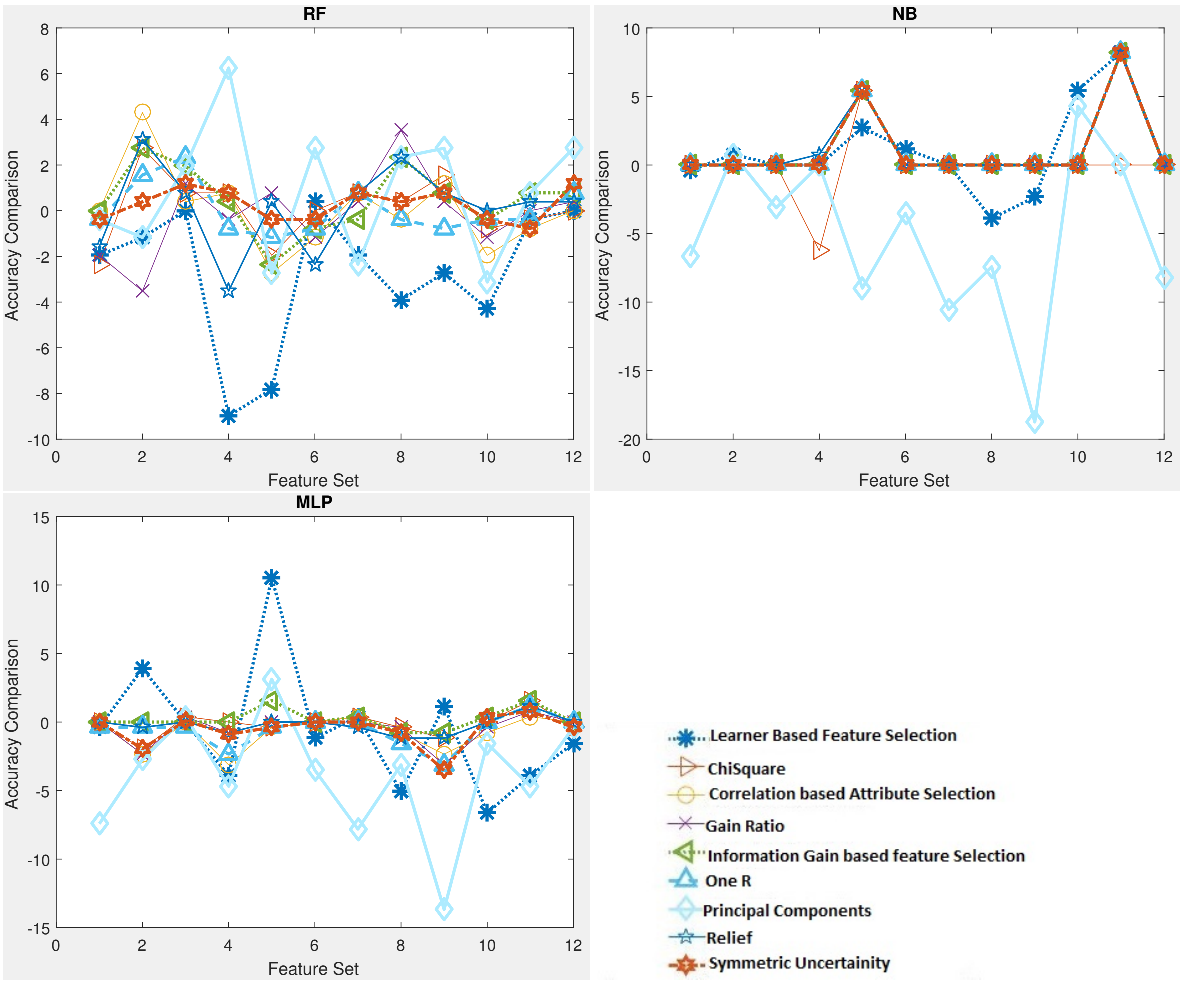


FIGURE 3.7: Accuracies with 100 traces per key bit

3.5.2 Naive Bayes

The results for Naive Bayes are shown in Fig. 3.7. It can be seen that there is not much variation in the output accuracies for all combinations. Principal Component Analysis, particularly, performed poorly in all cases, especially for Comb-9. Chi-square accuracy decreased by 7% for Comb-4. All feature extractors and selectors didn't show any improvement at all except for Comb-4 and Comb-11. It can be stated that, for Naive bayes, a combination of MAV, SSC, MFMD is a good choice of features.

3.5.3 MultiLayer Perceptron

For MLP, variations are seen just like the Random Forest case. Almost all features extractors and selectors behaved in a similar fashion, with insignificant accuracy gain. However, LBS showed surprisingly better performance for Comb-5 and decreased efficiency for the rest of them. Principal Component Analysis decreased the accuracy for Comb-9 by 14%. The accuracies of Comb-1, Comb-11 and Comb-12 are 90.6%, 91.4% and 91.4% respectively. With MLP, MAV alone, the combination of MAV, SSC, MFMD, and the combination of MAV, SSC, SSI, FR are recommended combinations of features.

3.6 Conclusion

In retrospect, after analyzing the results on the EM radiations obtained from the Kintex-7, we can conclude that, for different classification algorithms, the choice of features or combination of features would be different. For Random Forest, features MAV, combination of MAV, SSC, SSI, FR can be used along with Principal Components. However, for Naive Bayes, MAV and a combination of MAV, SSC, MFMD is best choice, if used with Symmetric Uncertainty and Information Gain. For MLP, MAV alone, the combination of MAV, SSC, MFMD, and the combination of Mav, SSC, SSI,FR are the recommended sets of features. The overall trend shows that a combination of time and frequency-domain features gives better performance for secret-key estimation.

Chapter 4

Hyper-parameter Optimization for Machine-Learning based Electromagnetic Side-Channel Analysis

This chapter is an adapted version of a published article (Publication II). It advances the work presented in the previous chapter and performs hyperparameter optimization to select the best trained model for efficient machine learning-based side-channel analysis. The optimization is performed for the symmetric cipher AES.

4.1 Abstract

Side-channel attacks are the class of attacks which exploits the physical leakages of the system to recover the secret key, based on the weakness induced due to implementation of algorithms on embedded systems. AES is mathematically secure but side-channel information can lead to key recovery. Over the last decade, machine learning has been introduced in parallel with traditional statistical side-channel analysis methods. Accurate classification using machine-learning-based approaches critically depends on various factors including, the precision of the input datasets which consist of the features, tuning of different parameters for that particular algorithm, per feature sample length, number of validation folds and feature extraction/selection methods. For analysis of leaked signals in this study, hyper-parameter tuning is carried out on the feature datasets formed on the basis of the time-domain and frequency-domain properties of the signals. The results provide the comparative analysis of

the best choices and lead to concrete selection of the parameters.

4.2 Introduction

Side-channel analysis (SCA) is a great threat to embedded systems. SCA includes a number of attacks based on the type of side-channel information being leaked, e.g. power consumed signals, electromagnetic radiations, timing information categorized on the basis of exploitation of the power consumed by the system, EM radiations from the system and timing of the processes, respectively. Various power analysis methods based on statistical and machine learning analysis have proven to be successful against practical systems having AES as primitive. However, EM analysis based on machine learning still needs to be considered in more depth. Moreover, there are very few online repositories for datasets, covering a limited number of platforms, like DPAContest[24]. Our research focus is on collecting leaked EM data while AES is running on Kintex-7, and performing analysis for key bit classification by exploiting the weaknesses using state-of-the-art machine-learning classification algorithms. Data input to these machine-learning algorithm is of paramount importance. It is the deciding factor for accurate classification accuracy. Wrong input or too large a dataset can lead to problems of under-fitting or over-fitting, hence resulting in inaccurate classification. To overcome the problems of under-fitting and over-fitting, it is essential that the best-fitted pre-processing and features extraction/selection mechanisms should be followed for the data-set under investigation. Various pre-processing techniques are recommended and used to shape target primitive input data to machine-learning algorithms, and most prominent of all is Principal Component Analysis (PCA) [86] [87] [76] [88]. Based on the fact that correct features selection can improve the classification accuracy, we have tried a few different selected features, out of captured raw electromagnetic signal samples, as an input to a classification algorithm. New features reveal different interesting facts as discussed further below. Moreover, the experiments are performed by changing the parameters for tested algorithms, to find the best suitable settings for better accuracy.

The rest of the paper is organized as follows. Section 2 discusses the existing literature related to key recovery of AES using SCA, Section 3 gives an overview of AES and briefs the techniques being used along with the classification algorithms, Section 4 discusses the experimental setup and data organization, Section 5 discusses the results and Section 6 concludes the paper.

4.3 Existing Literature

Side-channel attack was first introduced by Paul Kocher in 1996 [75]. Following his findings, the research community started unveiling other side-channel leakages which can be exploited to recover the key; to name a few, power being consumed by the system, EM radiations being emitted out, acoustic vibrations of components, timing of processes [7, 17, 37, 89]. Out of all these, the most promising are power analysis and electromagnetic attacks. They are based on the fact that a different amount of power is consumed when crypto operations are performed and Electromagnetic radiations will be emitted due to the electromagnetic field produced by a flowing current. Both these attacks can be categorized into Simple and differential attacks. In the simple analysis version, by merely looking at the signal waveform on an oscilloscope, one can deduce the operations performed with specific key bits [40] [90]. However, differential analysis involves complex statistical analysis, and proved to be successful for a number of embedded systems (FPGAs, Processors) for different primitives like AES and ECC [37]. With the advent of the modern machine-learning era, researchers came up with the idea of mapping machine learning onto SCA, as a template attack can easily be visualized as a machine-learning process [18]. In addition to simple machine learning, neural-network-based algorithms have been proposed to retrieve the symmetric key from crypto systems [19] [20].

For AES-128, the best known attack is exhaustive search, having a complexity of 2^{128}. One of the factors on which the security of AES depends is number of rounds. Schneier has recommended to use 16 rounds for AES-128 bit [91]. However, researchers have used modern state-of-art machine-learning classification algorithms to attack all variants of AES[21, 78]. Raw power signals have been analyzed for signs of vulnerability which can lead to key

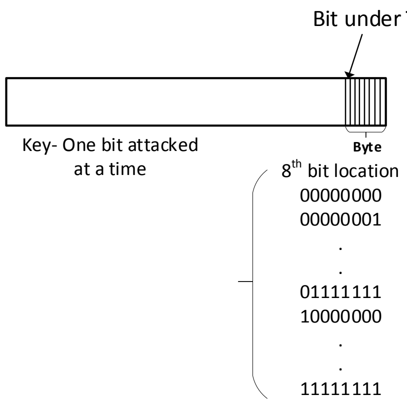


FIGURE 4.1: Byte Under Attack

recovery. It is suggested to use SVM and Random Forest for AES key bit classification from the data-set available at the DPA-Contest site [24]. Existing data-sets are limited to a few platforms and there exists no data for FPGA (Kintex-7). The security of AES on hardware FPGA platforms needs to be evaluated. The focus of this work is to collect and analyze AES EM leaked data from Kintex-7, using machine-learning techniques.

4.4 Attack and Analysis Methodology

The primitive under testing is the Advanced Encryption Standard (AES), standardized by NIST in 2001. It is the most widely used block cipher in embedded systems, to provide security, having three key sizes, 128-bit, 192-bit and 256-bit and 10, 12 and 14 rounds respectively. During each round, four operations of Substitution, Permutation, Mixcolumns and Addition are performed, except for the last round which doesn't have Mixcolumns.

Data samples are formed on similar lines as suggested by Lerman et al. in [21] and in our previous work [92] [93]. For attack purposes, one bit is attacked at a time from a particular target byte. Out of 128 key bits, 120 bits are selected at random and it stays same for the the rest of the experiment, however the last 8 bits are the target, as shown in Fig. 4.1. For the target bit location, data samples having the same key-bit value are grouped together, e.g. let's say that the target bit is the most significant 8th bit, then all data samples for which the 8th bit is '1' are marked as 'class 1' and all key values which have the 8th bit as '0' are

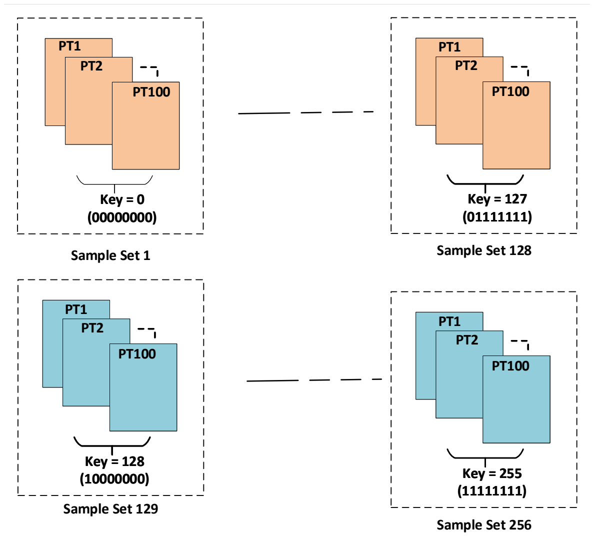


FIGURE 4.2: Sample Formation

marked as 'class0'. In total there will be 128 bits for each. Fig. 4.2 shows an overview of the data samples formation process. For grouped data samples, features are calculated based on signal properties. For instance, the following properties are treated as features.

- Time-Domain Properties
  - Mean (M)
  - Kurtosis (K)
- Frequency-domain properties
  - Median Amplitude Spectrum (MAS)
  - Median of Distribution (MedDis)
  - Mean of Distribution (Mdis)
  - Frequency Ratio of Distribution (FR)

These features are calculated for each collected sample, and samples are collated for further analysis using machine learning classification algorithms. Combination of time-domain features (Mean and Kurtosis) labelled as 'TDF' and all the mentioned frequency domain features labelled as 'FDF' are also tested to analyze if a combination of features can contribute better towards enhanced accuracy. For some of the cases, certain attribute selection/extraction methods are used on the feature datasets to analyze if the performance accuracy can be improved.

4.4.1 Classification Algorithms

Four algorithms are used to analyze the accuracy of key classification; three machine-learning supervised algorithms and one deep-learning artificial-neural-network based algorithm.

Random Forest

Random forest is a class of supervised learning algorithms which doesn't require tuning of hyper-parameters and still gives accurate classification. The algorithm is based on bagging of decision trees and forms a forest of random decision trees merged together to give better prediction [82]. Nodes split and a tree is formed using the features from data-sets. With increasing tree level, during node splitting, the model selects the best features from the random subset of features rather than selecting the best feature from the previous node, which helps in achieving diversity. That is why, it gives best accuracy in most of the cases due to the refined results. As the model grows, it calculates the feature importance while selecting the feature at node splitting which helps in rectifying the problem of over-fitting, as useless features are removed based on a feature importance score.

Naive Bayes

Naive Bayes is also a supervised-learning algorithm based on the Bayesian theorem, suitable for high-dimensional data [83]. It is based on the assumption that features are independent of each other, i.e. each property independently contributes towards the classification, and that is why it is known as Naive. Mostly, features combine together and are dependent on each other to predict the class. However, Naive Bayes performs best for large sets of input data. Posterior probability is calculated using Bayes theorem that helps in finding the odds, i.e. finding the probability for feature 'f' that it belongs to class 'c'. Equation (4.1) is used for classifying the posterior probability using Naive Bayes.

$$\begin{aligned} P(c|f) &= P(f|c)P(c)/P(f) \\ & where \\ P(c|f) &= Posterior Probability given feature f \\ P(c) &= Class Prior Probability \\ P(f) &= Predictor Prior Probability \\ P(x|f) &= Likelihood \end{aligned} \tag{4.1}$$

Support Vector Machine

The Support Vector Machine (SVM) also belongs to the supervised-learning category. It is mostly used for classification problems, however it can be used for regression as well due to its linear classification ability. SVM maps and represents the feature points in space so that a clear hyper-plane can be modeled to separate the classes.

Multilayer Perceptron

Multilayer Perceptron is a class of feed-forward neural network, using 'backpropagation' for training, which is a supervised learning technique. MLP consists of an input, output and hidden layers. Hidden layers can be more than one. It is a fully connected network with layers having specific weights 'w' and neurons having a linear activation function which maps the weighted inputs to outputs. These weight values are adjusted based on the output errors as compared to the expected value, and is achieved through back prorogation. The output error is represented using equation 6.1, where $e_i(n)$ is the error produced at node 'i' in nth data point, d is expected value at node i and y is the value produced by the perceptron.

$$e_i(n) = d_i(n) - y_i(n) \tag{4.2}$$

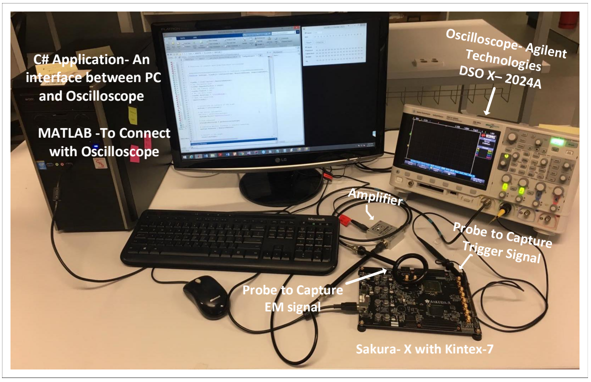

C# Application- An interface between PC and Oscilloscope
MATLAB -To Connect with Oscilloscope
Oscilloscope- Agilent Technologies DSO X- 2024A
Amplifier
Probe to Capture Trigger Signal
Probe to Capture EM signal
Sakura- X with Kintex-7

FIGURE 4.3: Hardware Setup for Data Collection

Equation 6.2 gives the function calculation on one hidden layer with weights 'w', bias vectors b(1) and b(2), and activation functions G and s.

$$f(x) = G(b^{(2)} + w^{(2)}(s(b^{(1)} + w^{(1)}x))) \tag{4.3}$$

Validation

While training a model using a machine-learning classification algorithm, it is important to validate the model to check if the model got pattern correctly and there doesn't exist any bias. To achieve this, K-fold cross-validation mechanisms are used for our analysis. In K-fold cross validation, the input datasets are divided into training and testing datasets. Model is trained using the training portion of the data and the error is estimated for the testing portion to see how well the model performed. This is known as the holdout method. In K-fold validation, hold out is performed k times, such that k-1 subsets are used for training and one subset is used for validation. This is repeated k times to get the better validation accuracy.

4.5 Experimental Setup

For our experiments, raw EM radiation data samples are collected using a Keysight Agilent Oscilloscope, while AES was running on a Kintex-7 FPGA [80]. For each bit value, 100 traces are collected, each consisting of 10k sample points, for 100 random plain-texts. The

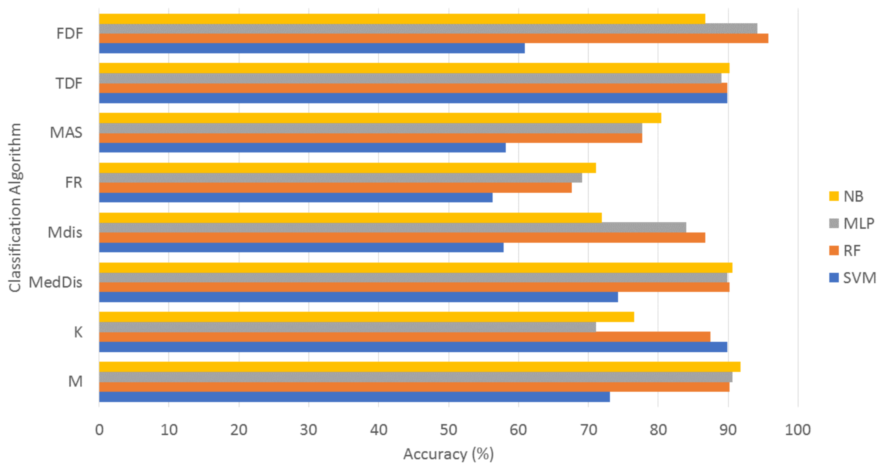


FIGURE 4.4: Comparison of all features for four classification Algorithms

oscilloscope is configured using MATLAB libraries in C#, which give an automated setup for data collection. Fig. 4.3 shows the hardware setup for data collection. The collected data samples are then processed using Java snippet and features are calculated using MATLAB, to transform the raw signals data into a feature-instances data format suitable for machine-learning algorithms. For machine-learning classification, Weka is used[85].

4.6 RESULTS AND ANALYSIS

Analysis is divided into four phases, each explained below along with the results.

4.6.1 Phase-1

For phase-1 analysis, accuracy is calculated for eight different feature-sets for four classification algorithms namely Naive Bayes, Random Forest, Multilayer Perceptron and Support Vector Machine, using 10-fold validation. Fig. 4.4 shows the resulting accuracy. It can be seen that, out of all the different feature-sets, a combination of frequency domain features (FDF) produced better results for all four classification algorithms.

4.6.2 Phase-2

For phase-2, as observed in previous literature for AES and DES, a SVM works better if the input data is pre-processed with PCA. For our AES data collected from Kintex-7, we have analyzed the pre-processing effect on classification accuracy. We have analyzed the effect of Principal Component Analysis and Chi-Square. It has been observed that use of principal components as pre-processing has introduced a significant increase in SVM accuracy because there are redundant features in data-set which do not contribute to classification while using SVM, as shown in Fig. 4.5. PCA did not show any significant improvement using any other classification algorithms. This is mainly due to the nature of the other three classification algorithms. Moreover, none of the algorithms showed a significant increase in accuracy using Chi-Square.

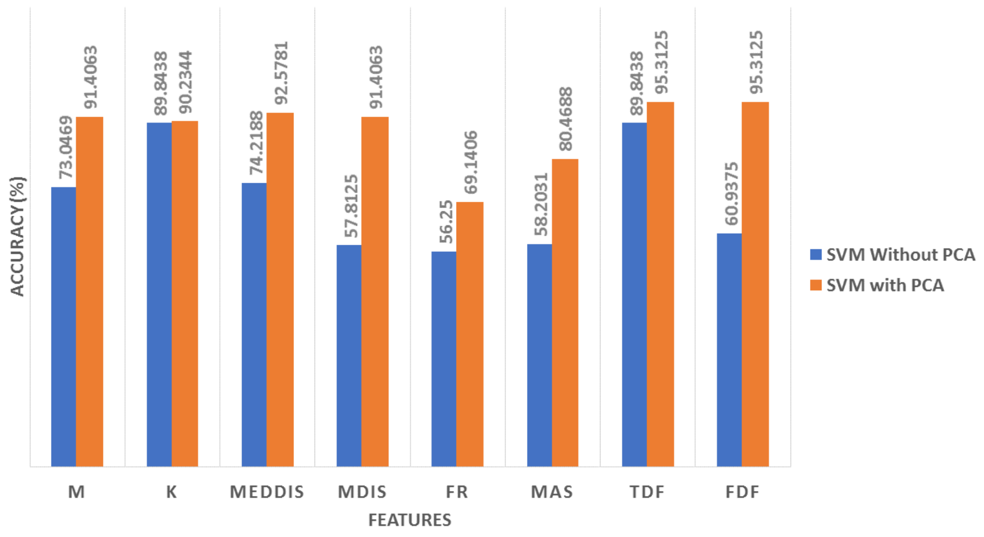


FIGURE 4.5: SVM with and without PCA

4.6.3 Phase-3

Based on the findings in phase-1, FDF is selected for further analysis. The aim is to examine the improvement due to a changed number of validation folds and number of samples per key bit classification. The goal of this analysis is to observe if the model can be trained and the key bit can be classified using less number of samples. Fig. 4.6 and 4.7 show the resultant

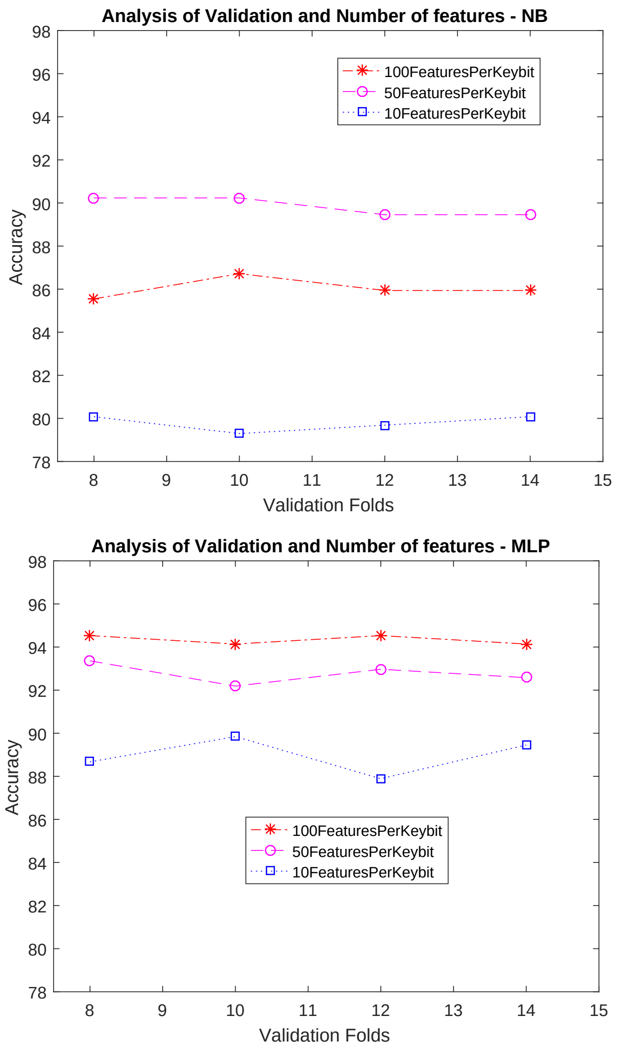


FIGURE 4.6: Effect of change of validation folds and number of features on Accuracy for NB and MLP

accuracy (%) after using different numbers of validation folds for varying samples sizes of 100, 50 and 10. It has been concluded that for Random Forest, for all sample sizes, after 10 folds the curve becomes flat. Moreover, the accuracy reduces with a reduced number of samples.

For MLP, again the accuracy drops as the number of samples is reduced from 100 to 10. Additionally, strange behavior is observed for sample size 50. For all sample sizes, 10 folds seem to be a good choice, but for sample size 50, it lowers the accuracy by 1%.

For SVM, after PCA pre-processing, sample size 50 gives the same accuracy as sample size 100, for 10-fold validation, while sample size 10 reduces the accuracy significantly.

For Naive Bayes, again like the others, sample size 10 is not sufficient to classify the key

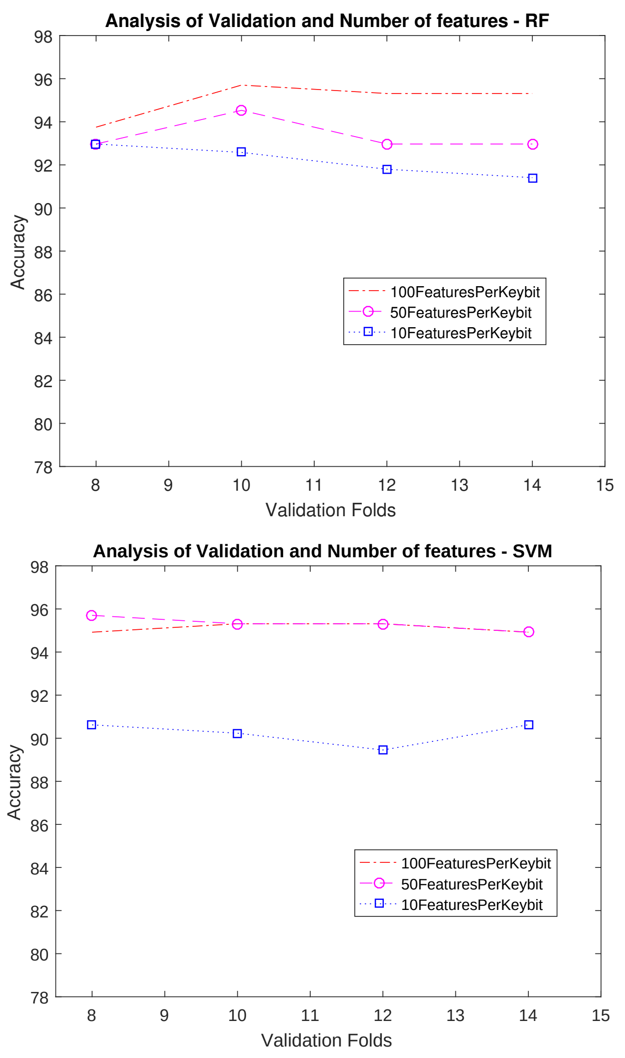


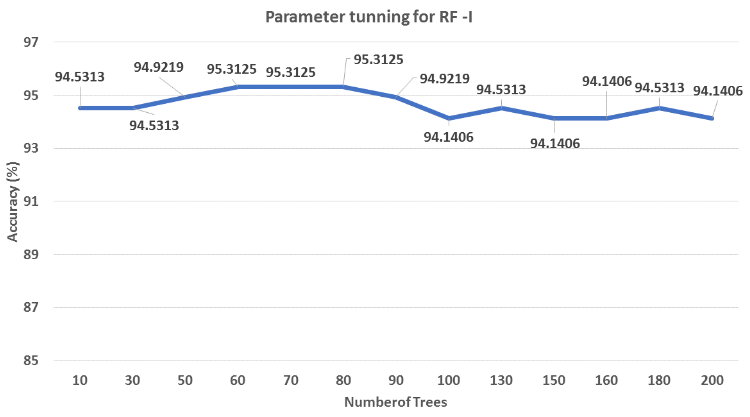
FIGURE 4.7: Effect of change of validation folds and number of features on Accuracy for RF and SVM

bit. However, a drastic change is observed; just as was the case in SVM, it increases the classification accuracy with sample size 50. This is still less than the accuracy with Random Forest and SVM but, within NB, 50 samples improves the accuracy as compared to 100 samples.

4.6.4 Phase-4

This phase deals with the tuning of parameters for NB, RF, MLP and SVM. The effect of each parameter is discussed below.

For RF, experiments are performed to analyze the number of trees which gives better accuracy. Moreover, the number of features considered by each tree while splitting a node are



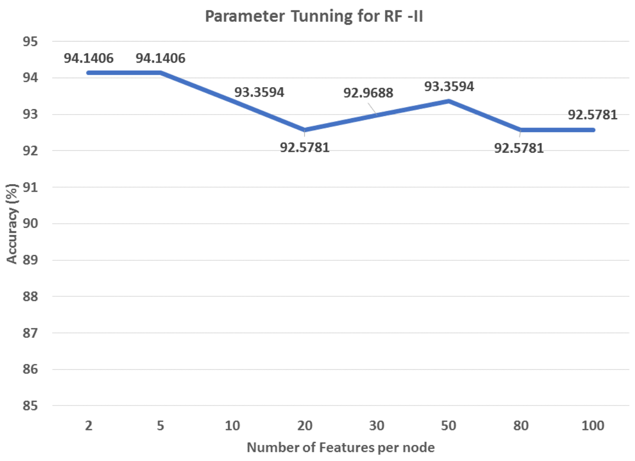


FIGURE 4.8: Parameter Tuning for RF

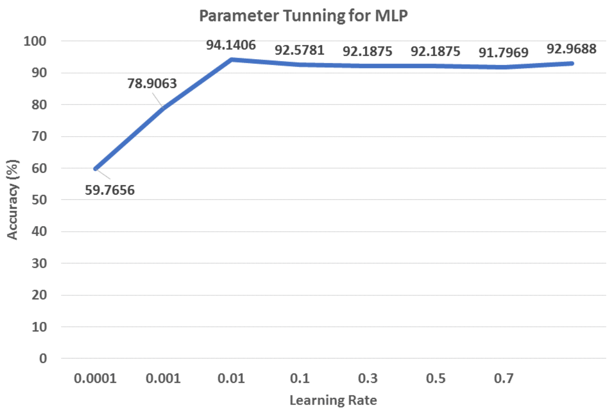


FIGURE 4.9: Parameter Tuning for MLP

changed and effect is observed. Approximately, 94% accuracy is achieved if the number of features per node is 2-5, and 95.3% accuracy is obtained if the number of trees in the random forest is 60-80.

For MLP, the effects of changes of batch size and learning rate are observed. The batch

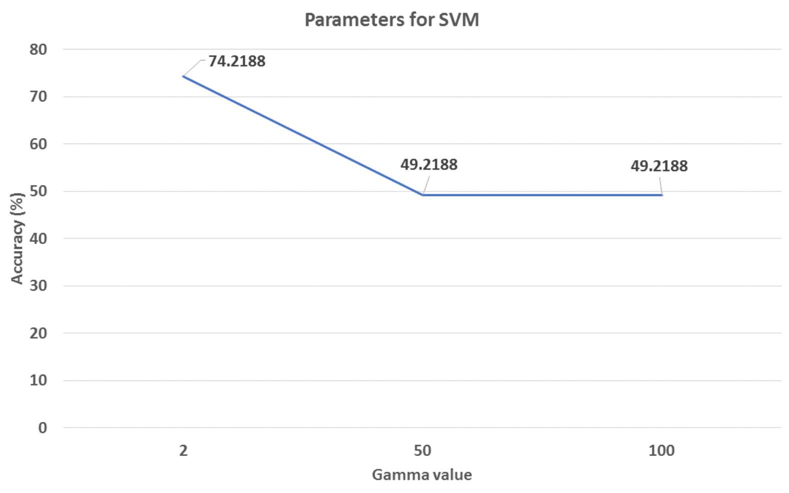


FIGURE 4.10: Parameter Tuning for SVM

size is the number of samples which will be propagated through the network at a time and the learning rate is the rate at which the model trains itself with new data. Generally, if the learning rate is too high or too low, the model keeps on bouncing and it is hard to accurately train the model, meaning that the model's decision making is ambiguous and inaccurate. Fig. 4.9 shows that the best accuracy results of 94% are achieved when the learning rate is set to 0.01. The accuracy decreases significantly with lower and higher learning rates. It is also seen from the experiments that changing the batch size has no or an insignificant effect on the accuracy.

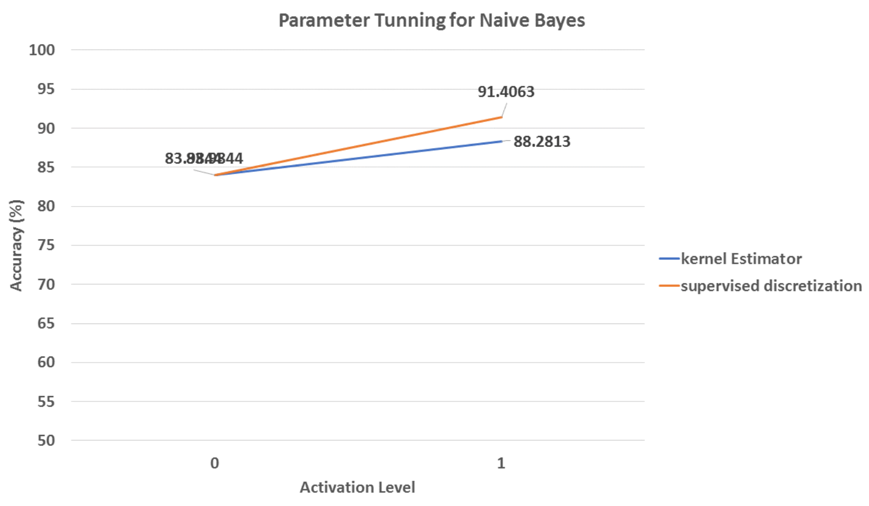


FIGURE 4.11: Parameter Tuning for NB

SVM uses nonlinear kernel functions such as the Gaussian radial basis function or a polynomial kernel, having gamma as a free parameter. A small value of gamma means low bias and high variance in the trained model and vice versa. For SVM, parameter gamma is changed and it is observed that the accuracy decreases with higher values of gamma, as can be seen from Fig. 4.10. It can also be seen that if the data is normalized before classification then the accuracy decreases by 20%.

In Naive Bayes, the kernel is a weighting function used for density estimation of random variables. Turning on kernel estimation will have different effect on the training model. The effect of a kernel estimator and supervised discretization are analyzed with and without applying these parameters on the training data. It is observed that the accuracy is increased by 5% and 8% for kernel estimator and supervised discretization, respectively, as shown in Fig. 4.11.

4.7 Conclusion

On the basis of the analysis results, it can be concluded that Random Forest is the best choice for key recovery using EM radiation analysis, with a sample size of 50 samples per key bit classification, if 'frequency-domain (FDF)' properties are selected as features. Using the 'mean' feature, the classification accuracy is above 90% but is low as compared to FDF. Moreover, it can be asserted that 10-fold validation is suitable for the leaked signal data. It has also been observed that PCA pre-processing worked with SVM classification only, so that models like Random Forest and Multilayer Perceptron handle the high-dimensional data well; even Chi-Square did not produce any significant improvement on the resultant accuracy. Multilayer Perceptron showed classification accuracy of more than 90%, suggesting that use of more complex deep-learning algorithms might improve the classification accuracy. It has also been concluded that various parameters can be changed to improve the accuracy.

Chapter 5

Machine-Learning-Based Side-Channel Evaluation of Elliptic-Curve Cryptographic FPGA Processor

This chapter is an adapted version of a published article (Publication III). In this chapter a hamming weight based attack methodology is presented which utilizes the proposed feature generation methodology (discussed in Chapter 3) to asymmetric ciphers along with the concept of using feature engineering techniques in hybrid mode. The proposed machine learning-based attack model is evaluated on Elliptic Curve Cryptography (ECC) always-double-and-add algorithm. A quantitative comparative analysis is performed, for the proposed attack, to recover the secret key from the raw unprocessed leakage traces and from the processed feature-engineered traces.

5.1 Abstract

Security of embedded systems is the need of the hour. A mathematically secure algorithm runs on a cryptographic chip on these systems, but secret private data can be at risk due to side-channel leakage information. This research focuses on retrieving secret-key information, by performing machine-learning-based analysis on leaked power-consumption signals, from Field Programmable Gate Array (FPGA) implementation of the elliptic-curve algorithm captured from a Kintex-7 FPGA chip while the elliptic-curve cryptography (ECC) algorithm is running on it. This paper formalizes the methodology for preparing an input dataset for further

analysis using machine-learning-based techniques to classify the secret-key bits. Research results reveal how pre-processing filters improve the classification accuracy in certain cases, and show how various signal properties can provide accurate secret classification with a smaller feature dataset. The results further show the parameter tuning and the amount of time required for building the machine-learning models.

5.2 Introduction

Security is the core requirement in embedded systems nowadays and is ensured by using secure cryptographic algorithms on the embedded chips inside these systems. When designing and standardizing cryptographic algorithms, it is ensured that no mathematical relationship can be found between the key, the plain-text, and the ciphertext. However, side-channel attacks are still a threat to the embedded system. In side-channel attacks, physical leakages of the system are exploited to recover the private secret key. Side-channel attacks were introduced by Paul Kocher in the 90s [3], which was followed by the discovery of more side-channel attacks on hardware implementation of popular algorithms like AES, DES, RSA and ECC [5, 6, 17, 79, 89]. All these algorithms are proven to be prone to various kinds of side-channel attacks including power-analysis attack (PA), electromagnetic-analysis attack (EMA), timing attacks (TA). In 2003, Standaert et al. presented a practical PA attack on a Field Programmable Gate Array (FPGA) implementation of AES (symmetric algorithm) [94], and during the same year Siddika et al. presented a power-analysis attack on an FPGA (Virtex 800) implementation of an elliptic-curve cryptosystem [95]. Mulder et al. have presented techniques of key recovery by capturing, processing, and analyzing EM radiations using statistical models [37]. Based on similar techniques, the authors in [7] performed side-channel analysis for retrieving secret information. To perform the side-channel-based key-recovery analysis, various statistical and mathematical methods are used [27, 86, 96–98]. However, noise in leaked signals is one of the main hurdles to the success of side-channel attacks, which leads to the need for huge data sets for accurate key recovery. To cater to this issue, researchers have proposed to use statistical tools like principal-component analysis

(PCA). PCA is used as the pre-processing step to eliminate the noise from side-channel leaked data, hence enhancing the success of differential power analysis (DPA) [76]. PCA can also act as a distinguisher [99].

Recently, researchers have performed machine-learning and neural-network-based analysis to improve side-channel attack efficiency, using various classifiers, to recover the key from the DES, AES and RSA hardware implementations [18–21, 78]. Some of the major challenges of machine learning are over-fitting and the curse of dimensionality, in which a model trains itself to the specific data so well that it fails to predict accurately with new unseen data. To solve this problem, various feature extraction and selection techniques are used [84, 100]. Some of them have been tested for AES data classification as well [93]. There is a very limited literature on machine-learning-based side-channel analysis of elliptic-curve cryptosystems, which is the standard for public-key cryptosystems and is ideally used for resource-constrained environments like IoT-based systems. The focus of this research is to analyze the resistance of the elliptic-curve cryptosystem algorithm, double-and-add-always (which is designed to be resistant against DPA), against a machine-learning-based power analysis attack. To check the immunity of an elliptic-curve cryptosystem against this attack; a hardware system was set up which is capable of capturing the power being consumed by the FPGA Kintex-7 chip while the elliptic-curve cryptography (ECC) algorithm is running on it (as ECC power-signal data does not exist for a Kintex-7) and then analyzing it against various machine-learning-based algorithms with a specifically designed feature dataset. We have chosen an FPGA for analysis because of its popularity for rapid prototyping. Our contributions in detail are listed below.

Our Contributions. Our contributions are threefold. Firstly, analysis of captured power signals is carried out using three machine-learning and neural-network-based classification techniques to classify and recover the secret-key bits of the ECC algorithm. The complete methodology is formulated for formation of the input datasets for further machine-learning analysis. In the existing literature, machine-learning-based side-channel attacks are launched on the raw samples, but no clear information is provided about the attack methodology and input feature datasets. Moreover, the ECC double-and-add-always (public-key) algorithm is selected for attack, as not much analysis is done on key recovery in the public-key domain.

Secondly, we propose to use signal properties as features for efficient analysis instead of using raw samples. This ensures the elimination of redundant data during processing, hence increasing the computational power and reducing the time to train and to test the network. The same proposed hypothesis has been tested for the AES (symmetric cryptography) algorithm in our previous work [93]. Based on the findings of our previous work for the symmetric key algorithm, we have tested the public-key algorithm for five particular signal properties only. Please note that this study is conducted for the ECC (public-key cryptography) algorithm.

Thirdly, our contribution is the application of filters to datasets, before processing them using the classification process. We have used Principal Components and Chi-square filters for pre-processing. The purpose of applying filters is to avoid the problem of wrong classification and to check if the accuracy can be improved by selecting/extracting important features. Again, we have selected one feature-selection and one feature-extraction algorithm, based on our previous findings from the research related to the symmetric algorithm AES. This double layer of pre-processing is applied just to verify if this can improve the accuracy further.

The rest of the paper is organized as follows. Section 5.3 describes related work, the ECC algorithm under test and the classification algorithms used for this analysis, Section 5.4 describes the implementation design of ECC (algorithm under analysis), Section 5.5 explains our attack methodology for key recovery using machine-learning-based classification techniques and describes the feature formation procedure, Section 5.6 outlines the hardware and software experimental setup, Section 5.7 gives an analysis of the results while Section 5.8 concludes the paper.

5.3 Background and Related Terminologies

5.3.1 Power-Analysis Attacks

The PA is a strong passive attack, meaning that the attacker does not need to manipulate the device in any way to extract the secret key. In fact, whenever a command is executed by the device, the consumed power is measured by putting a resistor between V_{ss} or V_{dd} and the true V_{dd}, for processors implemented in CMOS technology. The voltage drop by the current through the resistor is recorded. The voltage measurements are then analyzed using statistical methods to recover the secret key.

PAs can be categorized into simple (SPA) and DPA. The feasibility of a simple PA depends upon the assumption that each instruction will have a unique power trace, which is normally caused by key-dependent branching. For scenarios where traces are not related to the key and instructions but are related to the data key, such attacks are categorized as differential power-analysis attacks. In DPA, the results of hypothetical models are compared with the actual experimental results.

5.3.2 Classification Algorithms

For the analysis in this paper, four main classification algorithms are used—three machine-learning and one simple neural-network-based algorithm. These algorithms have been tested for similar nonlinear data, having independent features, for other symmetric and asymmetric algorithms.

Random Forest (RF)

RF belongs to the class of supervised machine-learning algorithm which is based on decision trees [82]. The outcome of each tree contributes towards the prediction which makes is more reliable and accurate. RF helps in overcoming the problem of over-fitting by using feature-bagging technique. It produces better results even without hyper-parameter tuning which we will verify for our leaked data as well.

Support Vector Machine (SVM)

The support vector machine is another supervised-learning algorithm, which maps and represents data points in n-dimensional spaces to create a clear hyper-plane to separate classes. High-dimensionality can be an issue with SVM which can be handled using feature-extraction methods like PCA.

Naive Bayes (NB)

NB is also a supervised-learning algorithm. It is based on Bayes theorem, in which a probability model is created for the possible outcomes. It is useful for large datasets and is based on the assumption that predictors are independent, i.e., the features present in a sample are completely uncorrelated with each other, which is true for our key classification problem feature set as well.

Multilayer Perceptron (MLP)

A multilayer Perceptron is a type of feed-forward neural network, which uses backpropagation for training. This supervised-learning algorithm is used for solving complex problems stochastically. It is a fully connected network with layers having specific weights 'w' and neurons having a linear activation function which maps the weighted inputs to outputs. These weight values are adjusted based on the output error as compared to the expected value and is achieved through backpropagation.

5.3.3 Validation

It is important to validate the model against the existence of bias, after training with a machine-learning classification algorithm. For our analysis, the k-fold cross-validation mechanism is applied for validation. In the k-fold cross-validation, a hold-out method is used in which the model is trained k times, using k-1 subsets of the training data, and an error is estimated for the testing portion (which is one subset of the data) to analyze the performance of the model. The process is repeated k times to get better validation accuracy.

5.3.4 Feature/Attribute Selection and Extraction

In a feature-selection procedure, several features/attributes are selected, from the existing feature dataset, which are then used in classification-model construction. However, in feature-extraction methods, a new feature/attribute dataset is formed based on the existing features. Both techniques help in reducing the features which helps in better classification. We have selected one feature-selection (Chi-Square) and one feature-extraction (PCA) method for our analysis. As mentioned before, PCA has proven to be the best choice for pre-processing if a support vector machine (SVM) algorithm is used before classification. One of the purposes of this research is to analyze the effect of this best-performing feature-extraction technique on our reduced proposed feature data set (which is formed based on signal properties). Chi-square is randomly selected from the list of feature-extraction techniques. The reason for this selection is that our previous machine-learning-based power analysis on AES data, showed that all feature-selection give almost similar results [92, 93]. We just picked one feature selection as the scope of analysis is wider than just analyzing the feature pre-processing.

5.4 Design and Implementation of Elliptic-Curve Cryptosystem F256 on FPGA

This section explains FPGA design of the elliptic-curve double-and-add-always algorithm (6.3.3) used for this analysis. The understanding of the implementation of the algorithm is important for re-launching the attacks for achieving the same results.

5.4.1 Power Analysis and ECC

ECC, introduced by Koblitz and Millers in the early 80s, is a preferred powerful public-key cryptosystem, especially for resource-constrained environments like smart cards, mobile phones, IoT-based devices, and RFIDs. In ECC, point multiplication is the resource-expensive operation in which a point on an elliptic-curve is added to itself successively. Let 'P' be the point and 'k' be the number of times 'P' is required to be added, then output 'Q' will be 'k' times point 'P' multiplication and is given by (5.1). Elliptic-curve point multiplication is also referred to as Elliptic-curve scalar multiplication (ECSM). Security of an elliptic-curve cryptosystem is based on the elliptic-curve discrete-logarithm problem, which relies on the fact that for an elliptic curve E and given points *P(x,y,z)* and *Q(x,y,z)*, it is hard to find the integer k such that $Q = kxP$.

$$Q = kxP \tag{5.1}$$

To compute ECSM, double-and-add is the simplest straightforward algorithm, in which operations are performed depending upon the 'k' key bits. If the key bit is '0' then only the point-double operation is performed. However, point-double and point-addition both are performed if the key bit is '1'. The simple double-and-add algorithm is susceptible to a simple power-analysis (SPA) attack; simply by analyzing the power consumption of the chip, scalar key 'k' can be resolved, by merely looking at the oscilloscope, without using any advanced processing. Countermeasures are proposed in the literature to help safeguard against SPA attacks. The simplest of all is to add an extra operation so that the double-and-add operations are performed always irrespective of the scalar k bit as can be seen from Algorithm 6.3.3. Double-and-add-always seems to be resistant against PA but is not secure against the safe-error attack, where an attacker introduces an error and examines if the output will show an error or not. Depending upon the output, the scalar key bit k is determined. However, double-and-add-always still seems to be feasible due to the low cost. Further details of the algorithm can be found in [31].

Algorithm 6.3.3. double-and-add-always

```
Input: $P, = k[n]$,
Output: $Q = k\dot{P}$,
1. $R_0 = P, R_1 = 0$
2. For $i = 1$ to $n - 2$
3.     $R_0 = 2\dot{R}_0$
4.     $R_1 = R_0 + P$
5.   If $(k_i = 1)$ then
 6.     $R_0 = R_1$
7.  end if
5. Return $Q = R_0$
```

5.4.2 Nist Standard for 256-Bit Koblitz Curve

The NIST curve (SECP256K1), used in this analysis, over prime fields F_p, is defined as E: $y^2 = x^3 + ax + b$ mod p, where $a = 0$ and $b = 7$ and $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$ [101]. The two main field operations in the double-and-add-always algorithm, point doubling and point addition in Jacobian coordinates over curve E, used for this study, are described in [102]. Jacobian coordinates are preferred over affine coordinates because inversions can be avoided while performing the addition or doubling operation, which is not the case in the affine coordinate system.

5.4.3 Point Doubling in Jacobian Coordinates

This section gives the formulas used for implementing point doubling.

Suppose: P(X_1, Y_1, Z_1) and

$$\alpha = 3X_1^2 + \alpha Z_1^4, \beta = 4X_1Y_1^2, \tag{5.2}$$

Point Q on curve E is defined as: $Q(X_2, Y_2, Z_2) = 2.P(X_1, Y_1, Z_1)$

$$X_2 = \alpha^2 - 2\beta, \tag{5.3}$$

$$Y_2 = \alpha(\beta - X_2) - 8Y_1^4, \tag{5.4}$$

$$Z_2 = 2Y_1Z_1, \tag{5.5}$$

5.4.4 Point Addition in Jacobian Coordinates

This section gives the formulas used for implementing point addition. Suppose: $P_1(X_1, Y_1, Z_1)$ and $P_2(X_2, Y_2, Z_2)$ are two points on curve (E) and

$$\gamma = Y_1Z_2^3, \lambda = X_1Z_2^2, \ \mu = Y_2Z_1^3 - Y_1Z_2^3, \xi = X_2Z_1^2 - X_1Z_2^2, \tag{5.6}$$

The new point P3 on Curve (E) such that: $P_3(X_3, Y_3, Z_3) = P_1(X_1, Y_1, Z_1) + P_2(X_2, Y_2, Z_2)$ is:

$$X_3 = \mu^2 - \xi^3 - 2\lambda\xi^2 \tag{5.7}$$

$$Y_3 = \mu(\lambda\xi^2 - X_3) - \gamma\xi^3, \tag{5.8}$$

$$Z_3 = Z_1Z_2\xi, \tag{5.9}$$

All calculations are to be done in finite field F_p, meaning that *mod p* reduction is applied to Formulas (6.1)–(6.8).

A modular reduction unit is designed based on an interleaved modular multiplier architecture similar to the one proposed in [103, 104]. Based on the implementation results in [103, 104], an interleaved modular multiplier has more efficient area and timing characteristics. For fast and area-efficient implementation of such a multiplier, we use just one CSA adder and a look-up table. The structure of our design is depicted in Fig. 5.1.

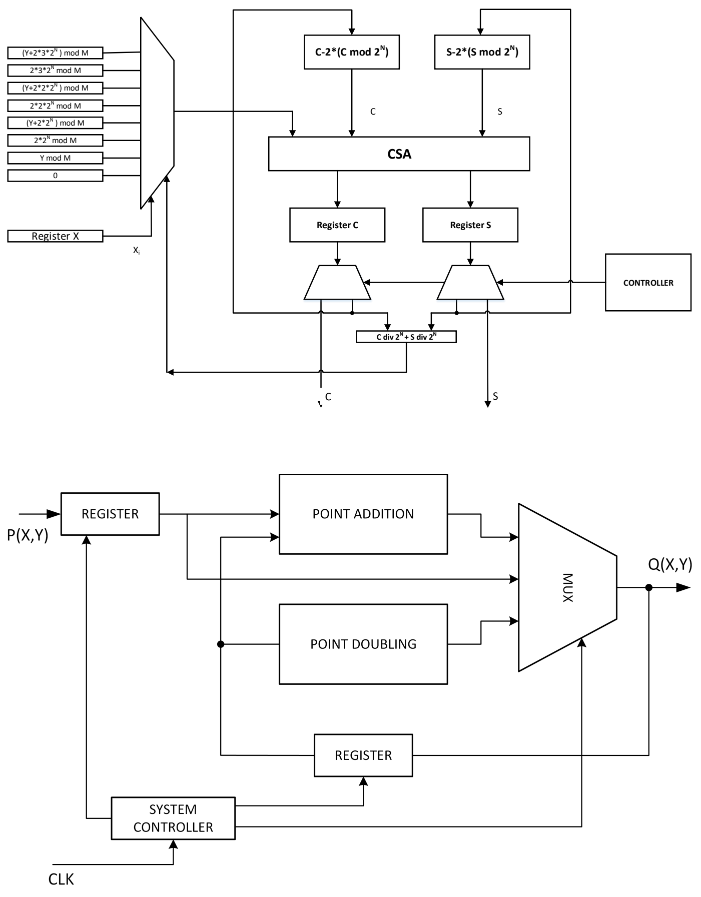


FIGURE 5.1: Interleaved Modular Multiplier (top) and ECC core Diagram (bottom).

The look-up table code is given by;

$$\text{LUT}(0) = 0;$$

$$\text{LUT}(1) = Y;$$

$$\text{LUT}(2) = 2 * 2^n \bmod M;$$

$$\text{LUT}(3) = (2 * 2^n + Y) \bmod M;$$

$$\text{LUT}(4) = 4 * 2^n \bmod M;$$

$$\text{LUT}(5) = (Y + 4 * 2^n)\text{mod } M;$$

$$\text{LUT}(6) = 6 * 2^n \text{ mod } M;$$

$$\text{LUT}(7) = (Y + 7 * 2^n) \text{ mod } M;$$

and

$$(S,C) = \text{CSA}(A,B,C) \text{ is}$$

$$S_i = A_i \bigoplus B_i \bigoplus C_i,$$

$$C_{i+1} = (A_i.B_i)V(A_i.C_i)V(B_i.C_i), C_0 = 0$$

The modular multipliers use one clock cycle to register inputs, 256 (n) clock cycles in the loop, one clock cycle to calculate the CSA addition and one clock to register output, so the calculation is done in just 259 n+3 clock cycles.

5.4.5 ECC Core Design

The ECC core design gets a point on the ECC curve in Jacobian coordinates **P(X,Y,Z)** and calculates point **Q = kxP** within the same coordinate system. Fig. 5.1 illustrates the ECC core design.

5.4.6 Elliptic-Curve Point Doubling—ECPD

Point doubling uses three modular multiplier units to calculate (6.1)–(6.4) in parallel. Ten modular multiplications are done in five stages that reduce the point-doubling calculation time to 5(n+3)+4 clock cycles.

For curve SECP256K1, as a = 0, the logic can be reduced. Using just one modular reduction unit, ECPD can be performed at 7 logic levels or 7(n+3)+2 clock cycles by the optimized-area ECPD. Fig. 5.2 shows the data-flow diagram of the ECPD doubling with and without optimized area.

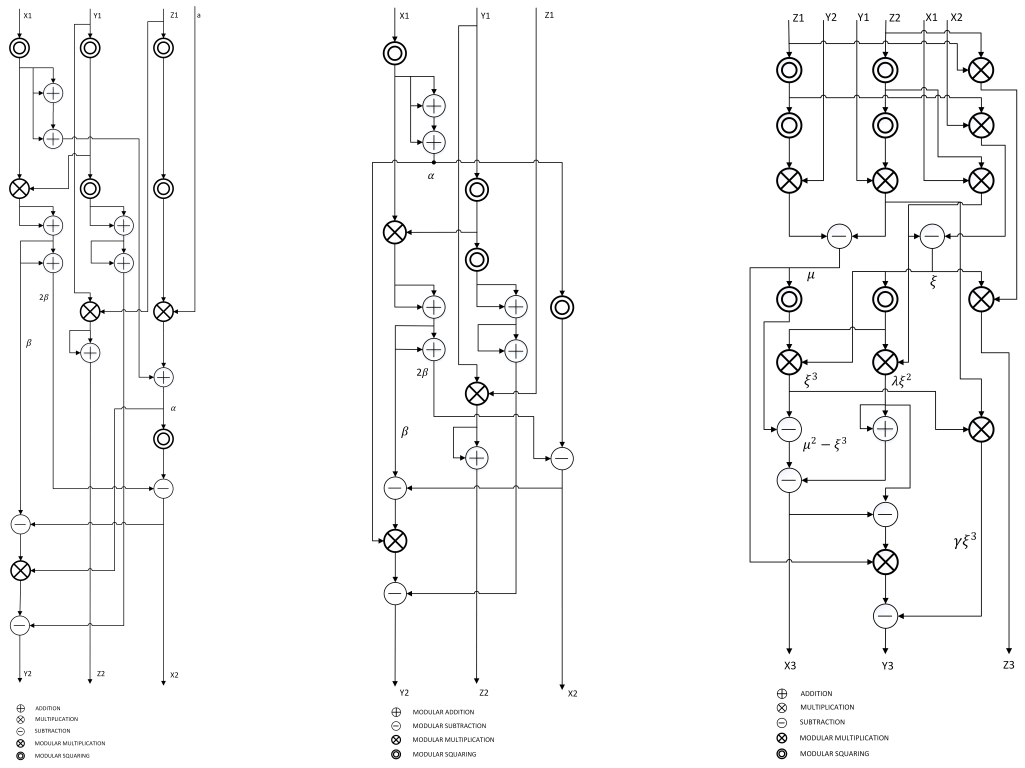


FIGURE 5.2: Data Flows (Left to Right) 1. Point-doubling ECPD for SEC2P256K1 curve, 2. Optimized area for point doubling, 3. Point Addition.

5.4.7 Elliptic-Curve Point Addition

Point addition uses three modular multiplier units to calculate point Q + P on the elliptic curve in parallel. Sixteen modular multiplications are done in seven stages as shown in Fig. 5.2, so the latency of point addition will reduce to 7(n+3)+5 clock cycles.

5.4.8 Scalar Factor (Private Key) k

The scalar factor 'k' is stored in internal RAM and can be changed via software command. To implement a point multiplication, the double-and-add-always algorithm is used as given in Algorithm 6.3.3. A point doubling is done followed by a point addition at every stage i, but the result of the point addition is used only when the *i*th bit of the scalar k is '1'. Otherwise, the result of point addition will not be used.

In this method, N times PD and PA are required (here N = 256). This algorithm uses the same hardware resources for the zero and one bits of the key k, so the power consumption during calculations is homogeneous. The resources consumed by the design of the interleaved multiplier are given in Table 5.1.

TABLE 5.1: Implementation results on Xilinx FPGA XC7k160tfbg676-1.

Resource	Utilization CORE AREA (LUT)
26,570	
ECPA (LUT)	14,382
ECPD (LUT)	11,760
CLK frequency	100.00 MHz
DYNAMIC POWER	0.20 W
TOTAL POWER	0.313 W

5.5 Attack Methodology

The purpose of this research is to capture and analyze the power-consumed signals of the FPGA (Kintex-7) while the ECC double-and-add-always algorithm is encrypting data with a secret key. The idea is to attack one bit at a time. For our analysis, we will attack the least-significant three bits of the nibble i.e., bit 2, bit 3 and bit 4. The bit at location one does not need to be attacked as it does not contribute to the encryption. To achieve this purpose, a random 31-bytes (which are the most significant 248 bits) fixed key is selected and the value of the last byte is changed in ascending order, from 2^1 till $2^4 - 1$. For further simplification, in this paper we have attacked bit locations 2, 3 and 4 only, and bit locations 5 to 8 are set to “0000“. From now on, ’key’ refers to the last nibble of the key as shown in Fig. 5.3.

For the analysis, machine-learning classification will be used. For classification using machine learning, the data samples should consist of the properly labeled features. We propose to use a different set of features as opposed to the raw samples’ amplitude, which leads to a division of our attack into two main steps:

- Step 1—Training dataset preparation
- Step 2—Classification using machine learning

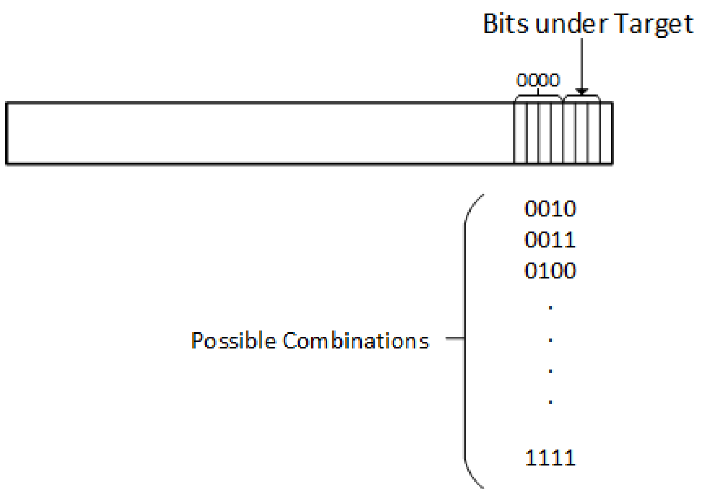


FIGURE 5.3: Key Bits Under Target.

5.5.1 Step 1—Training Dataset Preparation

Let N be the number of randomly selected ECC points, in the Jacobian coordinate system, from the elliptic curve E, and M represent the set of ECC points, then each ECC point in M, over curve E, can be represented as follows:

$$M = \{(X_i, Y_i, Z_i) \text{ where } i = 1, 2, ...N\} \tag{5.10}$$

Let K be the least-significant four bits of the 256-bit key. Out of the 4 LSBs, the three bits at locations 2nd, 3rd and 4th, are the target of this analysis. The first bit is not considered, as the double-and-add-always algorithm's implementation starts encryption using a second bit of the key. For each bit location, raw traces of length Len_{Trace} are collected and then processed to form samples. $S_{BitLoc} = N * S_pt$ samples are collected for N ECC points from the set M, where S_pt represents the number of samples for each ECC point from the pool of N ECC points. As the number of possible combinations for the last nibble is 2^4 and there is no point in attacking the first bit, so in total $S_N = S_{BitLoc} * (2^n - 2)$ samples are collected. For creating a training dataset for machine-learning classification, data samples need to be labeled. After data sample collection, labeling is an important task. To ease the process of attacking and labeling, we have divided the attack into three levels according to the bit location under attack and have categorized the samples into two groups. Each is further explained below.

Group Labeling

All data samples are divided into two groups ’GB0’ and ’GB1’. GB0 means that the sample represents a bit ’0’ and GB1 means that the sample represents a bit ’1’. Each attack level will have different samples marked as GB0 or GB1 according to the bit location.

Attack Levels

Attack levels are designed based on the bit locations under attack, called ’LB{b}’. LB stands for the ’Location bit’ and ’b’ represents the actual location of the bit. Based on this information, three bit levels are defined as follows.

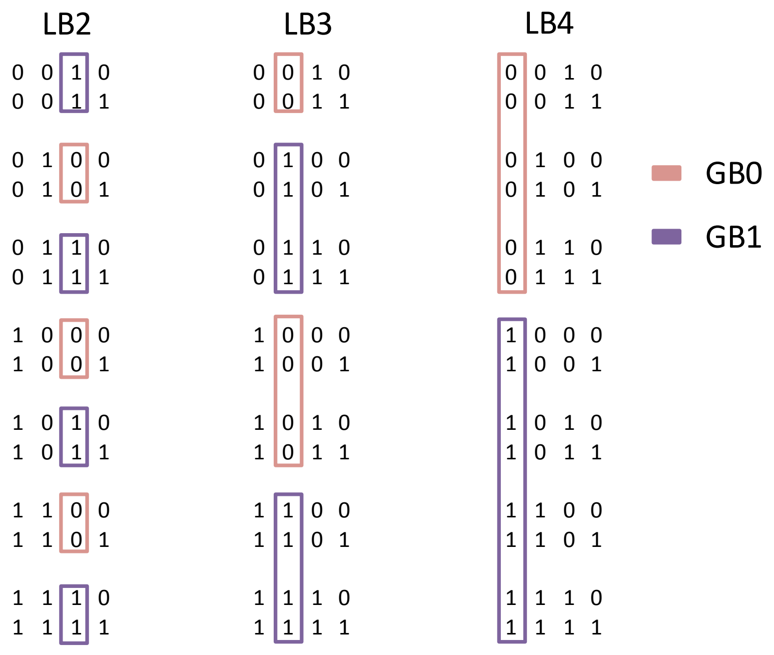


FIGURE 5.4: Sample Labeling.

- LB2—At this attack level, LSB ’2’ is targeted and each sample for the key having ’0’ at the second location is marked as ’GB0’ and all samples for the key byte having ’1’ at the second location are marked as ’GB1’.

- LB3—At this attack level, LSB '3' is targeted and each sample for the key having '0' at the third location is marked as 'GB0' and all samples for the key byte having '1' at the third location are marked as 'GB1'.
- LB4—At this attack level, LSB '4' is targeted and each sample for the key having '0' at the fourth location is marked as 'GB0' and all samples for the key byte having '1' at the fourth location are marked as 'GB1'.

The attack levels along with the labeling of the samples are shown in Fig. 6.2.

Features Dataset Formation

As all the raw samples have been labeled, the next step is to calculate the features. For our analysis, we have used time-domain and frequency-domain signal properties as features. The reason for selecting these particular signal properties is based on our previous analysis on AES leaked data. We selected and analyzed more than six signal properties and concluded that a combination of time-domain and frequency-domain signal properties leads to better classification [92, 93]. An explanation of each signal property (used in this work) is given below:

- Mean of Absolute Value (MAV)—Mean of the signal is calculated.
- Kurtosis (Kur)—For Kurtosis, Frequency-distribution curve peak's sharpness is noted.
- Median PSD (FMD)—For Median PST, median is calculated in frequency domain.
- Frequency Ratio (FR)—For Frequency ratio, frequency ratio of the frequencies is recorded.
- Median Amplitude Spectrum (MFMD)—For MFMD, the median amplitude spectrum of signals is calculated.

For all the captured S_N samples, the above-mentioned features are calculated, returning one sample value for each instead of Len_{Trace}, hence reducing the data sample size, which is the advantage of using the above-proposed features.

The overall training dataset preparation process is shown in Fig. 5.5.

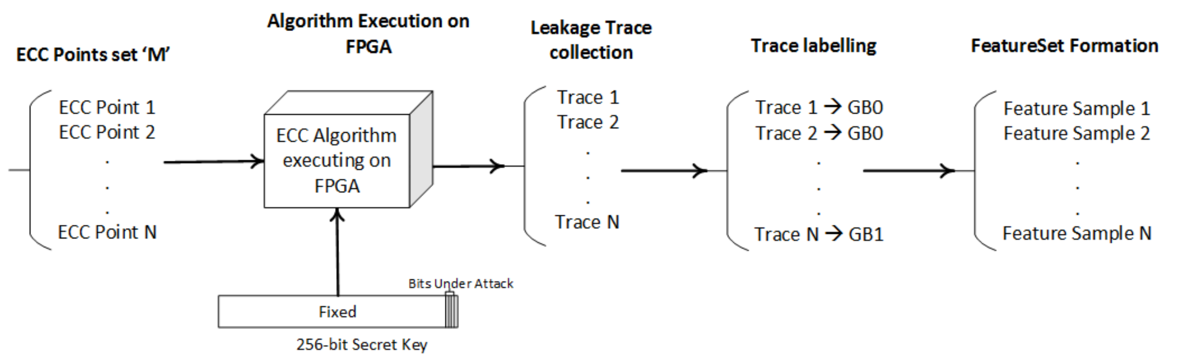


FIGURE 5.5: Dataset Preparation.

5.5.2 Step 2—Classification Using Machine Learning

Traditionally, statistical methods are used to perform the analysis but for this research machine-learning and neural-network-based classifiers are applied on the feature datasets, formed in Section 5.5.1. The classification algorithms selected for analysis are Support Vector Machines (SVM), Naive Bayes (NB), Random Forest (RF) and Multilayer Perceptron (MLP). An explanation of each is given in Section 5.3.2. According to author's knowledge, there is very little work done in the field of machine-learning-based power analysis on elliptic curves which includes analysis of ECC leaked data (from a FPGA) using PCA-SVM. Hence, in our analysis, the comparison is provided with respect to the machine-learning-based analysis only.

There are two parts of the analysis as given below.

Analysis without Pre-Processing

In the first phase of analysis, classification is performed on the feature datasets without any pre-processing. This analysis will help in identifying the impact of using signal properties as features.

Analysis with Pre-Processing

In the second phase of analysis, the feature dataset is first processed through a feature selection and extraction mechanism before training the model and is then subjected to the classification. The feature selection and extraction techniques used for pre-processing are

PCA and Chi-Square (Chi-Sq). Details are given in Section 5.3.4. The signals' noise makes the side-channel attacks harder to launch. The evaluators/selectors are used to filter out the features to overcome the problem of noisy signals, hence reducing the training time and computational complexity. Another benefit of using these extractors/selectors is to reduce the possibility of a wrong classification. For testing the trained model, another feature dataset is formed based on the same methodology. This is done to gain more confidence in the results, as it ensures that the model has never seen the test data before. The process of classification on the training feature dataset is shown in Fig. 5.6. Moreover, the effect of changing of various variables/parameters was observed. The time required to build the model has also been recorded.

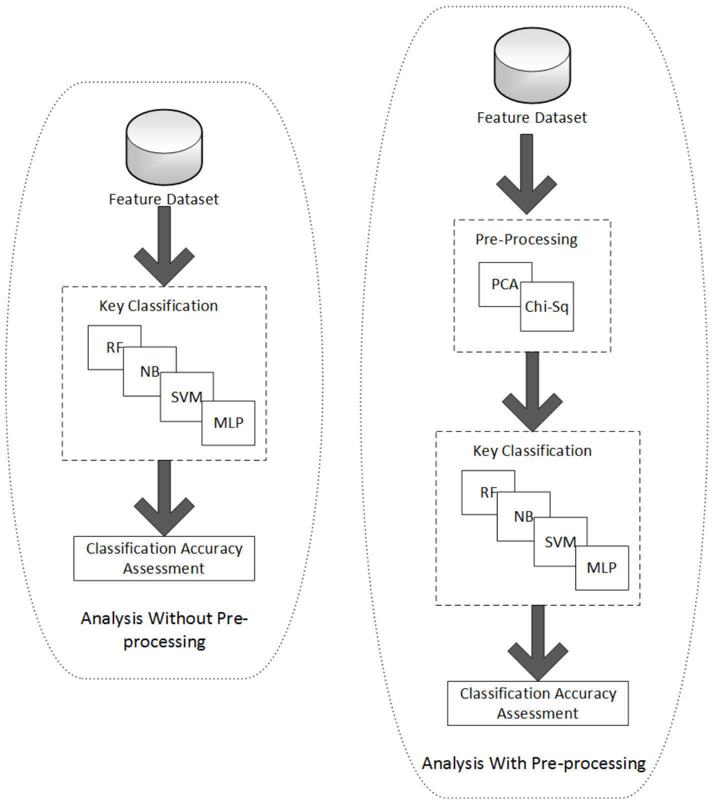

Feature Dataset
Key Classification
RF
NB
SVM
MLP
Classification Accuracy Assessment
Analysis Without Pre-processing
Feature Dataset
Pre-Processing
PCA
Chi-Sq
Key Classification
RF
NB
SVM
MLP
Classification Accuracy Assessment
Analysis With Pre-processing

FIGURE 5.6: Classification process without pre-processing (**left**) and with pre-processing (**right**).

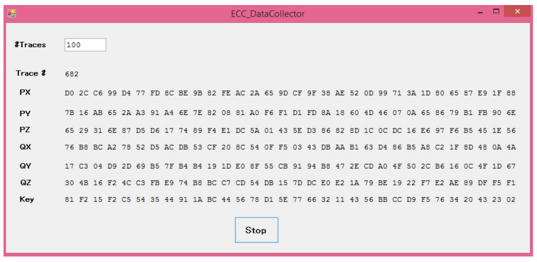
5.6 Experimental Setup

This section explains the hardware and software setup for testing the methodology explained in the above sections.

5.6.1 Step 1—Data Capture

To conduct our experiments, we must capture the leakage traces, as a power signals database for ECC does not exist. For the hardware setup, we captured the power signals for ECC FPGA (Kintex-7) implementation, operating at 24 MHz. For this research, specialized side-channel analysis board, named as SAKURA-X, is used [80]. On SAKURA-X, for calculating the power being consumed, a resistor is connected in series and a voltage is measured across that. The user does not need to tweak the board, as the connector is available to get the power signal directly. Traces are captured using a Tektronix oscilloscope having a 5 GS/sec sampling frequency and a 1 GHz bandwidth. We have acquired N = 100 traces for randomly selected ECC points from set M, and for each point S_pt = 10 traces were captured. Thus, in total S_N = 14,000 traces are collected where each trace has 10 k sampling points.

For the software side of the data-collection process, we have developed bespoke codes using C# and the MATLAB library to form an automated standalone application which requires little or no intervention from the user. The hardware setup and the application GUI is shown in Fig. 5.7. A few modules of the C# application provided by SAKURA are used to achieve the purpose [80]. The new bespoke C# application consists of three main units: control unit, data unit, and configuration unit as shown in Fig. 5.8, and an explanation of each is given below.



```
#Traces   100

Trace #   682
PX        D0 2C C6 99 D4 77 FD 8C BE 9B 82 FE AC 2A 65 9D CF 9F 38 AE 52 0D 99 71 3A 1D 80 65 87 E9 1F 88
PY        7B 16 AB 65 2A A3 91 A4 6E 7E 82 08 81 A0 F6 F1 D1 FD 8A 18 60 4D 46 07 0A 65 86 79 B1 FB 90 6E
PZ        65 29 31 6E 87 D5 D6 17 74 89 F4 E1 DC 5A 01 43 5E D3 86 82 8D 1C 0C DC 16 E6 97 F6 B5 45 1E 56
QX        76 B8 BC A2 78 52 D5 AC DB 53 CF 20 8C 54 0F F5 03 43 DB AA B1 63 D4 86 B5 A8 C2 1F 8D 48 0A 4A
QY        17 C3 04 D9 2D 69 B5 7F B4 B4 19 1D E0 8F 55 CB 91 94 B8 47 2E CD A0 4F 50 2C B6 16 0C 4F 1D 67
QZ        30 4B 16 F2 4C C3 FB E9 74 B8 BC C7 CD 54 DB 15 7D DC E0 E2 1A 79 BE 19 22 F7 E2 AE 89 DF F5 F1
Key       81 F2 15 F2 C5 54 35 44 91 1A BC 44 56 78 D1 5E 77 66 32 11 43 56 BB CC D9 F5 76 34 20 43 23 02
```

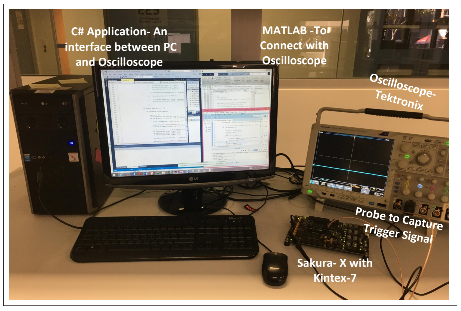

C# Application- An interface between PC and Oscilloscope
MATLAB -To Connect with Oscilloscope
Oscilloscope- Tektronix
Probe to Capture Trigger Signal
Sakura- X with Kintex-7

FIGURE 5.7: GUI for raw Sample Collection Application and hardware setup for power analysis data capture.

- Configuration Unit—The configuration unit uses MATLAB library support for C# and configures the oscilloscope through the C# application. This eliminates setting up the oscilloscope on every start up; the application automatically restores it to the settings required for the data capturing. The configuration unit communicates with the oscilloscope

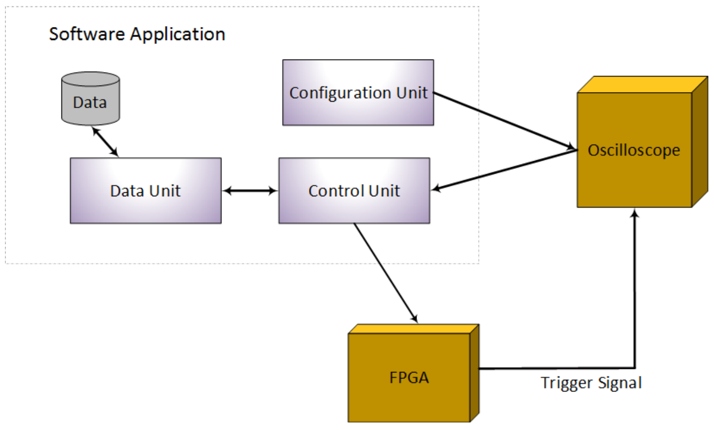


FIGURE 5.8: Software Setup Design.

only.

- Control Unit—The control unit has the role of sending the ECC points to the FPGA after taking them from the data unit. When the FPGA receives an ECC point, it starts the process of encryption and sends a trigger signal to the oscilloscope. As soon as the trigger signal is received at the oscilloscope, it will start collecting the leaked information from the FPGA and will transmit it to the control unit. The control unit then stores the information by communicating with the data unit. The control unit communicates with both the oscilloscope and the FPGA.

- Data Unit—The data unit handles the data. It is responsible for storing and retrieving the data in files. The data unit communicates with the control unit only.

5.6.2 Step 2—Feature Datasets Formation

After collection of the raw traces, samples are labeled according to the description given in Section 5.5.1, using a bespoke java snippet. After labeling, features (properties) are calculated using bespoke MATLAB code, and act as features for further classification.

5.6.3 Step 3—Analysis

Classification models are then trained, using the proposed feature datasets. Feature datasets are trained and tested with and without applying the pre-processing filters. For training and testing, tools like weka and organe3 are used [85]. Parameters settings for each classification algorithm are discussed in the results.

5.7 Results and Discussion

Results and discussion are divided into four sections. In each section, results are discussed with reference to classification algorithms.

5.7.1 Analysis Phase 1—Accuracy without Pre-Processing

In the first part of phase-1 analysis, the classification accuracy is calculated on the raw-signal feature data set. It is observed that RF gives an accuracy of 79% for LB4. For NB, SVM and MLP, the accuracy is even lower i.e., 52%, 55% and 71%. For LB2 and LB3, the accuracy is less than LB4, as shown in Table 5.2. These results clearly show that the data cannot be correctly classified due to the large number of features in the dataset.

TABLE 5.2: Accuracy for Raw sample analysis.

Algorithm	**LB4**	**LB3**	**LB2**
RF	79.2%	56.9%	58.0%
SVM	55.4%	49.3%	45.7%
MLP	71.9%	58.7%	55.6%
NB	52.5%s	57.0%	55.7%

In the second part of phase-1 analysis, the classification accuracy is calculated on the proposed processed feature datasets without any pre-processing (i.e., feature selector/extractor) for all three levels of attack (LB2-LB4), as given in Fig. 5.9. Models are trained and tested using the four classifiers SVM, RF, NB, and MLP. It can be seen that, without the pre-processing step, SVM does not perform well for any level of bit classification. However, for the fourth-bit classification, RF gives an accuracy of approximately 90% while NB and MLP give an accuracy of 85% and 88%, respectively. RF and NB perform well for the datasets in which the features are completely independent of each other. These results prove that the features in the signals feature datasets (for fourth-bit location) are independent of each other.

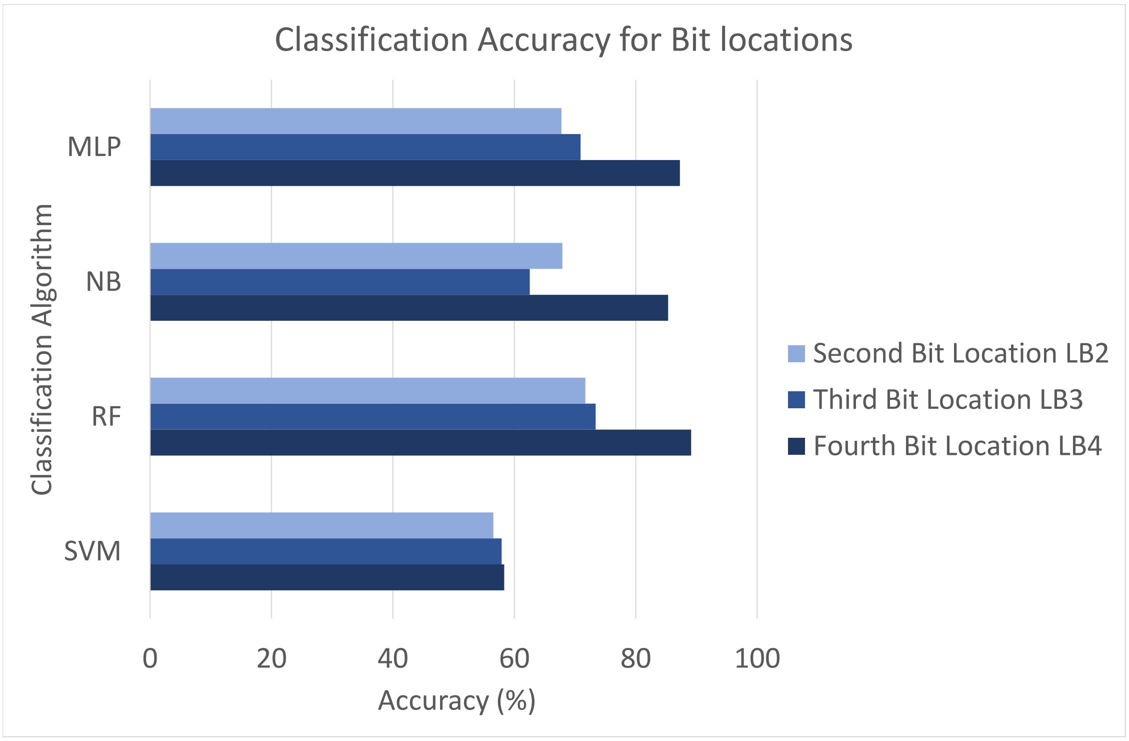


FIGURE 5.9: Classification Accuracy without any pre-processing.

For bit 2 and bit 3 classification, the maximum accuracy achieved is 71–73% with RF. Both SVM and MLP perform poorly in these cases. The reason for MLP's low performance could be the feature dataset size. For neural-network algorithms, the training data should be huge, roughly a hundred times more than the number of features in each trace/row. It is worth exploring if MLP or any other neural network can behave better if the number of samples is increased for better training. This analysis is out of the scope of this paper and is a future prospect of this particular research.

5.7.2 Analysis Phase 2—Accuracy with Pre-Processing

In the second phase of the analysis, the classification accuracy is calculated on the feature datasets after pre-processing them using the feature selectors/extractors. The results of 'LB2', 'LB3' and 'LB4' are given in Fig. 5.10. The results show that if PCA is applied then, for

SVM, the accuracy improves for all three cases. This happens because the obtained traces are noisy, having redundant information. PCA extracts the important features/components so when SVM is applied on the reduced feature set then the accuracy is improved. The maximum accuracy attained is 87% for 'LB4'. However, Chi-square did not show any improvement in any of the LBs. It is worth noting that the accuracy of RF got worst after pre-processing with PCA, because RF works on the assumption that there is no dependence between features. PCA reduces the number of features and at the same time removes the col-linear features from the feature dataset. For MLP, accuracy increases after applying filters in case of LB4 but strange behavior is observed in case of LB3 and LB2, which requires further analysis.

Saeedi et al. in [105] have obtained 96% accuracy after applying SVM on 4-bit implementation of ECC leaked data . Our results of classification algorithms are obtained after applying SVM on 256-bit key (out of which first 31 bytes of the key are fixed random numbers). Our results show that, with PCA-SVM, accuracy of around 86% can be achieved to recover the least-significant nibble from a 256-bit key.

5.7.3 Analysis Phase 3—Time to Build Models

It has been seen that the time taken to build the model with raw signals varies from 90–150 s for classifiers. However, the time taken to build the model on the proposed processed feature dataset is less. The reason is obviously that, with raw signals, the number of features per trace is 10 k times more than for the proposed processed feature datasets. In particular it was observed that the time taken to build the model for the MLP, with proposed processed feature dataset, is more than the time taken by SVM, RF and NB. After applying PCA, the time taken to build the model is reduced in all cases (LB2-LB4) for all algorithms, as can be seen from Fig. 5.11 and Figure 5.12. The reason for the decrease in the time required to build the model is that the number of features is reduced after applying filters.

5.7.4 Hyper-Parameter Tuning

Based on the analysis so far, the best-performing combination of filters and classifier is selected, and different parameters are tuned for LB4. The parameters used for analysis, using

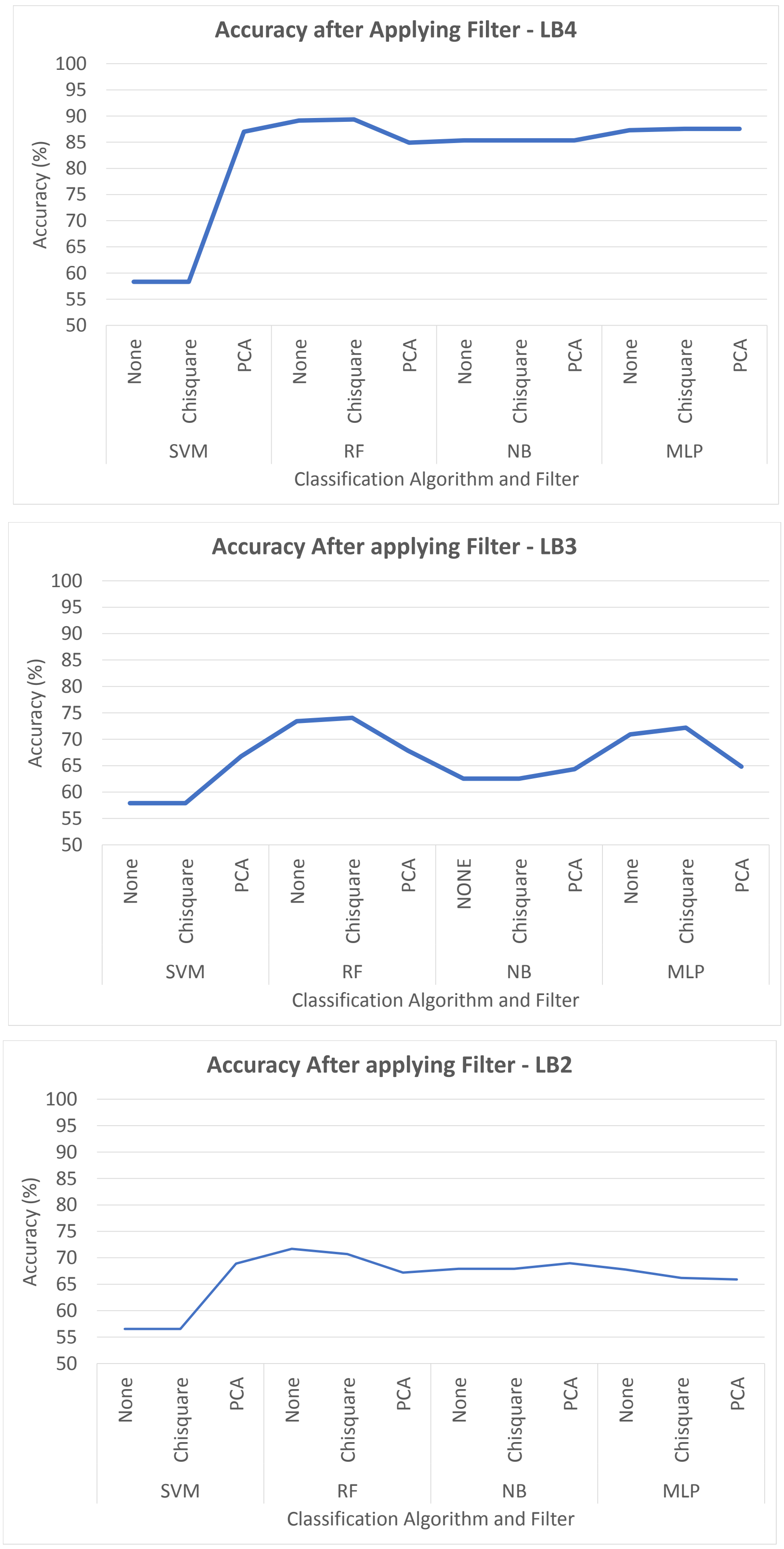


FIGURE 5.10: Classification Accuracy with pre-processing—LB2, LB3 and LB4.

four classification algorithms, are discussed below.

The parameters tuned for RF are the number of trees and the number of features per

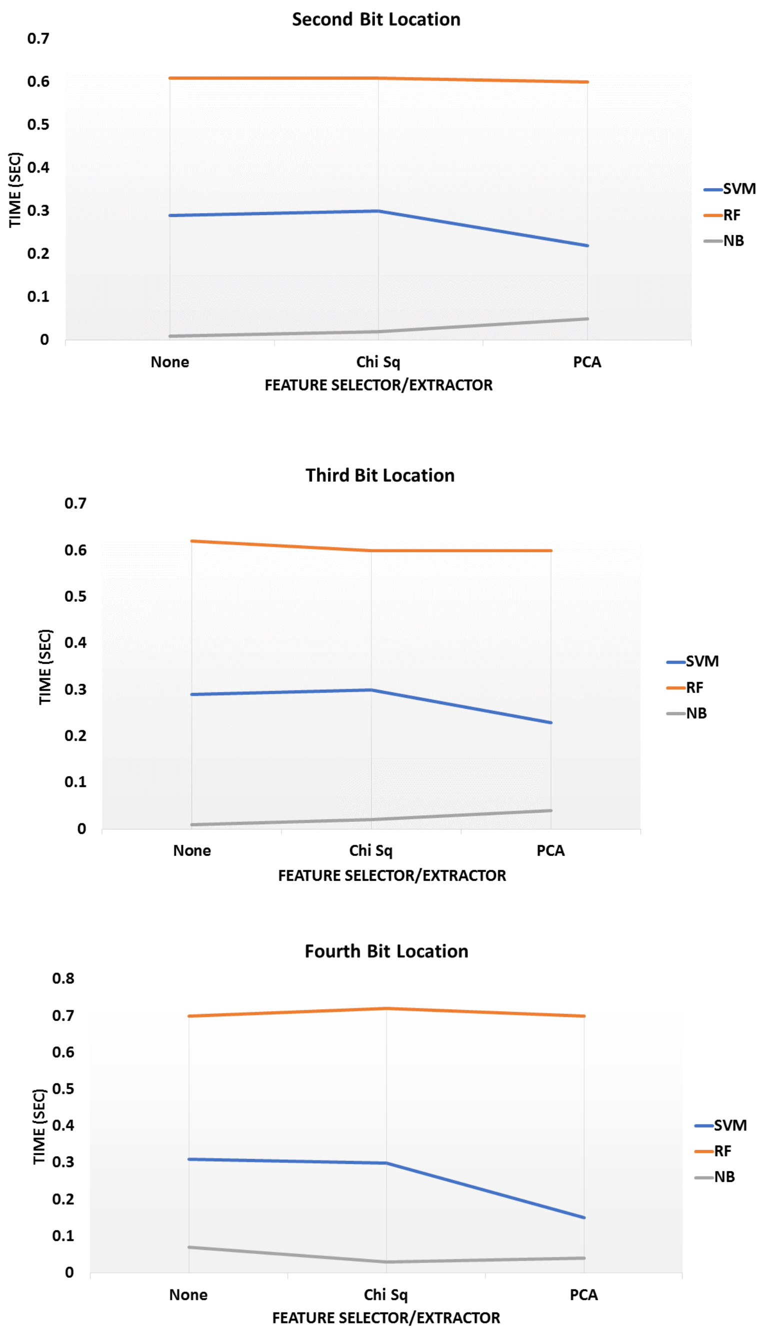


FIGURE 5.11: Timing Results for model building for LB2, LB3 and LB4.

tree. Experimental results show that with 90 trees within the forest, the highest accuracy of approximately 89.4%, is achieved. The accuracy decreases if the number of trees is increased or decreased beyond 90. Therefore, 90 trees are selected for further analysis, and number of features per tree are changed, when it is observed that maximum accuracy is obtained when the number of features per tree is 30, as shown in Fig. 5.13.

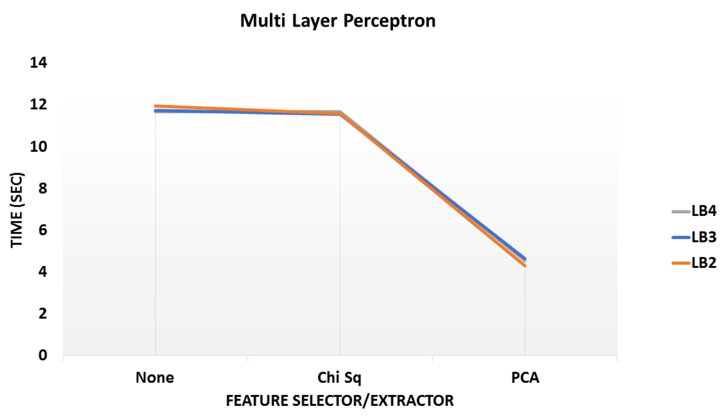


FIGURE 5.12: Timing Results for model building for MLP.

In NB, two important parameters are kernel estimator and supervised discretization. It was observed that turning on kernel estimator gives an accuracy of 86.77%. However, accuracy is increased if supervised discretization mode is turned on.

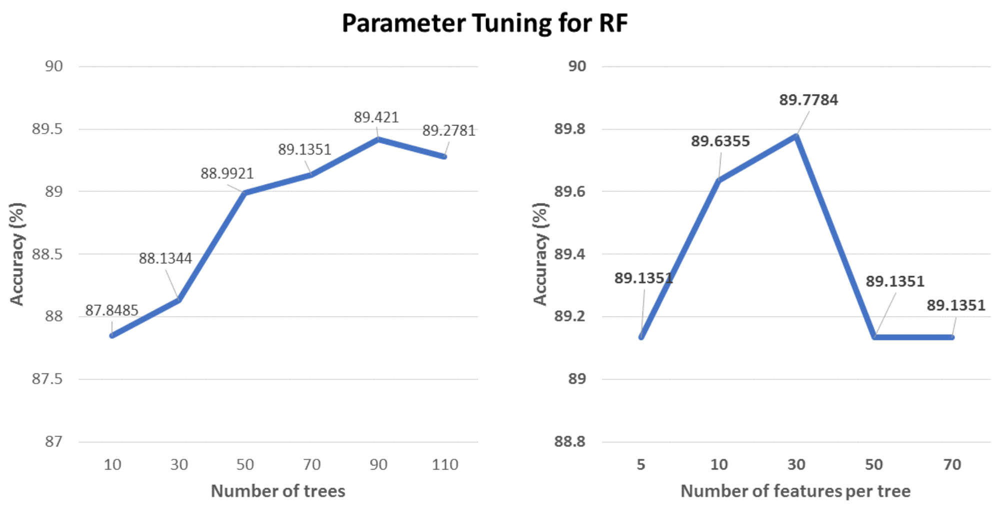


FIGURE 5.13: Parameter Tuning for RF.

For SVM, gamma is changed to see the effect on accuracy. As SVM uses nonlinear kernel functions, so a lower gamma value means low bias and high variance. It is seen that with higher values of gamma, the accuracy decreases as can be seen from Fig. 5.14.

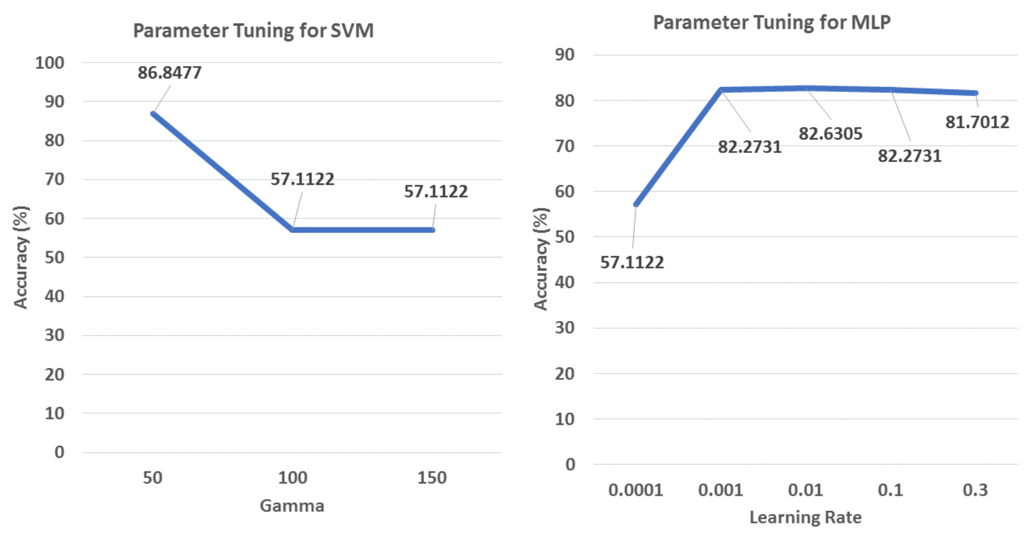


FIGURE 5.14: Parameter Tuning for SVM and MLP.

For any neural network, the two most important parameters to analyze are the change of learning rate and the batch size. For this analysis, the number of neurons is fixed, and one hidden layer is used. Learning rate is the rate of training with which the model is trained, and the batch size is the number of samples that are given to the model for one training period. It was observed that the batch size does not have any effect on the accuracy. However, the model is trained best with learning rate of 0.01, as can be seen from Fig. 5.14.

5.8 Conclusion

After analyzing the results on the power-consumed signals obtained from the Kintex-7, we can conclude that signal properties of the captured leaked power signals can be used as features, as they give an accuracy of approx. 90% with RF. This means that we can recover the secret key from the leaked signals with approx. 90% accuracy. For classification algorithms like RF, NB, and MLP, pre-processing did not show any improvement at all, because these classifiers already perform well for noisy data having redundant information. However, for SVM, using PCA as a pre-processing step improved the accuracy to 86%, as PCA extracted the important relevant features. SVM still shows less accuracy than the others. Moreover, the time taken for building the model has been analyzed and it is observed that the time for training the model is more for raw signals and for the neural-network-based MLP model. The parameters for all four classification algorithms have been tuned and the best recommendations are put forward in the paper.

Chapter 6

Electromagnetic Side-Channel Attack Analytics on Elliptic Curve Cryptography using Machine Learning in IoT Devices

This chapter is an adapted version of a prepared article to be submitted to a journal (Publication IV). In this chapter, three machine learning-based attack models are identified based on the attack categories (divide-and-conquer and analytic attacks [26, 27]) and are adapted for machine learning-based side-channel attack on ECC always-double-and-add algorithm. The attacks are evaluated on electromagnetic leakages from asymmetric key ECC algorithm for raw and feature-engineered datasets as proposed in Chapters 3-5. Generally, noise is introduced as a countermeasure to hide the relationship between the leakage and the secret key. Hence in this study, another challenge is introduced by performing analysis on the heavily noised leakage traces, in which 97% of the trace consists of noise. The comparative analysis is comprehensively reported.

6.1 Abstract

Security is one of the vital aspects of the emerging Internet of Things (IoT) systems. The implementation weakness of the secure algorithms on the IoT devices can lead to side-channel attack vulnerability. The use of machine-learning has exacerbated the vulnerability by improving attack efficiency. One of the challenges in using machine-learning for side-channel analysis is formulating the attack model suitable for the underlying target algorithm. We

have presented and investigated three machine-learning-based electromagnetic attack models to recover the secret key from the standard Elliptic curve-based algorithm's implementation leakages. Moreover, quantitative analysis is provided for the best attack model selection using standard machine-learning evaluation metrics. An analysis is performed on the raw unaligned data samples and reduced feature-engineered datasets to analyze the attack efficiency. The investigation concludes that the implementation can lead to the secret key recovery, with 96% accuracy, using a simple neural-network-based algorithm.

6.2 Introduction

Internet of Things (IoT) based systems are revolutionizing the industrial manufacturing infrastructure. Cognitive Industrial Internet of Things (IIoT) applications include cloud-connected smart factories that automate the manufacturing process, smart grids, intelligent vision, intelligent medical systems, and real-time power state estimation. Cognitive IoT is breeding smart factories built on top of artificial intelligence technology, with well-instrumented and interconnected devices, control systems, and sensors. When put to predictive modeling, the industrial automated system generated Big Data offers numerous benefits like early defects and production failure detection, 100% product verification instead of random sampling. In these IoT applications, there are a plethora of devices involved, from various vendors, connecting and communicating, incorporated with a variety of firmware and software, hence are exposed to higher security risk. Embedded chips are at the core of these IoT based systems. Various manufacturers, including Intel, are proposing Industry 4.0 autonomous solutions based on Field Programmable Gate Array (FPGA). FPGA chip based designs offer low production time and quick integration of new features in the existing systems due to their reconfigurable nature. One of the limitations of these embedded devices is the availability of on-chip resources. For the efficient implementations on these chips, security is often compromised. Breach of security in control systems can cause fatal financial and reputational damages and, above all, can jeopardize human lives. This leads to the fact that in all IoT based systems, security measures must be integrated at the design level [33]. For authentication and digital security, Elliptic Curve Cryptographic (ECC) based asymmetric algorithms are preferred in the resource-constraint environments due to the efficient processing and small key size [106–110]. The standard ECC algorithms are mathematically secure, but the weak implementation of the algorithm can introduce side-channel attack vulnerability.

Various studies prove that the underlying system (side-channel) information, including power consumption, electromagnetic emanations, cache-timing, acoustic vibrations, can be exploited to recover the secret information from embedded chips [3, 5, 6, 94]. The electromagnetic side-channel emissions, caused by the current flow within the chip, can be analyzed to determine the transitions of data from ‘0’ to ‘1’ or ‘1’ to ‘0’. Secret information can either be recovered by merely looking at a single signal trace on the oscilloscope or by utilizing the statistical methods, leading to univariate and multivariate analysis and hence simple and differential analysis [37, 98]. Recently, machine learning (ML) has been proposed and used for the analysis of side-channel leaked data. In earlier studies, mainly symmetric cryptographic algorithms like Advanced Encryption Standard (AES), Data Encryption Standard (DES), and Rivest–Shamir–Adleman (RSA) have been evaluated against machine-learning-based side-channel attacks [22, 70, 111]. The existing investigated classifiers include multi-class Support Vector Machines (SVM), Random Forest (RF), Decision Trees (Tree), simple Neural Network-based algorithms like Multi-Layer Perceptron (MLP), and Convolutional Neural Networks (CNN), on both protected and unprotected versions of the various cryptographic algorithms [112], [113], [114], [92],[21]. Classification response to asymmetric and symmetric algorithms might be different due to their inherent design differences. According to the no free lunch theorem, there is no single method that is suitable for all the problems in machine learning classification problems, that is, results from one classification algorithm on different cryptographic datasets can produce varying results, depending on the characteristics and properties of the data [115]. This leads to the need to analyze asymmetric Elliptic Curved-based algorithms against ML-based Side-channel attacks as well.

The success of machine-learning (ML) based attacks considerably depends on the attack design formulation. As the side-channel attacks exploit the relationship dependency of the leakage data with the processed secret data/key, hence, the same leakage trace might represent a single key bit, a key nibble, a key byte, or a particular algorithm operation (e.g., addition or doubling). The resulting design model leads to binary or multi-class classification problems, as has been presented in literature [18, 116, 117]. The hamming weight-based class labeling model is the most popular model for analyzing cryptographic algorithms. The majority of these machine-learning based Side-channel Analysis (SCA) models are presented for symmetric key algorithms. Due to the different ECC algorithm structure, various attack models can be formed that have not been tested previously, using machine learning for ECC leakage datasets. In this research, we have identified and analyzed these models to select the best attack model.

One of the challenges in applying machine-learning onto electromagnetic side-channel leakage data is achieving excellent performance efficiency on noisy leakage data. In certain cases, noise is induced intentionally in the side-channel leakages, to provide resistance against side-channel attacks. Moreover, various countermeasures have been proposed against weak ECC implementations [118]. These countermeasures can be weakened by using pre-processing techniques before applying traditional statistical methods or by aligning the traces and selecting a point of reference using CPA and SOST [119]. The point of references can be mapped onto features in machine-learning methodology. Machine-learning based side-channel attacks have significantly improved the performance of key recovery on raw data, without the need for pre-processing. Deep learning-based models are presented which can recover the secret information from the raw traces, without the requirement of data pre-processing or alignment [23]. Limited analysis exists for asymmetric algorithms. To improve ML-based SCA's efficiency, one of the techniques is feature engineering, i.e., to refine and select the critical features from the leakage traces that contribute the most towards the better classification model. The other method to improve the model efficiency is to perform hyperparameter optimization, and it has proven to enhance the performance of datasets in other fields [120]. In this research, we have performed both feature engineering and hyperparameter optimization to evaluate and improve the best-selected attack model's performance on protected and unprotected ECC scalar multiplication implementations.

This research aims to define and evaluate an efficient machine-learning-based electromagnetic attack model on ECC based embedded systems, used in the IoT environment, which will help embedded system designers to develop side-channel attack resistant secure systems. The overall research contributions are listed below:

- Three attack models are identified and analyzed for the efficient key recovery, from the asymmetric cryptographic algorithm implementations, using machine learning classifiers. The presented research is the first study to explicitly formulate and evaluate the machine-learning-based electromagnetic analysis attack models for Elliptic Curve implementations, comparatively. The design challenges are explored while mapping electromagnetic attacks on the machine-learning-based electromagnetic attacks.

- We have investigated machine learning-based attacks' performance for predicting secret keys from raw signals, without noise removal and without aligning the signals. The research evaluates the feasibility of the machine-learning-based attacks in a practical scenario, where a limited number of traces are available, along with highly induced noise due to neighboring components. Moreover, the reduced size datasets, with processed features (extracted utilizing the signal properties of the raw leaked electromagnetic traces), are also evaluated to compare the efficiency of the attack model performance.

- An automated uninterrupted electromagnetic side-channel leakage information collection process is designed to collect the data from the efficient FPGA implementation of the target cryptographic algorithm.

- Unprotected and protected ECC implementations, both, are evaluated against machine-learning-based electromagnetic attack, using six machine learning classifiers. The best performing classifier model is further tuned for hyperparameters to achieve the best performance.

- The performance comparison based on the quantitative analysis is presented for the proposed attack models.

The rest of the paper is organized as follows. Section 10.3 presents the related background information. Section 6.4 explains the machine-learning-based side-channel attack methodology along with all the steps involved and the attack models, Section 6.5 briefs the experimental setup, Section 6.6 gives the results and analysis, and Section 6.7 concludes the paper.

6.3 Background

6.3.1 Electromagnetic Side-Channel Analysis

The electromagnetic analysis is based on the fact that whenever information is processed within a chip, then due to current flow within the device, the electromagnetic field is produced, and electromagnetic (EM) radiations emit out. These EM radiations leak information about the transitions of data from '0' to '1' and '1' to '0'. Due to this leakage data dependency on the secret information, the relationship between leakage data and secret data can be exploited to recover the secret data or key.

6.3.2 Side-Channel Attacks and Machine learning

Side-channel attacks were first introduced by Kocher et al. [3]. Following that, various side channels have been exploited to extract the secret key, including timing information, power consumption, electromagnetic radiation emission, system vibrations [17, 79, 89]. Electromagnetic side-channel attacks have been exploited to attack the small devices having ARM processors and FPGAs [121–123]. Emissions from PCs and laptops have been analyzed to recover the useful secret information [124]. The efficiency of side-channel attacks improves using machine-learning techniques. Several cryptographic algorithms like AES, DES, RSA, and ECC have been analyzed using machine-learning based attacks, mainly based on Support Vector Machines (SVM), Random Forest (RF), and Deep learning algorithms like Convolutional Neural Network (CNN). In [116], Lerman et al. have presented a semi-supervised template attack on AES power signals leakages (captured from an 8-bit microcontroller ATmega328P), and proposed to recover the AES key by using a small set of known keys. In

[117], Prouff et al. have also used the hamming weight strategy for class labeling. In [18], Levina et al. have discussed the applicability of the machine-learning methods for secret key classification tasks, based on the bitwise and byte-wise analysis. All mentioned studies are majorly carried out on symmetric ciphers. For asymmetric cipher, Weissbart et al. in [125], and Mukhtar et al. in [126], have presented machine-learning-based attacks for ECC. However, the analysis using the attack models presented in this research has not been performed previously.

In machine-learning side-channel attacks, the leakage dataset is divided into two sets, training, and testing. The training set is used for training the model for future predictions, and the testing set is used to test the model. To have an unbiased and accurate model, testing set is never shown to the model during training. Further details on the structure and division of the leakage datasets are given in Section 6.4.

6.3.3 Algorithm Under Attack

ECC is the recommended asymmetric algorithm for resource-constrained environments like IoT, due to its higher efficiency as compared to other asymmetric algorithms. The security of ECC relies on the hard mathematical problem, i.e., discrete logarithm problem (DLP), which states that provided the point ‘P’ and ‘Q’, it is computationally infeasible to find the key ‘k’ which is used in the encryption of ‘Q’, where $Q = kP$.

In the ECC algorithm, the main operation is the elliptic curve scalar multiplication (ECSM). To perform ECSM, double-and-add is the simplest algorithm used. In double-and-add, operations are performed based one the key bit value. If the key bit is ‘1’, then double and add both operations are performed. However, if the bit is ‘0’, only a doubling operation is performed. Due to this dependent branching on the key bits, this algorithm is susceptible to side-channel attacks. As a countermeasure, an improved version of the same algorithm is proposed, known as double-and-add-always. In this version, the branching dependence is removed, and all doubling and adding operations are performed irrespective of the scalar key bit value, as given in the Algorithm 6.3.3. This algorithm uses the same hardware resources for the ‘0’ and ‘1’ bits of the key k, so the power consumption during calculations is homogeneous.

Simple power and electromagnetic analysis attacks become infeasible due to homogeneous power consumption, which leads to the analysis of more sophisticated methodologies like machine-learning. Machine-learning based electromagnetic attack strength has not been analyzed thoroughly, using additional countermeasures. Further details of the algorithm can be found in [31].

Algorithm 6.3.3 : double-and-add-always

```
Input: $P, = k[n]$,
Output: $Q = k\dot{P}$,
1. $R_0 = P, R_1 = 0$
2. For $i = 1$ to $n - 2$
3.     $R_0 = 2\dot{R}_0$
4.     $R_1 = R_0 + P$
5.   If $(k_i = 1)$ then
6.       $R_0 = R_1$
7.   end if
5. Return $Q = R_0$
```

For the analysis in this paper, we have selected NIST curve SECP256K1 and Interleaved Modular multiplier algorithm, as both are suitable for resource constraint environments. The NIST SECP256K1 curve is a short Weierstrass elliptic curve defined as E: $y^2 = x^3 + ax + b$ mod p , where $a = 0$ and $b = 7$ and $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$, and provides efficient implementation of group operations. Moreover, the interleaved modular multiplier is area efficient and possesses low power consumption characteristics [127] [103].

Elliptic curve point doubling (ECPD) and elliptic curve point addition (ECPA) are performed in Jacobian coordinates and finite field F_p [102]. Advantage of working in Jacobian coordinate system is that the field inversion can be avoided, which is a costly operation. Suppose $P_1(X_1, Y_1, Z_1)$ and $P_2(X_2, Y_2, Z_2)$ are two points on curve E, then formulae used for performing an ECPD and ECPA operations in Jacobian coordinates are given in 6.1 - 6.4 and 6.5 - 6.9.

$$\alpha = 3X_1^2 + aZ_1^4, \beta = 4X_1Y_1^2. \tag{6.1}$$

Point Q on curve E is defined as:

$Q(X_2, Y_2, Z_2) = K \cdot P(X_1, Y_1, Z_1)$ such that:

$$X_2 = \alpha^2 - 2\beta. \tag{6.2}$$

$$Y_2 = \alpha(\beta - X_2) - 8{Y_1}^4. \tag{6.3}$$

$$Z_2 = 2Y_1 Z_1. \tag{6.4}$$

$$\gamma = Y_1 {Z_2}^3, \lambda = X_1 {Z_2}^2, \tag{6.5}$$

$$\mu = Y_2 {Z_1}^3 - Y_1 {Z_2}^3, \xi = X_2 {Z_1}^2 - X_1 {Z_2}^2 \tag{6.6}$$

The new point P_3 on Curve E defined as: $P_3(X_3, Y_3, Z_3) = P_1(X_1, Y_1, Z_1) + P_2(X_2, Y_2, Z_2)$ can be obtained using :

$$X_3 = \mu^2 - \xi^3 - 2\lambda\xi^2. \tag{6.7}$$

$$Y_3 = \mu(\lambda\xi^2 - X_3) - \gamma\xi^3. \tag{6.8}$$

$$Z_3 = Z_1 Z_2 \xi. \tag{6.9}$$

6.4 Machine-Learning based Attack Methodology

Overview of the machine-learning-based side-channel attack on embedded systems is explained in this section. The purpose of the proposed machine learning-based electromagnetic (EM) side-channel attack is to analyze the electromagnetic leakage data to recover the secret key. This section briefs the algorithm under attack, modes of proposed attack, attack models.

6.4.1 Leaked Data Structure

The input dataset plays an important role in machine learning analysis and consists of two components; trace instances and target class.

- Trace instances represent the actual data samples and carry information about the secret data. In our case, the traces are the electromagnetic signals. Each sample value within the trace represents the feature of that particular instance. For electromagnetic analysis, features are the amplitude level of the electromagnetic signals.
- Target class is a label for that particular instance which delineates the particular key bit. The whole dataset can consist of two or more classes, leading to a binary or multi-class classification problem. A group of similar classes is used as training examples to train the model. Based on these trained examples, the classification model can predict the class of the unseen data sample. The relationship between leaked data (traces) and the secret (class) depends upon the data bit transition. It is critical to label the data collected from 256-bit key encryption accurately. Labeling is handled by the attack model, as explained in 6.4.5.

6.4.2 Modes of Attack and Analysis Process

The generic machine-learning-based side-channel attack can be segregated into two modes; passive mode and active mode. The proposed attack scenario refers to the class of multivariate attacks. A large number of traces are required to train the model in passive or offline mode. However, a secret key bit can be recovered with a single trace in active mode. For passive mode, it is assumed that an attacker/adversary is in possession of a device similar to the target device, which helps in obtaining the training data to train the classification model. Active mode refers to the scenario where attackers/adversaries can collect a single trace from the device and process it to recover the key. Machine learning pre-trained models makes it practically possible to launch an attack on an embedded system in real-time.

In each of the discussed attack models, passive mode involves two steps, EM data acquisition and model selection based on the machine-learning analysis.

For passive mode training data collection, consider that adversary, possessing a device similar to target device, is capable of recording the physical leakage information l_i while N encryption operations, $E_K(P_i)$, are performed on the plain text P_i with key K_j, where $i = 0 \ldots N$, $j = 0$ to K_{len} and K_j is a vector of key bits of length K_{len}. K_{len} represents the total key length. A single key-bit value or whole key byte might be targeted at a time, as per the attack model under analysis (as explained in 6.4.5). The leakage l_i for each trace consists of two parts; leakage value due to transitions of bits and noise due to surrounding components, as given in 6.10.

$$l_i = L_{s_i} + N_R \tag{6.10}$$

L_i denotes electromagnetic leakages due to sensitive variable, that is due to the processing of data and key bits k_j. The assumption is that noise N_R is independent of the sensitive variables, and the leakages L_i contains information about the secret key. Encryption is performed using N randomly selected ECC points, processed with a fixed random key, to generate the leakage signal. The target of this attack is the least significant nibble of the key byte. First, the least significant bit does not contribute towards encryption, so is ignored, and the rest of the three bits are attacked. A key of length $K_{len} = 256$ bits is sent to FPGA for ECC encryption process initiation $E_K(P_i)$. For this 256-bit key, 31 bytes are random and fixed, and the last byte is changed only. If one bit at a time can be recovered, then the same methodology can be extended to other bits based on the windowing technique. For this analysis, the last byte's four Most Significant Bits (MSBs) are set to '0000' while the first four Least Significant Bits (LSBs) are changed and sent in ascending order, i.e., from 2^1 till $2^4 - 1$. With fixed ECC point ($P = P_X, P_Y, P_Z$), N leaked electromagnetic traces samples of length Len_{Trace} are collected. So in total, $(2^4 - 2) * N$ samples are collected. The collected samples are then stored for further processing. For the rest of the paper, the key refers to the last nibble of the 256-bit key.

After raw data samples acquisition, the next step is to label the collected sample traces, based on attack model 6.4.5. The traces are labeled and grouped together to form an input dataset for machine-learning analysis. Each newly formed dataset will contain $(2^4 - 2) * N$

samples.

One of the problems in machine-learning-based analysis can be over-fitting. In the over-fitting scenario, model trains itself for the known data so well that it fails to generalize to the unseen, unknown data. To cater to this problem, the datasets are further divided into two sets; training and testing dataset, in the ratio of 80% to 20%. The test dataset is never used in the training process, so that model can be fairly evaluated on unseen data. Moreover, cross-validation is applied to training data, which helps in overcoming the problem of over-fitting. Cross-Validation is a technique that is used to validate the model by going through almost whole data by building 10 folds on non-overlapping data. If training and testing dataset size are denoted by 'TrainRatio' and 'TestRatio' respectively, then each training and testing dataset after division will contain $(2^4 - 2) * N * (TrainRatio/100)$ samples and $(2^4 - 2) * N * (TestRatio/100)$ samples.

6.4.3 Analysis for Attack Model Selection

The acquired leakage traces L are explicitly labeled according to one of the attack models (6.4.5), thus resulting in two different traces datasets. Using raw datasets without alignment and without pre-processing, creates a realistic attack scenario. After the preparation of the datasets based on the raw electromagnetic leakage traces, further analysis is carried out using machine learning to select the best attack model. The labeled data is used as an example to train the machine-learning classification model. Based on the learned examples, the trained classification model can predict the class (key bit value) of an unseen leakage data sample, during active attack mode. For analysis, the simplest neural network (NN), Multilayer Perceptron (MLP), is considered due to its outstanding results in other domains. It should be noted that L is the concatenation of all the leakage samples for all the key bits K_j, as given by the 6.11.

$$L = L_{K_0}||L_{K_1}||\dots L_{K_j} \tag{6.11}$$

We have collected $Len_{Trace} = 10k$ electromagnetic leaked signals for our analysis,

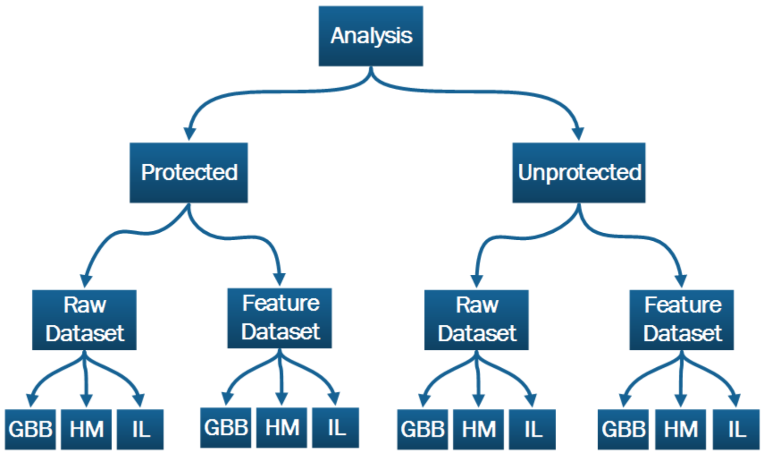


FIGURE 6.1: Analysis Hierachy

referred to as “raw datasets". Hence, the number of features per sample in raw datasets will be 10k. Firstly, the analysis is carried on raw datasets. For the second part of the analysis, new features are formed from the raw features based on time-domain and frequency-domain signal properties, e.g. Mean of the signal, Kurtosis, Median power spectral density, Frequency ratio, and Median Amplitude. These datasets are referenced as “Feature datasets". The analysis is performed for both protected and unprotected ECC algorithm versions on raw and feature datasets using both attack models. For the better understanding and visualization of the analysis hierarchy, an analysis tree is given in Fig. 6.1.

6.4.4 Training Model

Neural networks have shown tremendous performance improvement for secret key recovery. We have performed analysis using simple feed-forward artificial neural network (ANN), Multi-Layer Perceptron (MLP). MLP consists of at least three layers: one input layer, one output layer, and one or more hidden layers [128]. All layers are fully connected to each other. Neurons are one of the basic building blocks of the neural networks, which make simple decisions for complex problems simply by multiplying the assigned weight with the input. Input features are presented at the input layer to train the network using other layers' neurons and weights. The output layer neurons return the final class output. Except for the input layer, all layers consist of neurons that use nonlinear activation function to learn the patterns in the data using supervised learning backpropagation technique. Activation functions are a set of mathematical equations, which are one of the crucial importance in training a converged neural network. They activate/deactivate the output of a particular neuron/node based on the relevance of the input toward accurate model prediction. It also helps in normalizing the output in a specific range, for example, between 0 and 1, or -1 and 1. There are various activation functions that can be used during neural network training. We have tuned our network and selected the best activation function for the leakage traces under study. Further details on hyperparameter tuning are given in Sec. 6.5.3.

We have compared our MLP modeling results with other existing machine learning algorithms, including Support Vector Machines (SVM), Random Forest (RF), Naive Bayes (NB), Decision Tree (Tree), and K-Nearest Neighbors (KNN) [82, 129, 130] .

6.4.5 Attack Models

For this research, three attack models are used, which have previously not been evaluated for ECC algorithm using a machine-learning-based approach.

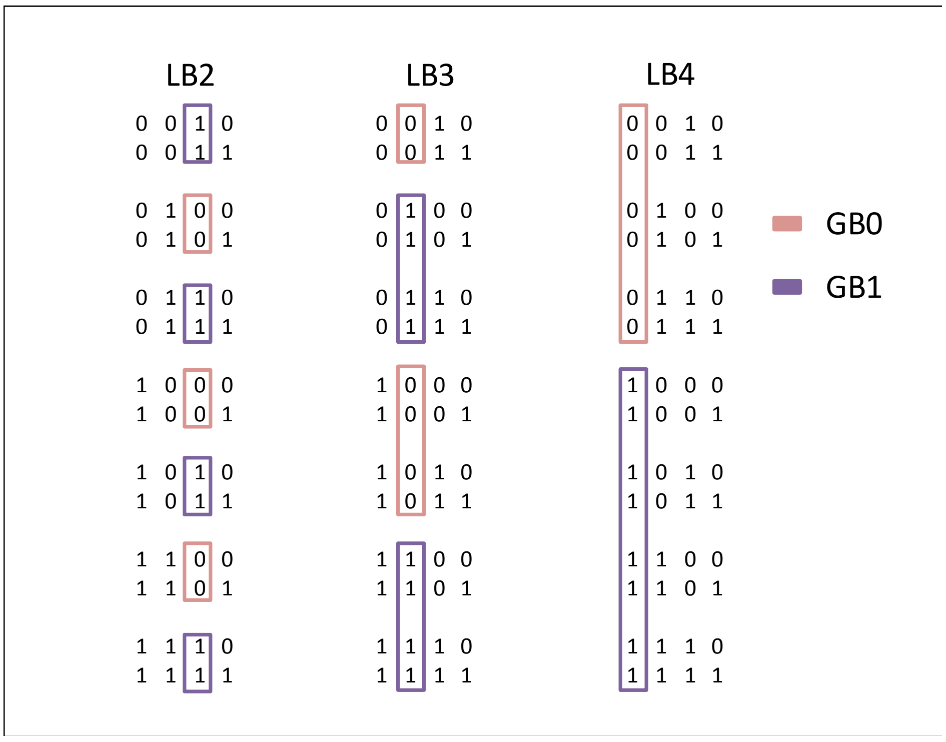


FIGURE 6.2: GBB Attack Model

Model 1 - Attack based on Bit Location

This attack model is based on the location specification of the key bit within the key vector k_j of length K_{len}. The trace samples are collected for complete key encryption rather than just a bit encryption. The hamming distance or the transitions of bits from bit 0 to bit 1 and bit 1 to bit 0 will have a different impact on the chip processing behavior, which will result in a variety of leakages. The raw collected EM traces are grouped together, based on the set key bit location, which leads to three levels of attacks, named as, LB2, LB3, and LB4. Each attack level represents an independent binary classification problem, where the target class can belong to either of the one class; $GB0$ or $GB1$. Description of attack levels is given below:

- LB2

  Samples are grouped based on the bit at 2^{nd} location. EM leakage traces (l_i) collected with key vector k_j having bit ‘1’ at LSB 2^{nd} location are marked as $GB1$ and others are marked as $GB0$.

- LB3

  Samples are grouped based on the bit at 3^{rd} location. EM leakage traces (l_i) collected with key vector k_j having bit ‘1’ at LSB 3^{rd} location are marked as $GB1$ and others are marked as $GB0$.

- LB4

  Samples are grouped based on the bit at 4^{th} location. EM leakage traces (l_i) collected with key vector k_j having bit ‘1’ at LSB 4^{th} location are marked as $GB1$ and others are marked as $GB0$.

We have used a similar approach against power analysis on FPGA power leakages and achieved successful attack outcome [126]. The labeling/grouping approach used for this attack model is shown in Fig. 6.2. This approach is named as “Group Bit Bins Model", due to the grouping of secret scalar bits instead of bytes, and will be addressed as ‘GBB’ throughout the rest of the paper.

TABLE 6.1: Labels used for Hamming Weight Approach

Key	Class Label	Key	Class Label
0010	HM1	0011	HM2
0100	HM1	0101	HM2
0110	HM2	0111	HM3
1000	HM1	1001	HM2
1010	HM2	1011	HM3
1100	HM2	1101	HM3
1110	HM3	1111	HM4

Model 2 - Attack based on Hamming Weight

Hamming weight is referred to as the number of bits set in a data word. In the m-bit processor, binary word data is coded as D_W, as given by 6.12, and the hamming weight $H(D_W)$ is given by 6.13. The hamming weight $H(D_W)$ value would be between 0 and $m-1$.

$$D_W = \sum_{n=0}^{m-1} d_j 2^j \tag{6.12}$$

$$H(D_W) = \sum_{n=0}^{m-1} d_j \tag{6.13}$$

D_W will have average hamming weight $\mu = m/2$ and variance $\sigma^2 = m/4$, if D_W contains m independent and uniformly distributed bits. Leakage traces having a uniform number of bits might have similar leakages, which can contribute to the information of retrieving the bits in secret information, hence reducing the search space for the ultimate key recovery. Moreover, for power analysis, the Hamming weight model is based on the fact that power consumption, and hence electromagnetic emanations, are correlated with the transition of bits. Hence if there is a transition from bit 0 to 1 or from bit 1 to 0, then power consumed by the device would be different, leading to different electromagnetic emanations out of the device, which might leak information about the relationship of the key, input data, and output data.

The EM leakage traces are labeled and grouped based on the key's hamming weight. As the key ranges from 2^1 to 2^4-1, four labels HM{i} will be used for each nibble, as shown in Table 6.1. With four labels/classes, it represents a multi-classification problem. It should be noted that for our analysis, the first 31 bytes have been selected randomly and are fixed throughout the data collection process. So they contribute similar leakage to each trace. The only difference in traces would be based on the key vector bits which have been changed. This “Hamming Weight based model" will be addressed as ‘HM’ throughout the rest of the paper.

Model 3 - Attack based on Key Identity

In this model, the attack is launched on the key byte in contrast to the key bit and key nibble as done in the attack model 1 and attack model 2 approach, respectively. In other words, the machine learning model will try to classify each key byte by forming a multi-class classification problem. The traces are collected for each key byte encryption, and the same key byte value is assigned as a label to the respective trace. For example, the trace sample collected for key bit vector "0000 0010" will be labeled as ’2’, key bit vector "0000 0011" will be labeled as ’3’ and so on. This approach has been used to analyze the effect of the same key bit vector on the leakage information. N plaintexts (ECC points) are used to perform encryption operations with the same key bit vector (key byte), so N samples will be labeled for each key bit vector. The dependency between keys and data will lead to leaking information about the key. As the labels are the same as key bytes hence the name "Identity Label based Model" and will be addressed as ’IL’ throughout the rest of the paper.

6.5 Experimental Setup

For experiments, side-channel electromagnetic radiations are collected from the FPGA (Kintex-7), while ECSM is performed. The collected raw data is then processed further to prepare for machine-learning classification.

6.5.1 ECC Implementation on FPGA

A low power consumption design is used for the attack analysis. The inputs are the scalar k, and a point on the elliptic curve $P(X, Y, Z)$. The output is the point $Q = k \cdot P$ on the elliptic curve. The scalar factor key k is stored in the internal RAM and can be changed via software command.

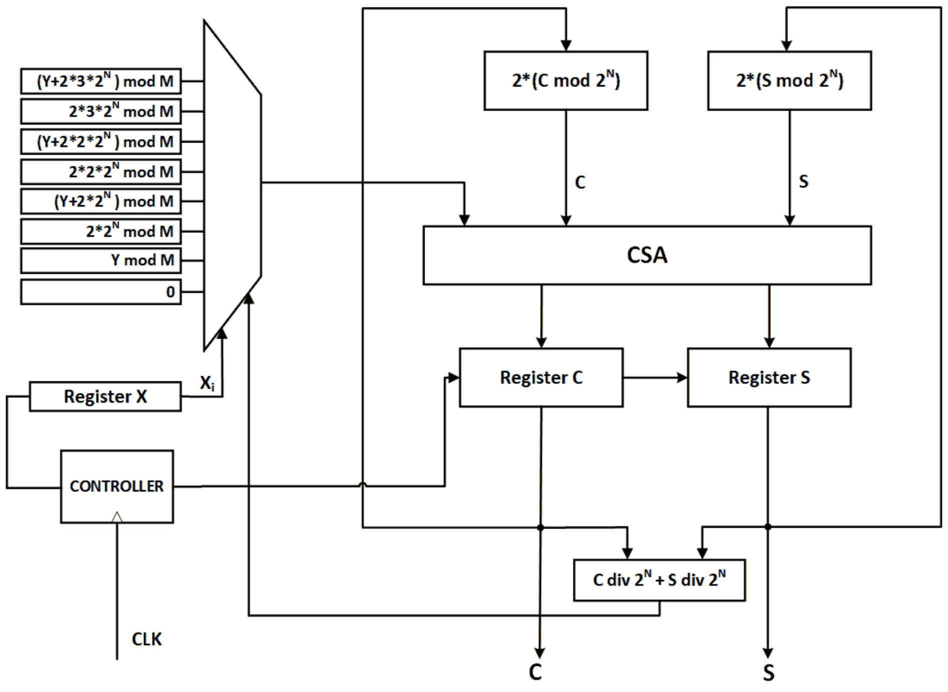


FIGURE 6.3: Interleaved Modular Multiplier

A point doubling is done, followed by a point addition at every stage i, but the result of the point addition is used only when the i^{th} bit of the scalar k is '1'. In this method, N_{OPS} ECPD and ECPA operations are required (here N_{OPS}=256). For the fast and area-efficient implementation of the multiplier, we use just one CSA adder and a lookup table. The structure of our design is shown in Fig. 6.3.

$$\begin{aligned} S_i &= X_i \oplus Y_i \oplus C_i, \\ C_{i+1} &= (X_i \cdot Y_i) \vee (X_i \cdot C_i) \vee (Y_i \cdot C_i), \\ C_0 &= 0. \end{aligned} \tag{6.14}$$

As depicted in Fig. 6.3 the modular multipliers take one bit of input X at every clock cycle. The loop latency is N_{OPS} clock cycles in total. There are three main operations; lookup table calculation, final registers output addition, and modulus p comparison and subtraction. Each operation takes one clock cycle, and the whole calculation takes 59 $(N+3)$ clock cycles.

ECPA is performed using one modular operation in parallel, and even modular multiplications are done in $7(N + 3) + 4$ clock cycles. Using just one modular reduction unit, ECPD can be performed in $7(n+3)+2$ clock cycles. The point addition uses three modular multiplier units to calculate point $Q + P$ on the Elliptic curve in parallel. Sixteen modular multiplications are done in seven stages. The latency of point addition is reduced to $7(N + 3) + 5$ clock cycles.

6.5.2 Data Collection - Hardware and Software

The acquisition of leakage data from the hardware implementations is a cumbersome task. Several hardware-based and software-based setups have been proposed for data collection [80] [131]. For data acquisition, hardware principally designed for side-channel evaluation, with mounted Kintex-7 FPGA (SAKURA) is used. FPGA operating frequency is 24MHz. Random ECC points are generated for encryption. Encryption is performed using $N = 100$ random ECC points, and thus $N = 100$ raw leaked signals, emitting out of Kintex-7 FPGA, are captured using Chip-Whisper EM Probe and Tektronix MDSO Oscilloscope, with a sampling rate of 400 microseconds. A 256-bit key is sent from the PC to the FPGA device. When the key is received at the FPGA device, a trigger signal is sent by the FPGA device to the PC application indicating the encryption process's initiation. Based on the trigger signal, the data collection process begins in C# application, which utilizes MATLAB libraries to connect to the oscilloscope. The whole data collection procedure is developed as an automated setup for an uninterrupted collection of raw leaked samples.

6.5.3 Data Analysis Specification and HyperParameter Optimization

Raw leaked samples are labeled using bespoke MATLAB code using three mentioned approaches, thus forming raw datasets. Moreover, new features are formed using MATLAB bespoke code as well. Machine learning analysis is performed using Orange software and Python Sklearn Libraries using resources of the AWS cluster supercomputer. For neural network MLP, 5-fold validation is used, and the tested hyperparameters are given in Table 6.2.

TABLE 6.2: Performance metrics for Protected and Unprotected ECC using GBB Attack Model

Parameter	Value Range
Activation	Relu, Identity, Logistic, tanh
Solver/Optimizer	ADAM, SGD, LBFGS
Epochs	300-700

6.6 Results and Analysis

Machine learning analysis results are interpreted in the form of a confusion matrix. The confusion matrix is a matrix that shows the number of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). For analysis in this paper, the following performance metrics are considered; accuracy, recall, precision, F1 Score, and Receiver Operating Characteristic (ROC). Accuracy is the ratio of the number of correct predictions/(TP) to the total number of predictions (TP+TN), recall is the ratio of TP to (TP+FP), precision is the ratio of TP to (TP+FN), and ROC is the graphical representation of True Positive Rate (TPR) and False Positive Rate (FPR). For balanced datasets, accuracy and ROC can correctly portray the results. However, for imbalanced datasets, recall, precision, and F1-Score are used to depict the trained model's true performance. Below are the results for ECC electromagnetic leakage data analysis, performed on raw datasets and feature datasets, prepared using GBB, HM, and IL attack models.

6.6.1 Analysis on Raw Datasets

This section presents the results on the raw traces datasets.

Group Bit Bins(GBB) Model

The results, based on all performance metrics, for protected and unprotected ECC algorithm analysis using GBB model LB4, are given in Fig. 6.4.

For recovering the key bit at location 4 (LB4), accuracy 91% is achieved with Neural Network MLP (without hyper parameter optimization), with F1 score, precision, and recall as 0.886, 0.890, and 0.875, if no countermeasure is used in the design. However, 89.5% and 88.7% accuracy is achieved with SVM and kNN. High recall and precision show that the percentage of true positives and true negatives are higher as compared to false positive and false negatives. The test accuracy is very close to training classification accuracy, which shows that the model is not over-fitting. It is seen that the accuracy was reduced by 16%, 17%, and 22% for analysis with NN, SVM, and kNN if the masked key countermeasure is used. All other classification algorithms performed worst than the mentioned three.

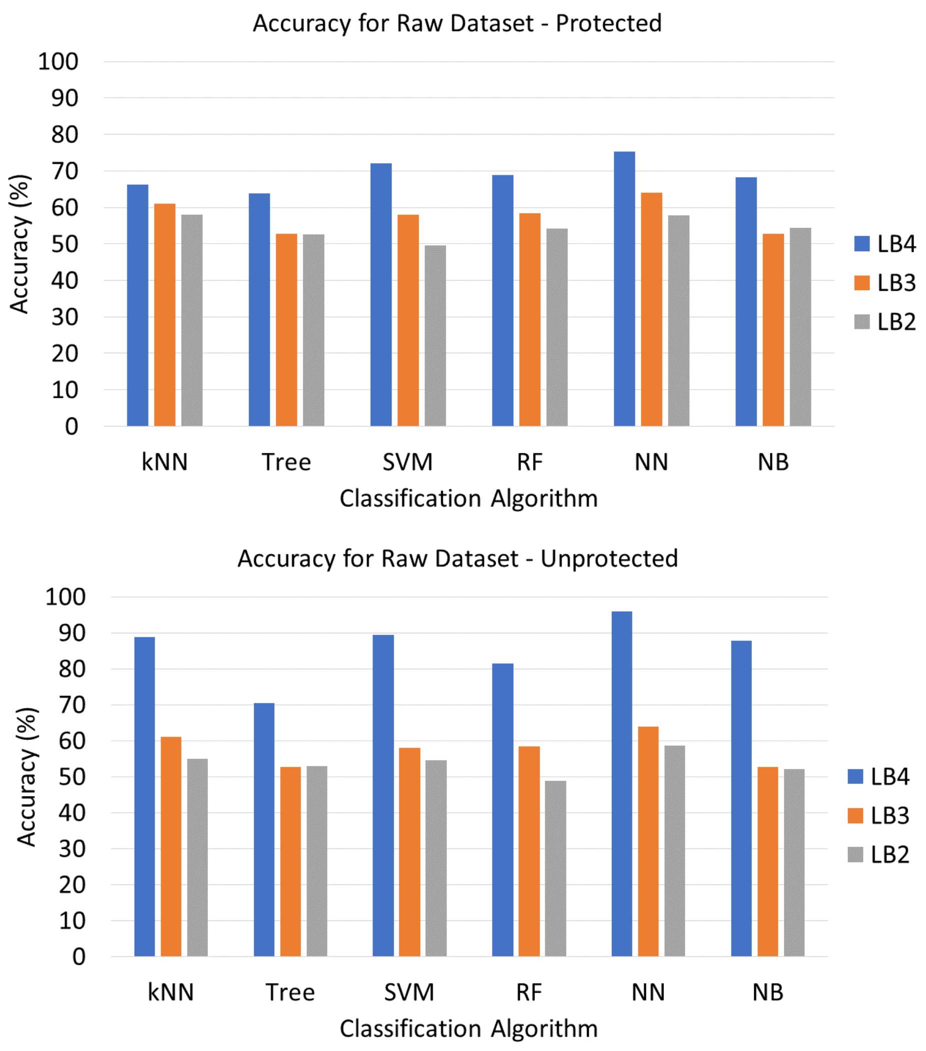


FIGURE 6.4: Accuracy for raw datasets for protected and unprotected Implementations

For recovering a key bit at location 3 (LB3), in the unprotected version, the accuracy of 56.5% is achieved with kNN, which is the highest among all the algorithms. The accuracy of 53.6% is achieved with NN. F1 score, precision, and recall is higher than 0.5. However, for the protected version, there is 5% increase in the accuracy with better F1 score, precision, and recall. It can be seen from Fig. 6.5 that the area under the curve (AUC) in ROC curve for

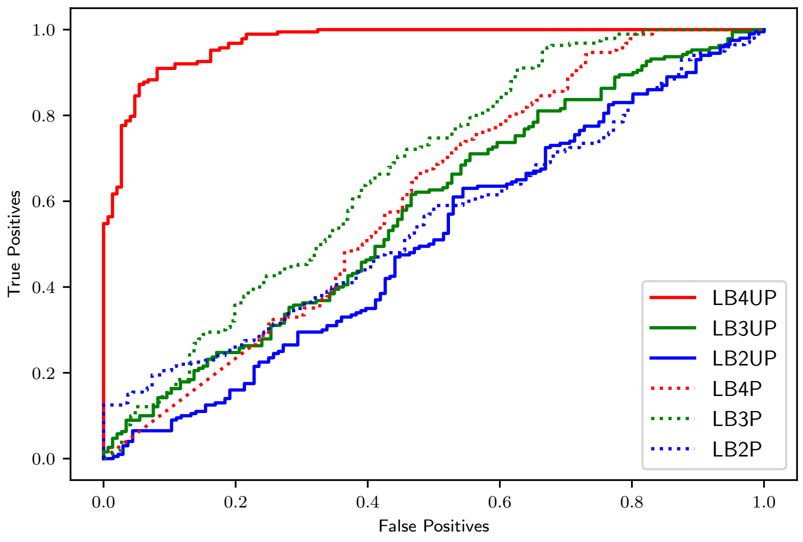


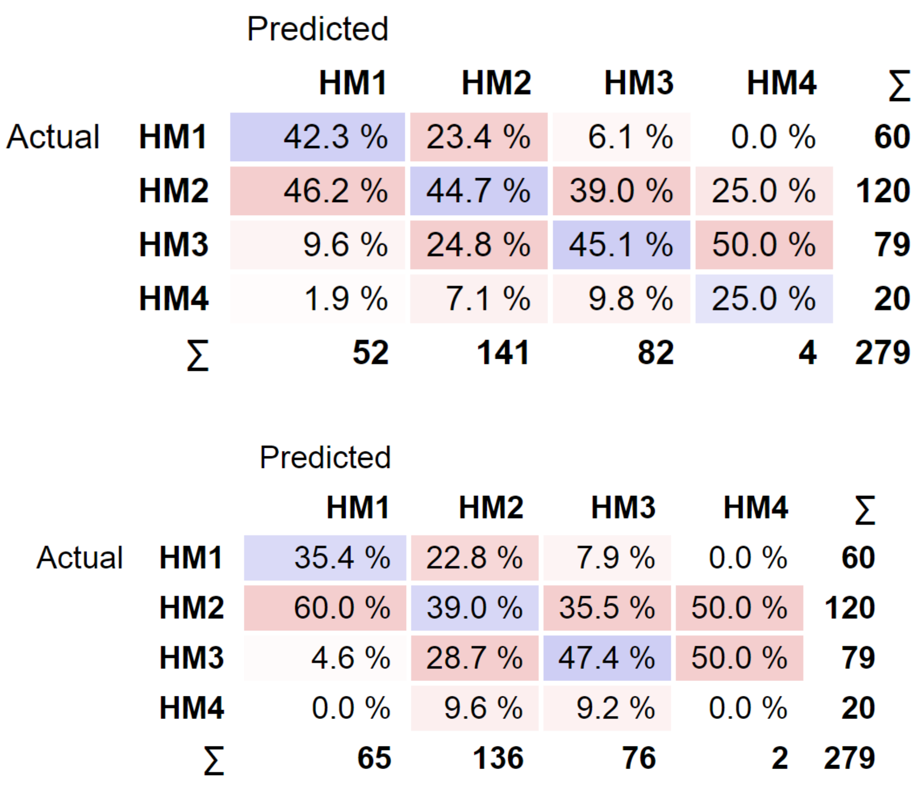
FIGURE 6.5: ROC for analysis using GBB model for protected and unprotected

LB3 is not adequate to justify the model as the best-trained model. It shows that the ratio of classifying secret bits correctly is higher as compared to incorrect classification i.e., bits that are ‘1’ will be classified correctly as ‘1’; however, the probability of getting false negatives is still high.

For recovering key bit at location 2 (LB2), the accuracy of 55% and 51.8% is achieved with kNN and NN, if no countermeasure is used. With the countermeasure protected version, accuracy decreases by 1-2% while training with all the classifiers. However, for NN, it is increased to 57.7%, which implies that the key bit at location 2 can be recovered with more than 50% confidence in both cases of with and without countermeasure protected ECC. Results show that the key recovery probability is lower for the least significant bits (LSB).

Hamming Weight based Model

This section presents the analysis results on the Hamming weight-based model. Four classes are formed for the data collected with the hamming weight-based model. For unprotected and protected cases, the accuracy obtained for HM1, HM2, HM3, and HM4 is in the range of 69-75%, 48-56%, 61-68%, and 62-92%, respectively. As the hamming weight dataset is an imbalanced/skewed dataset so accuracy is not the reliable performance metric and can lead to false conclusions. In this case, it is essential to take F1-score, precision, and recall into consideration as well. For HM1, the highest F1-score is achieved with kNN and NB; for HM2, RF and NN gave better F1-score, while for HM3 NN seems to be the best choice. For HM4, the F1-score is 0.05. HM4 needs further investigation by considering the traces for one byte rather than one nibble, which is out of the scope of this study. The confusion matrix, for NN, is shown in Fig. 6.6. SMOTE, the over-sampling technique, has been applied to balance the dataset, but no significant improvement is observed.



		Predicted				
		HM1	**HM2**	**HM3**	**HM4**	**∑**
Actual	**HM1**	42.3 %	23.4 %	6.1 %	0.0 %	**60**
	HM2	46.2 %	44.7 %	39.0 %	25.0 %	**120**
	HM3	9.6 %	24.8 %	45.1 %	50.0 %	**79**
	HM4	1.9 %	7.1 %	9.8 %	25.0 %	**20**
	∑	**52**	**141**	**82**	**4**	**279**

		Predicted				
		HM1	**HM2**	**HM3**	**HM4**	**∑**
Actual	**HM1**	35.4 %	22.8 %	7.9 %	0.0 %	**60**
	HM2	60.0 %	39.0 %	35.5 %	50.0 %	**120**
	HM3	4.6 %	28.7 %	47.4 %	50.0 %	**79**
	HM4	0.0 %	9.6 %	9.2 %	0.0 %	**20**
	∑	**65**	**136**	**76**	**2**	**279**

FIGURE 6.6: Confusion Matrix 1.Unprotected (UP) algorithm, 2.Protected (P) algorithm

Identity based Label (IL) Model

For an unprotected version of the ECC algorithm, most of the trained models give poor results for all the key classes except Key2. For class Key2, secret key can be retrieved with 70.6% accuracy with SVM.

For countermeasure protected ECC, it is observed that the key classes cannot be distinguished with high probability. In the majority of the cases (for key2 to key15), accuracy is

!h

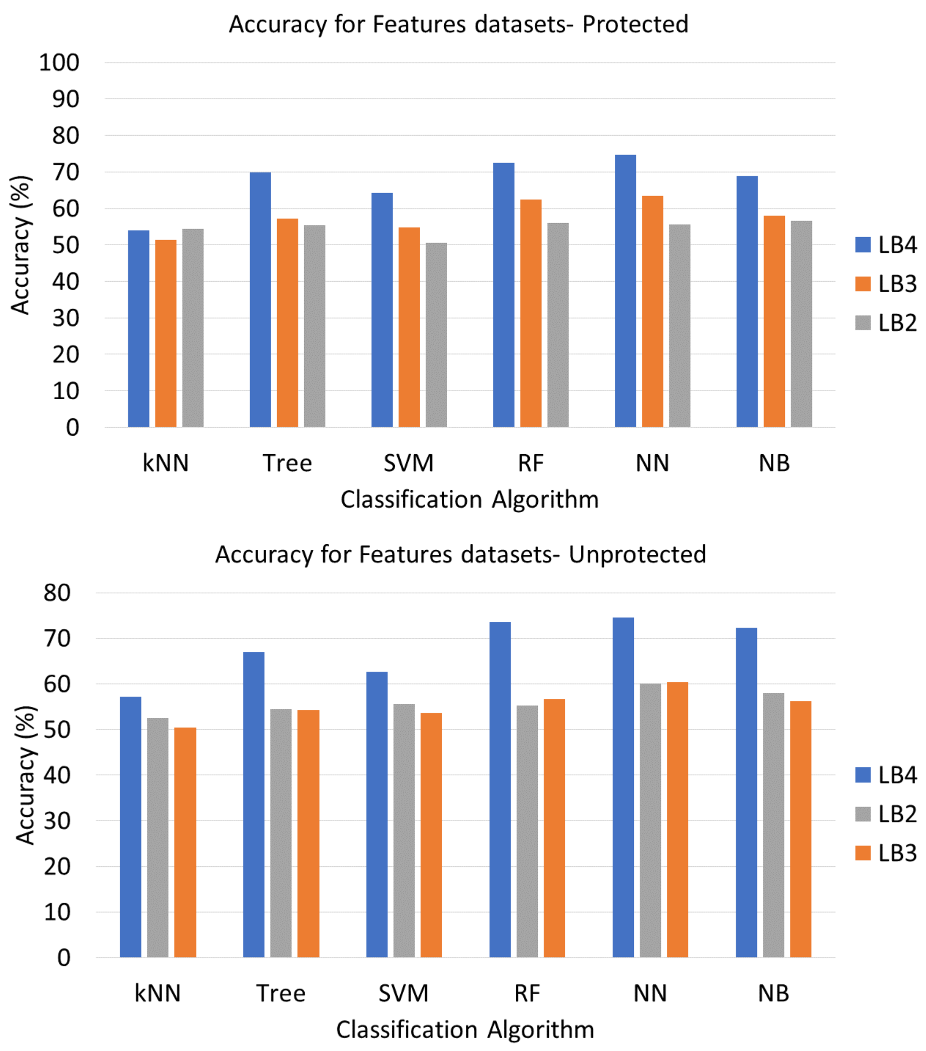


FIGURE 6.7: Accuracy for features datasets for protected and unprotected Implementations

above 90%, but the F1-score, precision, and recall are very poor, which demonstrates that the model has predicted more false positives and false negatives as compared to true positives and true negatives.

6.6.2 Analysis on Reduced Feature Datasets

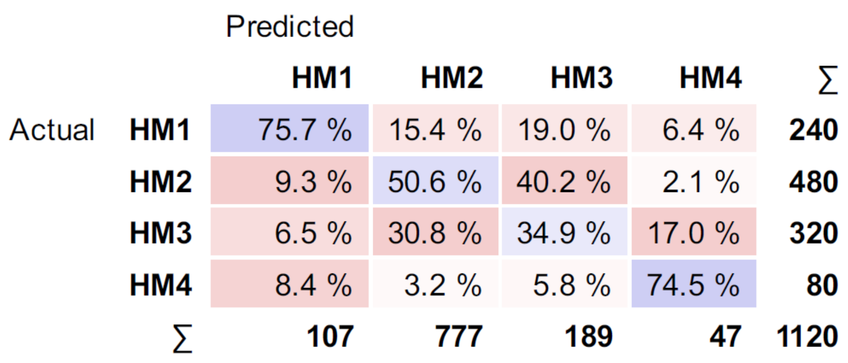
This section presents results on the reduced feature datasets.

Group Bit Bins(GBB) Model

To classify the key bit at location 4 (LB4), the highest accuracy, 74.8% is obtained with NN, for both protected and unprotected implementations, as shown in Fig. 6.7. However, for the rest of the classifiers, the accuracy is around 53-74%. The best performance is achieved, for both protected and unprotected implementations, using NN. For the rest of the attack levels (LB3 and LB2), accuracy has reduced significantly.

Hamming Weight based Model

For the attack model using a hamming weight, the resulting feature dataset is imbalanced. Hence accuracy alone can be deceptive. Other parameters, like F1-score, recall, and precision, have been closely monitored. HM1, HM2, HM3, and HM4 can be recovered with 75.7%, 50.6%, 34.9%, and 74.5%, respectively, as shown in Fig. 6.8. It has been observed that other parameters are very low, which means model did not fit well. This shows that while performing feature engineering, some useful information is lost, which is helpful for distinguishing the secret key.



		Predicted				
		HM1	HM2	HM3	HM4	∑
Actual	HM1	75.7 %	15.4 %	19.0 %	6.4 %	240
	HM2	9.3 %	50.6 %	40.2 %	2.1 %	480
	HM3	6.5 %	30.8 %	34.9 %	17.0 %	320
	HM4	8.4 %	3.2 %	5.8 %	74.5 %	80
	∑	107	777	189	47	1120

FIGURE 6.8: Accuracy on Feature Dataset

Identity Based Approach (IL) Model

For IL approach without countermeasure, Key2 can be recovered with accuracy 96-98% with NN, RF, Tree, and SVM. However, for other keys from key3 to key15, no information can be extracted with any other classification algorithm. However, for IL with a countermeasure, all keys from key3 to key15 have low classification score. However, for key2, it also reduced by 50-60% approximately. This shows that the IL-based attack model cannot recover the key from countermeasure protected implementations.

6.6.3 Discussions

Overall, in our presented analysis for the attacks models, we have observed that different models influence the machine-learning model performance differently. It has been seen that in contrast to the Hamming weight and Identity label models, GBB model provides an effective attack strategy which helps in retrieving secret key with a better probability using less number of data instances per target class (in range 100-130 only). This is due to the reason that GBB is based on a binary classification problem and each target class has an adequate number of traces. However, the Hamming weight model and identity label based attacks form a multi-class classification problem; hence they require more data for each target class. We are using a limited number of traces by assuming that the adversary is constrained in obtaining a large set of leakage measurements. This assumption has been set to analyze a realistic constraint environment in the IOT environment. However, more data can be collected from the device, or data can be generated by using existing synthetic data over-sampling or generative adversarial (GAN) techniques [132]. Moreover, we believe, for investigating machine-learning-based electromagnetic analysis using the Identity Label model, pre-processing using filters and wrapper methods might help in improving the results. As the scope of this paper is to scrutinize the machine-learning-based attack models, so filter based pre-processing is left for future work. The analysis also shows that the existing countermeasure is not providing the required level of immunity against machine learning-based side-channel attacks.

The presented GBB attack can be extended to attack all consecutive secret key nibbles by using appropriate windowing techniques. In all the presented attack scenarios, simple neural network, Multi-layer Perceptron (MLP), shows better performance as compared to other machine learning algorithms due to its ability to learn nonlinear relationships between the input and target class even in the presence of noise. Based on MLP performance, further analysis was carried and hyperparameters were tuned. The following are the combination of the hyperparameters that presented the best trained MLP model; Relu as activation function, Adam as an optimizer, with 100 hidden layer neurons and 300 Epochs. The accuracy increased by 3-5% in almost all cases. In GBB, the LB4 key bit can be recovered with 96% accuracy. Only the best-trained model is reported in Fig. 6.4.

It has also been observed that the performance of machine-learning-based attacks are better on the raw dataset as compared to on a reduced set of features. This illustrates that useful information is lost while processing features. The neural network learns the details from the raw data in an efficient manner, due to its inherent design capabilities of learning from noise. The results lead to the fact that if the raw samples can give accuracy between 50-70%, then the trained model with a specific portion of the samples of interest (using windowing techniques on 3% trailing samples) might improve accuracy significantly. This analysis is left for future work. The purpose of this research is to study the evaluation of the model with traces that have relevant information in only 3% of trailing samples, 96% samples are representing noise for that particular bit evaluation.

In the existing literature, symmetric cipher leakages have been extensively analyzed using machine learning-based side-channel attacks (ML-SCA) [23, 133]. However, few public-key leakage datasets are analyzed for ML-SCA based analysis [71, 105, 125, 126]. Saeedi et al. have presented the performance results for the secret key recovery from the same ECC algorithm FPGA implementations, in which the best performing neural network recovered the secret key bit with 88% accuracy [105]. However, our presented model outperforms, and the key bit can be recovered with 96% accuracy.

6.7 Conclusion

Embedded chip security is critical in embedded devices used in the Internet of things (IoT) based systems. A side-channel attack is one of the practical attacks that can exploit the side-channel leakages due to mathematically secure algorithms' insecure implementations.

Machine learning has improved the side-channel attack efficiency, and attacks can be launched successfully on unprocessed raw data samples in real-time. This paper presents three attack models for machine-learning-based electromagnetic side-channel analysis on the cryptographic chips, used in embedded systems, and an evaluation framework for the attacks. The proposed attack evaluation process performs machine-learning analysis on captured electromagnetic leakage data from cryptographic algorithm implementation on Field Programmable Gate Arrays, using six classification algorithms. Moreover, quantitative analysis is performed for the selection of the best attack model. Based on our experiments, on the three attack models, we were able to recover the key with up to 96% accuracy using GBB attack model. The high accuracy illustrates the vulnerability of the implementation against the GBB attack. We have also investigated the effect of feature engineering techniques on the attack models. Secret key from unprotected ECC design implementations can be recovered with better accuracy. However, ECC implementations, secured with masked key countermeasures, give poor classification results. The accuracy results are between 50-75%, which shows that the countermeasure does not provide the required security level. The attacks are performed in a realistic environment, on raw and reduced data samples, with relevant information in 3%samples (96% noise) only. In light of the findings, we expect system designers to propose efficient countermeasures for such devices at the design level for optimum information security.

Chapter 7

Machine-Learning assisted Side-Channel Attacks on RNS-based Elliptic Curve Implementations using Hybrid Feature Engineering

This chapter is an adapted version of a submitted article (Publication V). In previous chapters, in addition to unprotected algorithm versions, masked key countermeasure is also considered. For asymmetric ciphers, particularly ECC, the Residue Number System (RNS) is considered one of the strongest countermeasures. In this chapter a systematic machine learning-based evaluation methodology of RNS-ECC cryptosystem is presented and is evaluated on the electromagnetic leakages from an implementation of Montgomery Powering Ladder algorithm on ARM processor using machine and deep learning algorithms. Furthermore, a hybrid feature engineering methodology is proposed to select the most contributing features. For recovering the secret key from the asymmetric algorithm, two attack scenarios i.e. location dependent and data dependent attacks are investigated.

7.1 Abstract

Side-channel attacks based on machine learning have recently been introduced to recover the secret information from software and hardware implementations of mathematically secure algorithms. Convolutional Neural Networks (CNNs) have proven to outperform the template attacks due to their ability of handling misalignment in the symmetric algorithms leakage data traces. However, one of the limitations of deep learning algorithms is the requirement of huge datasets for model training. For evaluation scenarios, where limited leakage trace instances are available, simple machine learning with the selection of proper feature engineering, data splitting, and validation techniques, can be more effective. Moreover, limited analysis exists for public-key algorithms, especially on non-traditional implementations like those using Residue Number System (RNS). Template attacks are successful on RNS-based Elliptic Curve Cryptography (ECC), only if the aligned portion is used in templates. In this study, we present a systematic methodology for the evaluation of ECC cryptosystems with and without countermeasures against machine learning side-channel attacks using two attack models. RNS-based ECC datasets have been evaluated using four machine learning classifiers and comparison is provided with existing state-of-the-art template attacks. Moreover, we analyze the impact of raw features and advanced hybrid feature engineering techniques, along with the effect of splitting ratio. We discuss the metrics and procedures that can be used for accurate classification on the imbalance datasets. The experimental results demonstrate that, for ECC RNS datasets, the efficiency of simple machine learning algorithms is better than the complex deep learning techniques when such datasets are not so huge.

7.2 Introduction

Side-channel attacks (SCA) constitute an ever evolving technique of recovering secret information from the exploitation of the physical leakage that appears in cryptographic implementations (e.g. power consumption, electromagnetic emanations, timing, vibrations leakage [75, 134, 135]). From an information-theoretic point of view, profiled template attacks are one of the most powerful SCAs. The attacker in such attacks is assumed to have access not just to the target device, but also to an open copy of it for the profiling phase. Having control of the secret information, he creates a leakage profile that he can later use to retrieve an unknown secret (not under his control) from its collected leakage traces during a cryptographic operation [136]. Recently, machine learning (ML) based side-channel attacks have been proposed, as direct extension of template attacks, extending the concept of leakage templates into trained ML models. These models can be used for secret information predictions, thus providing an interconnection between the SCA and ML research field [20, 70, 137]. Furthermore, several researchers showed that machine learning and deep learning (DL) techniques, like Convolutional Neural Networks (CNNs) outperform traditional side-channel attacks since they are able to learn from misaligned data and, therefore, eliminate the need of pre-processing [23, 73].

CNNs can improve the performance of the attacks, however, various factors can introduce hindrance in the accurate classification using deep learning algorithms. Firstly, a huge amount of leakage traces (samples/instances) are required for training such a model. Secondly, the traces are collected using high sampling rate to capture the minor details in the leakages which results in high-dimensional noisy data traces/samples which might contain redundant features. Researchers have focused on reducing the number of traces required to retrieve the secret information while applying machine learning to SCA. However, the optimal number of samples required to achieve desired accuracy, known as sample complexity, grows rapidly (exponentially in some scenarios) with the higher number of noisy instances containing irrelevant features [138]. Selecting/extracting a small subset of features, can indeed reduce the sample size required to achieve a good problem generalization with the particular machine learning algorithm. Picek et al. have evaluated the impact of various feature engineering techniques on the profiled side-channel attacks on AES [139]. Mukhtar et al. [126], have presented side-channel leakage evaluation on protected and unprotected ECC Always-double-and-add algorithm using machine learning classifiers and proposed to use signal properties as features. Zaid et al. in [74] have shown the insights for the selection of features while building an efficient CNN architecture for side-channel attacks.

In the recent literature, there is a considerable amount of research works focused on ML and DL SCAs for symmetric-key algorithms. However, only few researchers have tumbled with the increased complexity and high number of samples in traces that exist in public-key cryptosystems [71, 73, 125] identifying the presence of a gap of attack analysis on public-key cryptographic algorithms. The few ML/DL based evaluation analysis that exists for public-key cryptography, do not yet consider the evaluation of cryptosystems under the presence of strong SCA countermeasures. According to the *no-free lunch theorem*, no two datasets will show the same results for the same classifier [115]. Thus, the ML analysis on SCAs provided for some symmetric-key implementations and even public-key cryptographic implementations (e.g. RSA) won't be of much use in other settings like RNS-ECC implementations.

Additionally, the complexity of the ECC computations makes the well known ML analysis concerns of under-fitting and over-fitting, occurring due to bias and variance in data, very crucial. In fact, the machine might learn from data so well or so poorly, that it is unable to generalize on the unseen data, thus making the training accuracy deceiving. To cater these concerns, an optimal number of data traces need to be identified, proper data splitting strategy must be chosen, and appropriate feature engineering techniques must be administered. These activities, though, are hard to specify as the cryptographic computations become more elaborate and include strong SCA countermeasures. Thus, all the above issues highlight the need for a concrete methodology to analyze ECC implementation datasets for ML-based profiling SCAs especially when such implementations have dedicated, strong, SCA countermeasures.

Elliptic curve cryptographic primitives have been widely studied for the efficiency and SCA resistance. Therefore, many efficiency enhancement techniques and SCA countermeasures have been devised. Among them, several researchers have proposed using Residue Number System (RNS) arithmetic representation as a way of decreasing scalar multiplication computation delay [140, 141] by transforming all numbers to the RNS domain before performing finite field operations [142]. In addition, RNS can be used for producing strong SCA countermeasures that can withstand simple and advanced SCAs [142] using the Leak Resistant Arithmetic (LRA) technique. Recently, a comprehensive study on RNS ECC implementations for Edwards Curves [143], using the Test Vector Leakage Assessment (TVLA) techniques [144], showed that the combination of traditional SCA countermeasures like Base Point randomization, scalar randomization etc. when combined with LRA based RNS countermeasures can considerably reduce information leakage. Also in [143] it was proven that profiled template attacks on RNS SCA protected implementation are partially successful (using location dependent and data dependent leakage attacks) thus implying that more powerful attacks may be able to compromise the RNS SCA countermeasures [143, 145].

In this paper, a concrete methodology for Machine Learning SCA resistance of RNS-based ECC cryptosystems is proposed, realized in practise and analyzed in depth using various ML model algorithms and feature engineering techniques in order to achieve optimal results. This study could serve as a guideline for RNS-based RSA implementations as well. The methodology is able to retrieve attack vulnerabilities even against noisy RNS-based implementations that include RNS and traditional SCA countermeasures. More specifically, we focus our evaluation plan on location dependent and data dependent leakage attacks (both with and without countermeasures). Our analysis includes several restrictions like misaligned and imbalanced datasets, as well as restricted number of traces. Furthermore, a comparison of attack models using four machine learning classifiers is made. We also discuss the criteria on the selection of optimized hyperparameters for each of the classifier. Once the optimally tuned model parameters are selected, then further feature engineering techniques are applied to analyze the attack performance with reduced number of features. In scenarios with limited number of samples in the datasets, data splitting ratio can be one of the attack performance affecting factors. Finally, we analyze the effect of three data splitting ratios on the overall attack performance. Analytically, the paper novelty is the following:

- A six stage methodology for launching a practical machine-learning based side-channel attack is proposed. Our analysis is based on assessing the SCA resistance of an RNS-based ECC implementation with and without countermeasures. For the first time in research literature, the effectiveness of a combination of RNS and traditional SCA countermeasures on an RNS ECC implementation against machine-learning based side-channel attacks is presented.

- Machine learning based side-channel attacks are presented for location and data dependent leakage models using four machine learning classifiers. For each classifier, hyperparameter tuning has been performed to extract the best-trained model for the underlying problem. Results are presented using standard machine-learning evaluation metrics. The implications of relying on the classification accuracy alone, in case of imbalance data, are also discussed.

- Various state-of-the-art hybrid feature engineering techniques, which have proven to offer performance improvement in other domains, are tested on side-channel leakage traces from RNS-based ECC implementations. Three hybrid feature engineering approaches are proposed in order to handle the complexity of public-key cryptographic trace. The impact of dimensionality reduction along with the filter and wrapper feature selection methods, is observed. A comparative analysis is performed to show the performance improvement due to the proposed hybrid method as compared to the filter, wrapper and dimensionality reduction methods alone.
- This work also investigates the effect of data splitting and validation folds on the attack efficiency for RNS-ECC dataset.
- An RNS-based ECC implementation is one challenging dataset, due to the RNS operation intrinsic parallelism. For RNS-based implementations, existing traditional template attacks are successful only if the aligned portion of the traces is used for the attack. This limitation makes the attack difficult to launch. However, in this study, quantitative analysis is performed to analyze the success of the machine learning-based attack by using the full trace length and the aligned part for training the model.

The rest of the paper is organized as follows. Section 7.3 presents the classifiers used for evaluation along with the algorithm under attack. Section 10.4 explains the attack methodology along with other evaluation strategies and datasets used for evaluation. Section 7.5 presents the results on RNS-based ECC leakage datasets. Section 10.7 concludes the paper.

7.3 Preliminaries and Related Literature

7.3.1 Potentials of RNS as Side-Channel Attack Countermeasure

The Residue Number System (RNS) is a non-positional arithmetic representation, where a number X is represented by a set of individual n moduli x_i ($X \rightarrow^{RNS} X : (x_1, x_2, ...x_n)$) of a given RNS basis $\mathcal{B} : (m_1, m_2, ...m_n)$ as long as $0 \leq x < M$, where $M = \prod_{i=1}^{n} m_i$ is the RNS dynamic range and all m_i are pair-wise relatively prime. Each x_i can be derived from x by calculating $x_i = \langle x \rangle_{m_i} = x \pmod{m_i}$. Since it can effectively represent elements of cyclic groups or finite fields there is merit in adopting it in elliptic curve underlined finite field operations. RNS hardware implementations of Montgomery multiplication for elliptic curves [146] and RSA [147] showed that RNS usage can increase scalar multiplication efficiency. Furthermore, RNS can be used to design SCA countermeasure as is observed in several research papers, for instance Bajard et al. in [142, 148], Guillermin [140], Fournaris et al. [149, 150]. RNS parallel processing of finite field operations apart from speed offers also different representation of the elliptic curve points, which may reduce SCA leakage. Also, RNS is a non positional system (single bit change in an RNS number's moduli can lead to considerable changes in the binary representation of finite field element) which intrinsically increases noise in the computational process [141]. Furthermore, in [142], the Leak Resistant Arithmetic (LRA) technique was proposed where it was proven that by creating a big pool of RNS basis moduli (at least $2 \times n$), then randomly choosing some of them to act as an RNS basis for representing finite field elements and a specific computation flow, randomly permuting this RNS basis, can be a potent SCA countermeasure. LRA has been applied to modular exponentiation designs in two ways, either by choosing a new base permutation once at the beginning of each scalar multiplication or by performing a random bases permutation once in each scalar multiplication round [150]. In this paper, the second approach is adopted.

7.3.2 RNS-based ECC Scalar Multiplication

The ECC scalar multiplication algorithm evaluated in this paper is based on a variation of Montgomery Power Ladder (MPL) for Elliptic Curves on $GF(p)$ [151]. Algorithm 7.3.2 uses the LRA technique by choosing a random base γ_i permutation and transforming all $GF(p)$ elements in this permutation in each MPL round i. After the end of the round the algorithm chooses a different base point permutation for the next round. This RNS SCA countermeasure is enhanced with the base point V randomization technique using an initial random point R [145]. All $GF(p)$ multiplications used in EC point addition and doubling are done using the

RNS Montgomery multiplication [142]. Apart from the above countermeasures as proposed in [143], a RNS operation random sequence approach is also followed i.e. the individual moduli operation for each RNS addition, subtraction or multiplication are executed in a random sequence. Furthermore, scalar randomization is used as a countermeasure. This is based on the concept of computing random multiples r of the order of the curve $\#E$ instead of computing directly the scalar multiplication $[e]P$ (i.e. one can compute the same point as $[e + r\#E]P$). The bits of scalar e are masked using a different random value r at each SM execution. In order to evaluate the potential of the above countermeasures, four variants of the algorithm were implemented, with different countermeasures activated each time.

Algorithm 7.3.2 : LRA SCA-FA Blinded MPL [152]

```
Input: $V, R \in E(GF(p)), e = (e_{t-1}, e_{t-2}, ...e_0)$,
Output: $e \cdot V$ or random value (in case of faults),
1. Choose random initial base permutation $\gamma_t$.
2. Generate random $r$ integer and randomize scalar $s = [e + r\#E]$
3. Transform V, R to RNS format using $\gamma_t$ permutation
4. $R_0 = R, R_1 = R + V, R_2 = -R$
5. Convert $R_0$,$R_1$,$R_2$ to Montgomery format
6. For $i = t - 1$ down to 0
7.     $R_2 = 2R_2$, performed in permutation $\gamma_t$
8.     Choose a random base permutation $\gamma_i$
9.     Random Base Permutation Transformation from $\gamma_{i+1}$ to $\gamma_i$ for $R_0$ and $R_1$
10.    $b = \hat{s}_i$ (where the hat symbol is logical inverse)
11.    $R_b = R_b + R_{s_i}$ and $R_{s_i} = 2R_{s_i}$ in permutation $\gamma_i$
12.end
13. Random Base Permutation Transformation from $\gamma_{i+1}$ to $\gamma_i$ for $R_0$ and $R_1$
14. If ($i$, $e$ not modified and $R_0 + V = R_1$) then
15.    Random Base Permutation Transformation from $\gamma_0$ to $\gamma_t$ for $R_0$
16.    return $R_0 + R_2$ in permutation $\gamma_t$
17. end
18. else
19.    return random value
20. end
```

7.3.3 Machine Learning Algorithms

In this paper, four different classifiers are used to create the trained ML-DL model of a Device Under Test (DUT) leakage information, Support Vector Machine, Random Forest, Multi-Layer Perceptron and Convolutional Neural Networks. In this subsection, each classifier is described in brief, the parameters that were identified as important for profiling SCAs are specified and the basic classified benefits are presented.

Support Vector Machine (SVM)

Support vector machines (SVMs) are one of the most popular algorithms used for classification problems in different application domains, including side-channel analysis [22, 129, 130]. In SVM, n-dimensional data is separated using a hyperplane, by computing and adjusting the coefficients to find the maximum-margin hyperlane, which best separates the target classes. Often, real-world data is very complex and cannot be separated with a linear hyperplane. For learning hyperplanes in complex problems, the training instances or the support vectors are transformed into another dimension using kernels. There are three widely used SVM kernels; linear, radial and polynomial. To tune the kernels, hyperparameters like ’gamma’ and cost ’C’ play a vital role. Parameter ’C’ acts as a regularization parameter in SVM and helps in adjusting the margin distance from the hyperplane. Thus, it controls the cost of misclassification. Parameter ’gamma’ controls the spread of the Gaussian curve. Low values of ’C’ reflect more variance and lower bias; however, higher values of ’C’ show lower variance and higher bias. However, higher gamma leads to better accuracy but results in a biased model. To find an optimum value of ’C’ and ’gamma’, gridsearch or other optimization methods are applied.

Random Forest (RF)

In Random forest (RF), data forest is formed by aggregating the collection of decision trees [153]. The results of individual decision trees are combined together to predict the final class value. RF uses unpruned trees, avoids over-fitting by design, and reduces the bias error. Efficiently modeling using random forests, highly depends on the *number of trees* in the forest and the *depth* of each tree. These two parameters have been tuned for an efficient model in this study.

Multi-Layer Perceptron (MLP)

Multi-Layer Perceptron (MLP) is a basic feed-forward artificial neural network that uses back-propagation for learning and consists of three layers: input layer, hidden layer, and a output layer [128]. Input layer connects to the input feature variables and output layers returns back the predicted class value. To learn the patterns from the non-linear data, non-linear activation function is used. Due to the non-linear nature of side-channel leakages, MLP appears to be the best choice, in order to recover secret information from learning patterns of the signals.

Convolutional Neural Network(CNN)

Convolutional Neural Network (CNN) is a type of neural network which consists of convolutional layers, activation layers, flatten layer, and pooling layer. Convolutional layer performs convolution on the input features, using filters, to recognize the patterns in the data [72]. The pooling layer is a non-linear layer, and its functionality is to reduce the spatial size and hence the parameters. Fully connected layers combine the features back, just like in MLP. There are certain hyperparameters related to each layer, which can be optimized for an efficient trained model. These parameters include learning rate, batch size, epochs, optimizers, activation functions, etc. In addition to these, there are a few model hyperparameters which can be used to design an efficient architecture. It should be noted that the purpose of this study is not to propose the architecture design of the convolutional neural network (CNN) but to analyze and test the existing proposed CNN design on the RNS-based ECC dataset. Therefore, the focus is on tuning the optimized hyperparameters rather than model hyperparameters.

7.3.4 Feature Engineering Techniques

Features play a key role in accurate machine learning analysis. Sample values in a trace T represent the features. Generally, machine learning model can be represented with the eq. 7.1, where F represents the feature matrix and w represents the weights learnt during learning steps that are used for predicting the class on unseen values.

$$y_i = w_0 + \sum_{j=1}^{F_n} F_{ij} w_j \tag{7.1}$$

It is evident from previous research that more is not better when it comes to features in the training dataset. The massive dataset containing irrelevant redundant features, probably due to the noise in side-channel leakage traces, can confuse the model during the learning process and offer several problems. It can 1. degrade the performance of the resulting machine learning model as the model learns from real and noisy data both, 2. can elevate the sample complexity problem and computational processing requirement by increasing the requirement of sample size to achieve the desired accuracy performance for a particular cryptographic algorithm under attack, 3. can introduce over-fitting leading to poor generalization and inaccurate analysis. Feature engineering techniques can play a distinct role in eliminating the irrelevant unnecessary features from the cryptographic dataset [154]. In this paper, our goal is to reduce the large number of irrelevant features and create an efficient, effective and accurate machine learning model for RNS ECC data. In all cases, number of features F_m are selected/extracted from a pool of features F_n, where inequality (7.2) holds. The focus of this study is to utilize existing feature engineering techniques and propose an efficient hybrid feature engineering scheme (Sec. 7.4.5). The effect of dimensionality reduction and other feature selection techniques has not been analyzed on RNS-based ECC implementation datasets before.

$$F_m < F_n \tag{7.2}$$

In this study, feature datasets are formed using filter methods, wrapper methods and a hybrid approach based on both methods.

Feature Extraction

As mentioned before, side-channel dataset consists of high-dimensional data due to high sampling rate. In the classical side-channel analysis, Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are used to reduce the data dimesnionality [87, 155–157]. In machine learning context, these techniques are grouped and named as feature extraction techniques. In this group of feature engineering technique, a new feature subset is formed by reducing the dimensionality of the existing feature dataset. Based on the transformation method being used, feature extraction can be further categorized into linear transformation and nonlinear transformation.

- Principal Component Analysis is a statistical procedure for data dimensionality reduction using orthogonal transformation, while retaining the maximum variance and internal structure of the sample [158]. However, the subspace vectors in low dimensional space might not be optimal as PCA does not involve sample classes. Technically, new variables are formed in a subspace, known as principal components (PCs), which are uncorrelated to each other and contain the maximum useful information about the underlying dataset. These PCs are the eigenvectors of the data's co-variance matrix and are computed by singular value decomposition.

- LDA is a supervised learning dimensionality reduction technique, in which distance between mean of each class is maximized by projecting the input data to a linear subspace [159, 160]. It helps in reducing the overlap between the target classes by minimizing the intra-class variance.

Feature Selection

In the feature selection techniques, a new feature dataset is formed by selecting the most contributing relevant features from the existing features dataset. The method of selecting the features distinguishes and categorizes the feature selection into three main categories: filter, wrapper and embedded methods. Generally, in all selection methods an empty subset is selected and then features that are not used before are added one-by-one from the pool of the existing features. In filter methods, intrinsic properties of the features are selected, based on the relevance and relationship between the features and the target class, using uni-variate statistical analysis [161]. In wrapper methods, each subset of features is used to train the model using a classifier algorithm and is cross-validated to check the performance. Optimal features are selected based on the algorithm performance by iteratively using a search algorithm [162]. Wrapper methods are further categorized into step forward feature selection and step backward features selection. Due to the k-fold cross-validation and huge learning steps, wrapper methods are more computationally expensive as compared to filter methods and take longer to select the feature from multi-dimensional data. In the proposed hybrid method, the characteristics of both filter and wrapper methods are combined together.

In this study, filter and wrapper methods used for analysis are Chi-Square Test (Chi2), Pearson's Correlation Coefficient (PCorr), Mutual Information (MI), F-test, and T-test, Recursive Feature Elimination using Random Forest (RFE-RF) and sequential Feature Selection using Random Forest (RF-Imp) are used. For the last two, random forest is selected as an induction algorithm because of its easy interpretability, low over-fitting and better predictive performance. A brief introduction of each filter and wrapper method is given below.

- Chi-Square Test (Chi2) - Chi2 measures the deviation between the expected and observed value of the feature and response key class. Based on this measured value, it is decided if the feature is dependent on the response key class. Higher the chi2 values, higher is the dependence between feature and response class and hence feature is selected.

- Pearson's Correlation Coefficient (PCorr) - Pearson's correlation coefficient measures the degree of association between the features or between features and the response class. It returns a co-variance matrix, which holds correlation values. A value near to 1 or -1 means high positive and negative correlation between the values and 0 means no correlation. It should be noted that with large datasets, the correlation value near to 0 (let's say 0.3) might not exactly mean no correlation. Correlation measures help in finding the classifier to be used, for example, for non-linear correlations, algorithms like SVM, and RF would be the best choice.

- Ftest - Ftest helps in measuring the significance of each feature in the dataset with the help of hypothesis testing. Two models X and Y are created. Model X is created with a constant only, however, model Y is created with a constant and a feature. The least-square errors of both models are calculated and compared. The difference between model errors helps in deciding if the feature is significant or not. Ftest shows poor performance for non-linear relationships.

- Mutual Information (MI) - Mutual information measures the dependence of features on the response class. If the feature and the response class are independent, then MI will be 0, meaning, no information can be extracted about response class, if feature is known. If the response class is a deterministic function of features, then response class can be determined from the feature function, with MI as 1. MI works well for non-linear relationships.

- Ttest - Ttest is another correlation test. It measures the statistical significance between the features and the target class. It is often used in side-channel analysis to indicate the potential leakage detection on a device by correlating the expected values with the measured values of a cryptographic implementation.

- Recursive Feature Elimination (RFE-RF) - As the name implies, this method follows backward brute force approach for feature selection. A model is created on an entire feature dataset and an importance score is calculated for each feature predictor. Feature predictors are ranked based on their score and least important predictors are removed. The model is re-built and procedure is repeated again for the desired number of cycles. It is seen that random forest works best in combination with recursive feature elimination [163].

- Sequential Feature Selection using Random Forest (RF-Imp) - In this wrapper method, a subset from the features set is selected and is evaluated using the chosen algorithm (Random Forest in this case). Based on the resulting performance, the best performing subset is selected. Step forward approach is followed in which empty pool is filled with best performing features one-by-one. For evaluation, k-fold validation is used.

7.4 Machine Learning based Evaluation Methodology for ECC RNS Scalar Multiplication

For an Elliptic Curve Cryptography(ECC)-based cryptosystem, the main target of SCAs is the scalar multiplication (SM), and more precisely in our case the scalar multiplication in Montgomery Powering Ladder (MPL). The RNS approach introduces significant differences in the finite field computation approach followed in each point operation [164] that impacts the side-channel traces. Enhancing this approach, with traditional and RNS SCA countermeasures, makes possible SCA attacks as well as SCA assessment hard to implement. Based on the work of [143], profiling attacks are the only SCAs that can only partially compromise an SCA resistant RNS-ECC SM implementation (using data-dependent and location-dependent template attacks). However, there is no indication if such RNS implementations (protected or unprotected) can withstand potent ML-based profiled SCAs. Thus, in this paper, the template attack approaches have been extended to utilize the pattern learning capability of the machine learning algorithms, in order to evaluate the amount of secret information that can be recovered. For recovering the secret key bit, the ML-based attack formulation leads to the binary classification problem. Moreover, hybrid feature engineering techniques are proposed to improve the attack performance. The need for a solid tailor-made methodology to access RNS-ECC SM implementation stems from the unique characteristics of the SM under attack combined with the fact that ML models are adapted to the problem at hand. In this paper, such a concrete ML-based profiling SCA methodology is proposed and analyzed in detail. Initially, we collect leakage traces following specific attack scenarios that match possible leakage of the RNS-ECC SM implementation. Then the collected raw data are aligned and cleared from noise using pre-processing and then are split into separate training and testing datasets. Both datasets are separately processed using feature engineering techniques. The reduced feature training dataset is used to train the machine learning model, and a reduced feature testing dataset is used to test the trained model for the recovery of the scalar key bits by predicting the key bit class. An overview of the complete methodology is given in Fig. 7.1. The methodology is split into the following six distinct stages:

1. **Attack Scenario Specification:** This constitutes the first stage of the methodology plan. In this stage the possible targets on the RNS-ECC SM algorithm are identified. More specifically, as in all SM MPL variations, the most evident information leakage can be observed from the scalar bit depended sample difference when updating R_0 or R_1 storage areas and/or the scalar bit depended trace difference when point doubling operation is executed on R_0 or R_1. Following the approach, carried in [143], two attack scenarios can be identified for ML SCAs: data-dependent attacks and location-dependent attacks. It should be pointed out that the RNS structure of all involved numbers (each number is split in several independent moduli) makes the power or EM variations due to different memory storage, more complex since the point coordinates are no longer single numbers to be handled by a big number software library (that may lead to R_0 or R_1 storage in contiguous memory blocks) but, on the contrary, small numbers that may be stored independently in memory.

2. **Raw Trace Preprocessing Mechanism:** The fact that RNS operations are performed as individual, autonomous moduli operations, thus triggering execution optimizations (parallel processing, pipelining etc.) along with the fact that the algorithm 1 RNS ECC SM implementation has several powerful SCA countermeasures and the fact that software implementations lead to noisy and misaligned traces, highlight the need for a trace preprocessing stage before using them for ML model training and profile attacking.

3. **Data Splitting:** At this stage, the preprocessed collected raw data are split into separate training and testing datasets. In side-channel data analysis, the available leakage data traces might be limited. Splitting data with 50-50 ratio might produce a very small training dataset. Insufficient training data traces might result in over-fitted or under-fitted model. On the other hand, having too little testing dataset might not evaluate the trained model correctly. A trade-off value is required to train and test the model. To cater for this real-world side-channel analysis limitation, at this state, the appropriate data splitting is studied and the impact of different data splitting ratios for training and

testing data and deduce the best data split ratio is determined.

4. **Feature Selection and Processing:** Another important aspect of machine learning analysis is the features. Redundant features can lead to over-fitting and curse of dimensionality, which ultimately results in an inaccurate model. At this stage, appropriate feature engineering techniques and feature processing combination models are proposed in order to choose the optimal features for ML model training. Also, a combination of feature processing models and designed experiments are proposed in order to test the proposed feature processing combination models.

5. **ML Classification model training:** At this stage, the ML classifier models are trained using an optimal set of parameters. The machine-learning algorithms, described in Sect. 7.3.3, are used at this stage i.e Support Vector Machines (SVM), Random Forest (RF), Multiplayer Perceptron (MLP) and Convolutional Neural Networks (CNN). The algorithms have been tuned to achieve the best performance.

6. **Key Prediction:** The final stage of the overall methodology is devoted to the usage of the ML trained models on the trace testing set in order to evaluate the SCA resistance of the RNS ECC SM implementation against ML profiling attacks.

In the following subsections, the methodology stages are described in more detail. Also, we propose how each stage should be used in order to analyze and assess the ML-SCA resistance of algorithm 7.3.2 with and without the presence of countermeasures. The parameter settings used for the algorithm under study are mentioned in each stage.

7.4.1 Trace Collection Experimental Setup

All trace Datasets for the following analysis are collected by executing algorithm 7.3.2 RNS-based ECC SM implementation (in two variants) on a BeagleBone Black that use an ARM Cortex A8 processor operating at 1GHz. Samples were collected using EMV Langer probe LF B-1, H Field (100KHz- 50MHz), and Lecroy Waverunner 8404M-MS with 2.5GS/sec sampling rate.

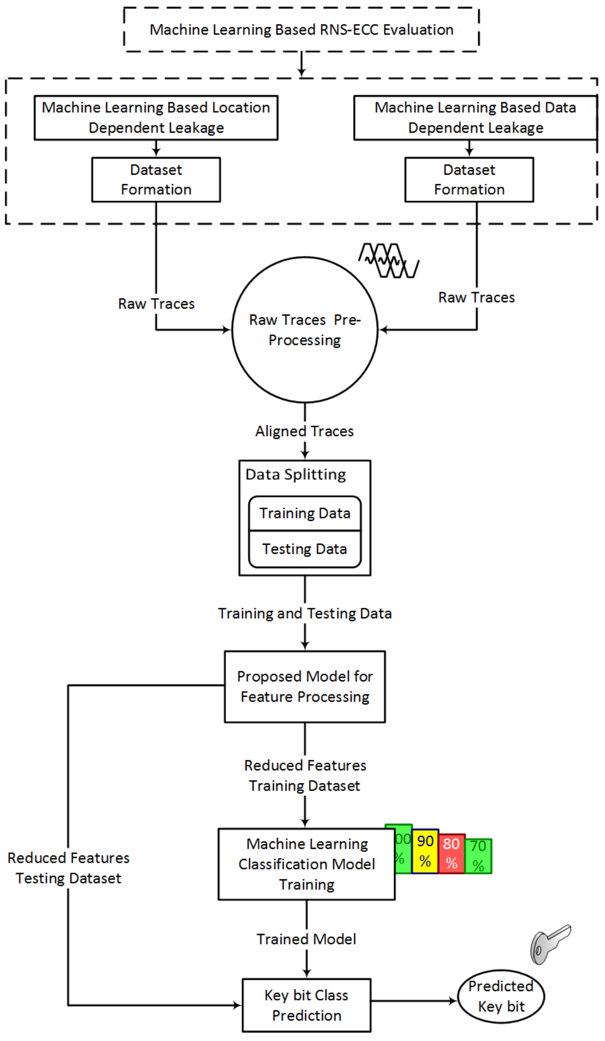

Machine Learning Based RNS-ECC Evaluation
Machine Learning Based Location Dependent Leakage
Machine Learning Based Data Dependent Leakage
Dataset Formation
Dataset Formation
Raw Traces
Raw Traces
Raw Traces Pre-Processing
Aligned Traces
Data Splitting
Training Data
Testing Data
Training and Testing Data
Proposed Model for Feature Processing
Reduced Features Training Dataset
Reduced Features Testing Dataset
Machine Learning Classification Model Training
100 %
90 %
80 %
70 %
Trained Model
Key bit Class Prediction
Predicted Key bit

FIGURE 7.1: Machine Learning based Evaluation Methodology for RNS-ECC

The RNS-ECC SM algorithm 7.3.2 implementation was taken from a public repository [152] and was customized according to the requirement for data collection and attack scenario determined in the proposed methodology stages. For data collection and formatting, Matlab R2019 and Inspector 4.12 provided by Riscure was used [165]. For machine learning analysis, a Python environment with Keras and Scikit learn libraries has been used [166]. All features selection/extraction methods have been taken from Scikit learn [167] except T-test which was implemented in-house.

To meet computation extensive needs of machine learning algorithms, NCI (National Computational Infrastructure) Australia high-performance supercomputing server has been used [168].

7.4.2 Attack Scenarios Specification

Machine Learning based Data-dependent Leakage Analysis ($MLDA$)

In data-dependent attack scenario, the adversary can monitor the power or electromagnetic emission (EM) fluctuations due to the processing of a different value of the $i-th$ scalar bit e_i. This is reflected in processor instructions corresponding to line 9 of the ECC scalar multiplication algorithm (Alg. 7.3.2), where performed operations depend on the value of secret key bit e_i resulting in registers R_0 and R_1 updated differently. R_0 contains the addition result and R_1 contains the doubling result if the scalar secret key bit $e_i = 1$ and in reverse order if $e_i = 0$ (R_1: addition, R_0: doubling). Since the data determine the register that is used and therefore causes the leakage, we refer to this analysis as “data-dependent leakage". Such data leakages should also be observable using protected scalar bit countermeasures if the scalar bits under attack are retrieved from a memory location in a clear view.

For the purpose of analysis, we have collected the leakages traces of the first few algorithm 7.3.2 rounds for a $233-bit$ scalar. As explained, data leakage LD is labeled as ‘1’ if the scalar e_i =‘1’ and is labeled ‘0’ otherwise in round i. Only one instruction was observed and 50k traces, each of 700 samples, were collected; out of which around 3k-7k were utilized after alignment in the other stages of the proposed methodology.

Machine Learning based Location-dependent Leakage Analysis ($MLLA$)

In location-dependent attack scenario, key-dependent instruction leakages are exploited, utilizing the storage structure information. More precisely, it is assumed that based on the storage content, the leakages for a particular operation will be distinguishable. It can be observed that in each round i of algorithm 7.3.2 only two operations have key-dependent instruction; that is, addition and doubling. Both operations are performed in the same order, irrespective of the value of the scalar key bit e_i. However, the storage content differs according to the scalar bit value. The storage register R_0 is doubled when the scalar key bit is ‘0’, otherwise R_1 is doubled. Based on the fact that there is no memory address randomization, we can exploit the vulnerability by collecting the leakage data for doubling operation. The data will be labeled and classified based on the content of storage registers R_0 and R_1. Such memory access leakage has also been exploited for RNS-based RSA in [169]. Papachristodoulou et al. in [143], have exploited a similar vulnerability for ECC SM by utilizing a small sample window of 451 samples (out of 3k samples per trace) for training and classification for template profiling SCAs. Identifying the specific samples for training purposes requires more in-depth knowledge of the underlying system and requires a lot of signal processing, which might be discouraging for the attacker. The work of Andrikos et al. performed location-based attacks using machine/deep learning but those were focused on accessing different SRAM locations and are not algorithm-specific [170]. In our work, we have used the ML approach to classify the scalar key bit e_i, exploiting the doubling operation leakage, by using the whole trace rather than the small sample portion of 451 samples. We have achieved similar results, which proves that the machine learning attack is realistic and practical from an attacker point of view. For the location-based analysis, we have labeled leakage data LD as ‘0’ if R_0 is doubled and labeled LD as ‘1’ if R_1 is doubled. We collected 50k traces (each of 3k samples long), out of which 14k traces are used after stage 2 (preprocessing) of the proposed methodology

Datasets

For a detailed evaluation of an RNS-ECC SM approach against the above two ML-based attack scenarios, all potential countermeasures that can be applied on the implementation should be evaluated using the proposed RNS-ECC SM evaluation methodology. To achieve that, two implementation variants of the algorithm 7.3.2 SM can be identified for each ML attack scenario, one with all SCA countermeasures enabled (protected version) and one with all SCA countermeasures disabled (unprotected version). In line with the above rationale, for the evaluation of algorithm 7.3.2 the trace datasets of Table 7.1 can be identified, denoted and collected.

TABLE 7.1: Trace Dataset Categories

Name	Countermeasures	Notation	Trace Length
Protected Data Dependent Leakages	RNS LRA technique, base point randomization, scalar randomization countermeasure and random RNS operation sequence	DD_P	700
Unprotected Data Dependent Leakages	no countermeasure	DD_{UP}	700
Protected Location Dependent Leakages	RNS LRA technique, base point randomization, scalar randomization countermeasure and random RNS operation sequence	DL_P	3000
Unprotected Location Dependent Leakages	no countermeasure	DL_{UP}	3000

7.4.3 Raw Trace dataset Pre-processing

Trace Alignment

Alignment plays an important role while using machine learning techniques especially on raw leakage samples. In raw leakage samples or row instances, each data point in a particular sample will be treated as a feature and then the feature columns are used to train the model. Having misaligned features might scatter the useful feature information all across the columns, hence making it difficult for the ML classifier to learn from the scattered haphazard data. Misalignment generally occurs due to the noise of the neighboring components in the device. However, in some cases, noise is intentionally induced to the system as a countermeasure to increase side-channel attack resistance. Executing a software implementation in an embedded system operating system (as is used in this paper experimental setup) will result in trace collection of noise that is unexpectedly added from the other processes of the operating system. Common signal processing technique can be used in order to reduce the noise like low pass or band pass filter. In the collected traces, the application of a low pass filter approach was chosen. Initially, the dominant frequencies are measured using Fast Fourier Transform (FFT), as shown in Fig. 7.2 and it is observed that the maximum energy lies between 0-300MHz, with the highest frequency at 1GHz. Based on the observation, a low-pass filter is applied and the resulting clear patterns are used for alignment.

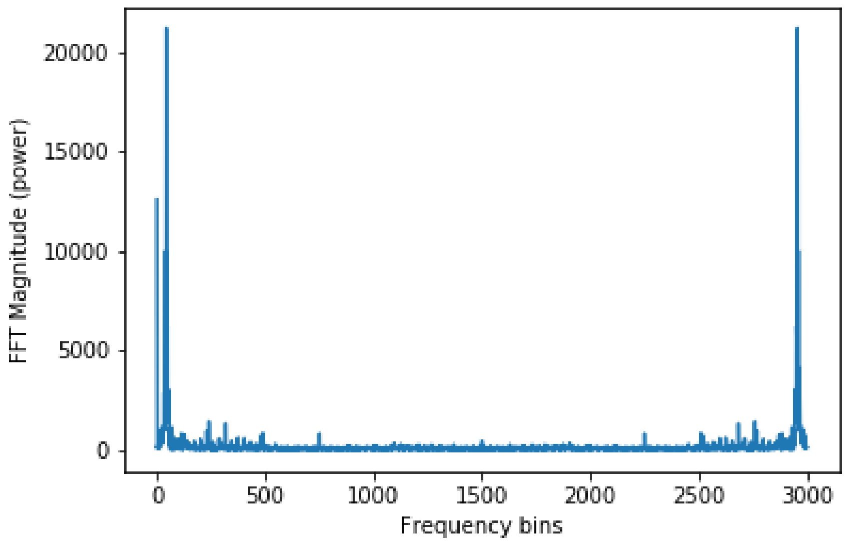


FIGURE 7.2: Fast Fourier (FFT) of the leakage samples

Skewed or Imbalanced Datasets

For a good performing trained model, it is imperative to have a balanced dataset. Skewed or imbalance dataset is the one in which the traces for one class label are more than the other. The trained model will be biased due to the dominating class and will not be able to classify the unseen data accurately. To emulate the problem of imbalance and observe its impact in the experimental process of the RNS-ECC SM assessment, after traces where collected and aligned, we produced both balanced and unbalanced dataset outcomes. Datasets DD_P, DD_{UP} and DL_P were almost balanced, having approximately 1050, 1500, and 3800 traces (for both 1's and 0's), respectively. These three datasets had ideal balanced data for modeling. However, dataset DL_{UP} traces were collected to be highly skewed. i.e. the number of traces for class key bit '0' was higher compared to class key bit '1' (10150 and 42 traces respectively). To handle the skewness and minimize its impact, Synthetic Minority Oversampling Technique (SMOTE) was used as it outperformed for other cryptographic datasets [132]. SMOTE synthesizes new instances for the minority class traces and balances the data [171].

7.4.4 Data Splitting and Validation Strategy

Machine learning-based side-channel attacks are based on the template attack approach. In template attacks, two datasets are used; template and test datasets. The template dataset (pre-defined examples) is used to train the system, and then the test (unknown) dataset is used to evaluate the attack [136]. Similarly, in ML SCAs, the leakage data set LD is divided into the training dataset, D_{Train}, which is used to train the machine learning model and the test D_{Test} dataset. Unlike, template attacks, though, another dataset is introduced in ML analysis known as Validation D_{Val} dataset. In this stage of the methodology, the above described dataset splitting and its role is analyzed below:

- D_{Train} dataset is used during the model fitting process and helps model learn the patterns from data.
- During the evaluation, D_{Val} is used to fine-tune the model using model hyperparameters. The model never directly learns from the validation data, but it can occasionally see the data during the learning process. Hence it provides biased evaluation and changes the model structure based on the validation data results.
- D_{Test} dataset is completely unknown to the system and is never used in the training process. D_{Test} provides an unbiased evaluation of the model.

One of the important aspects in machine learning is to decide the dividing ratio of the training, validation, and testing sets. The bigger the dataset, the better the trained model will be. It becomes a huge problem, especially with the datasets having a small number of instances (traces). To evaluate the effect of data division on secret information recovery, in this paper, three proportions are tested. The ratios used for training and testing datasets are 90-10%, 80-20%, and 50-50%. Datasets are shuffled before splitting for spreading the instances in the space.

At this methodology stage analysis, we suggest in this paper, the use of k-fold cross-validation which is a resampling procedure used for evaluation of machine learning trained model. After the initial dataset split into two sets, i.e. training and testing, the training dataset is further split using k-fold validation scheme into training and validation. In this validation procedure, data samples are split into k groups. One group is a holdout or validation dataset and rest of the data is used for training the model. Model is fitted on the training group set and evaluated on the holdout/validation set. This ensures that the whole dataset undergoes a proper validation process. For the k-fold validation, 5 and 10 folds are the most recommended values as they neither give high variance nor high bias in the resulting validation error estimate [172]. However, high number of validation folds can lead to increased training time. This processing time can be reduced by using an optimal number of folds, yet still achieving a reliable trained model.

7.4.5 Feature Processing and Engineering: Proposed Hybrid Approaches

In this methodology step, we propose an analysis approach to deduce the impact of feature selection/extraction techniques, extensively used in other machine-learning based applications, (Sec. 7.3.4) on ML classification model of RNS-based ECC dataset. Our proposed hybrid approach combines the characteristics of the two or more feature extraction/selection techniques to improve the learning performance and efficiency. To provide a comparison with the application of the classical feature techniques only (which is not tested for RNS-ECC dataset before), we have explicitly split the analysis into three sections as explained below.

- **Approach A:** In the first approach, features dataset is processed using the feature selection and extraction methods as explained in Sec. 7.3.4, that is, Ftest, T-test, Chi2, MI, P_Corr, PCA, LDA, RFE-RF, and RF-Imp. There are total F_n features for location-dependent leakages ($MLLA$) and data-dependent leakages ($MLDA$). Out of F_n, F_m features are selected. The selected output features are directly given as input to the machine learning models for training.

- **Approach B:** In the second approach, features datasets are processed (Tier 1) using filter methods (Ftest, T-Test, Chi2, MI, P_Corr), and the output features are further reduced (Tier 2) using PCA and LDA dimensionality reduction techniques. For Tier 1 feature selection, F_m features are selected from F_n pool of features, for both $MLLA$ and $MLDA$. However, for Tier 2, F_o PCA components (features) are selected from

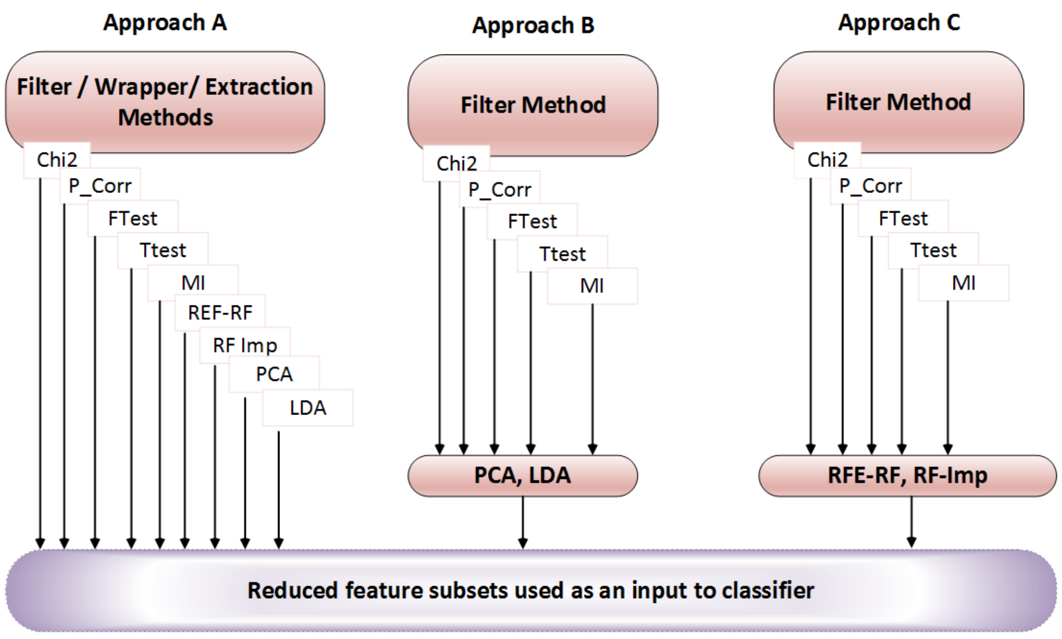


FIGURE 7.3: Hybrid Feature Engineering Approaches

F_m features dataset. For binary classification, LDA projects F_m features onto one dimension.

- **Approach C**: In the third approach, in Tier 1 features F_n are ranked according to the relevance, resulting in a subset of features consisting of F_m features. In Tier 2 processing, features are further selected based on the classifier algorithm performance using RFE-RF and RF-Imp, and are reduced to feature subset consisting of F_o, for both $MLLA$ and $MLDA$. The RFE-RF and RF-Imp methods recursively eliminate the redundant features which do not contribute towards classification.

Our proposed approaches help in tackling the drawbacks of filter and wrapper methods. In filter methods, the target response class is not involved in the selection process. To involve the target class, the relevant uncorrelated features are selected using filter methods and are further reduced by recursively searching through the feature pool or by using dimensionality reduction. Moreover, directly applying the wrapper methods or the dimensionality reduction techniques are computationally expensive due to the huge number of features in the dataset. This approach helps in eliminating the least correlated redundant feature, thus reduces the time required for processing, whereas preserves the overall useful information in the leakage dataset. The graphical description of the proposed approaches is presented in Fig. 7.3.

TABLE 7.2: Parameter tuning for SVM, RF, MLP and CNN

Classifier	Parameter	Value Range
SVM	C	[0.1, 0.01, 0.5, 1.0]
	gamma	[1,10,30,40,50]
	kernel	[Poly, Sigmoid, RBF]
MLP	Learning Rate	[0.001,0.0001]
	Solver	[adam, sgd]
	Batch Size	[32]
	Activation Function	[tanh,relu,identity,logistic]
	Epochs	[200]
RF	Trees Depth	[5,10,20,30]
	Number of Trees	[10,50,100,200]
CNN	Learning Rate	[0.001,0.01,0.1, 0.5]
	Epochs	[300]
	Activation function	[relu,selu,elu]
	Optimizer	[Adam, Nadam, RMSprop,Adamax]
	Init Mode	[uniform, normal]
	Batch Size	[32, 100, 400]

7.4.6 ML Model Training: Parameter Tuning

At this stage of the RNS-ECC SM evaluation methodology, the ML models are trained using the features selected from the hybrid feature extraction process. The four classification algorithms described in Sect. 7.3 are used to evaluate the effectiveness of the location-dependent and data-dependent attacks and also to evaluate the performance of the features subset, i.e. Support Vector Machines (SVM), Random Forest (RF), Multi-Layer Perceptron (MLP) and Convolutional neural network (CNN). There are certain parameters in each classifier algorithm, as mentioned in 7.3.3, that needs tuning. For the systematic evaluation of RNS-ECC SM, the hyperparameters are tuned using gridsearch to obtain the best possible trained model. The tuned hyperparameters are shown in the Table 7.2.

7.5 Results and Discussions

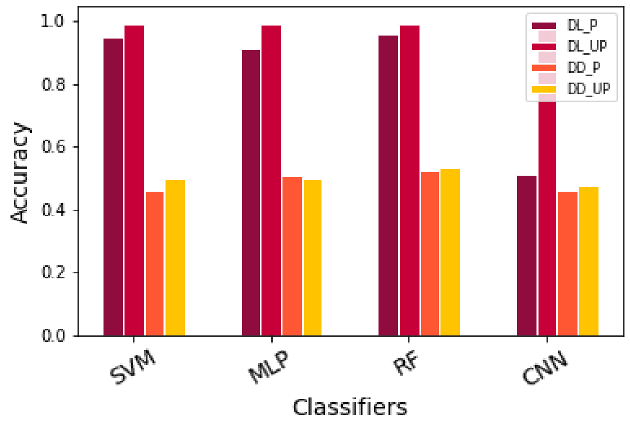
Manifesting the proposed methodology for the experimental process described in Sect. 7.4.1 for the RNS-ECC SM implementation of algorithm 7.3.2 as described in the previous section, the performance of our proposed approach and its outcomes-results can be evaluated and analyzed. There are various evaluation metrics which can be used to evaluate the performance of machine learning models including Accuracy (Acc), Precision (specificity), Recall (sensitivity), F1 score, Receiver Operating Characteristics (ROC), and Area Under Curve

(AUC). For binary classification problems on balanced dataset (as is our case), accuracy is sufficient evaluation metric. Accuracy is the ratio of correct predictions to the total number of predictions. Hence, it exhibits the reliability of the model in a practical real-world scenario on unseen data.

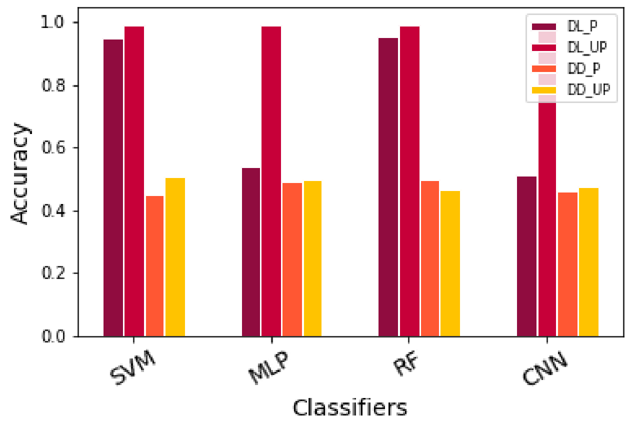
As described in Sect. 7.4.2, four datasets of protected and unprotected leakage traces are evaluated using four machine learning classifiers. In this section, the results are presented in four sections, for better understanding. Sect. 7.5.1 presents the classifier's performance on raw features, without applying any feature engineering, Sect. 7.5.2 presents results after applying feature engineering techniques as explained in Sect. 7.4.5 approach A, Sect. 7.5.3 exhibits comparison results for Sect. 7.4.5 approach A, B and C, and Sect. 7.5.4 depicts the affect of data splitting size. For the sets of experiments conducted in Sec. 7.5.1-7.5.3, the models are trained with the raw traces using four classifiers, for all four datasets. For comparative analysis with existing studies, analysis is divided into two sub-cases. In case a, machine learning analysis has been performed on the full length traces that is, all the trace samples (trace length 0-699 and 0-2999 for $MLDA$ and $MLLA$, respectively) are used as features for training the model. However, in the case b, features dataset is reduced and only the aligned part of the traces (precisely, 550-900 for DD_P, 1150-1950 for DD_{UP}, 80-250 for DL_P, 190-250 for DL_{UP},) is used for training the models.

7.5.1 Classifier's Performance on Raw features

Fig. 7.4a and 7.4b show the accuracy of the trained classifiers for the case a and case b, respectively. The plotted accuracy is achieved by tuning the hyperparameters as given in Table 8.4. Best selected parameters are also given in Table 7.3. It can be observed that for location-dependent attacks ($MLLA$) in case a, the secret can be recovered with 94-100% accuracy for the protected and unprotected implementations. However, for data-dependent attacks ($MLDA$), the best accuracy, approximately 54%, is achieved with RF. In some cases, imbalance dataset scenarios are created and SMOTE is applied before applying machine learning classifiers, to balance the datasets. In addition to accuracy, recall, precision, and F1 score has been closely monitored as well, which is less than 0.5 in case of CNN, but greater



(a) Trace Dataset with all samples



(b) Trace Dataset with aligned reduced samples

FIGURE 7.4: Accuracy of four classifiers on raw samples without feature processing

than 0.9 for other classifiers.

It has also been observed that the complex deep learning model (CNN) did not perform well for all the datasets, which was the expectation because datasets have a small number of traces and large number if irrelevant features. It is expected that with a huge dataset, the performance, using complex networks like CNN might improve, but the collection of the huge dataset and high computational cost, might be highly discouraging for the attacker. Scope of this study is to analyze the affect of limited size datasets with computationally efficient classifiers. It has also been noticed that a simple neural network like MLP gives good accuracy if complete trace length is used, however, it cannot classify the target key bit (accuracy around 53%) with the reduced trace length, for case b. This shows that an amount of useful information is contained in the unaligned portion of the trace as well.

Due to the inherent design capability of dealing with redundant features, in both SVM and RF, reducing the features per trace does not affect the classification accuracy. RF, by design, constructs unpruned trees and removes the unnecessary redundant features during the training process, hence produces an efficient model without using any feature engineering technique. In SVM, Radial Bias Function (RBF) kernel transforms the data and creates new features that are separable in high dimensional space so by design it retains the most contributing features and eliminates unnecessary ones. It appears that the RNS-ECC SM location-based leakage is linearly separable in higher dimension space. However, this is not the case with RNS-ECC SM data-dependent leakages.

To analyze the possibility of under-fitting and over-fitting, training, validation and testing accuracy, all are closely monitored in all cases. For SVM with RBF, it is observed that lower values of parameter 'C' and higher values of parameter 'gamma' provide the best results. The validation curve for gamma parameter tuning is given in Fig. 7.5. For RF, 50 and 100, trees along with varying tree depth of 5-20 present good results. For MLP, batch size 32, activation function 'relu' and optimizer 'adam' give the best results for $MLLA$ analysis. However, for $MLDA$ analysis, activation function 'tanh' and optimizer 'sgd' and 'adam' provide the best results for protected and unprotected leakage datasets, respectively.

Given the above results, a comparison between ML analysis and the state-of-the-art template attack results (based on the perceived information (PI)) on RNS-ECC SM implementation, can be made. For template attacks, PI utilizes practical leakages to estimate the Probability Density Function (PDF) of the algorithm 7.3.2 implementation. Steps explained in [173], are followed to estimate the PI of RNS implementation leakages from BeagleBone. First profiling traces are collected to estimate the leakage model and then PI is estimated for the actual test leakages from the chip. The leakage model is estimated based on profiling traces and then PI is estimated for the collected test traces. The estimation and assumption errors are calculated to evaluate the attacking model. It is observed that machine learning performs better than the template profiling attacks on the RNS-ECC SM implementation datasets. For template attacks, the classification success rate for the location-based attacks is 87-99% for unprotected implementation and for implementations with one countermeasures

TABLE 7.3: Best parameters for SVM, RF, MLP and CNN

DataSet	Classifier	Feature No	Parameters
DL_P	SVM	All	C: 1.0, gamma: 40, kernel: rbf
	MLP	All	activation: relu, batch_size: 32, solver:adam
	RF	All	max_depth: 20, n estimators: 100
	CNN	All	Act: Relu, Optimizer: Adam, Learning_Rate:0.001
	SVM	Reduced	C: 0.1, gamma: 40, kernel: poly
	MLP	Reduced	activation: relu, 'batch_size: 32, solver: adam
	RF	Reduced	max_depth: 30, n_estimators: 50
	CNN	Reduced	Act: Relu, Optimizer: Adam, Learning Rate:0.001
DL_{UP}	SVM	All	C: 0.1, gamma: 1, kernel: rbf
	MLP	All	activation: relu, batch_size: 32, solver: adam
	RF	All	max_depth: 5, n estimators: 50
	CNN	All	Act: Relu, Optimizer: Adam, Learning Rate:0.001
	SVM	Reduced	C: 0.01, gamma: 10, kernel: poly
	MLP	Reduced	activation:relu, batch_size: 32, solver: adam
	RF	Reduced	max_depth: 5, n_estimators: 10
	CNN	Reduced	Act: Relu, Optimizer: Adam, Learning Rate:0.001
DD_P	SVM	All	C: 0.5, gamma: 50, kernel: rbf
	MLP	All	activation: logistic, batch_size: 32, solver: sgd
	RF	All	max_depth: 20, n estimators: 100
	CNN	All	Act: Relu, Optimizer: Adam, Learning_Rate:0.001
	SVM	Reduced	C: 0.5, gamma: 10, kernel: rbf
	MLP	Reduced	activation: tanh, batch_size: 32, solver: adam
	RF	Reduced	max_depth: 20, n_estimators: 10
	CNN	Reduced	Act: Relu, Optimizer: Adam, Learning Rate:0.001
DD_{UP}	SVM	All	C: 0.5, gamma: 1, kernel: sigmoid
	MLP	All	activation: tanh, batch_size: 32, solver: sgd
	RF	All	max_depth: 20, n estimators: 100
	CNN	All	Act: Relu, Optimizer: Adam, Learning Rate:0.001
	SVM	Reduced	C: 0.5, gamma: 1, kernel: rbf
	MLP	Reduced	activation: logistic, batch_size: 32, solver: adam
	RF	Reduced	max_depth: 10, n_estimators: 10
	CNN	Reduced	Act: Relu, Optimizer: Adam, Learning Rate:0.001

activated. When a combination of countermeasures is used, then this percentage falls to 70-83%. For machine learning analysis the classification accuracy is 95% and 99.5% for protected (DL_P) and unprotected (DL_{UP}) RNS-ECC SM implementations, respectively. In [143], template attack on RNS-ECC implementation is successful only if the specific sample window from each trace is selected for training. However, in machine learning-based side-channel attack, the model trained with the complete trace length gives equal or better results. Isolating and selecting the aligned part only for the training phase, might not be an easy task for an attacker thus making the template attack difficult. However, it is more convenient to train with the complete raw trace, which implies that machine learning attacks are less complex from an attacker perspective.

!h

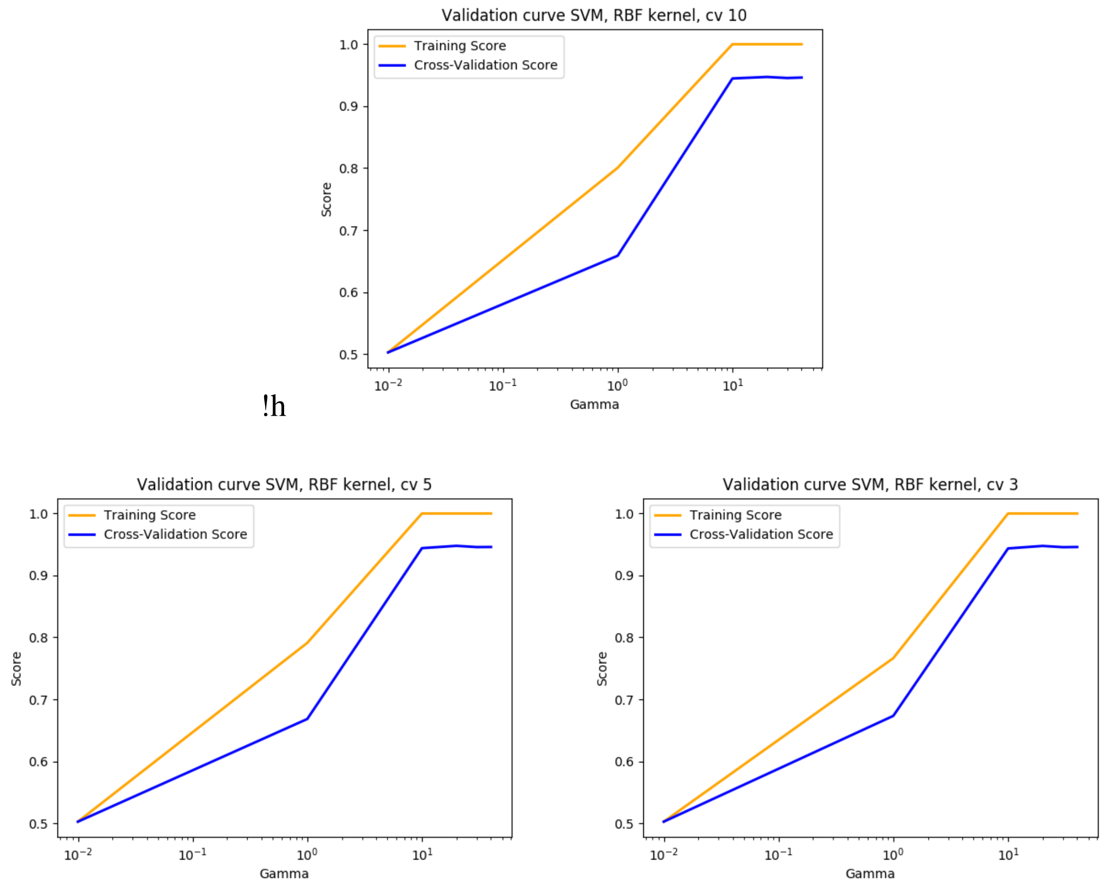
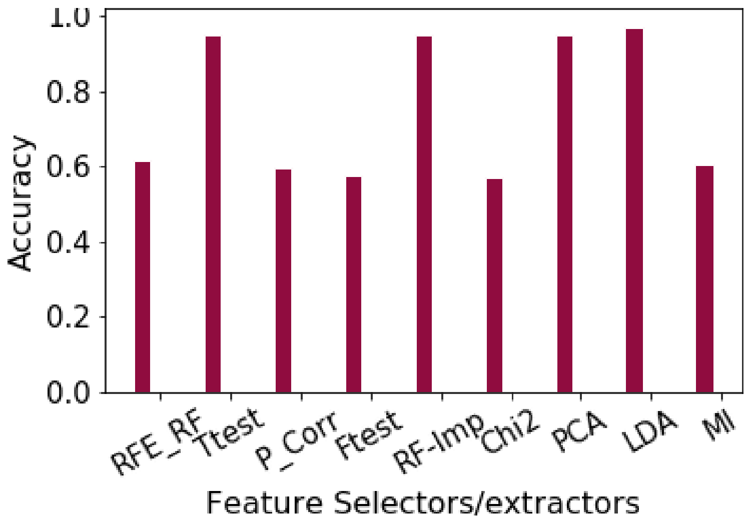


FIGURE 7.5: Gamma Parameter Tuning

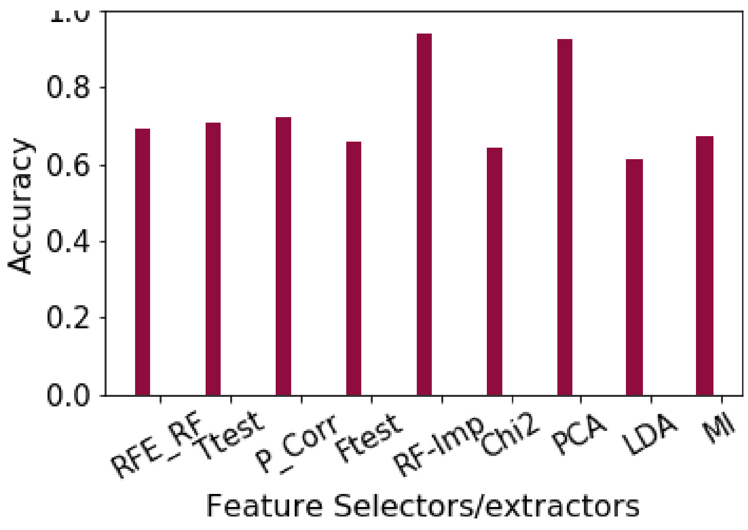
7.5.2 Impact of Feature Engineering

In this section of experimental analysis, advance feature engineering techniques, based on wrapper and filter methods as explained in Sect. 7.4.5 approach A, are applied to analyze the impact of feature reduction on the trained model performance. $F_n = 50$ features have been selected from the full length (features $F_m = 3k$ and $F_m = 700$ respectively) and reduced length (varying numbers of features depending upon the aligned portion) traces, except T-test. For T-test threshold is set to 0.5 and the resultant 1299 features are selected for further analysis. Results for SVM trained model on RNS-ECC protected datasets ($MLLA$) are shown in Fig. 7.6.

The purpose of applying feature engineering techniques is to find the optimal numbers of features for the bias-variance trade-off. Variance in machine learning is the type of error that occurs due to the model's sensitivity to small fluctuations in the training dataset. High



(a) Trace Dataset with all samples



(b) Trace Dataset with aligned reduced samples

FIGURE 7.6: Performance comparison for $MLLA$ using SVM with feature extraction/selection techniques

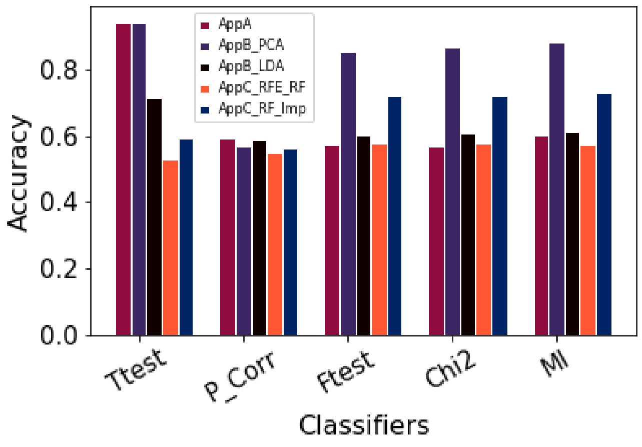
variance leads to over-fitting as the model might learn from the noise in the data. Bias, on the other hand, is the type of error that occurs due to erroneous assumptions in the learning algorithm. High bias leads to under-fitting as a model might miss relevant information between features and the target key class. Both the errors are inter-linked, minimizing one error will increase the other one. Neural nets (high capacity models) can lead to high variance problems as they might learn from the noise in the data. Regularization, early stopping, and drop-out has been used to avoid the problem in our evaluation. For RF, pruning deals with the above issues, so feature engineering is not required. However, for SVM finding an optimal number of features will improve the model's accuracy.

In the case of RNS-ECC datasets, there is a higher bias than a variance. When PCA is applied, the variance is increased thus bias is reduced. Usually, the variance is increased to a level so that the model doesn't overfit. The suitable variance threshold (with classification accuracy 100%) is achieved when a number of features are selected to be $F_m = 50$ for PCA. For case a, model performance stays same or has improved by using Ttest, RF-Imp, PCA and LDA. For case b, improvement is observed for RF-Imp and PCA. However, performance decreases when analysis is performed after reducing features using LDA. LDA uses classifier and fails to extract the relevant features as some of the information, required to identify the relationship between the target class and the feature dataset, is lost while the traces are trimmed during alignment process.

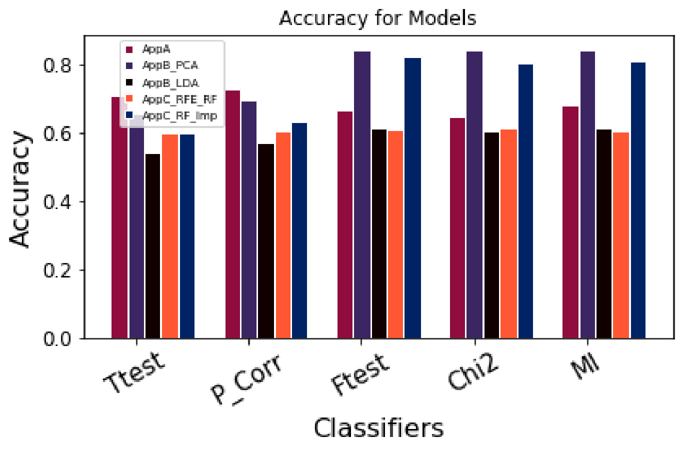
7.5.3 Hybrid Feature Selection Techniques

In this section, comparative analysis is performed, based on the evaluation results of the hybrid approaches of the proposed methodology on $MLLA$, as explained in Sect. 7.4.5 approach B and C. For all hybrid methods, feature selection filter methods have been applied to reduce the bias in the input data by selecting the independent $f_n = 300$ features from the complete pool of the features $f_m = 3k$ ($MLLA$) and $f_m = 700$ ($MLDA$) and then only $f_o = 50$ features are selected from the reduced pool of features using extraction techniques, for both case a and case b.

For case a (Fig. 7.7a), T-test gives best results using approach A and B. Generally, the trend is seen that the combination of feature selection using filter method with the recursive feature elimination, reduces the model accuracy. One of the reasons could be that features are highly correlated with each other rather than with the target class. Approach 2 with PCA returns the accuracy greater than 80%. For Ftest, MI, and Chi2, there is an increase of 13-30% in the resultant accuracy using hybrid approach C. For case b, some of hybrid methods have shown improvement in accuracy as compared to the Fig. 7.4b.



(a) Trace Dataset with all samples



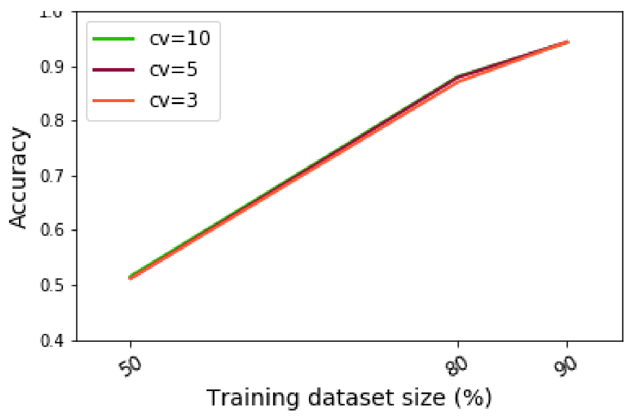
(b) Trace Dataset with aligned reduced samples

FIGURE 7.7: Performance comparison of hybrid feature processing approaches

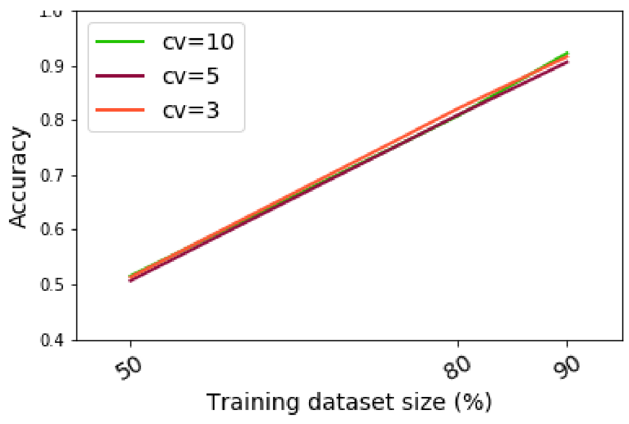
7.5.4 Impact of Data Splitting Size

In this analysis phase, we have performed quantitative analysis, as described in Sect. 7.4.4. For analysis, out of the best performing feature selection techniques (having accuracy greater than 95%), we have chosen one randomly (i.e. PCA on protected dataset DD_P) to further investigate the impact of varying data splitting ratios for RNS ECC Dataset. It can be seen that the best results are obtained with data splitting ratio of 90:10 for training and testing data.

In [133], for symmetric ciphers, in total 60,000 instances are used for training and testing, out of which 50,000 are for training and 10,000 are for testing. As expected, the huge set of traces is ideal for training with deep learning algorithms like CNN. However, the required training time in this case will be high too. In this study, we have evaluated the effect of having a small number of traces useful of key retrieval. We have seen that location dependent attack



(a) Trace Dataset with all samples



(b) Trace Dataset with aligned samples only

FIGURE 7.8: Impact of Data Splitting Size on Model Accuracy

is successful in recovering the key with few traces in less time using validation folds as low as 3.

7.6 Conclusion

In this paper, we have presented the evaluation methodology of machine learning-based side-channel attacks on an elliptic curve RNS-based scalar multiplier implementation with and without RNS and traditional SCA countermeasures. Each stage of the methodology was described along with a practical experimental realization. A detailed analysis of the RNS-ECC SM implementation proposed methodology results was also provided in four different phases of analysis. Comparison has been provided with the state-of-the-art template attacks on the RNS-ECC balanced and imbalanced datasets. It can be concluded that the machine learning-based side-channel attacks require less prepossessing and give better performance results for location-based profiling attacks, hence, leading to a time-efficient realistic attack scenario. The secret key can be recovered from unprotected and protected RNS ECC SM implementations, using location-based attack, with 99% and 95% accuracy, respectively.

The impact of advance feature engineering techniques has been analyzed using feature extraction and feature selection methods. Moreover, several hybrid approaches were also evaluated. It has been observed that PCA, LDA, T-test, RF-based feature selection provides improved accuracy results.

We have also evaluated the effect of training the model with the small dataset, that is dataset containing reduced aligned samples only, to classify RNS-ECC key bits using machine-learning based side-channel attacks. We have observed that for location based attacks, SVM and RF can successfully distinguish the scalar key bit with more than 95% accuracy for both full length and reduced length aligned trace datasets. Trace sample window does not affect the classification results using SVM and RF, due to their inherent characteristics of eliminating redundant features during the training process. However, MLP can distinguish and classify the scalar key bit correctly only if the full trace length dataset is used. If the reduced trace, based on the aligned part, is used for training an MLP network, then some useful information is lost during alignment process and the model fails to classify the scalar key bit. This reduces the complexity of the attack and increase the attack success rate in real world scenario. RNS-ECC implementations showed resistance against Machine-learning based data dependent attacks.

Machine-learning based side-channel attacks on PKC provide a realistic efficient attack scenario to recover the secret information as they require less pre-processing compared to template attacks on RNS ECC implementations.

Chapter 8

Improved Hybrid Approach for Side-Channel Analysis using Efficient Convolutional Neural Network and Dimensionality Reduction

This chapter is an adapted version of a published article (Publication VI). This chapter addresses the concern of requirement of huge datasets for successful deep learning-based side-channel attacks (DL-SCA). The existing DL-SCA are applied on datasets consisting of around 60,000 traces while each trace consisting of 700-1500 samples. The number of instances in target classes directly affects the system's capability to determine the pattern between the secret key and the leakage data. With more instances for all the target classes, a better pattern can be learnt. Moreover, in certain cases, the sampling frequency during leakage data acquisition is set to high to capture the small details about the sensitive information. if the number of instances are low with high samples per traces (lets say more than 35k), then machine learning models perform poorly on raw data. In this scenario a better model is required. This chapter presents a deep learning CNN-based architecture which utilizes the dimensionality reduction and class imbalance techniques for improving the efficiency and performance. In previous chapters, for asymmetric ECC, two algorithms always-double-and-add and Montgomery Powering Ladder (for RNS) side-channel leakages are analyzed for NIST SECP256K1 curve and Edward curve. However, in this chapter the presented model is evaluated on ECC Montgomery Power Ladder (MPL) protected (BEC) and unprotected implementation leakage datasets which have low instance-feature ratio. A comparative analysis is provided to assess the presented model by analyzing the impact of dimensionality reduction and class imbalance alone and by comparing the performance results with the existing complex models.

8.1 Abstract

Deep learning-based side channel attacks are burgeoning due to their better efficiency and performance, suppressing the traditional side-channel analysis. To launch the successful attack on a particular public key cryptographic (PKC) algorithm, a large number of samples per trace might need to be acquired to capture all the minor useful details from the leakage information, which increases the number of features per instance. The decreased instance-feature ratio increases the computational complexity of the deep learning-based attacks, limiting the attack efficiency. Moreover, data class imbalance can be a hindrance in accurate model training, leading to an accuracy paradox. We propose an efficient Convolutional Neural Network (CNN) based approach in which the dimensionality of the large leakage dataset is reduced, and then the data is processed using the proposed CNN based model. In the proposed model, the optimal number of convolutional blocks is used to build powerful features extractors within the cost limit. We have also analyzed and presented the impact of using the Synthetic Minority Over-sampling Technique (SMOTE) on the proposed model performance. We propose that a data-balancing step should be mandatory for analysis in the side channel attack scenario. We have also provided a performance-based comparative analysis between proposed and existing deep learning models for unprotected and protected Elliptic curve (ECC) Montgomery Power ladder implementations. The reduced network complexity, together with an improved attack efficiency, promote the proposed approach to be effectively used for side-channel attacks.

8.2 Introduction

Embedded device security in the internet of things (IoT) based systems is of paramount importance, and security measures should be integrated at the design level [33]. Public Key (asymmetric key) algorithms like Elliptic Curve Cryptography (ECC) are recommended for such resource-constraint environments [106–110]. These algorithms are theoretically and mathematically secure, but their weak implementations can lead to security breaches through side channel attacks. Side channel attacks can exploit the secure algorithm implementations by analyzing the side-channel leakages, including power signals, electromagnetic emanations, timing information, etc. [75, 134, 135, 174]. Traditionally, profiled-based template attacks are considered one of the strongest side-channel practical attacks. In these attacks, the adversary has access to the open copy of the target device [136]. Successful practical template side-channel attack designs have been proposed over the past decade [175]. Machine learning (ML) analysis has been proposed as a mechanism to improve the side-channel attacks due to the similarities between template attacks and machine learning-based data analysis [20, 70, 137].

Elliptic Curve Cryptography (ECC) based public-key algorithms are the preferred choice for authentication, digital signatures, certificates etc. in the resource-constraint environments due to their efficient processing and small key size [106–110]. ECC algorithms are mathematically secure but their weak implementations can introduce many exploitable side-channel attack vulnerabilities. Machine Learning attacks have been extensively performed and studied for side channel leakages from symmetric key algorithms (for example Advanced Encryption Algorithm, AES). However, very limited analysis exists for the asymmetric key-based public-key algorithms like RSA and Elliptic Curve Cryptography (ECC) [125] [126]. Some of these attacks use simple machine learning algorithms (including Random Forest or Support Vector Machine) [126]; however, deep learning (DL) techniques seem more promising for side channel analysis due to the noisy nature of the side channel leakage signals.

Additionally, in most of the scenarios side channel leakages are misaligned, and require pre-processing to exploit the leakages for recovering the secret information, which can be a tedious and possibly discouraging task for an attacker. Convolutional Neural Network (ConvNet) based deep learning technique constitutes an ideal candidate for eliminating the leakage traces' excessive noise. More specifically, the convolutional layer in ConvNet reduces the leakage trace samples by extracting and learning from only essential features by assigning weights and eliminating noise. Cagli et al. have proposed to use ConvNets for side-channel attacks and have shown successful results on data, for symmetric algorithm implementations, without requiring any pre-processing or alignment [23]. Kim et al. have shown the impact of adding noise to existing samples, which helps recover the secret information with reduced samples [73].

Furthermore, selecting important features or points of interest is crucial while launching side-channel attacks. Traditionally, various methods are proposed to select POIs [119]. Recently, Picek et al. and Mukhtar et al. have proposed feature engineering techniques to achieve optimal results by analyzing the impact of using the feature engineering techniques on side-channel leakages and processing them further by using machine learning classifiers [139] [176].

However, using ConvNets for side channel analysis still suffers from several problems. Firstly, ConvNets require a huge amount of traces/instances to extract sensitive information from the side channel leakages. This requirement becomes more exacting in complex cryptography algorithm implementations (like public-key cryptography algorithms), where the high sampling frequency is needed to ensure that enough leakage information is acquired. Hence, generating an enormous leakage dataset that is processed further to recover the secret information by utilizing the deep learning classifiers' pattern recognition capability, as proposed by various studies without applying any pre-processing or alignment on the data [73] [133]. This removes the need to use any pre-processing at a considerable computational complexity cost. The huge datasets lead to increased computational complexity and hardware resource usage of the deep learning based side channel attacks which, in turn, leads to substantial time to train the model and launch the attack. In several security scenarios where the secret information's life span is important, this delay in retrieving the secret might be unacceptable. One possible solution in such scenarios is to reduce the input dataset size by using feature extraction techniques. In non Machine Learning (classical) side channel analysis, principal component analysis (PCA) has been proposed as a pre-processing step to select the important features [155]. For machine learning-based side-channel attacks, Golder et al. have presented results for using PCA as a pre-processing step for classification using ML on symmetric ciphers [156]. However, all the existing machine learning-based side channel analysis with PCA pre-processing are applied to symmetric cipher datasets, and no substantial work has been done on public-key cryptosystems. Moreover, the number of samples (or features) per instance is generally small (ranging from 400 - 6000) in the existing studies [73, 133]. However, in the presented case of an asymmetric cipher, the number of samples per trace/instance is very large (33000 precisely).

Secondly, another aspect that can create problem while training side-channel leakages with the deep learning algorithms is the amount of data instances/traces per target class. If class data is highly imbalanced, it can hinder accurate modeling by giving rise to an accuracy paradox. Traditionally, there are data-level and algorithm-level data balancing techniques that can balance the target classes and improve the trained model performance [171] [177]. Picek et al. have recommended using the Synthetic Minority Over-sampling Technique (SMOTE) to balance data, based on the experimental findings for symmetric-key algorithm leakage information [132]. However, there is no analysis using SMOTE for side channel leakages of the public-key cryptography algorithm implementations.

Contributions - In this paper, we provide solutions to the above problems and offer a thorough study for performing ConvNet based deep learning side channel attacks on Elliptic Curve Cryptography scheme implementations, efficiently. We propose a hybrid deep learning-based attack methodology and an analysis framework to improve the side-channel attacks on imbalanced leakage datasets by using the combination of dimensionality reduction and class imbalance techniques along with the proposed simple Convnet model. The optimal number of convolutional blocks are used to build the powerful features extractor within the cost limit. The proposed efficient ConvNet-based approach has been evaluated for both protected and unprotected ECC scalar multiplication Montgomery Power Ladder (MPL) implementations. High sampling frequency was used during the data collection process to fully capture the side channel leakage of the public-key ECC implementations. Thus, 33000 samples per trace for analysis were collected, resulting in a massive dataset with a low instance-feature ratio. To handle the high computational complexity of the attack due to this massive dataset, we proposed a time-efficient model for analyzing public-key cryptographic (PKC) schemes (ECC), based on the dimensionality reduction. Moreover, in line with the findings for symmetric ciphers, to solve the imbalance problem in traces per class, we have analyzed SMOTE's impact on the public-key ECC implementations using our proposed attack architecture. Based on the findings, it is determined that the data balancing should be included as a mandatory step for a reliable attack model for public-key cryptosystem. Our proposed method enables the network to train much faster with better performance than the existing state-of-the-art methods, as shown by our performed comparative analysis between the proposed and other traditional existing models.

The rest of the paper is organized as follows. Section 8.3 provides some background information and briefs the techniques and models used in this study. Section 8.4 explains the implementation approach and introduced countermeasures of the PKC algorithm under analysis. Section 10.4 describes the proposed methodology and analysis framework. Section 8.6, explains the experimental setup. Section 8.7 presents the results and discussions on both (protected and unprotected) ECC dataset using proposed ConvNet architecture with PCA and SMOTE. Section 10.7 concludes the paper.

8.3 Background and Preliminaries

Assuming P is a point on the Elliptic Curve $E(F)$ defined over a finite field [1] F then this point is characterized by its coefficients x, y i.e. $P : (x, y)$ where $x, y \in F$. Scalar multiplication (SM) i.e., $e \cdot P$, where e is an integer, is the main operation used in Elliptic Curve Cryptography, and it has been widely studied for its side channel attack resistance. SM relies on the repetition of many point addition and point doubling operations that themselves are implemented using finite field arithmetic operation like modular addition, subtraction, inversion, and multiplication (in $GF(p)$ or $GF(2^k)$). Since modular inversion is a computationally complex operation, most designers exchange it with several modular multiplications

[1]In cryptography, prime finite fields, GF(p), and Binary extension fields, $GF(2^k)$, are used

and addition/subtractions by transforming the Elliptic Curve and its points from the affine coordinate domain to the projective coordinate domain [178]. In the projective coordinates domain, each point is represented by three coordinates i.e $P : (X : Y : Z)$ where $X, Y, Z \in F$.

8.3.1 Side Channel Attacks

Traditional algorithms implementing SM (e.g., double-and-add algorithm) have serious imbalances associated with the value of each bit of the secret scalar (e); thus strong association of SM computation can be made with the secret scalar been processed. There is a broad range of SM focused SCA attacks both simple and advanced or horizontal and vertical [179] [180] and [181].

Simple SCAs can be easily mounted in the double-and-add algorithmic approach followed in SM and are typically horizontal type of attacks i.e., they can be mounted using a single leakage trace that is processed in time. Such simple SCAs can be easily countered by using highly regular SM algorithms i.e. algorithms in which each round's operations are unrelated to the scalar bit that they are processing (e.g Double and always Add algorithm or Montgomery Power Ladder (MPL)[182]). However, there are a series of SCAs, known as comparative SCAs (focused initially on Power attacks (PAs) but also extended to Electromagnetic emission (EM) attack) that still manage to overcome the above regularity by manipulating the base point input of the SM (doubling attack (collision-based attack) [183] and its variants [184] or the chosen plain text attack in [185] (also known as 2-Torsion Attack (2-TorA) for ECC).

There are, however, more advanced attacks (advanced SCAs) on EC SM both of vertical and horizontal nature (where the attack needs many traces or a single trace, respectively). Differential Attacks (DSCA), originally proposed by Kocher in [186] for power consumption leakage, is the most widely known such attack. These attacks appear in many variations based on the used hypothesis distinguishers, leading to sophisticated DSCAs like Correlation SCA (requiring less traces to reveal the secret than DSCA) [187] and collision correlation attack [188] [189] [190]. These attacks are possible even when a single trace is available (horizontal attacks) [191] or the Horizontal Collision Correlation attack (HCCA) [181], [180].

Whitnall et al. [192] suggest that there are considerably more potent SCAs than DSCAs, known as profiling attacks. Such attacks rely on a profiling phase on the device under attack. In the profiling phase, an attacker identifies the leaking operation (Point of Interest, PoI) and produces all possible different states of this operation by feeding the device with all possible secret key value inputs (e.g., one byte or one bit). These states are statistically analyzed to create an identifiable profile for each secret key value. An attack phase then follows where the attacker targets a device with an unknown secret scalar and collects PoI leakage traces for various inputs using the same trace collection mechanism and parameters as in the previous phase. Using the profile and some discriminator, the attacker tries to identify the appropriate leakage trace from the profile that has a high probability of matching the unknown secret leakages and retrieves the secret. The most common type of such profiling attacks are template and online template attacks (TA) [175] [87] [193].

The concept of profiling a device to create a leakage model based on labeled leakage traces has been explored further by researchers using ML techniques for creating a profile. Using ML, the attacker does not need to create a perfect leakage model but rather lets an ML algorithm be trained with a non-exhaustive series of leakage traces (that can be associated/labeled to some, instead of all, secret block values). As the leakage noise increases (possibly also due to masking or hiding countermeasures), the ML profiling approach tents to provide better results as compared to traditional attacks [20].

8.3.2 Montgomery Ladder Algorithm and Countermeasures

Given the above analysis, an EC SM implementation should include appropriate countermeasures to be protected against a broad range of attacks. Profiling SCAs, as various researchers highlighted, [193] [175] [194], can overcome several existing countermeasures, indicating the need for a more sophisticated randomization throughout the whole computation flow of the scalar multiplication algorithm.

One of the most popular such variations of secure scalar multiplication algorithms is the Montgomery Power Ladder (MPL) algorithm. As seen in Algorithm 1, it has strong regularity in each round of operations (step 2 of Algorithm 1) and minimal interference from the secret scalar bit value. This MPL regularity is manifested by the constant number of identical point operations performed in each scalar round regardless of the corresponding secret scalar bit [182] and prohibits an attacker from performing simple horizontal and vertical SCAs. Apart from that, MPL favors parallelism since step 2a or step 2b operations (point addition (R_0+R_1) and point doubling ($2R_0$ or $2R_1$)) can be performed in parallel. Apart from the performance benefit that such a feature offers, it can potentially scramble side channel signals to identify each one of those two operations that can become difficult for an attacker. However, the MPL algorithm still leaks some information about the scalar bit since the outcome of point addition and point doubling in each MPL round is stored in the different storage area (eg. registers) depending on the processed scalar bit value. For example point doubling result is stored in R_0 when $e_i = 0$ and in R_1 otherwise. This subtle irregularity can be identified and exploited using profiling SCAs (e.g., template SCAs and ML SCAs) to retrieve the secret scalar e [194].

Algorithm 1. Simple SCA resistant MPL algorithm

```
Input: $P$ : EC base point $\in EC(F)$,
$e = (e_{t-1}, e_{t-2}, ...e_0) \in GF(2^k)$
Output: $e \cdot P$
1. $R_0 = O$, $R_1 = P$
2. For $i = t - 1$ to 0
    If ($e_i = 0$) then
      (a) $R_1 = R_0 + R_1$, $R_0 = 2 \cdot R_0$
    else
      (b) $R_0 = R_0 + R_1$, $R_1 = 2 \cdot R_1$
    end if
3. Return $R_0$
```

To remedy the MPL SCA problems, SCA countermeasures fitting into two different categories can be used, leakage hiding or leakage masking [195] [196]. In the hiding approach, appropriate measures are included in SM computation flow so that the leakage of point addition/doubling operation or storage area is made indistinguishable from random noise. To achieve that algorithmically within the EC SM we can introduce dummy operations in the computation flow or modify the algorithm, so traces of one MPL operation are very similar from traces of another operation (or their result storage process). Since MPL favors parallelism that enables a designer to merge operations in time (more than one point operation is processed in each time frame), hiding can be achieved by scrabbling the operation traces at the same time frame.

Masking aims at disassociating the sensitive information from the leakage trace. This approach relies on some form of randomization (additive or multiplicative) on the sensitive information associated with a leaky MPL point or storage operation. As originally proposed by Coron in [197] and later extended and adapted by various other researchers [196] EC SM masking techniques aim to randomize the EC multiplication secret scalar (e), the input point P (base point blinding), or the input point's projective coordinates (X, Y, Z). The most easily applicable are the first and the last (scalar blinding and projective coordinate blinding) since the point blinding technique requires the introduction and storage of a random point in each scalar multiplication [25].

8.3.3 Data Imbalance Technique - SMOTE

Data imbalance, meaning the number of instances of each class is not equal, leads to misclassification and can give rise to an accuracy paradox. In application domains of machine learning, there are various techniques to handle imbalance classes for accurate modeling. One of the techniques to address the class imbalance issue is to modify the input training data distribution to decrease the imbalance ratio of the target classes. There is no guarantee to have an equal amount of leakage bit information in side-channel leakage data, especially in multi-class classification problems. In this research, we have studied SMOTE's effect on improving the secret data recovery attack efficiency.

Generally, for imbalance datasets, under-sampling and over-sampling techniques are used. In under-sampling, majority class data instances are removed to bring it to the minority class level. In over-sampling, more samples are added for the minority class instances. Under-sampling discards data, which might contain important information required for accurate classification. On the other hand, over-sampling increases computation time and can cause over-fitting. To address these issues, numerous intelligent under-sampling and over-sampling techniques have been introduced to preserve sensitive information. Kubat et al. have presented a method for removing noise and redundant data from the majority class using one-sided selection [198]. Algorithms based on K-nearest neighbors (K-NN) classifiers are proposed to remove the majority samples based on their distance from minority samples [199]. Among all sampling techniques, the over-sampling technique of SMOTE is the most popular one. SMOTE generates artificial samples for the minority class synthetically, using minority samples and their minority neighbors [171]. Picek et al. presented results for SMOTE's performance on the side channel leakages from symmetric ciphers [132].

8.3.4 Principal Component Analysis

In the principal component analysis, the dimensionality of data is reduced to increase interpretability. It uses an orthogonal linear transformation to re-position the data onto a new coordinate system, and a new reduced smaller feature dataset is formed based on the existing feature space [158]. The feature that explains the maximum amount of variance is positioned at the new dataset's first location. PCA helps discard the features that capture similar information and thus aids in creating a more parsimonious model.

8.3.5 Convolutional Neural Networks

CNN is a deep learning algorithm that takes input signals data and learns the differentiating aspects of the target class by assigning weights and importance. Generally, CNN consists of convolutional layers, a flatten layer, a pooling layer, and fully connected layers [72, 200]. Activation functions are used in each layer to deal with the non-linearity. The convolutional layer performs convolution on the input features, using filters/kernel to recognize the data's patterns. This filter hovers over the complete data trace from left to right, based on the set stride, and reduces the input features dimensionality by convolution. Generally, dimensionality can be reduced or stays the same depending upon the padding being used. For this layer, kernel and stride are the hyperparameters which can be tuned further to obtain a good performing model. The pooling layer is an approach to reduce the sample size by downsampling the features from the feature map by summarizing features in patched regions. The intuition of using a pooling layer is to select a dominating feature from a particular layer in a particular region. If the feature is not dominating, then the resulting value will be small and will wear out with further pooling in the next layer. Hence, it helps in reducing the computational complexity, combats over-fitting, and encourages translational invariance. It takes a filter, but instead of applying convolution, it either takes the maximum value from the feature map region or takes the average. Based on this, there are two main widely used pooling methods; max-pooling and average pooling. A combination of the convolutional layer and pooling layer forms a pair i-th layer in a neural network architecture. The number of such layers can be increased to capture the minor low-level details. Increased convolutional layers enhance the overall model's computational complexity, which takes a longer time in training. The existing proposed architectures for side-channel analysis are complex, consisting of numerous layers with a large number of filters [23, 73]. We have selected one such complex ConvNet architecture for comparison in this study. The existing architecture has been evaluated for AES leakages. However, we have tested the same network for ECC leakage data, and then we have presented results by evaluating with our proposed architecture.

8.4 Hardware Design and Implementations

As a target of the proposed deep learning side channel attacks, two EC SM hardware implementations of Binary Edwards curves (BEC) on $GF(2^k)$ has been chosen, based on the work of Fournaris et al. in [25]. Both implementations have the BEC intrinsic protection against SSCAs[2], use the MPL algorithm described as Algorithm 1 and exploit the parallelism in step 2a or 2b of the algorithm in order to achieve efficiency and side channel attack resistance. In [25], point addition and point doubling operation are decomposed in their basic finite field operations, as shown in Table 8.1, and those operations are examined for their data dependability. Those operations that are data-independent (they do not rely on the result of some other finite field operation) are grouped in stages to be computed in parallel using some constrained number of parallel processing elements. In the architecture design of [25], three modular multiplier processing elements and three modular adder processing elements are used, operating in parallel, thus producing 11 parallel stages of grouped finite field operations

[2]They offer completeness and uniformity

(shown in Table 8.2.

TABLE 8.1: Point Operations Partial Results [25]

Point Addition $(X_3 : Y_3 : Z_3) = (X_1 : Y_1 : Z_1) + (X_2 : Y_2 : Z_2)$	Point Doubling $(X_{3D} : Y_{3D} : Z_{3D}) = 2(X_1 : Y_1 : Z_1)$
$A = X_1 \cdot X_2$	$DA = X_1 \cdot X_1$
$B = Y_1 \cdot Y_2$	$DC = Y_1 \cdot Y_1$
$C = Z_1 \cdot Z_2$	$DE = Z_1 \cdot Z_1$
$D = d_1 \cdot C$	$DB = DA \cdot DA$
$E = C \cdot C$	$DD = DC \cdot DC$
$F = d_1 d_1 \cdot E$	$DH = DA \cdot DE$
$G_1 = X_1 + Z_1$	$DI = DC \cdot DE$
$G_2 = X_2 + Z_2$	$DL = DE \cdot DE$
$G = G_1 \cdot G_2$	$DF = d_1 \cdot DL$
$H_1 = Y_1 + Z_1$	$DJ = DH + DI$
$H_2 = Y_2 + Z_2$	$DO = d_2 \cdot DJ$
$H = H_1 \cdot H_2$	$DM = DB + DD$
$I = A + G$	$DG = d_1 d_2 \cdot DM$
$J = B + H$	$DK = DG + DO$
$K_1 = X_1 + Y_1$	$DL_1 = DF + DJ$
$K_2 = X_2 + Y_2$	$DL_2 = DH + DD$
$K = K_1 \cdot K_2$	$DL_3 = DI + DB$
$L = d_1 \cdot K$	$DX_3 = DL_2 + DK$
$U_1 = K + I$	$DY_3 = DL_3 + DK$
$U_2 = J + C$	$DZ_3 = DL_1 + DG$
$U_3 = U_1 + U_2$	
$U_4 = L \cdot U_3$	
$U_5 = F + U_4$	
$U = C \cdot U_5$	
$V_1 = A \cdot B$	
$V_2 = G \cdot H$	
$V_3 = d_1 \cdot E$	
$V_4 = V_1 + V_2$	
$V_5 = V_3 + V_4$	
$V_6 = L \cdot V_5$	
$V_7 = D \cdot F$	
$V_8 = V_7 + V_6$	
$V = Z_3 + V_8$	
$M_1 = A + D$	
$N_1 = G + D$	
$O_1 = M_1 \cdot N_1$	
$M_2 = B + D$	
$N_2 = H + D$	
$O_2 = M_2 \cdot N_2$	
$P_1 = D \cdot O_1$	
$P_2 = D \cdot O_2$	
$X_3 = V + P_1$	
$Y_3 = V + P_2$	
$Z_3 = U$	

As can be observed in Table 8.2, the parallel finite field operations provided in each stage are not associated with only one point operation (point addition or point doubling) from Table 8.1 of an MPL round. This scrabbling mechanism can potentially prohibit identifying the performed point operation and/or storage since both point addition and doubling are performed in parallel using the same structural blocks (processing elements). The storage pattern (meaning, which intermediate results are stored in which register) is very similar in every round except some multiplexer units at the end of the computation, as described in [25]. This approach was designed to provide resistance against advanced SCAs and template attacks. However, by performing a Welch's t-test [3], in [25], the authors discover that there is still non-trivial leakage of the secret scalar key during the SM computation. So, while the computation processing in each MPL round (all stages) and its storage pattern is fairly regular regardless of the secret scalar bit, there are still indications that some attack could potentially succeed in recovering the secret scalar key. In this paper, given the above remark, we use the proposed DL/ML attack methodology on an implementation (denoted as unprotected implementation) that is produced through the above-described process.

To solve the above-described leakage issue, in [25], a mechanism based on random operations is introduced in the parallel stages to mask the values stored in the implementations' registers. Using a random number generator integrated in the hardware implementation, a random value *r* is generated in each MPL round. This *r* is used in two extra stages of parallel operations that are introduced in Table 8.2 in order to multiplicatively mask the values of all performed finite field operations involved in an MPL round. The additional operations, colored in blue in Table 8.2, are following the random coordinates countermeasure approach, but instead of performing randomization once per SM, the countermeasure is expanded by re-randomizing the coordinates at every MPL round. This effectively masks/eliminates any processing leakage association between the scalar bits and the processed MPL round parallel operations. The Welch's t-test on such an implementation, denoted as protected implementation, applied in measurements of [25] indicates that indeed there is only trivial leakage of the secret key in all MPL rounds. However, although advanced SCAs may fail to retrive the secret scalar in the protected implementation (due to trivial leakage), profiling attacks (e.g., ML SCAs) may be successful since the storage leakage pattern (not assessed using TVLA) is mostly the same compared to the unprotected implementation of [25]. In this paper, we evaluate the proposed DL/ML attack methodology on this protected implementation to identify its efficiency, accuracy and to explore the limits of the TVLA test as a trusted SCA assessment approach in the presence of DL/ML attacks.

[3]The constant versus random scalar methodology was used following the Test Vector Leakage Assessment technique

TABLE 8.2: Paralleling BEC point addition and doubling $GF(2^k)$ operations

Inputs		$(X_1 : Y_1 : Z_1)$		$(X_2 : Y_2 : Z_2)$		
Stage	**M1**	**M2**	**M3**	**Ad1**	**Ad2**	**Ad3**
1	A	B	DA	G1	G2	K1
2	G	DC	DB	H1	H2	K2
3	H	DD	DE	I	-	-
4	C	DH	DI	J	-	DM
5	V2	V1	K	DJ	DL2	DL3
6	DO	E	DG	U2	V4	U1
7	D	V3	L	DK	-	U3
8	DL	F	U4	M1	V5	N1
9	V7	V6	O1	M2	N2	U5
10	DF	O2	U	V8	DX3	DY3
11	-	P2	P1	DL1	-	V
12	rDY3	rZ3	rDX3	DZ3	Y3	X3
13	rY3	rX3	rDZ3	-	-	-
Outputs		$(X_3 : Y_3 : Z_3)$		$(X_{3D} : Y_{3D} : Z_{3D})$		
Rand Outputs		$(rX_3 : rY_3 : rZ_3)$		$(rX_{3D} : rY_{3D} : rZ_{3D})$		

- : idle r: random number

8.5 Proposed Attack Methodology and Evaluation Framework

To launch a deep learning-based side-channel attack, assume the adversary is in possession of the open copy of the device and has computational and resource capacity to obtain and process the side channel leakage information. However, we assume that the adversary wants to recover the secret information in requisite attack time T_A, from the obtained leakage traces L_T. The proposed deep learning-based attack methodology is systematically divided into five steps. In step 1, the leakage data L_T is formatted and labeled to identify the target class for each trace. In step 2, the prepared dataset is processed to balance the target class instances synthetically. In step 3, the dimensionality is reduced using PCA, and in the last step, classification is performed using the proposed CNN model. Each step is further elaborated in Sec. 8.5.1 - 8.5.4. However, Sec. 8.5.5 describes the strategy followed in this study to evaluate the proposed approach.

One of the critical concerns in neural networks is over-fitting while dealing with the side-channel noisy leakages. In over-fitting, the model learns from the data so well, or we can say it learns from the noisy patterns as well, that it creates a model with high variance. The resulting model will fail to generalize on the unseen data. To handle the problem of over-fitting, we have taken specific measures at various stages of the analysis. Each measure is explained in the respective section.

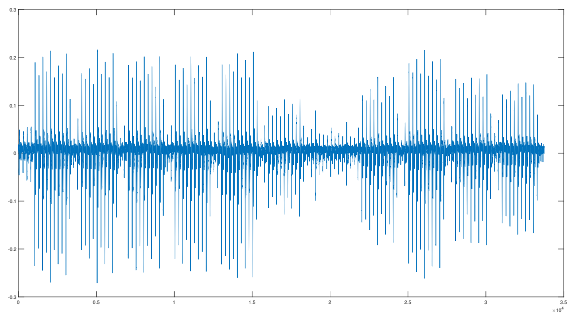
8.5.1 Step 1 - Dataset Preparation

For analysis in this research work, both protected and unprotected implementations of the EC MPL algorithm are analyzed, as explained in 8.3.2. For both implementations, to launch bit level machine learning based side channel attack, at first data traces T, of length S_T, are collected for each bit operation, and then each trace is labeled as target class '0' or '1', based on the processed bit during leakage collection. The formed datasets are then processed through a machine learning classifier to train the model. The trained model is finally tested on unseen data to predict the key bit used for the encryption. The resulting labeled signals are shown in Fig. 8.1. The trace where collected, following the approach in [201], using a PicoScope 5000D Series Oscilloscope with a sampling rate of 1GS/s that was connected through a pre-amplifier to a resistor onboard a SAKURA-X FPGA board. The description of both the datasets is given below:

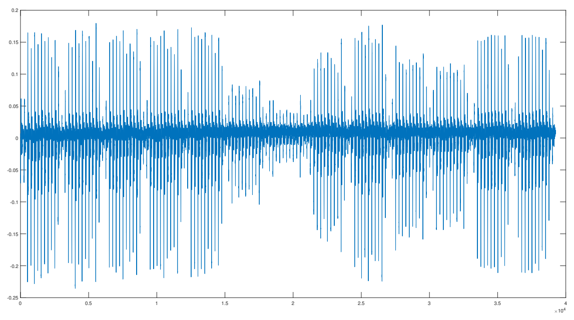
- Leakage Dataset Unprotected LD_{UP} - This dataset consists of side-channel leakage traces for MPL implementation on FPGA and is not protected by any countermeasure. This dataset consists of $T = 5,000$ data traces (instances), and each instance consists of $S_T = 33750$ samples.

- Leakage Dataset Protected LD_P - This dataset consists of side-channel leakage traces for MPL implementation on FPGA, which are protected by the countermeasures described in Sec. 8.4. This dataset consists of $T = 5,000$ data traces (instances) and each instance consist of $S_T = 39250$ samples.

To handle over-fitting, we have divided our datasets further into three subparts; training data, validation data, and testing data, in the ratio of 60:20:20 %, respectively. Training and validation data is used during training the model. However, test data is held back and is never shown to the model during training, which ensures that the test data's analysis produces reliable results during the testing phase.



(a)



(b)

FIGURE 8.1: Obtained Raw Data signals after labeling (bit 0) for (a) Unprotected and (b) Protected Implementations

8.5.2 Step 2 - Handling Class Imbalance

After obtaining the data and forming the datasets, the next step is to balance the target class instances. An imbalance dataset can lead to an accuracy paradox by misclassifying data due to the empowering majority class. For analysis of the side-channel leakage data, we propose to use the class balancing technique as a mandatory step to balance the classes before applying a machine learning classifier, for a better reliable trained model. Our presented case of bit-level attack is a binary classification problem where two class key bits '0' and '1' need to be classified. To analyze the impact of the class imbalance technique, we have generated the datasets with less number of 1's and more number of 0's. To be precise, there are 1500 and 2600, samples for 1's and 0's, respectively. After generating the dataset, SMOTE is applied. As explained in 8.3.3, SMOTE is a synthetic oversampling technique; we have increased our samples for the under presented class, which is '1'. After applying SMOTE, both classes have an equal number of instances. The new generated samples have the same characteristics as those of the training dataset samples.

8.5.3 Step 3- Dimensionality Reduction and Data Visualization

After handling class imbalance, we have reduced the number of features using the dimensionality reduction technique, Principal Component Analysis (PCA). PCA has been used for traditional analysis with regards to side-channel leakage data. PCA can capture and highlight the dataset's maximum variance in just a few principal components, hence, transforming the useful information by eliminating the redundant features.

In our presented case, as the number of instances (traces) is less than the number of features (no of samples per trace), so use of pre-processing or feature engineering, to reduce the number of samples/features, can aid in reducing the computational complexity and also will help in training a better-trained model. The extra features certainly contain redundant information and noise. Usually, deep learning is expected to pick up the data anomalies, but that might not always be true, especially if the ratio of instances to features is very low. In some instances, this can give rise to over-fitting, where the model learns from the noise instead of learning from the relationship between the secret information and leakage traces. Due to the noisy nature of the leakage information, for machine learning-based side-channel analysis, it is of crucial importance to select the most contributing features. Training the model with reduced feature dataset has various benefits, including reducing training complexity and accurate trained model.

PCA can achieve this goal because it tries to find a linear subspace that best fits our data. The aim is to minimize the sum of square of orthogonal distances or maximize our data's spread within low dimensional subspace. Let X be our mean subtracted data. To find subspace w such that our data have maximum spread in this subspace, we use,

$$w = \underset{\|w\|=1}{\arg\max} \left\|w^T X\right\|_2^2 \tag{8.1}$$

$$w = \underset{\|w\|=1}{\arg\max}\, w^T X X^T w \tag{8.2}$$

If we take the Lagrangian of w and then its derivative, we got

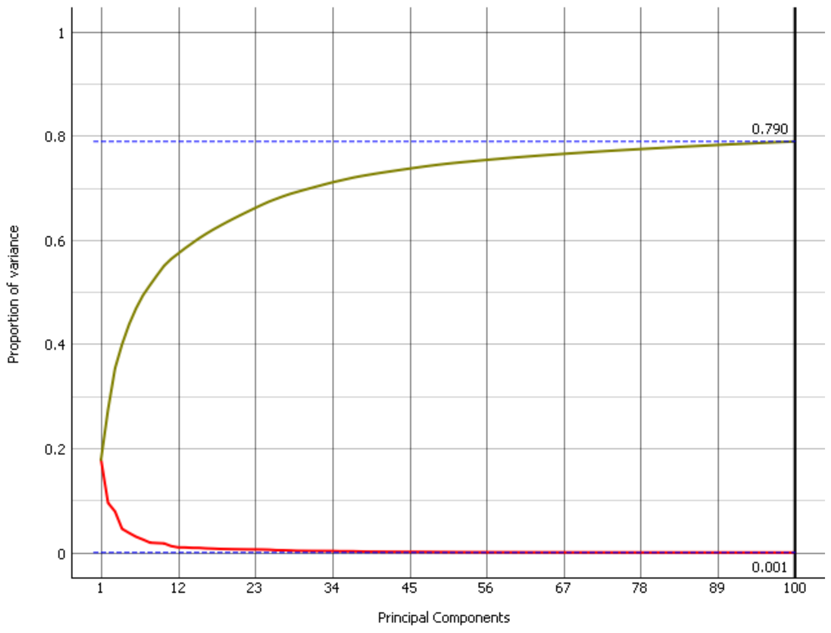
$$X X^T w = 4\lambda w \tag{8.3}$$

This ends it up with an eigenvalue problem. If we solve Eq.8.3 our solution w will give first principal component (largest eigenvalue). To get other eigenvalue, we need to subtract the largest value (already found) from X and find out the next largest value and so on.

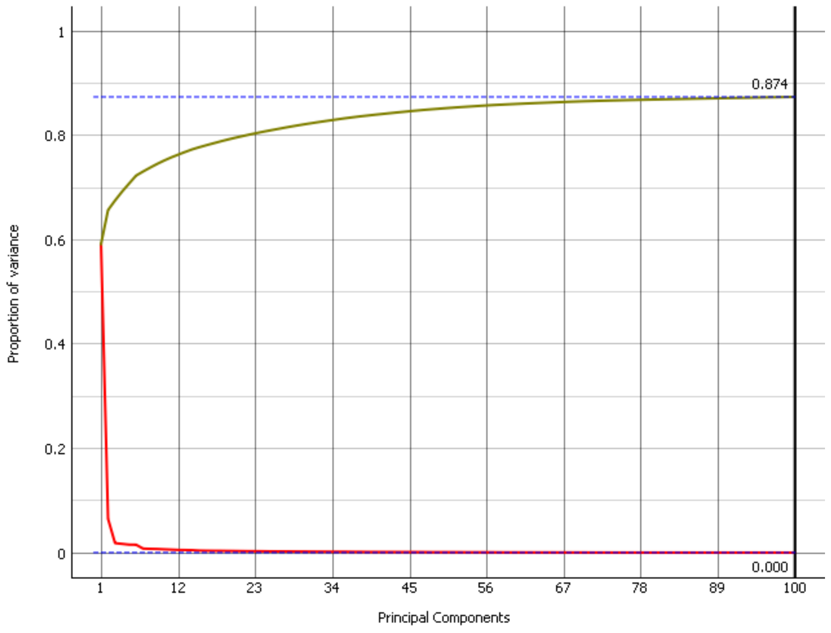
Fig. 8.2 shows the proportion of variance due to PCA components for both protected and unprotected leakages. It can be seen that the variance of 79% and 87% is covered with 100 PCA components. The maximum variance will be covered if PCA principal components are selected beyond 100. To analyze the effect of the principal components' various sizes, we have performed analysis using the number of components from the group PCA_{CG} where PCA_{CG} = 200, 400, 600, 800, 1000 and 1200 principal components. Analysis results are given in the results section.

8.5.4 Step 4 - Modeling using Deep Learning Classifier

The instances of the target class '0' and '1' are balanced using the synthetic data balancing approach, SMOTE, and then the dimensionality of the data traces is reduced by applying PCA, based on the conclusions deduced from the observations in 8.5.3. In the last step, machine learning analysis is performed using the proposed Convolutional Neural Network (ConvNet/CNN) architecture. The proposed architecture is simple compared to the complex existing architectures and produces the same accuracy level in less time.



(a) a



(b) b

FIGURE 8.2: Proportion of Variance for PCA components for (a) Protected and (b) Unprotected

CNN Proposed Architecture

The proposed simple CNN architecture is shown in Fig. 8.3.

There are three combinational layers in our proposed design, as shown in the summary Table 8.3. Each combinational layer consists of a pair of convolutional layers and a pooling layer. However, in the last two combinational layers, an extra convolutional layer is added to extract more information before the pooling layer. There are five convolutional layers in

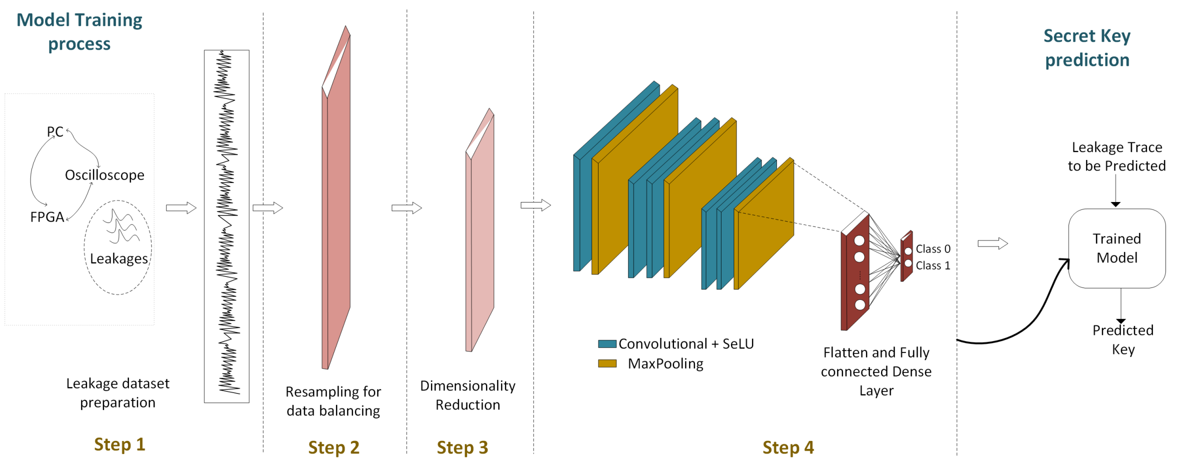


FIGURE 8.3: Proposed Hybrid Convolutional network-based system for the secret key recovery from the acquired side channel leakages. The figure shows the overall proposed system which consists of Convolutional network layers, dimensionality reduction module and a class imbalance module.

the proposed architecture, consisting of 4,8,8,16, and 16 filters with specific kernel size and stride, which enables the model to distinguish the secret key bit. Kernel size and stride are varied between 4 and 8. In our proposed architecture, we have used max-pooling, in which the maximum number is selected from a particular region. It has two primary hyperparameters, filter and stride. Once these hyperparameters are fixed, they do not change during the learning process. For our case, the value is set to 1-2.

Activation functions are used in convolutional layers to deal with the non-linearity of the data. We have tested various activation functions for our analysis, as listed in the Table 8.4, and selected Scaled Exponential Linear Unit (SeLU) as it produced the best results. Because of its ability to self-normalise, SELU has shown improved performance in various classification tasks using feed-forward neural networks [202]. One of the advantages of SeLU is that its internal normalisation is faster than external normalisation, which means that the network converges faster. As the gradient problem of vanishing and exploding is impossible in SeLU, it can be a reason for its improved performance on our netwrok.

The previous max-pooling layer's output is flattened to form a column vector and is then connected to the fully connected layer. Fully Connected Layer is the final layer that takes the output of the previous flatten layer as input and then outputs N dimension vector where N is the number of the output target classes ($N = 2$ in this case), and then back-propagation

TABLE 8.3: Proposed CNN Model Summary

Layer (type)	Output Shape	Number of Parameters
$Conv1d\_1$	199x4	20
$conv1d\_2$	66x8	136
$conv1d\_3$	21x8	264
$conv1d\_4$	6x16	528
$conv1d\_5$	1x16	1040
$dense\_1$	2	34

is applied to each iteration of training during the epoch. After training over a few epochs, the model is able to learn from the provided features and classifies them using the softmax classification technique. We have trained our model for a longer time for 200 epochs, which provides enough batch training cycles to analyze the model performance. In our results, it is seen that the model performance becomes stable before 50 epochs.

Normalization and Over-fitting

Having huge differences between the maximum and minimum value in the data might degrade the learning process. Normalization is performed to speed up the learning process, and the model converges quickly, which results in an accurate trained model. As mentioned before, over-fitting is one of the issues in noisy side-channel leakages. The model can learn data patterns along with the noise. Such a model performs well on the training data but fails to generalize on the test (unseen) data. Specific techniques can be used to avoid over-fitting, including dropout and regularization. We have tried both and found better results with L2 regularization. It manages the weights and keeps them small in order to avoid over-fitting. In addition to learning from the noise, the duplicate instances within the training dataset can also result in an over-fitted biased model. To avoid this, duplicate rows are removed from the training dataset.

TABLE 8.4: Parameter tuning CNN

Parameter	Value Range
Learning Rate	[0.001,0.01,0.1, 0.5]
Epochs	[200]
Strides	4-6
Kernel Size	5-8
Pool Size	1-2
Pool Stride	1-2
Activation function	[relu,selu,elu,tanh,softplus]
Optimizer	[Adam,Nadam,RMSprop,Adamax,sgd]
Initialization Mode	[uniform,normal]
Batch Size	[32, 100]

Hyperparameter tuning

There are certain hyperparameters related to each layer, which can be tuned to improve the CNN performance. Table 8.4 shows the lists of parameters that are tuned to select the best performing model. Grid search functionality, available in the Scikit library, is used. In grid search, exhaustive search is performed, using all possible parameter combinations, and the best performing parameters are selected based on the model accuracy.

8.5.5 Evaluation Strategy

In order to systematically analyze the affect of the proposed neural network based side channel attack on the leakage data, analysis is further divided into four sets, as given below.

- Analysis on unprotected implementation dataset LD_{UP} using existing model ($A1$)
- Analysis on unprotected implementation dataset LD_{UP} using proposed model using varying PCA Components sizes ($A2$)
- Analysis on protected implementation dataset LD_P using existing model ($A3$)

- Analysis on protected implementation dataset LD_P using proposed model using varying PCA Components sizes ($A4$)

For analysis set $A1$ and $A3$, the collected raw traces/instances from FPGA implementations are processed through machine learning classifier CNN for both unprotected and protected implementations, respectively, as proposed in the existing literature. We have chosen the simplest existing CNN model for SCA. For analysis set $A2$, and $A4$, analysis is performed using our proposed model (explained in 8.5.4) for both unprotected and protected implementations. For these sets, data has been over-sampled using SMOTE, and then PCA is applied to change the dimensions of the data. For analysis in this study, we have tested the various number of principal components from PCA_{CG} group.

The accuracy and model training time is reported along with Receiver Operating Characteristic (ROC) curves. The outcome of the analysis will help in devising a time and resource-efficient mechanism for attacking FPGA implementations on PKC.

8.6 Experimental Setup

For implementations of the proposed deep learning model, python platform is used along with Keras and scikit-learn libraries [166, 167]. The computation requirement of hyper parameter tuning for deep learning processing is high, so NCI (National Computational Infrastructure) Australia high-performance super-computing server has been used [168]. However, for comparative analysis stand alone system equipment with GPU GEFORCE GTX 1080 Ti, memory 32GB and CPU Intel Core i7 (@3.4GHz) processor is used.

8.7 Results and Discussions

Based on the proposed framework, results and analysis is presented in this section for both LD_{UP} and LD_P datasets. Results are presented for all four analysis sets.

8.7.1 Results on Unprotected Implementations ($A1$ and $A2$)

The results for unprotected implementations, using both existing and proposed models, are presented here. For analysis on full length raw traces using existing models ($A1$), accuracy of 100% is achieved for all analysis sets. It takes 3.6 hours on a GeForce GPU system, as shown in Table 8.5. The results are obtained after fine-tuning the model with the hyper-parameters as mentioned in 8.5.4. For some of the optimizers, in the initial few epochs, training and validation accuracy curves are flat because the high number of features slows down the training process. Best accuracy is achieved with Adamax, Selu, and 0.001, as an optimizer, activation, and learning rate, respectively.

TABLE 8.5: Timing for Unprotected LD_{UP} using existing and proposed models

	Analysis Set	Time (sec)	Accuracy
Existing Model	$A1$	13248.42	100
Proposed Model	$A2$	425.46	100

For analysis set $A2$, and $A4$, firstly SMOTE is applied to balance the data instance, then the dimensionality of the raw data is reduced by pre-processing with PCA. Out of $S_T = 33750$ samples or features, only 800 features are selected based on the presented visual representation in 8.5.3. It has been observed that the same resulting high accuracy is achieved in just 425.46 seconds, with Adamax, Relu, and 0.001, as an optimizer, activation function, and learning rate, respectively.

8.7.2 Results on Protected Implementations ($A3$ and $A4$)

For analysis on $A2$, the raw data trace leakages from the protected implementations are analyzed using the existing complex model. It has been observed that using existing model, 62.1 % accuracy is obtained in 3.46 hours. For analysis on the set $A4$, the raw data trace leakages from the protected implementations are resampled using SMOTE, transformed using PCA, and then processed with the proposed CNN network. With PCA processing, features are reduced from $S_T = 39250$ to 800 only. The accuracy of 67.91% is achieved in only 321.61 seconds, as shown in Table 8.6. To further tune the model performance, hyper-parameters are optimized. Both the best performance results are obtained using Adamax and Selu as optimizer and activation function, respectively. For most of the optimizers, including Adagrad and Adadelta, delayed learning is observed for analysis with existing models, which happens due to the large number of features per-instance, whereas the total number of instances is small.

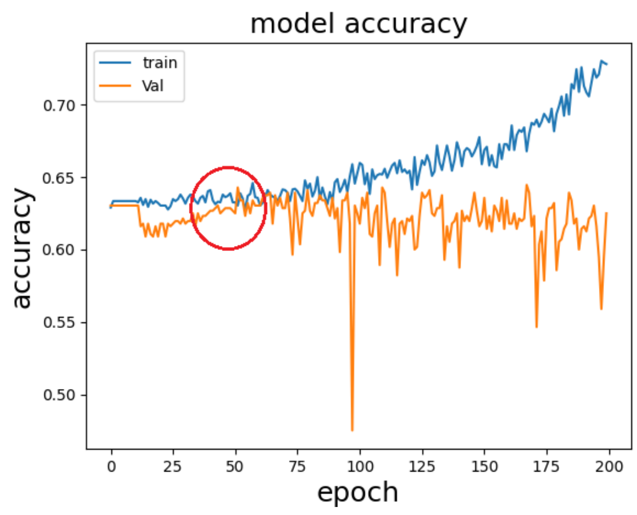
TABLE 8.6: Timing for Protected LD_P using existing and proposed models

	Analysis Set	Time (sec)	Accuracy
Existing Model	$A4$	12473.28	62.1
Proposed Model	$A5$	321.61	67.91

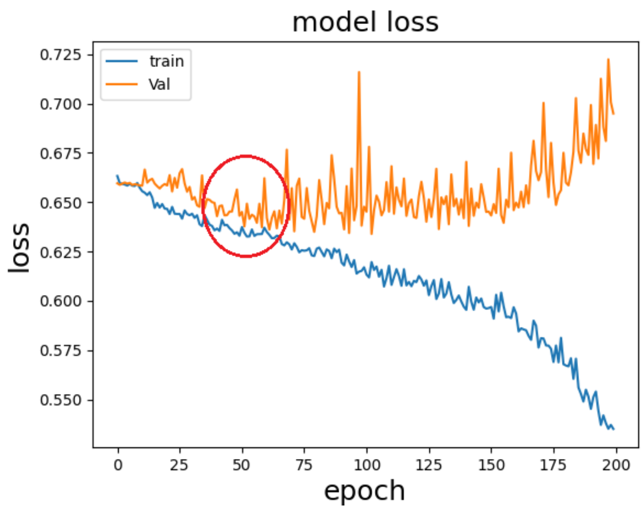
Training and validation, accuracy and loss, for training with the existing model on protected design is shown in Fig. 8.4. It can be seen that the training loss is decreasing, but the validation loss starts increasing after 50 epochs, which shows that the model performs poorly and might cause over-fitting. The over-fitting phenomenon can be confirmed from the accuracy plot as around epoch 50; the validation accuracy slightly goes higher than the training accuracy.

Training and validation accuracy and loss, for training with 200 epochs with the proposed model with SMOTE is shown in Fig. 8.5. It can be seen that the training loss is decreasing throughout the learning process. However, the validation loss decreases in the initial 25 epochs only, and after that no variation is seen, which means that the model is not learning any further and might cause over-fitting. So the best possible results achieved, with the protected design implementations under consideration, are 67.91%. It is also observed that the overall training and validation loss is smaller in the proposed model than the existing model training on the ECC Datasets.

To compare the impact of integrating SMOTE and PCA with our presented attack model, we have also performed experiments without having SMOTE or PCA in the pipeline. The results are depicted in Fig. 8.6. It has been seen that the validation accuracy is fluctuating drastically, and the loss in certain cases goes beyond the training loss which shows a poor model performance. We have also seen that applying only PCA before the classifier does



(a)

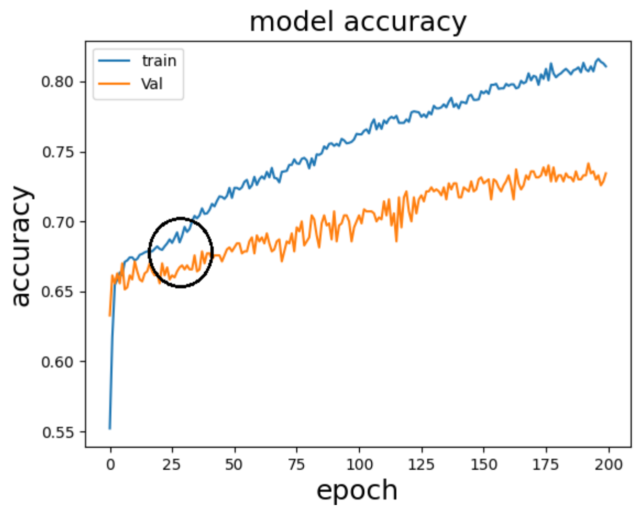


(b)

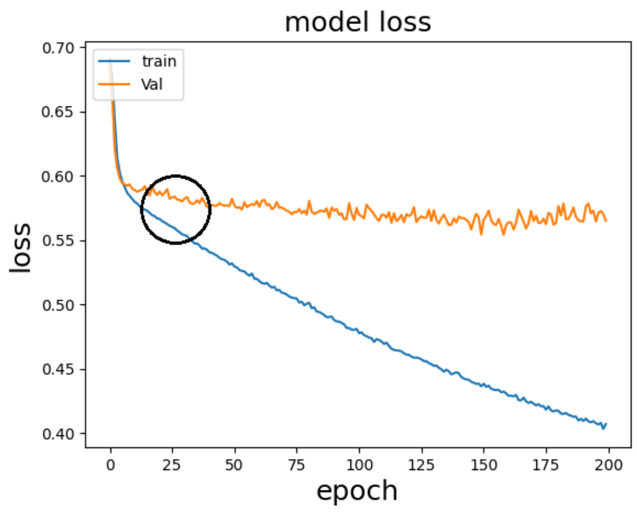
FIGURE 8.4: Training and Validation (a) Accuracy and (b) Loss for the existing model on LD_P

not return impressive results as well. This shows that the combination of all, PCA and over-sampling techniques for imbalance data and our proposed CNN model, provides better performance results than the existing complex models.

To further analyze the improvement produced by the proposed approach, Receiver operating characteristic (ROC) curves, obtained on test data evaluation, are plotted as shown in Fig. 8.7. ROC curves are the graphical plots, illustrating the classifier's diagnostic ability by displaying the True Positive Rate (TPR) and False Positive Rate (FPR). It can be clearly seen that the ratio of TPR to FPR, and the area under curve obtained using analysis performed on the proposed model is better as compared to the ratio of TPR to FPR, and the area under



(a)

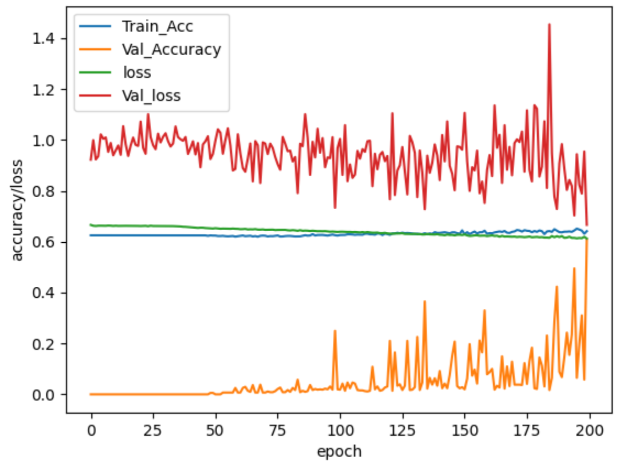


(b)

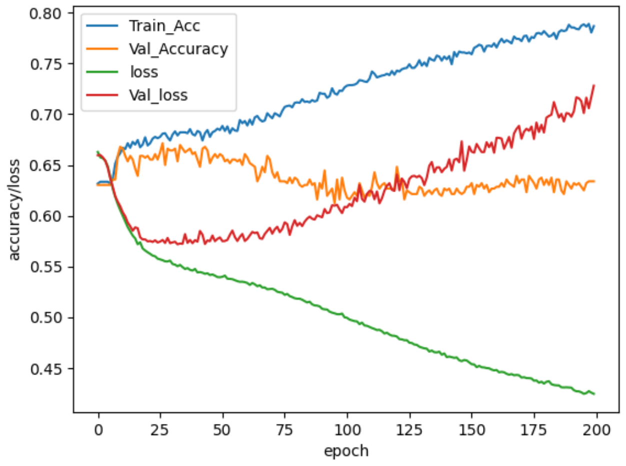
FIGURE 8.5: Training and Validation (a) Accuracy and (b) Loss for proposed model on LD_P using SMOTE

curve of the analysis of the existing model analysis. It has also been observed that the best performance has been achieved with 800 PCA components among all the test groups of PCA components, as explained in 8.5.3.

Based on the above results, it can be seen that the time efficiency of the attack has significantly improved using the proposed model. It is also observed that for both protected and unprotected implementations, the accuracy either stays the same or improves, in less training time as compared to the existing complex neural networks.



(a) a



(b) b

FIGURE 8.6: Accuracy Vs Loss plot for (a) SMOTE only and (b) PCA only

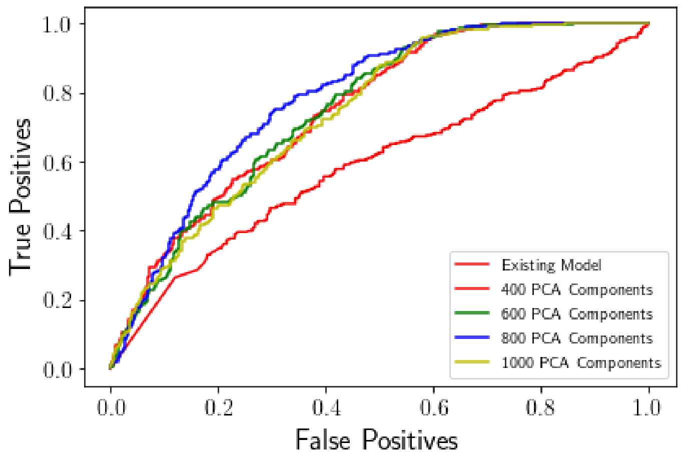


FIGURE 8.7: ROC of Protected Implementations for (a) Existing Model and (b) Proposed Model

8.8 Conclusion

This research work has proposed a hybrid deep learning-based side channel model based on CNN, PCA, and SMOTE, having an optimal number of convolutional layers. Our proposed model is computationally less complex than the existing deep learning-based models and performs better or the same in time-efficient manner. As a test case study, we have selected a variety of the ECC Montgomery Power Ladder Scalar Multiplication algorithm as minimal side-channel analysis exists on ECC from a machine learning perspective. We have used four analysis sets for our evaluation methodology, two for each protected and unprotected ECC implementations. Our experimental results have observed that accuracy improves by 6% using our proposed approach for protected implementations (which is 67%) and stays the same for the unprotected implementation that is 100%. We have also observed the effect of using SMOTE on the proposed model. It is also observed that the overall training and validation loss is less for the proposed model than the existing model training on the ECC datasets. Overall, it can be concluded that the proposed ConvNet enables the network to train much faster with better performance by consuming fewer hardware resources as compared to the existing state-of-the-art methods.

Chapter 9

Data Augmentation using Generative Adversarial Networks - Synthesizing Realistic Leakage Signals

This chapter is an adapted version of a prepared article (Publication VII). In previous chapter, the problem of small dataset, having less number of instances with high samples per instance, is addressed by reducing the number of features. In this chapter, another avenue is explored, i.e. the number of instances in the dataset are increased by generating fake leakage signals using Conditional Generative Adversarial Network (CGAN) and Siamese networks. This approach is particularly useful in scenarios where the number of instances are very low or where number of sample for one class are higher than other (case of imbalance dataset). The proposed model is evaluated on the symmetric and asymmetric ciphers' FPGA implementation leakages.

9.1 Abstract

Deep learning-based side-channel analysis performance heavily depends on the size of the datasets and the number of instances in each target class. Both small and imbalance datasets might lead to unsuccessful side-channel attacks. Such dataset issues might arise due to the adversary's constraint to collect a specific limited amount of the leakage traces in the given time. The attack performance can be improved by generating traces synthetically or artificially from the inadequate data instances, instead of collecting from the target device under attack. Generating the artificial traces, having characteristics of the actual leakage traces, using random noise is challenging. This research proposes a data augmentation architecture, based on conditional generative adversarial networks (CGAN) and Siamese networks, enhancing the attack capability. The paper presents a quantitative comparative machine learning-based side-channel analysis between a real raw signal leakage dataset and an artificially augmented leakage dataset. The analysis is performed on the leakage datasets of symmetric and asymmetric cipher implementations. We also investigate non-convergent networks' effect on the generation of fake leakage signals using two CGAN based deep learning models. The analysis shows that the proposed data augmentation model results in a well-converged network that generates realistic leakage traces according to the dataset under evaluation.

9.2 Introduction

Recently, deep learning-based attacks have been extensively studied for improving the profiling attacks. Profiling attacks are the class of side-channel attacks in which the adversary is assumed to have access to the target device's open copy. The deep learning model's performance can lower if adequate data is not provided during training, especially in scenarios where the adversary has additional constraint of collecting limited leakage measurements. Picek et al. have presented a profiling side-channel framework with restricted access to generate the measurements in profiling attack [203]. Moreover, scenarios where class imbalance exists, the training model can lead to biased results. Different approaches can be used to cater to the requirement of the data generation. Picek et al. have presented the results of using traditional machine learning-based data augmentation techniques and concluded that Synthetic Minority Over-sampling Technique (SMOTE) can aid in data generation for the symmetric ciphers, which results in an attack model with better performance [28]. Generative Adversarial Networks (GAN) is another popular data augmentation technique that is widely used in the image processing domain for generating fake images, which significantly improves the machine learning model's performance [204]. Only one existing study presents the realization of using GAN-generated fake signals as leakage signals for machine-learning-based side-channel analysis on symmetric cipher only [205]. However, authors have used the recommended practices and have mainly focused on the application of CGANs for the successful side-channel attack. This study presents a new CGAN-based architecture for data generation and provides insights for selecting the best suitable GAN-based model for data generation for both symmetric and asymmetric algorithm implementations.

GAN's performance for generating fake images/signals depends on the generator's progressive learning based on the discriminator's response. The design and selection of a GAN play an important role in generating realistic leakage signals. A well-convergent GAN network will generate traces carrying relevant/significant features similar to the original data samples. Designing GAN-based model with optimum convergence or equilibrium point is one of the greatest challenges for generating fake signals that contain the characteristics of real leakage traces. Several techniques, presented for fake image generation, can help achieve convergence, including feature matching, conditional GAN (CGAN), semi-supervised learning [206]. We propose a convergent GAN-based data augmentation model for generating fake leakage signals by utilizing information from the existing datasets. Our presented approach is inspired by the fake image generation presented in [207] and combines the characteristics of the Siamese network and Conditional GAN (Siamese-CGAN). We provide a comparative performance analysis of presented approach with the existing model. Moreover, an actual attack is launched to show the attack's practicality based on generated fake signals. Selecting a suitable GAN model might give better performance as compared to the real noisy leakages. Our contributions are explicitly listed below:

- We have presented a layered approach for generating the leakage 1-dimensional fake signals for machine learning-based side channel analysis (ML-SCA). Our presented approach combines Siamese network and conditional GAN characteristics with an extra model loss monitoring layer introduced to detect the model convergence. The performance of the proposed data augmentation technique is analyzed visually as well.
- We have provided a quantitative comparative analysis exhibiting the fake leakage trace datasets' effect on the side channel training model performance. These fake leakage

traces are generated from various converging points during proposed Siamese-CGAN model training, which helps analyze the importance of well-converged models.

- The proposed Siamese-CGAN model is trained on datasets from symmetric and asymmetric algorithm implementations both, using two different neural networks for generator and discriminator. Best performing neural networks are further selected for analysis.
- Performance of the Siamese-CGAN data augmentation model is evaluated by applying the actual machine learning-based side channel attack on the generated leakage traces using two neural network models that is MultiLayer Perceptron (MLP) and Convolutional Neural Network (CNN).

The rest of the paper is organized as follows. Sec. 9.3 gives an overview of the related literature, including profiled and machine learning-based side channel attacks, CGAN background, machine learning and cryptographic algorithms used for analysis, Sec. 9.4 explains the layered GAN approach for leakage traces generation for ML-SCA, Sec. 9.5 discusses the experiments conducted along with the results, and Sec. 9.6 concludes the paper.

9.3 Preliminaries

9.3.1 Profiled Attacks and Machine Learning Side Channel Attacks (ML-SCA)

Profiled attack is the most powerful side channel attack in which the adversary is considered to have access to the device's open copy under attack. There are two phases of the attack; profiling phase and attack phase. In the profiling phase, the adversary creates a profile of the device with all the possibilities for the data leakages and then can use them in the attack phase to distinguish/predict the unknown key bit [192]. Two main subsets are Template attacks [44] and stochastic model [69]. Machine learning- based side channel attacks (ML-SCA) are similar to profiling attacks. They also have two phases; training phases (profiling phase) and testing phase (attack phase). The adversary can train the model with the leakage examples collected from the device under the target and then evaluate the trained model using unlabeled or labeled key examples.

9.3.2 Generative Adversarial Networks (GAN)

GAN was first introduced by Goodfellow et al. in 2014 [204]. Since then, many variations of GAN have been proposed, including Conditional GAN (CGAN), Deep Convolutional GAN (DCGAN), Information Maximizing (InfoGAN), Stacked GAN (StackGAN) [208] [209] [210] [211]. There are various other deep learning-based GAN architectures, which are mainly based on DCGAN.

Generative Adversarial Network (GAN) is an architecture of the generative model for generating plausible data. It consists of two neural networks, discriminator D and generator G. Generator G network generates the fake data, with random noise input z and discriminator D network discriminates between real and fake data. Real data, is labeled as '1' and artificially generated fake data is labeled as '0'. The discriminator's task is to distinguish the real and generated data instances, whereas the generator's task is to improve the model based on the feedback from discriminator. GANs are based on the concept of a zero-sum non-cooperative game where one network (discriminator) is trying to minimize the loss, and the other network (generator) is trying to maximize the loss, meaning the discriminator is trying to distinguish and classify the data instances are real or fake, however, the generator is trying to generate data traces which are alike. This makes it hard to find a good convergence point. GAN converges when both D and G reach a Nash equilibrium, meaning that one (D/G) will not change its actions anymore, no matter what opponent (D/G) does. This is the optimal point adversarial loss in GANs aims to optimize the min-max problem given by equation 9.1. There are different techniques used to improve the performance of GANs by improving the convergence. One of such techniques is Conditional GAN (CGAN).

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))] \tag{9.1}$$

9.3.3 Machine Learning Algorithms for Analysis

Based on the deep learning-based side channel attacks (DL-SCA) performance on various cryptographic algorithms [23, 125, 126], we have tested our newly generated datasets using the state-of-art machine learning algorithms that is, MultiLayer-Perceptron (MLP) and Convolutional Neural Network (CNN).

MultiLayer-Perceptron (MLP)

MultiLayer-Perceptron (MLP) is a class of feed-forward neural network-based algorithm consisting of one input layer, one or more hidden layers, and one output layer. Each layer consists of nodes/neurons, except the input layer. These nodes utilized a nonlinear activation function to learn the patterns in the data. MLP uses supervised learning-based backpropagation to change the weights on the connections during data processing. A batch of data is presented to these fully connected networks during each epoch for learning.

Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN or ConvNet) is a class of deep learning neural networks and is principally based on the convolutions. A CNN architecture consists of an input and an output layer and few hidden layers. Hidden layers are usually convolutional layers having an activation function, followed by pooling layers that help reduce the dimension of data. CNN has the capability of learning the patterns from noisy side channel leakages without any preprocessing [23].

9.3.4 Cryptographic Algorithms Under Analysis

For our analysis, we have selected unprotected FPGA implementations of the Advanced Encryption Standard (AES) and Elliptic Curve Cryptographic (ECC) primitive. The leakage traces have been evaluated in [176] and [126] for hamming weight-based attacks, in which each leakage trace or instance is marked as 0 or 1 based on the key bit value, which confines our analysis to binary classification. It should be noted that the purpose of this research is to analyze the effect of artificially generated features in data traces for environments where the adversary has an additional constraint on collecting leakage traces to form a dataset. The presented methodology can be extended to produce and test the fake leakage traces for any other cryptographic algorithms, including the protected implementations.

9.3.5 Siamese Neural Network

Siamese neural network (also called twin/identical neural network) is an artificial neural network architecture which consists of two similar neural networks (having same weights and parameters) and is capable of processing two different input vectors to produce comparable output vectors [212]. The two neural networks are feedforward perceptrons which work in tandem and trained in back-propagation manner. The idea behind Siamese neural network is not to learn to classify the classes but to learn to discriminate between the input vectors. Hence a special loss function, contrastive loss or Triplet Loss is used for the training the network. For training the network, the pairs (x_i, x_j) of input vectors are prepared; few pairs consisting of similar vectors and few pairs consist of dissimilar vectors. The similar vector pair is labeled as y='1', whereas the dissimilar pair is labeled as y='0'. Each pair is fed to the Siamese network and distance is computed to check the similarity. The output vectors from each network are compared using cosine or Euclidean distance and can be considered as a semantic similarity between projected representation of the input vectors [213]. One of the best feature of Siamese network is that it can be used in transfer learning. Siamese networks are at the core of few-shot learning using various learning methods including matching networks [214].

9.4 Proposed Approach

This section explains the proposed layered GAN model used for selecting the optimal model for generating leakage signals from the random noise.

9.4.1 Data Splitting

The leakage data traces L are collected from the FPGA core while AES and ECC algorithms are performing the encryption E using the secret key K. The labeled collected traces are then divided into two sets; Training and Testing sets. Training set is used for training Siamese-CGAN, and testing set is used for evaluating the performance of the trained model. For fair evaluation of the trained Siamese-CGAN model, testing set is never shown to the network during training process.

9.4.2 Siamese-CGAN Model for Data Augmentation

In contrast to the standard GANs, conditional GANs (CGANs) perform conditional generation of the fake data, based on the class label rather than generating signals blindly. The labels c of the data traces/instances are used to train GANs in zero-sum or adversarial manner to improve the learning of the generator (G). As mentioned before, generator's G task is to generate the leakage signals that carry similar properties as of the original traces using the random noise z and the latent space input ls. The discriminator D task is to distinguish real and fake signals. In the CGAN training process, first the discriminator D is trained with the labeled real data traces T_{real}, and then the discriminator D is trained with the fake generated signals $G(z)$ or T_Gen. The objective function for CGAN is given by equation 9.2.

$$L_{GAN} = E_{x \sim T_{Real}}[log(D(x|c))] + E_z[log(1 - D(G(z|c)))] \tag{9.2}$$

In our proposed design of Siamese-Conditional Generative Adversarial Network (Siamese-CGAN), we are combining CGAN with the Siamese network concept. Siamese network is an architecture in which two identical/twin networks, carrying the same weights, are trained with two different inputs. In the proposed model, two generators $G1$ and $G2$ take two random input noise vectors $z1$ and $z2$, and generate fake signals $G(z1)$ and $G(z2)$, respectively, in Siamese fashion. Both the generators share the same network weights, and only the input is different. The discriminator D is first trained with the labeled real data T_{real} and then with the fake data T_{gen}, originating from the two twin generator networks $G1$ and $G2$.

As mentioned before, convergence is a challenging issue in training GANs. In some cases, the model converges and then starts diverging again; that is, it forgets its learned examples. Several techniques include memory-based learning, to handle such scenarios [215]. Training the model simultaneously with the random noise from two sources, can help obtain a better-converged model. Moreover, to analyze the impact of convergence, we have introduced another layer in the two-step CGAN model to analyze the model performance for leakage traces. This layer monitors the real traces loss L_{real}, generated traces loss L_{gen}, and GAN model Loss L_{GAN}. Let $D_{GAN \to R}$ represent the loss difference between L_{GAN} and L_{real}, and $DGAN \to G$ represent the loss difference between L_{GAN} and L_{real} then the average of loss differences over last t iterations will be given by

$$Loss_{Avg} = \frac{1}{t} \sum_{i=1}^{t} (|D_{GAN \to R}| + |D_{GAN \to G}|) \tag{9.3}$$

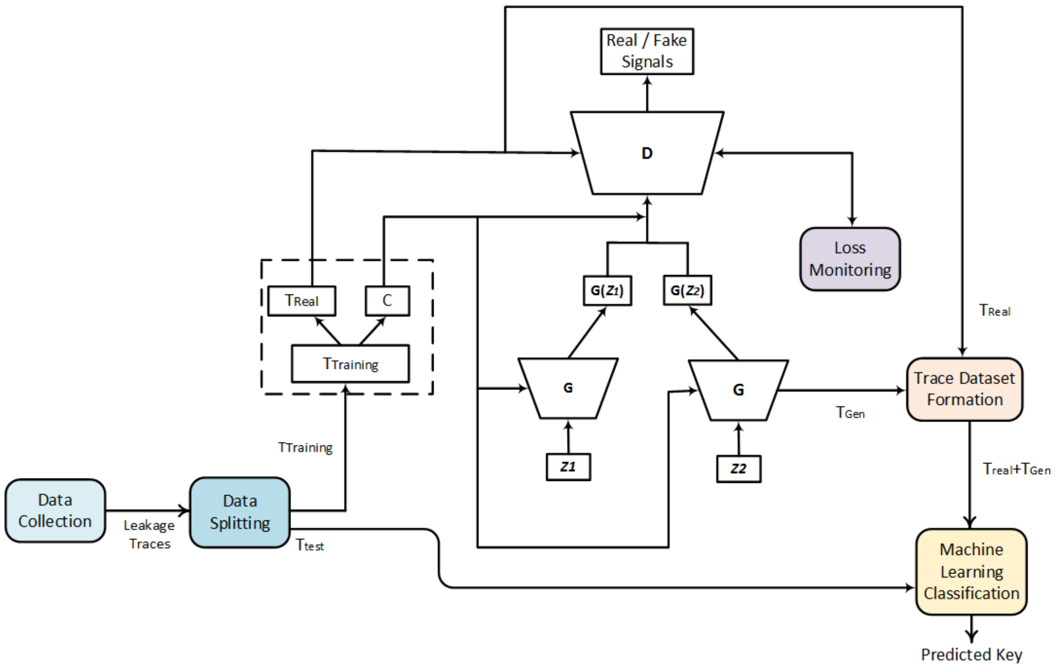


FIGURE 9.1: Proposed Siamese-CGAN architecture for ML-SCA

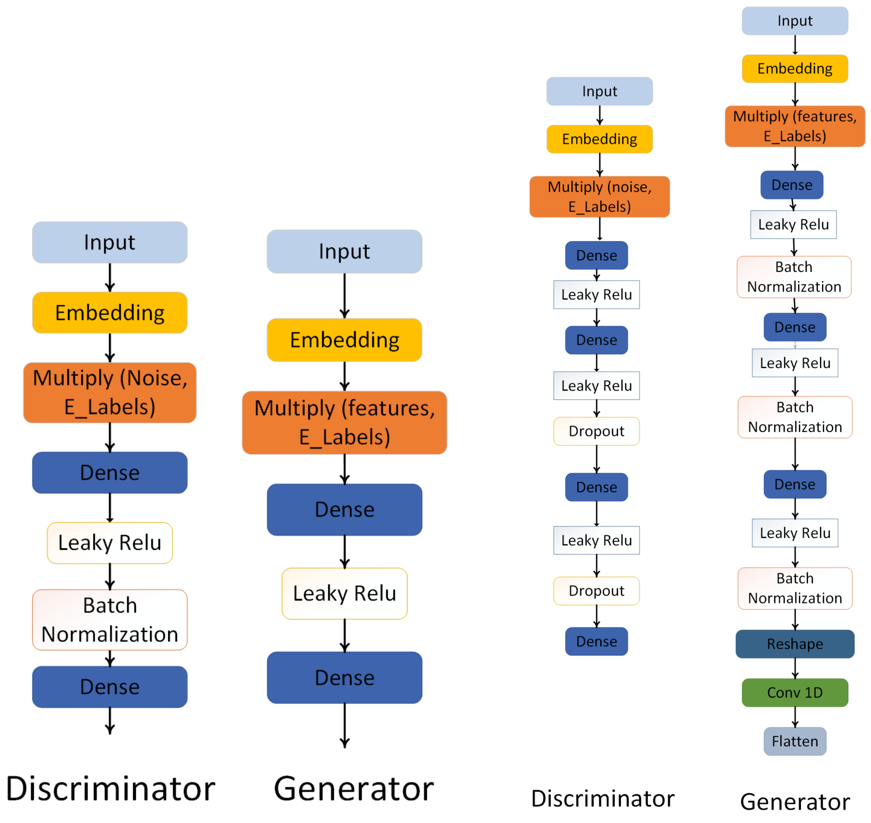
The model stops training when the average model loss $Loss_{Avg}$ over the last t iteration is less than the average loss over the last $t * 2$ iterations. The trained Siamese-CGAN model is then used to generate the n_g fake traces T_{gen}, containing features similar to the original signals. T_{gen} and T_{real}, having n_g and n_r instances respectively, are combined together to form a resultant dataset. This dataset is then used to train the machine learning model to analyze the generated dataset's performance on ML-SCA. A test set is set aside for a fair evaluation before adding the generated traces into the training dataset. The test set is never shown to the neural network during training. Proposed Siamese-CGAN specific for ML-SCA is shown in the Fig. 9.1.

9.4.3 CGAN Models for Discriminator and Generator

For evaluating the trained Siamese-CGAN model performance, two neural networks (MLP and CNN) are used for the generator and discriminator. These networks are addressed as $ModelA$ and $ModelB$. $ModelA$ (MLP-based) is based on two fully connected layers for the generator and the discriminator. However, $ModelB$ (CNN-based) has a more complex architecture with four connected and one CNN layer. Fig. 9.2 shows the structure of both the models. Batch normalization, LeakyRelu, and dropout layers are introduced, which help achieve a better stable performing model and help avoid over-fitting. For $ModelA$ discriminator, there is one dense layer with 512 neurons and last dense layer with Sigmoid function. For $ModelA$ generator, there are two dense layers, consisting of 256 and 700 neurons. For $ModelB$ discriminator, there are three dense layer with 512 neurons and a last dense layer with Sigmoid function. For $ModelB$ generator, there are three dense layers, consisting of 256, 512, and 1024 neurons. The last convolutional layer uses tanh activation function.

9.5 Experiments and Results

In our experiments, we have reduced the size of the AES and ECC leakage datasets intentionally to analyze the effect of the artificially generated data traces on small datasets. Experiments are performed on the leakage traces collected from AES and ECC FPGA implementations, as discussed in Sec. 9.3. It should be noted that for all the experiments, in the artificially generated $n_g = 150$ traces, there are 75 traces for each class (class bit 0 and class bit 1). For impartial analysis, we have divided our datasets into two parts; training



(a) Model A (b) Model B

FIGURE 9.2: *Siamese – CGANModels*

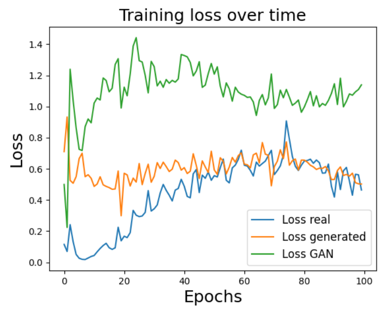
and testing, and then used training datasets for further training the proposed Siamese-CGAN networks to generate the fake leakage signals. We have performed experiments five times and have reported the best results obtained for each model. We have divided our experimental analysis into two sections. Firstly, in Sec. 9.5.1, we have compared the performance of our proposed CGAN-based model with the existing CGAN models and analyzed them visually. Secondly, in Sec. 9.5.2, we have provided the machine learning-based side channel analysis of the real dataset and the dataset consisting of real and fake leakages by training with two neural networks (MLP and CNN). For this analysis, we have also shown the comparison of generating leakage signals from the non-converging network and a converging network. To achieve this, we have generating fake signals from various points while training the Siamese-CGAN network. Precisely, we had generated the signals when the model converged the best (after 400 epochs) and generated the signals when the model was least convergent (initial epochs).

9.5.1 Analysis of Existing and Proposed GAN based approaches

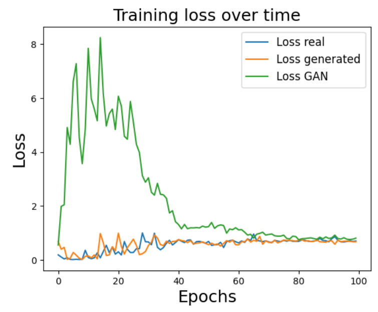
Firstly, we have performed a comparative analysis of our approach with the existing CGAN model. The existing CGAN network (without layered Siamese-CGAN setting) is addressed as $CGANModelA/B$, whereas our proposed models are addressed as $Siamese - CGANModelA/B$. Model loss for the best performing models, for both AES and ECC, is shown in Fig. 9.3 and 9.4, respectively. Fig. 9.3 a, and b presents the real, fake, and GAN loss for training with $CGANModelA$ and $CGANModelB$ without Siamese setting, respectively. However, Fig. 9.3 c, and d presents the real, fake, and GAN loss for training with $Siamese - CGANModelA$ and $Siamese - CGANModelB$ with the Siamese setting. Similarly, Fig. 9.4 presents the model training loss for ECC. Siamese-CGAN and CGAN models are trained for 1000 epochs, and history is stored after every 100 epochs, so the x-axis is scaled down by 100. It can be seen that the proposed $Siamese - CGANModelB$ architecture provides the best convergence for both AES and ECC. For AES, all model converges perfectly except the $CGANModelA$. It can also be seen that the proposed $Siamese - CGANModelB$ (Fig. 9.3 d) performs better and converges early around 450 epochs as compared to the existing $CGANModelB$ (Fig. 9.3 b), which converges around 800 epochs, on same AES dataset. For ECC, the losses decrease initially, and then the model starts diverging. However, the best convergence is obtained with the proposed $Siamese - CGANModelB$ in the initial 200 epochs (Fig. 9.4 d).

9.5.2 Analysis of Proposed Siamese-CGAN for ML-SCA

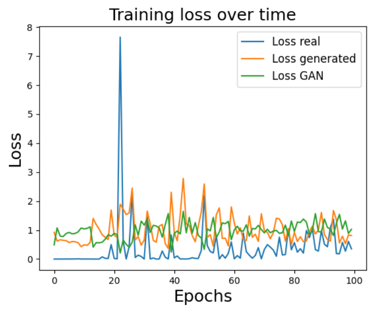
To compare the performance of the artificially generated traces datasets with the performance of the dataset consisting of real captured traces only, we have formed three new datasets and divided our experimental analysis into the three categories, which are discussed below. This analysis is an actual machine learning-based side channel attack using two neural networks, MLP and CNN. Table 9.1 and 9.2, show the performance accuracy results of both the GAN models using neural network algorithms MLP and CNN.



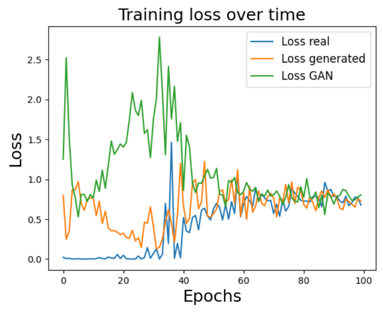
(a) a



(b) b



(c) a



(d) b

FIGURE 9.3: CGAN Model Training Loss for AES Leakages (a) $CGANModelA$ (b) $CGANModelB$ (c) $Siamese - CGANModelA$ (d) $Siamese - CGANModelB$

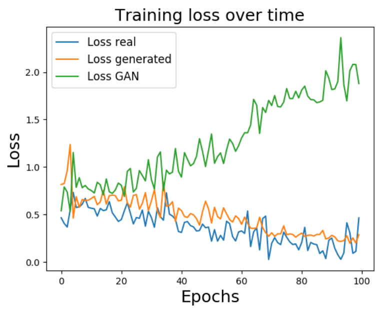
Analysis on Real traces Dataset

In this category of analysis, we have performed experiments on the real data traces only. We have selected $n_r = 300$ leakage traces that are collected from the device while the encryption is performed with the algorithm under target. No artificial/fake traces are included in the training dataset, so $n_g = 0$.

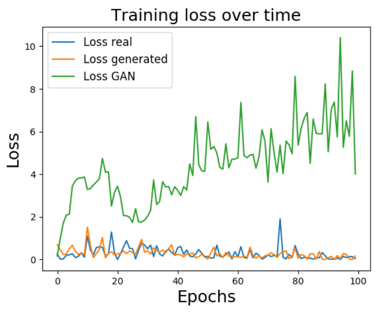
For AES, it has been observed that secret key can be recovered with 92% and 91% accuracy using MLP and CNN, respectively. For ECC, analysis of raw real data traces shows that the secret key can be recovered with 70% accuracy using MLP and CNN. The low accuracy of CNN is because we have intentionally reduced the number of traces to 300 only. It should be noted that preprocessing and alignment have not been applied on these datasets.

Analysis on Real and Generated traces dataset with Maximum Convergence

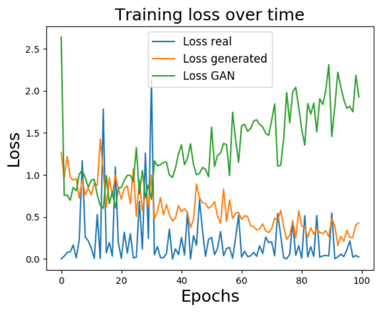
In this set of analysis, the training datasets consists of equal proportion of the real traces and the artificially generated traces $n_r = 150$ and $n_g = 150$. However, traces are collected for the epochs during which GAN model achieves maximum convergence (Max-Conv). In the



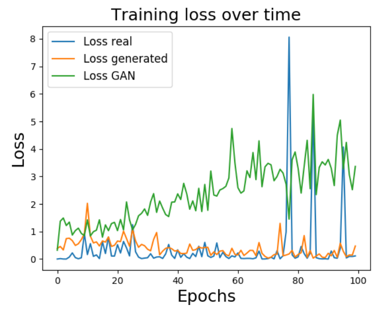
(a) a



(b) b



(c) c



(d) d

FIGURE 9.4: Siamese-CGAN Model Training Loss for ECC Leakages (a) $CGANModelA$ (b) $CGANModelB$ (a) $Siamese - CGANModelA$ (b) $Siamese - CGANModelB$

case of AES, it has been seen (Fig. 9.3) that the maximum convergence is achieved around 700 epochs for both $Siamese - CGANModelA$ and $Siamese - CGANModelB$. For our experiments, data traces are generated around 450-500 epochs for this analysis. It is observed that with the generated traces, MLP and CNN model gave the best performance, and secret key can be classified with 100% accuracy, which is better as compared to the accuracy achieved with the real traces only. It should be noted that the total number of traces in each set is still 300. However, the artificially generated traces contains significant information, based on the real traces, which improved the ML-SCA performance.

It can be seen from Fig. 9.4 that the model for ECC leakages does not converge well. Convergence seems to happen around epoch 200, but then divergence is observed. The model trained with the proposed approach shows a better convergence curve. The traces with maximum convergence analysis are generated around 200 epochs as the model did not converge well, hence the generated traces did not perform well. The secret key can be recovered with an accuracy range 51 - 70% with MLP and CNN using both models, which is nearly the same accuracy as is achieved on the real traces.

TABLE 9.1: Performance results for AES Leakages using MLP and CNN.

Dataset	GAN Model	Acc MLP(%)	Acc CNN(%)
Real 150, Generated 150 (Max-Conv)	Siamese Model A	100	100
Real 150, Generated 150 (Min-Conv)	Siamese Model A	53	97
Real 150, Generated 150 (Max-Conv)	Siamese Model B	53	100
Real 150, Generated 150 (Min-Conv)	Siamese Model B	53	99
Real 300	None	92	91

TABLE 9.2: Performance results for ECC Leakages using MLP and CNN.

Dataset	GAN Model	Acc MLP(%)	Acc CNN(%)
Real 150, Generated 150 (Max-Conv)	Siamese Model A	66	70
Real 150, Generated 150 (Min-Conv)	Siamese Model A	29	59
Real 150, Generated 150 (Max-Conv)	Siamese Model B	51	70
Real 150, Generated 150 (Min-Conv)	Siamese Model B	25	66
Real 300	None	70	70

Analysis on Real and Generated traces dataset with Minimum Convergence

In this analysis, we have combined the real traces with the artificially generated traces in equal proportion, that is $n_r = 150$ and $n_g = 150$. However, traces are collected for the epochs during which the GAN model showed minimum convergence. For AES, minimum convergence is observed around 200 epochs. Artificial 150 traces are generated and stored around this point and are further combined with the real 150 traces to train the ML model. It has been observed that even with minimum convergence, the ML model is able to recover the secret key with 100% accuracy with both GAN models. For ECC, the performance accuracy is very low (25-30%) with artificial traces, which is expected because traces are generated when the model did not reach full convergence. The performance is lower as compared to the performance on the real trace dataset.

9.6 Conclusion

Leakage trace dataset with an insufficient number of traces can be a hurdle in accurate attack modeling using machine learning-based side channel analysis. For such scenarios, data augmentation using Generative Adversarial Network (GAN) can be useful. We have proposed a layered architecture (Siamese-CGAN) based on CGAN and Siamese network that presents a well-convergent model to generate the artificial traces that possess similar characteristics like the real traces.

We have performed two sets of analyses. In the first set of analysis, we have performed experiments and presented visual comparative analysis between the performance of the proposed model and the existing CGAN based models for leakage signal generation. For this analysis, two neural network-based models (MLP and CNN) have been used for modeling the generator and the discriminator networks. The best model is selected based on the comparative analysis. In the second set of analysis, we have evaluated the generated fake dataset by applying the actual machine learning-based side channel attack on the leakage datasets from AES and ECC algorithm implementations. For this evaluation, two state-of-the-art neural networks, MLP and CNN, are used. We have also provided the comparative analysis between the performance of the dataset consisting of data generated from the well-convergent network and data generated from the non-convergent network.

The proposed Siamese-CGAN model performed better than the existing simple CGAN models for both symmetric and asymmetric datasets. The quantitative analysis results show that the well-converged Siamese-CGAN network produces fake leakage traces which are similar to the real collected traces. Hence, they produce a better machine learning-based model for side-channel attacks. We have also observed that the CNN trained models performed better than MLP for secret key recovery. We conclude that leakage traces/instances, with significant contributing features, can be generated artificially in less time if a huge set of traces are not available for ML-SCA. However, selecting a fully converging model might vary for each cryptographic algorithm.

Chapter 10

Leakage Assessment Evaluation Model based on Few-Shot Learning

This chapter is an adapted version of a prepared article (Publication VIII). In this chapter, a deep learning based leakage assessment scheme is presented for detecting the leakages in black-box scenarios. In previous chapters, methodologies are provided for improving the machine learning-based side-channel attacks for recovering the secret key from profiled attack (white-box scenario). However, in this chapter, the black-box scenario is considered for detecting the leakages only. The prominent feature of the proposed system is that new leakages can easily be integrated without retraining the model.

10.1 Abstract

Traditional side-channel analysis has been extended to utilize the advanced machine learning-based techniques to recover the key from the implementations of the mathematically secure algorithms by eliminating the need of pre-processing or alignment in profiled based scenario. For non-profiled scenarios, especially for leakage assessment, only one deep learning model has been proposed which is based on simple neural network (Multi-layer Perceptron). The existing methodology assumes that the input dataset should be balanced and requires a large number of training samples. In contrast to the existing methods, we have proposed a leakage assessment methodology which can successfully detect the leakages with just one training example. Moreover, multiple algorithm leakages can be integrated in a single system, providing a standalone leakage assessment system for evaluating a device. Our approach is

based on few-shot learning. We have evaluated our proposed methodology on the side-channel implementation leakages of symmetric (AES) and asymmetric (ECC) algorithms having varying levels of leakages. Our results demonstrate that leakages can be detected with high confidence.

10.2 Introduction

Advancements in machine learning has open a new era of possible threats to the hardware systems. One of the major threat is the exploitation of the side-channel leakage information acquired due to the weak implementations of the cryptographic algorithm. The security risk has increased with the introduction of a machine learning-based side-channel analysis (ML-SCA) which has reduced the effort required by adversary to successfully launch these attacks. The situation has given rise to the need of an advance evaluation system which can certify if the device is providing adequate amount of security. One of the possibility of evaluating a device is to have an exhaustive evaluation against all known attacks, which is practically infeasible. Traditionally, a device is evaluated using leakage assessment methods which are based on point-wise statistical tests including Welch's t-test and Pearson's χ^2-tests [216, 217]. Both the tests can distinguish two groups (fixed-random) based on a confidence value. However, they work best for univariate leakages only but gives impaired results for multivariate or horizontal attacks as they analyze individual points in a leakage trace independently. Existing leakage assessment techniques require internal details to successfully detect the leakages, hence open a plethora of possibilities when evluating on a new algorithm.

Wegener et al. have presented deep learning based leakage assessment (DL-LA) method to address the shortcomings of the existing leakage assessment statistical tests [218]. However, the presented approach has limitations of computational complexity due to the requirement of a large training set. The learning task might become more difficult when the available dataset is very small or is imbalanced. Deep learning networks have certainly outperformed classical methods in different domains including side-channel analysis, due to their ability of learning the data patterns using complex statistics. However, their efficiency is still not to the level of human perception. A human brain can learn from an image if shown one time only. To shorten the gap, concept of few short learning is introduced, in which network can learn even if few images are presented.

In this study, we propose a universal leakage assessment methodology based on deep learning and few-shot learning ([212]) to detect the leakages from the various groups of leakages acquired from multiple channels (symmetric and asymmetric) and multiple algorithms, by training with just a single example for each class. Such a system will be helpful in device certification scenarios. The proposed scheme has been evaluated on the standard implementations of AES and ECC. For both the algorithms, leakages are acquired at different sampling frequency to analyze if leakages can be detected from the samples having less information about the sensitive variables.

Rest of the paper is organized as follow. Sect. 10.3 explains the related literature and necessary background. Sect. 10.4 explains the proposed methodology. Sect. 10.6 presents the results along with discussions. Sect. 10.7 concludes the paper.

10.3 Background and Related Work

This section introduced the necessary background related to state-of-the-art leakage assessment, deep learning based side-channel analysis with profile and non-profile attack scenarios, siamese networks and few-shot learning.

10.3.1 Leakage Assessment

Since the introduction of the side-channel attacks [75], various research dimensions have been explored. One is to investigate the possible attacks to recover the actual underlying sensitive variable. Other is to assess the physical vulnerability or resistance of a device against known side-channel attacks through exhaustive verification. However, the exhaustive verification based assessment procedure became less practical due to the introduction of a multitude of side-channel attacks and countermeasures against these attacks. Another concern about this procedure is that the certified 'secure-device' might be at the risk of security vulnerability and trigger a key recovery in certain scenarios, which happens because a test case vector might have been missed during assessment process [219]. Hence, realizing the need of a generic leakage assessment methods, National Institute of Standards and Technology (NIST) conducted "Non-Invasive Attack Testing Workshop" to stimulate the development of evaluation tools and test methods to measure the effectiveness of migitations against non-invasive attacks on cryptographic chips [220]. Leakage detection or assessment focuses on the detection of the leakages only, without considering the underlying secret parameter details, hence providing a black-box testing scenario. In the leakage detection methods, different inputs are given to the system and leakage behaviour is recorded to further observe the difference between the leakages. Such leakage detection system proves to be a feasible evaluation method for the labs for certification of a device's physical security without the requirement of testing it against all the possible existing attacks. Few of the proposed evaluation methods include Welch's t-test and Pearson's χ^2-test [216, 217]. Both are hypothesis tests which are used to conclude if the groups of side-channel leakages can be distinguished with a confidence. Traditionally, for side-channel leakage evaluations, two groups of measurements are acquired from the device under test (DUT) with fixed and random inputs, which are then subjected to evaluation using the tests.

Welch's t-test

Welch's t-test is an adaptation of student's t-test for unequal sample distribution. It is designed to distinguish the means of two distributions and thus in side-channel context it is used to test that the hypothesis that the two leakage groups have equal means. It can be applied to first-order univariate analyses. Let Q_0 and Q_1 represent two groups of measurements with the fixed and random inputs, with cardinality n_0 and n_1, means μ_0 and μ_1, and standard deviations s_0 and s_1 respectively. The t-statistics, degrees of freedom v, confidence p to accept the hypothesis using student's t probability density function, can be calculated using following formulas.

$$t = \frac{\mu_0 - \mu_1}{\sqrt{\frac{s_0^2}{n_0} + \frac{s_1^2}{n_1}}}, \tag{10.1}$$

$$v = \frac{(\frac{s_0^2}{n_0} + \frac{s_1^2}{n_1})^2}{\frac{(\frac{s_0^2}{n_0})^2}{n_0-1} + \frac{(\frac{s_1^2}{n_1})^2}{n1-1}}, \tag{10.2}$$

$$p = 2\int_{|t|}^{\infty} f(t,v), dt, \tag{10.3}$$

$$f(t,v) = \frac{\gamma(\frac{v+1}{2})}{\sqrt{\pi v}\gamma(v/2)}(1 + \frac{t^2}{v})^{-\frac{v+1}{2}} \tag{10.4}$$

In side-channel context, t-statistics is evaluated by assessing the distinguishability based on the confidence threshold. The null hypothesis is rejected is $|t| < 4.5$. For $|t| > 4.5$ and $v > 1000$, the confidence p for accepting the hypothesis is smaller than 0.00001, which means it can be concluded with 99.999% confidence that the two leakages does not belong to the same group. If the assumption of $v > 1000$ does not hold then false positives might be observed. However, in case of side-channel leakage evaluation, this is rarely seen.

Pearson's χ^2-test

Due to the limitations and shortcomings of the moment-based nature, Welch's t-test can be used for first-order analyses mainly due to moment-based nature. For higher-order analyses of the countermeasure protected masked implementations, Moradi et al. [217] suggested to use Pearson's χ^2-test. This hypothesis test prevents the possibility of false positives by analyzing the full distributions and capturing information from multiple statistical moments. For evaluation with Pearson's χ^2-test, a contingency table F is constructed from the two groups Q_0 and Q_1 having r rows and c columns. The χ^2-statistics x, the degrees of freedom v, and the confidence p to accept the hypothesis using χ^2 probability density function, is given by following formulas.

$$x = \sum_{i=0}^{r-1} \sum_{j=0}^{c-1} \frac{(F_{i,j} - Ei.j)^2}{Ei.j}, \tag{10.5}$$

$$v = (r-1).(c-1), \tag{10.6}$$

$$E_{i,j} = \frac{(\sum_{k=0}^{c-1} F_{i,j}).(\sum_{k=0}^{r-1} F_{k,j})}{N}, \tag{10.7}$$

$$p = \int_{x}^{\infty} f(x,v), dx, \tag{10.8}$$

$$f(x,v) = \begin{cases} \frac{x^{v/2-1} e^{-x/2}}{2^{v/2} \gamma(v/2)} & x > 0 \\ 0 & otherwise \end{cases} \tag{10.9}$$

The procedure can easily be extended to more than two data groups. Generally, a better confidence level achieved with χ^2-test as compared to the t-test, can help in identifying the leakages from the masked implementations with low noise levels.

10.3.2 Profiled and Non-Profiled Attacks

Side-channel attack, proposed by Paul Kocher in 1996 [3], is a class of cryptanalytic attacks in which the side-channel leakages of the mathematical secure algorithms are exploited to recover the key. Variety of side-channel leakages including power consumption by the cryptographic module, electromagnetic radiations emitted by the chip, timing information of specific operation, and vibrations produced by the components due to specific operations [4–6, 8]. In profiled side-channel attacks, and attacker has access to the open copy of the device. She can acquire the leakages from the cryptographic device that is performing encryption E with a secret key k. The leakages are used to estimate a profile leakage model in profiling phase, which is then utilized to predict the key for the unknown leakage in the attack phase. Two subsets of profiled attacks are template attacks and stochastic model [44, 69]. Template attack is theoretically the most powerful side-channel attack and is based on the

Baye's theorem and uses the generative model strategy (like machine learning) to estimate the leakage function $L|(P,K)$,having multivariate Gaussian distribution that is parameterized by its mean ($\mu_{p,k}$) and covariance ($\sum_{p,k}$), for each template (p,k). In the non-profiled attacks, the evaluator has limited access to the closed copy of the device under target. She can only control and observe the input and leakages of the system in a black box scenario. Generally, this black box scenario is ideal from the evaluator's perspective for the certification of the embedded cryptographic device.

10.3.3 Deep Learning based Leakage Assessment

Deep learning belongs to the broader family of machine learning, which allows the system to automatically learn the data features representations using artificial neural networks and concepts of statistics. In recent years, machine learning (especially deep learning) has been explored to improve the side-channel attack success probability with keen focus on recovering the secret key with template attacks in profiled scenario [19–22, 71]. However, there is only one study by Wegenner et. al that proposed the use of deep learning for leakage assessment [218] and demonstrated the performance on implementation leakages of the PRESENT algorithm. Authors have presented a deep learning (Multi Layer Perceptron) based leakage assessment methodology which detects the leakages irrespective of the location,alignment and statistical order of the leakages. In the presented approach group imbalance can lead to false negatives due to the probability in the Binomial distribution.

10.3.4 Few-Shot Learning

Though humans have created artificial neural networks which have marvelous learning capability similar to the the learning ability of human beings but still far from human competence of learning instantly based on previous experience. The huge data requirement for efficient learning hinders the performance and efficiency of artificial neural networks. Human brain can memorize information based on just one example or can predict something whose example is never directly shown, whereas the previous knowledge about the generic domain can help in correctly determining the object. This human perceptual learning phenomena is still a mystery. Research community has been trying to understand and has proposed techniques based on transfer learning. In transfer learning, abstract knowledge from familiar concepts are applied to learn about the new unknown entity. This paves the path to few-shot, one-shot, and zero-shot learning in which exactly 1-5, 1, and 0 training examples, respectively, are required for learning about the underlying data.

Concept of one-shot learning was introduced by Li et al. [221] for a computer vision task in which a model was formed based on the prior knowledge transferred from previously built models. Few-shot learning is a variant of transfer learning and is based on Siamese networks and metric learning. Few examples of the class are presented to the network for training and then network is evaluated on the unknown inputs and unknown classes. Few-shot learning has numerous applications in computer vision, facial recognition, drug discovery [222, 223].

Generally, in the leakage assessment models similarity between the leakage vectors is measured using hypothesis testing, e.g. Welch's t-test and χ^2-test (as explained above). These tests might not be ideal for more complex data having features with high dimensionality. Siamese networks offer the same similarity computations using two parallel similar networks. Siamese neural network (also called twin/identical neural network) is an artificial neural network architecture which consists of two similar neural networks (having same weights and parameters) and is capable of processing two different input vectors to produce comparable output vectors [212]. The two neural networks are feedforward perceptrons which work in tandem and trained in back-propagation manner. The idea behind Siamese neural network is not to learn to classify the classes but to learn to discriminate between the input vectors. Hence a special loss function, contrastive loss or Triplet Loss is used for the training the network. For training the network, the pairs (x_i, x_j) of input vectors are prepared; few pairs consisting of similar vectors and few pairs consist of dissimilar vectors. The similar vector pair is labeled as y='1', whereas the dissimilar pair is labeled as y='0'. Each pair is fed to the Siamese network and distance is computed to check the similarity. The output vectors from each network are compared using cosine or Euclidean distance and can be considered as a semantic similarity between projected representation of the input vectors [213]. One of the best feature of Siamese network is that it can be used in transfer learning. Siamese networks are at the core of few-shot learning using various learning methods including matching networks [214].

10.4 Methodology

In this section, we describe overview of our proposed generic approach based on the Siamese networks and few-shot deep learning techniques and also explain the details with respect to the specific non-profiled and profiled scenarios. It should be noted that the proposed generic attack is applied on the location dependent ECC, AES leakages without any preprocessing and ASCAD AES data. However, it can easily be extended to other algorithms' data leakages.

10.4.1 Core Idea

The core idea of the proposed leakage assessment system is to determine if the attacker can extract information or detect the leakages through the labelled leakage data traces. The label does not necessarily need to be labelled as the actual key value (as is the case of profiled attack). Infact, the label can represent some leakages due to the processing of the distinct fixed inputs. Our proposed methodology is generic and can be adopted to any new leakage groups (more than two) from different algorithm implementations as well.

The basic aim of the proposed system is to distinguish the different groups of side-channel traces. A generic neural network acts as a distinguisher and is trained with the labelled data to classify the unseen data traces from the validation dataset. However, the task of the neural network in the proposed settings is not to learn the patterns from the leakages directly (regular binary or multi-class classification task), rather the model learns from the differences between the two input leakages traces' patterns, by utilizing the concept of few-shot learning.

For training and evaluating the proposed system, the acquired data traces are divided into distinct datasets; training and validation/test. Training set consists of T_l traces and is used for training the model in training phase, whereas, validation/test dataset consists of T_v traces and is used to evaluate the trained model. Obviously, the test dataset is never shown to the model during training for a better generalized model by avoiding over-fitting. Moreover, the labels of test data are not shown to the network during validation process, whereas they are known to the evaluator for assessing the success of the presented attack.

Success of the presented system depends on the number of the traces in training set and is measured by the success rate on the test datasets through prediction accuracy. In traditional leakage assessment, after detecting the leakage, the exact point of interest/leakages can be found by performing sensitivity analysis (SA). SA identifies the leakage points by quantifying how much the points contribute to the leakage function through the learnt neural network. The contributing points/features/pixels have been detected in image processing and in for side-channel analysis [217].

10.4.2 Attack Model Architecture

Consider an adversary/evaluator has acquired a group of leakage traces $Q = Q_0, Q_1, ...Q_{N_G}$ (where N_G represents the total number of distinct groups) for power consumption or EM radiations of the algorithms' implementations under analysis. The input groups are divided equally to form a training and a validation datasets consisting of total N and M instances, respectively. Leakage instances of all the group instances consist of same size $n_{samples}$ samples. The instances of the each group are labeled (l) same as group name to form a dataset. Next step is to form input pairs (X_i, Y_i), where X_i represents the input pair of leakages $[x_a, x_b]$ (x_a and x_b represent an instance from the group set Q, with same or different class label l_a or l_b), and Y_i represents the pair label. Pair label Y_i is different from the target class labels and is further explained in the Sec. 10.4.3. In each pair, class label of x_a is fixed and is combined with all the possible combinations of other classes x_b. The instances x_b are randomly selected by using random function on index. No other affine distortions or noise is added to the signals, because each collected traces is by-default carrying varying random noise. In Siamese network, there are two identical/twin networks which carry same weights and share same parameters. The input pair $[x_a, x_b]$ is processed by the two Siamese ConvNets and output tensors $[Z(x_a), Z(x_b)]$ are produced as given by eq. 10.10, where w, f, and b represent the tensor weights, activation function and the bias.

$$Z = f(\sum w_i . x_i + b) \tag{10.10}$$

If the two networks are similar in every respect, then the out Z for similar inputs x, should produce similar feature vectors. For example, if Q_O instance is given as input to ConvNet and another instance of same class group Q_O, but a different instance, is given as input to ConvNet, then resulting feature vector would be similar. However, if the two leakages belong to different classes then the output feature vectors will be different too. The element-wise similarity score would be different in both mentioned cases, which means the similarity score produced by the sigmoid layer (predicted output $\boldsymbol{p}$) will be different as well. In summary, this helps in distinguishing if the data at the input tensors was same or different. The similarity score between the output tensor feature vectors $[Z(x_a), Z(x_b)]$, is computed using Euclidean distance. The output prediction p of the last sigmoidal layer, which joins both the Siamese twins, is given by eq. 10.12, where L represents the layer number, and α represents the other parameters learnt during the training process.

$$p = \sigma(\sum_j \alpha |Z(x_a)^j_{L-1} - Z(x_b)^j_{L-1}|) \tag{10.11}$$

A threshold is defined to distinguish the predicted score p from the randomly generated difference. Let D represent the difference between $Z(x_a)$ and $Z(x_b)$, which will be near to 0 or 1 if the vectors x_a and x_b are similar or dissimilar, respectively. The computed sigmoid function (given by eq. 10.12) for $D(> 10^{-4})$ would be $S(D)$ 0.5. For $|D| > 10^{-4}$, the vectors are accepted as similar by the network.

$$S(x) = \frac{1}{1 + e^{-x}} \tag{10.12}$$

The above presented general idea stays same for profiled and non-profiled scenarios excepts the label of groups are changed, as discussed in section below. It should be noted that in both scenarios the purpose is to assess if the leakage exists or not. Aim is not to recover the key. The overall attack model and evaluation scheme is shown in Fig. 10.1.

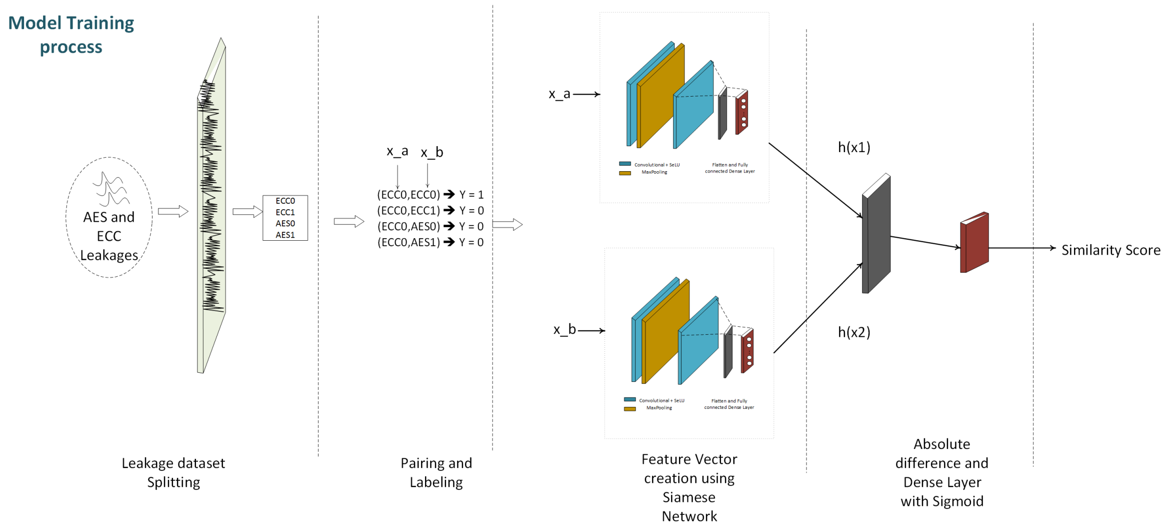


FIGURE 10.1: Propose Methodology Overview

Attack in Non-Profiled Scenario

In non-profiled case, the attacker/evaluator has limited access to the closed copy of the device under target, and cannot profile the device, thus offers a black box scenario. She can only control and observe the input and leakages of the system. In this case, the leakages are obtained with the fixed inputs to form the datasets for groups Q. Lets say she takes the leakages for two fixed inputs and forms groups Q_0 and Q_1, where instances x_a and x_b in pair x_a, x_b represent instances for two same or distinct class Q_0 and class Q_1. Based on the explanation presented in 10.4.2, she can form one set of pairs with same class label, that is instances x_a and x_b both belong to group Q_0. The other set of pairs will contain instances from both groups, that is x_a and x_b consist of instances for classes belonging to the group Q_0 and Q_1, respectively. Choice of size of a and b depends on the evaluator and can vary between $1, 2, ...N$. The formed dataset is then used for training and validation as explained in Section .10.4.3 and 10.4.4, respectively. The trained model can distinguish that if the unseen data instance, from validation dataset, belongs to fixed input group Q_0 and class Q_1 by calculating the similarity score. If similarity score is high and threshold validation success is more than 50%, then it means the leakage exists in the system and the implemented algorithm is not providing the required level of security.

Attack in Profiled Scenario

In profiled case, the attacker/evaluator has access to the open copy of the device. Consider an adversary is capable of performing leakage data acquisition while the target cryptographic algorithm is performing an encryption operation E on the device with the known key K, and produces a cipher text C, where $C = f(P, K)$. Adversary is interested in retrieving the key K, only by analysing the leakage information. The acquired data is for two fixed keys or key bits, lets say K_0 and K_1 for group Q_0 and Q_1, respectively. Further details about the leakage data acquisition are given in 10.5. To launch a machine-learning based side-channel attack, adversary needs to form instances pairs as explained in 10.4.2. One set of pairs with consist of the instances belonging to the same key class key, that is instances x_a and x_b both belong to key group K_0. The other set of pairs will contain instances from both key groups, that is x_a and x_b consist of instances for classes belonging to the group K_0 and K_1, respectively. Choice of size of a and b depends on the evaluator and can vary between $1, 2, ...N$. The formed dataset is then used for training and validation as explained in Section .10.4.3 and 10.4.4, respectively. The trained model can distinguish if the unseen data instance from validation dataset belongs to key class K_0 or K_1, based on the similarity score.

Attack Extension

The presented scenarios above are discussed for only two independent groups, whereas it can be easily extended to more than two groups as shown by experimental analysis in Sec .10.6.

10.4.3 Attack Model Training

The presented model is different from other classical deep learning-based side-channel analysis models as the classifier's output does not classify the leakage signal directly based on the probability, rather it calculates the similarity score denoting that the leakage signals belong to the either of the class from group Q. The network learns on the basis of the differences between the paired examples and is able to distinguish the unknown trace when presented in the evaluation phase. To model the problem as a supervised learning binary classification task, pairs $[x_a, x_b]$ are labeled as $Y =' 1'$ if both leakage signals are same and are labeled as $Y =' 0'$ if the leakage signals belong to a different target classes.

10.4.4 Attack Model Evaluation/Validation

The section describes the model evaluation strategy for the validation of the presented model. The described problem is different from the regular classification task so a different evaluation methodology is adopted to transform the problem into a supervised learning task. We have followed N-way k-shot validation methodology to evaluate the model performance. Training examples k are selected for all N classes for training the model, forming a support set. Query or test set consists of N classes which are required to be predicted. Traditionally for image processing tasks, unknown classes are given in test set to validate if the model can predict the correct class.

Labeled ($Y =' 1'$ for similar and $Y =' 0'$ for dissimilar pair entities) test pairs (x_a, x_b) are chosen randomly from the test (query) set and are given as an input to the trained Siamese network. Leakage x_a is kept same and x_b is chosen from the set Q, forming the four pairs. Lets say when we train the Siamese network, then there are four possible similarity scores outputs of the sigmoid layer, O_{S1}, O_{S2}, O_{S3}, and O_{S4}. Lets say $O_S 1$ is the output similarity score for the similarity pair only. If model is trained correctly then out of these scores, maximum score will be observed for the similarity pair only. This maximum similarity score O_{S1} is considered as correct class prediction and rest as incorrect predictions and is given by the eq.10.16.

$\mathrm{Y}_{predicted} = argmax_c \boldsymbol{p}$

For this analysis, the network is validated N_{val} times. Let $N_{correct}$ represent the correct class prediction then test set correct percentage is given by ed. 10.13.

$$Correct_{Percentage} = (100 * N_{correct})/N_{val} \tag{10.13}$$

We assume that the number of training and validation samples, T_l and T_v, are equal to maximize the statistical confidence value obtained during evlauation. Evaluator needs to select a upper bound that a false positive occurs which generally is $p_{th} = 10^{-5}$ in SCA evaluations. Let v be the validation accuracy then the total number of correct classifications can be represented by eq. 10.15. Considering a null hypothesis H_0 where the neural network does not learn anything and just predicts the class randomly then the total number of correct guesses would be a random variable from the binomial distribution H_0 : X Binom(M, 0.5) [218]. The probability that at least S_M correct classification occur in random fashion is given by eq. 10.15 and information is leaked if P(X ≥ S_M) <p$_{th}$.

$$S_M = v.T_v \tag{10.14}$$

$$P(X \geq S_M) = 0.5^M \sum_{k=S_M}^{M} \binom{M}{k} \tag{10.15}$$

10.4.5 Proposed Network Architecture

We have proposed a generic network which does not required tuning for leakages from the different algorithms as is the case with most of the existing deep learning leakage based SCA models.

For analysis in this study, two ConvNets are analyzed, named as ConvNet1 and ConvNet2. ConvNet1 consists of three convolutional layers and one dense layer with filters 8, 16 and 32, respectively, along with Max Pooling layer. ConvNet2 consists of four layers with filters 64, 128, 256, and 512, respectively, along with average pooling after each layer.

In all the network convolutional layers, Rectified Linear Units (ReLU) activation function is used. Before feeding the input data to the network, the input data features are normalized by scaling with the maximum absolute value. We initialized all the weights in the convolutional layers with glorot uniform distribution. We have tried different learning rates from fast to slow learning, for example 0.001, 0.0001, 0.00001 and 0.00006, and found that the network was able to converge to local minima quickly when trained with 0.00006. We obtained desired results by training the twin models with as low as 20 epochs and minibatch size of 32 with L_2 regularization (value = $2e^{-4}$) for convolutional layers which helps in better generation of the model. For the loss function, binary loss is used. A loss function computes the cross-entropy loss between the true labels and predicted labels. Let $Y(x_a, x_b)$ be the label and $p(Y(x_a, x_b))$ be the predicted probability based on the measure of similarity between x_a and x_b for N points. For the inputs being same (label $Y(x_a, x_b) =' 1'$, similar classes), log probability of being positive is added, that is $log(p(y(x_a, x_b)))$. Conversely, log probability of being negative (label $Y(x_a, x_b) =' 0'$, different classes) is $log(1 - p(y(x_a, x_b)))$. Hence the loss can be represented by the eq. 10.16.

$$Loss = \frac{-1}{N} \sum_{i=1}^{N} y(x_a^i, x_b^i).log(p(y(x_a^i, x_b^i)) + (1 - y(x_a^i, x_b^i)).log(1 - p(y(x_a^i, x_b^i))) \quad (10.16)$$

10.5 Datsets

The presented methodology is evaluated on two datasets consisting of the side-channel leakages from two standard cryptographic algorithm implementations; symmetric (AES) and asymmetric (ECC) algorithms.

10.5.1 AES Dataset

Dataset for AES power consumption leakage is collected from the FPGA implementation, using Sakura-X evaluation board [80], and Tektronix MDO series oscilloscope (having 1GHz bandwidth and 5GS/s sampling frequency). Sakura-X is a specialized board designed for the side-channel leakage data acquisition. A resistor is connected in series with the FPGA core to collect the power consumed by the system. The operating frequency of FPGA is reduced to $F_s = 3MHz$ and the sampling frequency of oscilloscope is set to $F_s = 2.5KHz$. Key byte k_i (where i = 2,3,..,254) is sent to the device, which triggers the encryption process. The leakage data traces for power consumption are collected while encryption E is performed with plain text P_m, where (where m = 1,2,..,10). Each T_{PWi} for ith key bit, of trace length T_len, consists of S_j samples (where j = 1,2..,T_{len}).

For data collection, a stand alone application is developed using C and Matlab libraries.

Data collection process is completely automated which requires just initial input parameter settings from the user. Data collection process has be divided into the following steps, performed using the developed application.

- Step 1 - Take input parameters (number of samples, key length) from the user;
- Step 2 - Configure the oscilloscope using Matlab libraries;
- Step 3 - Send the ROM address of the key from PC to FPGA through the control unit interface;
- Step 4 - Send the trigger signal to start encryption on FPGA;
- Step 5 - Initiate the process of leakage data collection and store in data file;
- Step 6 - After FPGA completes ECC processing, receive the encryption output and store in file.

10.5.2 ECC Dataset

The second dataset selected for ECC evaluations is the power consumption leakages of scalar multiplication operation of ECC implementations presented by Weissbart et al in [125]. The dataset is collected from the Ed25519 implementation of WolfSSL 3.10 (open source library written in C) on Pinata development board using Lecroy Waverunner z610i oscilloscope. The Pinata board for SCA evaluations by Riscure, consists of 32-bit STM32F4 microcontroller with an ARM-based architecture, running at 168 MHz clock frequency.

10.6 Experiments and Results

In this section, we present the experimental verification of the suitability of our proposed methodology for side-channel leakage assessment and classification strategy. We attempt to provide a real world scenario as the leakage traces are not preprocessed, and are not considering the circumstances of theoretical relevance, that is for AES, S-box only measurements are not targeted, rather a complete key is classified. We are bench-marking the presented model performance by performing experiments of existing and acquired datasets. Moreover, we have performed experiments on different algorithms implementations datasets to analyze the impact with respect to both symmetric and asymmetric ciphers. We have divided our experiments into three case studies; first set of analysis is performed to analyze the performance on the ECC and AES datasets using ConvNet1. Second set of experiments analyze the impact on four groups from both symmetric (AES) and asymmetric (ECC) leakage with underlying ConvNet2 model. For the last set of experiment, we have analyzed the performance on the classes which are present in query set only and were never presented in the support set during training.

For all the experiments each model is trained for 1-shot, 5-shot and 10-shot learning with 1,5, and 10 training examples for each class, respectively. N_{val}=200 validation query sets are tested to evaluate the model. We have performed four set of experiments by forming four combinations C_j (where $j = 4$) datasets of the pairs to train and evaluate the presented model. Each j combination datasets contains all four group labels, whereas one group label is selected as a maximum similarity score at a time. These four independent models are trained for the experimental analysis only to analyze the affect of keeping one pair as $Y = 1$ on classification of others. For the first combination $C1$ of the pairs, x_a is set to $AES0$ and x_b is taken from the group $Q_0 = AES0, Q_1 = AES1, Q_2 = ECC0, Q_3 = ECC1$, one-by-one. Pair $(AES0, AES0)$ is labeled as $Y =' 1'$, as it represents the true similarity score, named as 'similarity pair'. The rest of the pairs ($(AES0, AES1)$, $(AES0, ECC0)$, $(AES0, ECC1)$) are labeled as $Y =' 0'$, representing dissimilar classes, named as 'dissimilarity pair'. For the second combination $C2$, pair $(AES1, AES1)$ is labeled as $Y =' 1'$, and rest of the pairs ($(AES1, AES0)$, $(AES1, ECC0)$, $(AES1, ECC1)$) are labeled as $Y =' 0'$. Same is repeated for the rest of the two classes. Each combination of pairs is used to train the network, resulting in independent trained network. Trained networks for all these four combinations are evaluated one-by-one independently using the scheme presented in 10.4.4.

For the proposed system, a standalone python-based application is developed which takes input leakage datasets from the multiple sources. As a test case, AES, and ECC leakage datasets are presented to the system for above mentioned three set of experiments. The application splits data and forms separate vectors for all the classes in the group Q. Pairs (x_a, x_b) of the classes are formed based on these vectors for each class labels. Traces for forming the pairs are chosen randomly.

The results are presented in Table 10.1.

Case Study 1: ECC and AES datasets with ConvNet1 and ConvNet2

For this case study, ConvNet1 and ConvNet2 architectures are used for Siamese network. The validation score is computed for all four classes from the group. Support and query set have training and testing samples for the same classes. For the ConvNet1-based Siamese network, it is observed that for the classes belonging to ECC groups the traces can be distinguished with high probability. Training with 10-samples per class gives the best results. However, training with 1 and 5 samples gives results above 90% for both classes. For AES groups classes, one of the class group can be distinguished with above 90%, whereas other one gives better results with 1-shot learning as compared to other two.

For the ConvNet2-based Siamese network, ECC group classes can be distinguished with a better accuracy if trained with single training example only. However, poor performance is observed for AES group classes. Results on AES group classes show that the leakage exists as it has shown similarity with other traces 50-70.5% times. There are two reasons of poor validation accuracy on AES; one is the low sampling frequency selected for the AES leakage data acquisition, other is the selected points of interest. For AES, leakages for only one encryption round are selected for analysis, which does not contain sufficient information about the secret sensitive entity. It is observed that the overall performance has dropped while training with a complex neural network.

Case Study 2: Testing on Unseen Classes

In previous two cases, it is seen that the proposed attack model can successfully distinguish between the traces for both AES and ECC datasets. To analyze the model effect on unknown classes, in this case study we have given the classes in the query set whose samples are not shown to the network during training. It is observed that the classes whose samples are not

TABLE 10.1: Accuracy on the test dataset

Dataset	Class	1-shot	5-shot	10-shot
ECC-AES-ConvNet1	$ECC0$	91.5	96	96.5
	$ECC1$	94.5	94	98.5
	$AES0$	70.5	64	65
	$AES1$	90.5	88	89
ECC-AES-ConvNet2	$ECC0$	97	97.5	87
	$ECC1$	87	90	91
	$AES0$	53.5	48.5	50.5
	$AES1$	44.5	57	51
ECC-AES-Unseen Classes	$ECCR1$	57.5	60	66
	$ECCR2$	64.5	71	66.5
	$AESR1$	84.5	81.5	81.5
	$AESR2$	100	100	100

shown to the model, can still detect the leakages with more than 57% in all cases (more than 84% in AES case). It is also observed that in some cases training with 10 examples can improve the accuracy by 2-3% which is trivial.

Our proposed approach produces a single evaluation metric for distinguishibility of the leakages. However, in existing leakage assessment approach based on statistical analysis, individual univariate statistical tests are required to be performed. To lower the chances of false positives, we have performed analysis on 200 different random traces acquired with random input plain texts. Validation results showing above 50% accuracy shows that the a leakage can be detected from the random samples as well. In secure product evaluation false positive would be damaging. We do not claim that false positive will not occur. We have presented a model which eliminates the need of various separate test for various attacks, rather we stress that various algorithms can be combined in a single evaluation kit to identify if the leakage exist. If the system found a leakage then sensitive analysis can be performed to find the exact point of leakage. In contrast to the existing DL-LA methodology, which requires M training samples, our proposed methodology can identify the leakages using just few training examples.

10.7 Conclusion

We have introduced a novel methodology for leakage assessment which eliminates the requirement of huge datasets for training the neural network. Moreover, it provides a unified platform where multiple algorithms' implementation leakages can be combined in one system with a standard one metric, which eliminates the need of individual univariate statistical tests. Our proposed methodology can be used for detection of leakages from the white-box (profiled (open access to the device)) or black-box (non-profiled attacks (closed access to the device)) scenarios. Our proposed method can successfully identify the leakages from classes which are never used in training but belong to same group population. Leakages which are acquired with high sampling frequency demonstrated better leakage detection results as compared to the leakages obtained using low sampling frequency.

Chapter 11

Conclusion and Future Work

11.1 Conclusion

Weak implementations of a mathematically secure cryptographic algorithm imposes a great security risk to the cryptographic devices deployed in the IoT-based systems. With the introduction of machine learning to side-channel analysis along with the high-speed processing systems, this risk has dramatically increased. Machine learning and artificial intelligence are ever-evolving and new standards and techniques are constantly being introduced to improve the performance. This thesis investigated different avenues of side-channel attacks, and proposed powerful and promising side-channel attack methodologies exploiting emerging machine learning and deep learning techniques. The proposed methodologies are assessed on the real hardware-acquired side-channel leakages from different cryptographic chips and analysis is performed through developed hardware and software systems using different lanaguage platforms and libraries including Python, C#, MATLAB, Keras and Scikit-learn. The key contributions of this dissertation can be summarized as follows:

- A novel hybrid feature engineering scheme based on time-domain and frequency-domain signal properties, and feature extraction/selection is proposed. The proposed scheme is evaluated for both symmetric (Publication I) and asymmetric (Publication III) algorithms. For the symmetric algorithm AES, the different signal properties' combinations returned varying results for all three machine learning classifiers. However, best performance of 91.4% accuracy is obtained using Multi-Layer perceptron for the frequency-domain features set. Moreover, feature extraction (PCA) mainly improves the results for SVM classifier. Also, the secret key can be recovered from the asymmetric ECC leakage traces with 88-90 % accuracy using Random Forest and simple neural network (Multi-Layer Perceptron). Overall trend shows that the proposed feature engineering techniques with frequency-domain features can recover the secret key from the highly noisy data. This accuracy reaches 100% if only the aligned noise-free portion of the trace is used.

- Hyperparameter optimization plays a vital role in improving the accuracy of the machine learning trained models. In the presented research work, various hyperparameters for the machine learning algorithms have been tuned carefully to improve the performance. Chapter 4 presented the detailed hyperparameter tuning for the AES implementation leakages. However, the hyperparameter optimization is performed for all the models and schemes presented in this research. The final reported results are the ones obtained with the best trained model.

- Depending on the targeted key location (either a key bit or a key byte), various machine learning-based attack models can be formulated, leading to binary class or multi-class classification problems. In this study, three machine learning-based attack models are proposed (Publication IV) and are evaluated on the electromagnetic leakages of the countermeasure-protected and unprotected ECC algorithm FPGA implementations, along with the quantitative analysis for selecting the best attack model. In addition to assessing these attacks on the raw leakage traces, these attacks have also been evaluated on feature-engineered datasets. Moreover, keeping in view the noisy nature of the leakages in the real world scenario, leakage traces were selected to have more noise than useful information, i.e. 3% of the trace contains the relevant, useful information and 97% of the trace consists of the irrelevant noise. Based on the results, the secret key from the unprotected implementations can be recovered with 96% accuracy using GBB attack model. However, for the ECC implementations secured with the masked countermeasures, the key can be recovered with 50-75% accuracy for the presented attacks. This demonstrates that the relationship between the secret key and the leakage data can be detected using the presented attack and hence the countermeasure is not providing the required security level.

- For asymmetric ciphers, particularly ECC, the residue number system (RNS) is considered to be a strong countermeasure due to its parallel processing moduli. We have proposed a machine learning-based evaluation methodology for data-dependent and location-dependent attacks, along with the efficient novel hybrid feature engineering techniques, to recover the secret key from the dataset consisting of high-dimensional complex features due to the inherent parallel processing of RNS (Publication V). It is observed that the presented ML-SCA requires less preprocessing and gives better results for location-based profiling attacks, hence leading to realistic attack scenarios. The secret key can successfully be recovered from the unprotected and protected RNS-ECC SM implementations with 99% and 95% accuracy, respectively. Moreover, the impact of state-of-the-art and proposed hybrid feature engineering techniques have been analyzed as well. It is seen that PCA, LDA, T-test, and RF-based feature selection/extraction provide improved accuracy results. For this analysis, a comparative analysis is given with state-of-the-art template attacks. In previous studies, template attacks are performed on RNS-ECC, and the key is recovered by attacking the aligned portion of the trace only. However, our proposed scheme outperforms the existing template attack because it can recover the secret key by processing the complete trace instead of the aligned portion only. We conclude that machine-learning based side-channel attacks on PKC provide a realistic efficient attack scenario to recover the secret information as they require less pre-processing than the template attacks on RNS-ECC implementations.

- Efficient machine learning classification greatly depends on the structure/mechanics of the input datasets. Datasets having a low instance-feature ratio may result in a poorly-fitted model that might fail to generalize. To address this challenge, we have proposed a hybrid approach based on the CNN and dimensionality reduction and provided comparison with the existing complex schemes (Publication VI). Using our proposed CNN-based attack model, the attack efficiency improved by 6% for the protected ECC MPL SM BEC implementation leakages and 100% for the unprotected MPL leakages. The overall model training loss for our presented efficient architecture is less in comparison to the existing complex models. Our proposed ConvNet enables the network to train much faster and perform better by consuming fewer hardware resources, unlike the existing state-of-the-art methods.

- Leakage dataset with an insufficient number of leakage instances/traces in general, or an insufficient number of instances per class, can be a hurdle for an inaccurate attack modeling using machine learning-based side-channel analysis. To manage this issue well, we have proposed a data augmentation scheme based on CGAN and Siamese network (Siamese-CGAN), capable of generating artificial/fake data for each class that possesses similar characteristics as that of the original leakages (Publication VII). The presented scheme's performance has been evaluated using two popular neural networks, MLP and CNN, on the artificially generated data from the well-convergent and non-convergent networks for both symmetric and asymmetric algorithm implementation. Our analysis results show that the presented scheme results in a well-converged network that generates realistic leakage traces according to the evaluation dataset.

- We have also presented a leakage assessment and evaluation model that can take input leakages from multiple channels and multiple sources and successfully distinguish the leakages (Publication VIII). The presented system has been trained using the symmetric (AES implementation from FPGA) and asymmetric (ECC leakages from 32-bit STM32F4 microcontroller with an ARM-based architecture) leakages, and it is found that the leakages can be easily identified with high probability. Moreover, the presented system is flexible and leakages from more sources and different algorithms can be easily integrated.

11.2 Future Work

Machine learning-based side-channel analysis is an emerging research area, which has left the traditional (classical) side-channel analysis far behind. In this dissertation, we have explored and presented new side-channel attacks by exploiting state-of-the-art machine learning techniques. In future, the following aspects of ML-SCA can be further explored.

- In this study, simple machine learning algorithms (including SVM, NB, RF, Decision Trees) and deep learning algorithms (CNN and MLP) are explored. However, more deep learning models and architectures like Recurrent Neural Network (RNN) should be investigated for improving side-channel attacks especially Long Short-Term Memory (LSTM). LSTM is a special kind of artificial recurrent neural network. It differs from the standard feedforward networks in that it has memory and utilizes feedback information when making decisions. It would be interesting to explore if the memory-based neural networks can help in classifying the secret information in real-time.

- In this research, we have presented a hybrid leakage assessment model using few-shot learning for processing datasets from three sources. This idea can be extended to add the data from multiple algorithm sources and multiple channels to provide a universal evaluation tool.

- Transfer learning is a machine learning technique in which a network is trained for one task and is then re-used for some another task, instead of training from scratch for the second. This provides a more straightforward generalization of the network on the new dataset. We have proposed transfer learning in the form of few-shot learning task. However, the idea can be extended for combining the power and electromagnetic side-channel leakages. A neural network trained for leakages of one source/channel can be re-used to exploit the leakages' weights from the other source. This can lead to an efficient attack model due to the similar signal properties of power and electromagnetic signals.

- We have proposed a data augmentation technique based on CGAN. Further GAN based techniques can be explored, for example, Fisher GAN, Autoencoder instead of discriminator (EBGAN), Mean/Covariance Minimization GAN (McGAN) etc. Moreover, according to the No Free Lunch theorem, various datasets might produce varying results for the same machine learning algorithm. Consequently, various further cryptographic algorithms can be explored.

- With the advances being made in neural networks and the processing units' increased processing capability, an efficient standalone desktop and web-based application can be built, which can classify the given input leakages using various existing machine learning and deep learning-based architectures. There are currently many classical side-channel analysis commercial tools with limited integrated ML-SCA functionality. However, we are planning to develop an open-source, easy-access tool which can exploit the feature engineering techniques and machine/deep learning to ease the process of secure cryptographic algorithm evaluation.

- Moreover, based on the proposed efficient less resource-consuming attacks, it is possible to design and develop a mobile-based or a small hardware-based system using Tensorflow lite TinyML.

References

[1] T. Gomes, S. Pinto, T. Gomes, A. Tavares, and J. Cabral. *Towards an fpga-based edge device for the internet of things*. In *2015 IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA)*, pp. 1–4 (2015).

[2] S. Biookaghazadeh, F. Ren, and M. Zhao. *Are fpgas suitable for edge computing?* (2018).

[3] P. Kocher. *Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems*. In N. Koblitz, ed., *Advances in Cryptology — CRYPTO '96*, pp. 104–113 (Springer Berlin Heidelberg, 1996).

[4] E. De Mulder, P. Buysschaert, S. B. Ors, P. Delmotte, B. Preneel, G. Vandenbosch, and I. Verbauwhede. *Electromagnetic analysis attack on an fpga implementation of an elliptic curve cryptosystem*. In *EUROCON 2005 - The International Conference on "Computer as a Tool"*, vol. 2, pp. 1879–1882 (2005).

[5] S. A. Kadir, A. Sasongko, and M. Zulkifli. *Simple power analysis attack against elliptic curve cryptography processor on fpga implementation*. In *Proceedings of the 2011 International Conference on Electrical Engineering and Informatics*, pp. 1–4 (2011).

[6] P. Kocher, J. Jaffe, and B. Jun. *Differential power analysis*. In *Annual international cryptology conference*, pp. 388–397 (Springer, 1999).

[7] J. Longo, E. D. Mulder, D. Page, and M. Tunstall. *Soc it to em: Electromagnetic side-channel attacks on a complex system-on-chip*. In T. Güneysu and H. Handschuh, eds., *Cryptographic Hardware and Embedded Systems – CHES 2015*, pp. 620–640 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2015).

[8] D. Genkin, I. Pipman, and E. Tromer. *Get your hands off my laptop: Physical side-channel key-extraction attacks on pcs*. In L. Batina and M. Robshaw, eds., *Cryptographic Hardware and Embedded Systems – CHES 2014*, pp. 242–260 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2014).

[9] J.-S. Coron and I. Kizhvatov. *An efficient method for random delay generation in embedded software*. In C. Clavier and K. Gaj, eds., *Cryptographic Hardware and Embedded Systems - CHES 2009*, pp. 156–170 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2009).

[10] N. Veyrat-Charvillon, M. Medwed, S. Kerckhof, and F.-X. Standaert. *Shuffling against side-channel attacks: A comprehensive study with cautionary note*. In X. Wang and K. Sako, eds., *Advances in Cryptology – ASIACRYPT 2012*, pp. 740–757 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012).

[11] J. Irwin, D. Page, and N. P. Smart. *Instruction stream mutation for non-deterministic processors*. In *Proceedings IEEE International Conference on Application- Specific Systems, Architectures, and Processors*, pp. 286–295 (2002).

[12] T. Messerges. *Securing the aes finalists against power analysis attacks*. In G. Goos, J. Hartmanis, J. van Leeuwen, and B. Schneier, eds., *Fast Software Encryption*, pp. 150–164 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2001).

[13] M.-L. Akkar and C. Giraud. *An implementation of des and aes, secure against some attacks*. In C. Koc, D. Naccache, and C. Paar, eds., *Cryptographic Hardware and Embedded Systems — CHES 2001*, pp. 309–318 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2001).

[14] J. Blomer, J. Guajardo, and V. Krummel. *Provably secure masking of aes*. In H. Handschuh and M. Hasan, eds., *Selected Areas in Cryptography*, pp. 69–83 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2005).

[15] N. P. E Oswald, S. Mangard and V. Rijmen. *A side-channel analysis resistant description of the aes s-box*. In H. Gilbert and H. Handschuh, eds., *Fast Software Encryption*, pp. 413–423 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2005).

[16] M. Rivain and E. Prouff. *Provably secure higher-order masking of aes*. In S. Mangard and F.-X. Standaert, eds., *Cryptographic Hardware and Embedded Systems, CHES 2010*, pp. 413–427 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2010).

[17] R. Rivest. *Cryptography and machine learning*, pp. 427–439 (Springer Berlin Heidelberg, Berlin, Heidelberg, 1993).

[18] A. Levina, D. Sleptsova, and O. Zaitsev. *Side-channel attacks and machine learning approach*. In *2016 18th Conference of Open Innovations Association and Seminar on Information Security and Protection of Information Technology (FRUCT-ISPIT)*, pp. 181–186 (2016).

[19] H. Maghrebi, T. Portigliatti, and E. Prouff. *Breaking cryptographic implementations using deep learning techniques*. pp. 3–26 (2016).

[20] R. Gilmore, N. Hanley, and M. O'Neill. *Neural Network Based Attack on a Masked Implementation of AES*. In *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 106–11 (Institute of Electrical and Electronics Engineers (IEEE), 2015).

[21] L. Lerman, G. Bontempi, and O. Markowitch. *A machine learning approach against a masked aes*. Journal of Cryptographic Engineering **5**(2), 123 (2015).

[22] E. D. M. I. V. G. Hospodar, B. Gierlichs and J. Vandewalle. *Machine learning in side-channel analysis: a first study*. Journal of Cryptographic Engineering **1**(4), 293 (2011). URL https://doi.org/10.1007/s13389-011-0023-x.

[23] C. D. E. Cagli and E. Prouff. *Convolutional neural networks with data augmentation against jitter-based countermeasures*. In *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference*, pp. 45–68 (2017).

[24] URL http://www.dpacontest.org/home/.

[25] A. P. Fournaris, C. Dimopoulos, A. Moschos, and O. Koufopavlou. *Design and leakage assessment of side channel attack resistant binary edwards elliptic curve digital signature algorithm architectures*. Microprocessors and Microsystems **64**, 73 (2019).

[26] A. Bogdanov and I. Kizhvatov. *Beyond the limits of dpa: Combined side-channel collision attacks*. IEEE Transactions on Computers **61**(8), 1153 (2012).

[27] M. Renauld, F. Standaert, and N. Veyrat-Charvillon. *Algebraic side-channel attacks on the aes: Why time also matters in dpa*. In C. Clavier and K. Gaj, eds., *Cryptographic Hardware and Embedded Systems - CHES 2009*, pp. 97–111 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2009).

[28] S. Picek, A. Heuser, A. Jovic, S. Bhasin, and F. Regazzoni. *The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations*. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(1), 209 (2018). URL https://tches.iacr.org/index.php/TCHES/article/view/7339.

[29] N. I. of Standards and Technology. *Advanced encryption standard*. NIST FIPS PUB 197 (2001).

[30] D. ankerson and S. Vanstone. *Guide to Elliptic Curve Cryptography* (Springer, Berlin, Germany, 2004).

[31] G. S. I. Blake, G. Seroussi and N. Smart. Elliptic Curves in Cryptography, vol. 265 (Cambridge University Press, 1999).

[32] M. Joye and S. Yen. *The montgomery powering ladder*. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 291–302 (2003).

[33] C. A. Lara-Nino, A. Diaz-Perez, and M. Morales-Sandoval. *Elliptic curve lightweight cryptography: A survey*. IEEE Access **6**, 72514 (2018).

[34] G. de Meulenaer and F. Standaert. *Stealthy compromise of wireless sensor nodes with power analysis attacks*. In P. Chatzimisios, C. Verikoukis, I. Santamaría, M. Laddomada, and O. Hoffmann, eds., *Mobile Lightweight Wireless Systems*, pp. 229–242 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2010).

[35] X. Wang, Q. Zhou, J. Harer, G. Brown, Z. D. S. Qiu, C. A. G. J. Wang, A. Hinton, and P. Chin. *Deep learning-based classification and anomaly detection of side-channel signals*. In I. V. Ternovskiy and P. Chin, eds., *Cyber Sensing 2018*, vol. 10630, pp. 37 – 44. International Society for Optics and Photonics (SPIE, 2018). URL https://doi.org/10.1117/12.2311329.

[36] A. Sayakkara, N. Le-Khac, and M. Scanlon. *Leveraging electromagnetic side-channel analysis for the investigation of iot devices*. Digital Investigation **29**, S94 (2019).

[37] E. de Mulder, S. B. Ors, B. Preneel, and I. Verbauwhede. *Differential electromagnetic attack on an fpga implementation of elliptic curve cryptosystems*. In *2006 World Automation Congress*, pp. 1–6 (2006).

[38] T. Kasper, D. Oswald, and C. Paar. *A versatile framework for implementation attacks on cryptographic rfids and embedded devices*. Transactions on Computational Science **10**, 100 (2010).

[39] C. Clavier, B. Feix, G. Gagnerot, C. Giraud, M. Roussellet, and V. Verneuil. *Rosetta for single trace analysis*. In S. Galbraith and M. Nandi, eds., *Progress in Cryptology - INDOCRYPT 2012*, pp. 140–155 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012).

[40] C. Clavier, D. Marion, and A. Wurcker. *Simple power analysis on aes key expansion revisited*. In L. Batina and M. Robshaw, eds., *Cryptographic Hardware and Embedded Systems – CHES 2014*, pp. 279–297 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2014).

[41] S. Mangard. *A simple power-analysis (spa) attack on implementations of the aes key expansion*. In *Proceedings of the 5th International Conference on Information Security and Cryptology*, ICISC'02, p. 343–358 (Springer-Verlag, Berlin, Heidelberg, 2002).

[42] P. C. Kocher, J. Jaffe, B. Jun, and P. Rohatgi. *Introduction to differential power analysis*. J. Cryptogr. Eng. **1**(1), 5 (2011). URL https://doi.org/10.1007/s13389-011-0006-y.

[43] S. A. Kadir, A. Sasongko, and M. Zulkifli. *Simple power analysis attack against elliptic curve cryptography processor on fpga implementation*. In *Proceedings of the 2011 International Conference on Electrical Engineering and Informatics*, pp. 1–4 (2011).

[44] S. Chari, . J.R. Rao, and P. Rohatgi. *Template attacks*. In B. Kaliski, C. K. Koç, and C. Paar, eds., *Cryptographic Hardware and Embedded Systems - CHES 2002*, pp. 13–28 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2003).

[45] J. Rabaey. *Digital Integrated Circuits : a Design Perspective* (Englewood Cliffs, N.J. :Prentice Hall, 1996).

[46] V. Carlier, H. Chabanne, E. Dottax, and H. Pelletier. *Generalizing square attack using side-channels of an aes implementation on an fpga*. In *International Conference on Field Programmable Logic and Applications, 2005.*, pp. 433–437 (2005).

[47] J.-B. Coron, P. Kocher, and D. Naccache. *Statistics and secret leakage*. In Y. Frankel, ed., *Financial Cryptography*, pp. 157–173 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2001).

[48] T. Messerges. *Using second-order power analysis to attack dpa resistant software*. In C. Koc and C. Paar, eds., *Cryptographic Hardware and Embedded Systems — CHES 2000*, pp. 238–251 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2000).

[49] J. Waddle and W. D. *Towards efficient second-order power analysis*. In M. Joye and J-J.Quisquater, eds., *Cryptographic Hardware and Embedded Systems - CHES 2004*, pp. 1–15 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2004).

[50] E. Brier, C. Clavier, and F. Olivier. *Correlation power analysis with a leakage model.* In M. Joye and J.-J. Quisquater, eds., *Cryptographic Hardware and Embedded Systems - CHES 2004*, pp. 16–29 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2004).

[51] S.B. Örs, E. Oswald, and B. Preneel. *Power-analysis attacks on an fpga – first experimental results.* In C. Walter, C. Koç, and C. Paar, eds., *Cryptographic Hardware and Embedded Systems - CHES 2003*, pp. 35–50 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2003).

[52] K. Gandolfi, C. Mourtel, and F. Olivier. *Electromagnetic analysis: Concrete results.* In C. Koç, D. Naccache, and C. Paar, eds., *Cryptographic Hardware and Embedded Systems — CHES 2001*, pp. 251–261 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2001).

[53] J.-J. Quisquater and D. Samyde. *Electromagnetic analysis (ema): Measures and counter-measures for smart cards.* In I. Attali and T. Jensen, eds., *Smart Card Programming and Security*, pp. 200–210 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2001).

[54] J. R. S. Chari, C.S. Jutla and P. Rohatgi. *Towards sound approaches to counteract power-analysis attacks.* In M. Wiener, ed., *Advances in Cryptology — CRYPTO' 99*, pp. 398–412 (Springer Berlin Heidelberg, Berlin, Heidelberg, 1999).

[55] E. Prouff and M. Rivain. *Masking against side-channel attacks: A formal security proof.* In *EUROCRYPT* (2013).

[56] T. Popp and S. Mangard. *Masked dual-rail pre-charge logic: Dpa-resistance without routing constraints.* In J. Rao and B. Sunar, eds., *Cryptographic Hardware and Embedded Systems – CHES 2005*, pp. 172–186 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2005).

[57] S. Mangard. *Hardware countermeasures against dpa – a statistical analysis of their effectiveness.* In *Topics in Cryptology - CT-RSA 2004*, vol. 2964 of *Lecture Notes in Computer Science*, pp. 222–235 (Springer, 2004).

[58] S. Nagashima, N. Homma, Y. Imai, T. Aoki, and A. Satoh. *Dpa using phase-based waveform matching against random-delay countermeasure*. In *2007 IEEE International Symposium on Circuits and Systems*, pp. 1807–1810 (2007).

[59] J. G. J. van Woudenberg, M. Witteman, and B. Bakker. *Improving differential power analysis by elastic alignment*. In *Proceedings of the 11th International Conference on Topics in Cryptology: CT-RSA 2011*, CT-RSA'11, p. 104–119 (Springer-Verlag, Berlin, Heidelberg, 2011).

[60] L. Goubin and J. Patarin. *Des and differential power analysis the "duplication" method*. In Ç. K. Koç and C. Paar, eds., *Cryptographic Hardware and Embedded Systems*, pp. 158–172 (Springer Berlin Heidelberg, Berlin, Heidelberg, 1999).

[61] J.-S. Coron, E. Prouff, M. Rivain, and T. Roche. *Higher-order side channel security and mask refreshing*. vol. 8424 (2013).

[62] K. Schramm and C. Paar. *Higher order masking of the aes*. vol. 8424 (Springer, Berlin, Heidelberg, 2006).

[63] M. Joye, P. Paillier, and B. Schoenmakers. *On second-order differential power analysis*. In J. Rao and B. Sunar, eds., *Cryptographic Hardware and Embedded Systems – CHES 2005*, pp. 293–308 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2005).

[64] M. V. . A. Svoboda. *Circuit operators*. Stroje na. Zpracovani Informaci Sb. **111**, 247 (1957).

[65] B. Parhami. *Computer Arithmetic* (Oxford University Press, 2010).

[66] N. S. Szabo and R. H. Tanaka. *Residue arithmetic and its applications to computer technology* (1967).

[67] L. Papachristodoulou, A. P. Fournaris, K. Papagiannopoulos, and L. Batina. *Practical evaluation of protected residue number system scalar multiplication*. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2019**(1), 259 (2019). URL https://doi.org/10.13154/tches.v2019.i1.259-282.

[68] P. V. A. Mohan. *Residue Number Systems : Theory and Applications*.

[69] W. Schindler, K. Lemke, and C. Paar. *A stochastic model for differential side channel cryptanalysis*. In J. J.R. Rao and B. Sunar, eds., *Cryptographic Hardware and Embedded Systems – CHES 2005*, pp. 30–46 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2005).

[70] O. Markowitch, L. Lerman, and G. Bontempi. *Side channel attack: An approach based on machine learning*. In *Constructive Side-Channel Analysis and Secure Design, COSADE* (2011).

[71] M. Carbone, V. Conin, M. Cornélie, F. Dassance, G. Dufresne, C. Dumas, E. Prouff, and A. Venelli. *Deep Learning to Evaluate Secure RSA Implementations*. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019, Issue 2**, 132 (2019). URL https://tches.iacr.org/index.php/TCHES/article/view/7388.

[72] L. B. Y. LeCun, P. Haffner and Y. Bengio. *Object recognition with gradient-based learning*. In *Shape, Contour and Grouping in Computer Vision*, p. 319 (Springer-Verlag, Berlin, Heidelberg, 1999).

[73] A. H. S. B. A. H. J. Kim, S. Picek. *Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis*. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(3), 148 (2019).

[74] A. H. G. Zaid, L. Bossuet and A. Venelli. *Methodology for efficient cnn architectures in profiling attacks*. IACR Transactions on Cryptographic Hardware and Embedded Systems **2020**(1), 1 (2019). URL https://tches.iacr.org/index.php/TCHES/article/view/8391.

[75] P. Kocher, J. Jaffe, and B. Jun. *Differential power analysis*. In M. Wiener, ed., *Advances in Cryptology — CRYPTO' 99*, pp. 388–397 (Springer Berlin Heidelberg, Berlin, Heidelberg, 1999).

[76] B. L., J. Hogenboom, and J. G. J. van Woudenberg. *Getting more from pca: First results of using principal component analysis for extensive power analysis*. In D. O., ed., *Topics in Cryptology – CT-RSA 2012*, pp. 383–397 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012).

[77] L. Lerman, S. F. Medeiros, G. Bontempi, and O. Markowitch. *A machine learning approach against a masked aes*. In *CARDIS* (2013).

[78] L. Lerman, G. Bontempi, and O. Markowitch. *Power analysis attack: An approach based on machine learning*. Int. J. Appl. Cryptol. **3**(2), 97–115 (2014).

[79] D. Genkin, L. Pachmanov, I. Pipman, E. Tromer, and Y. Yarom. *Ecdsa key extraction from mobile devices via nonintrusive physical side channels*. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, p. 1626–1638 (Association for Computing Machinery, New York, NY, USA, 2016). URL https://doi.org/10.1145/2976749.2978353.

[80] URL http://satoh.cs.uec.ac.jp/SAKURA/index.html.

[81] (2001). URL NIST,FIPS-197:AdvanceencryptionStandard.

[82] L. Breiman. *Random forests*. Machine Learning, year = 2001 **45**, 5.

[83] T. M. Mitchell. *Generative and discriminative classifiers: Naive bayes and logistic regression machine learning* (2015).

[84] K. Kira and L. A. Rendell. *A practical approach to feature selection*. In D. Sleeman and P. Edwards, eds., *Machine Learning Proceedings 1992*, pp. 249 – 256 (M. Kaufmann, San Francisco (CA), 1992). URL http://www.sciencedirect.com/science/article/pii/B9781558602472500371.

[85] (2018). URL http://www.cs.waikato.ac.nz/ml/weka/.

[86] D. Oswald and C. Paar. *Improving side-channel analysis with optimal linear transforms*. In S. Mangard, ed., *Smart Card Research and Advanced Applications*, pp. 219–233 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2013).

[87] C. Archambeau, E. Peeters, F.-X. Standaert, and J. J. Quisquater. *Template Attacks in Principal Subspaces*, pp. 1–14 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2006).

[88] L. Bohy, M. Neve, D. Samyde, and J. jacques Quisquater. *Principal and independent component analysis for crypto-systems with hardware unmasked units*. In *In Proceedings of e-Smart 2003* (2003).

[89] D. Genkin, A. Shamir, and E. Tromer. *Rsa key extraction via low-bandwidth acoustic cryptanalysis*. In J. A. Garay and R. Gennaro, eds., *Advances in Cryptology – CRYPTO 2014*, pp. 444–461 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2014).

[90] O. Lo, W. Buchanan, and D. Carson. *Power analysis attacks on the aes-128 s-box using differential power analysis (dpa) and correlation power analysis (cpa)*. Journal of Cyber Security Technology **1**(2), 88 (2017). https://doi.org/10.1080/23742917.2016.1231523, URL https://doi.org/10.1080/23742917.2016.1231523.

[91] B. Schneier. URL https://www.schneier.com/blog/archives/2009/07/another_new_aes.html.

[92] N. Mukhtar and Y. Kong. *Secret key classification based on electromagnetic analysis and feature extraction using machine-learning approach*. In R. Doss, S. Piramuthu, and W. Zhou, eds., *Future Network Systems and Security*, pp. 80–92 (Springer International Publishing, Cham, 2018).

[93] N. Mukhtar and Y. Kong. *On features suitable for power analysis — filtering the contributing features for symmetric key recovery*. In *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, pp. 1–6 (2018).

[94] F.-X. Standaert, l. van Oldeneel tot Oldenzeel, D. Samyde, and J.-J. Quisquater. *Power analysis of fpgas: How practical is the attack?* In P. Y. K. Cheung and G. Constantinides, eds., *Field Programmable Logic and Application*, pp. 701–710 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2003).

[95] S. B. Örs, E. Oswald, and B. Preneel. *Power-analysis attacks on an fpga – first experimental results*. In C. Walter, C. K. Koç, and C. Paar, eds., *Cryptographic Hardware and Embedded Systems - CHES 2003*, pp. 35–50 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2003).

[96] L. Batina, B. Gierlichs, E. Prouff, M. Rivain, F. Standaert, and N. Veyrat-Charvillon. *Mutual information analysis: A comprehensive study*. J. Cryptol. **24**(2), 269–291 (2011). URL https://doi.org/10.1007/s00145-010-9084-8.

[97] S. Bhasin, J.-L. Danger, S. Guilley, and Z. Najm. *Side-channel leakage and trace compression using normalized inter-class variance*. In *Proceedings of the Third Workshop on Hardware and Architectural Support for Security and Privacy*, HASP '14 (Association for Computing Machinery, New York, NY, USA, 2014). URL https://doi.org/10.1145/2611765.2611772.

[98] R. Willi, A. Curiger, and P. Zbinden. *On power-analysis resistant hardware implementations of ecc-based cryptosystems*. In *2016 Euromicro Conference on Digital System Design (DSD)*, pp. 665–669 (2016).

[99] Y. S. M. N. S. G. J. D. F. Flament. *First principal components analysis: A new side channel distinguisher*. In *In Proceedings of the International Conference on Information Security and Cryptology*, pp. 407–419 (2010).

[100] C. Yun, D. Shin, H. Jo, J. Yang, and S. Kim. *An experimental study on feature subset selection methods*. In *7th IEEE International Conference on Computer and Information Technology (CIT 2007)*, pp. 77–82 (2007).

[101] Standards for Efficient Cryptography (SEC): Recommended Elliptic Curve Domain Parameters (2000). URL http://www.secg.org/.

[102] D. Hankerson, A. Menezes, and S. Vanstone. *Standards for efficient cryptography (sec): Recommended elliptic curve domain parameters* (2000). URL http://www.secg.org/.

[103] D. Narh Amanor, C. Paar, J. Pelzl, V. Bunimov, and M. Schimmler. *Efficient hardware architectures for modular multiplication on fpgas*. In *International Conference on Field Programmable Logic and Applications, 2005.*, pp. 539–542 (2005).

[104] A. Fattah, A. B. Eldin, and H. M. A. Fahmy. *An efficient architecture for interleaved modular multiplication*. World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering **3**, 2129 (2009).

[105] Y. Kong and E. Saeedi. *The investigation of neural networks performance in side-channel attacks*. Artificial Intelligence Review **52**(1), 607 (2019). URL https://doi.org/10.1007/s10462-018-9640-4.

[106] R. Azarderakhsh, J. Kimmo U., and M. Mozaffari-Kermani. *Efficient algorithm and architecture for elliptic curve cryptography for extremely constrained secure applications*. IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I-REGULAR PAPERS **61**(4), 1144 (2014). VK: crypto.

[107] Z. Liu, X. Huang, Z. Hu, M. K. Khan, H. Seo, and L. Zhou. *On emerging family of elliptic curves to secure internet of things: Ecc comes of age*. IEEE Transactions on Dependable and Secure Computing **14**(3), 237 (2017).

[108] K. Mahmood, S. Chaudhry, H. Naqvi, S. Kumari, X. Li, and A. Sangaiah. *An elliptic curve cryptography based lightweight authentication scheme for smart grid communication*. Future Generation Computer Systems (2017).

[109] S. Chaudhry, H. Naqvi, K. Mahmood, H. Ahmad, and K. Khan. *An improved remote user authentication scheme using elliptic curve cryptography*. Wireless Personal Communications **90**, 1 (2016).

[110] A. Höller, N. Druml, C. Kreiner, C. Steger, and T. Felicijan. *Hardware/software co-design of elliptic-curve cryptography for resource-constrained applications*. In *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6 (2014).

[111] M. Carbone, V. Conin, M.-A. CornÃ©lie, F. Dassance, G. Dufresne, C. Dumas, E. Prouff, and A. Venelli. *Deep learning to evaluate secure rsa implementations*. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(2), 132 (2019). URL https://tches.iacr.org/index.php/TCHES/article/view/7388.

[112] A. Heuser and M. Zohner. *Intelligent machine homicide*. In W. Schindler and S. A. Huss, eds., *Constructive Side-Channel Analysis and Secure Design*, pp. 249–264 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012).

[113] R. Gilmore, N. Hanley, and M. O'Neill. *Neural network based attack on a masked implementation of aes*. In *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 106–111 (2015).

[114] T. Bartkewitz and K. Lemke-Rust. *Efficient template attacks based on probabilistic multi-class support vector machines*. In S. Mangard, ed., *Smart Card Research and Advanced Applications*, pp. 263–276 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2013).

[115] D. Wolpert. *The supervised learning no-free-lunch theorems* (2001).

[116] L. Lerman, S. Medeiros, N. Veshchikov, C. Meuter, G. Bontempi, and O. Markowitch. *Semi-supervised template attack*. In E. Prouff, ed., *Constructive Side-Channel Analysis and Secure Design*, pp. 184–199 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2013).

[117] E.Prouff, R. Strullu, R. Benadjila, E. Cagli, and C. Dumas. *Study of deep learning techniques for side-channel analysis and introduction to ASCAD database*. IACR Cryptology ePrint Archive **2018**, 53 (2018). URL http://eprint.iacr.org/2018/053.

[118] A. Bellezza. *Countermeasures against side-channel attacks for elliptic curve cryptosystems* (2001).

[119] G. Fan, Y. Zhou, H. Zhang, and D. Feng. *How to choose interesting points for template*

attacks more effectively? In M. Yung, L. Zhu, and Y. Yang, eds., *Trusted Systems*, pp. 168–183 (Springer International Publishing, Cham, 2015).

[120] M. U. Yaseen, A. Anjum, O. Rana, and N. Antonopoulos. *Deep learning hyper-parameter optimization for video analytics in clouds*. IEEE Transactions on Systems, Man, and Cybernetics: Systems **49**(1), 253 (2019).

[121] J.-J. Quisquater and D. Samyde. *Electromagnetic analysis (ema): Measures and counter-measures for smart cards*. In *International Conference on Research in Smart Cards*, pp. 200–210 (Springer, 2001).

[122] K. Gandolfi, C. Mourtel, and F. Olivier. *Electromagnetic analysis: Concrete results*. In *International Workshop on Cryptographic Hardware and Embedded systems*, pp. 251–261 (Springer, 2001).

[123] D. Agrawal, B. Archambeault, J. R. Rao, and P. Rohatgi. *The em side-channel(s)*. In *Revised Papers from the 4th International Workshop on Cryptographic Hardware and Embedded Systems*, CHES '02, p. 29–45 (Springer-Verlag, Berlin, Heidelberg, 2002).

[124] A. Zajić and M. Prvulovic. *Experimental demonstration of electromagnetic information leakage from modern processor-memory systems*. IEEE Transactions on Electromagnetic Compatibility **56**(4), 885 (2014).

[125] L. Weissbart, S. Picek, and L. Batina. *One Trace Is All It Takes: Machine Learning-Based Side-Channel Attack on EdDSA*. In S.Bhasin, A. Mendelson, and M. Nandi, eds., *Security, Privacy, and Applied Cryptography Engineering - 9th International Conference, SPACE 2019, Gandhinagar, India, December 3-7, 2019, Proceedings*, vol. 11947 of *Lecture Notes in Computer Science*, pp. 86–105 (Springer, 2019). URL https://doi.org/10.1007/978-3-030-35869-3_8.

[126] N. Mukhtar, M. M. andY. Kong, and A. Anjum. *Machine-learning-based side-channel evaluation of elliptic-curve cryptographic fpga processor*. Applied Sciences **9**, 64 (2018).

[127] V. Bunimov and M. Schimmler. *Area and time efficient modular multiplication of large integers*. In *Proceedings IEEE International Conference on Application-Specific Systems, Architectures, and Processors. ASAP 2003*, pp. 400–409 (2003).

[128] J. Schmidhuber. *Deep learning in neural networks*. Neural Netw. **61**(C), 85–117 (2015). URL https://doi.org/10.1016/j.neunet.2014.09.003.

[129] Z. Zeng, D. Gu, J. Liu, and Z. Guo. *An Improved Side-Channel Attack Based on Support Vector Machine*. In *2014 Tenth International Conference on Computational Intelligence and Security*, pp. 676–680 (2014).

[130] C. Cortes and V. Vapnik. *Support-vector networks*. Mach. Learn. **20**(3), 273–297 (1995). URL https://doi.org/10.1023/A:1022627411411.

[131] URL https://www.newae.com/chipwhisperer.

[132] S. Picek, A.Heuser, A. Jovic, S. Bhasin, and F. Regazzoni. *The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations*. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2019**(1), 209 (2019). URL https://doi.org/10.13154/tches.v2019.i1.209-237.

[133] R. Benadjila, E. Prouff, R. Strullu, E. Cagli, and C. Dumas. *Deep learning for side-channel analysis and introduction to ASCAD database*. Journal of Cryptographic Engineering (2019).

[134] E. D. Mulder, S. B. Örs, B. Preneel, and I. Verbauwhede. *Differential Power and Electromagnetic Attacks on a FPGA Implementation of Elliptic Curve Cryptosystems*. Comput. Electr. Eng. **33**(5–6), 367–382 (2007). URL https://doi.org/10.1016/j.compeleceng.2007.05.009.

[135] D. Genkin, A. Shamir, and E. Tromer. *RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis*. In J. A. Garay and R. Gennaro, eds., *Advances in Cryptology – CRYPTO 2014*, pp. 444–461 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2014).

[136] S.Chari, J. R. Rao, and P. Rohatgi. *Template attacks.* In B. S. K. Jr., Ç. K. Koç, and C. Paar, eds., *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, vol. 2523 of *Lecture Notes in Computer Science*, pp. 13–28 (Springer, 2002).

[137] H. Maghrebi, T. Portigliatti, and E. Prouff. *Breaking cryptographic implementations using deep learning techniques.* IACR Cryptology ePrint Archive **2016**, 921 (2016). URL http://eprint.iacr.org/2016/921.

[138] P. Langley and W. Iba. *Average-case analysis of a nearest neighbor algorthim.* In *Proceedings of the 13th International Joint Conference on Artifical Intelligence - Volume 2*, IJCAI'93, p. 889–894 (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993).

[139] S. Picek, A. Heuser, A. Jovic, and L. Batina. *A Systematic Evaluation of Profiling Through Focused Feature Selection.* IEEE Transactions on Very Large Scale Integration (VLSI) Systems **27**, 2802 (2019).

[140] N. Guillermin. *A Coprocessor for Secure and High Speed Modular Arithmetic.* IACR Cryptology ePrint Archive (2011).

[141] P. Martins and L. Sousa. *The role of non-positional arithmetic on efficient emerging cryptographic algorithms.* IEEE Access **8**, 59533 (2020).

[142] J.-C. Bajard, L. Imbert, P.-Y. Liardet, and Y.Teglia. *Leak Resistant Arithmetic.* In M. Joye and J.-J. Quisquater, eds., *Cryptographic Hardware and Embedded Systems - CHES*, vol. 3156 of *Lecture Notes in Computer Science*, pp. 62–75 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2004).

[143] L. Papachristodoulou, A. P. Fournaris, K. Papagiannopoulos, and L. Batina. *Practical evaluation of protected residue number system scalar multiplication.* IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(1), 259 (2018). URL https://tches.iacr.org/index.php/TCHES/article/view/7341.

[144] J. J. G. Goodwill, B. Jun and P. Rohatgi. *A testing methodology for side channel resistance validation* (NIST noninvasive attack testing workshop, 2011).

[145] A. P. Fournaris, l. Papachristodoulou, and N. Sklavos. *Secure and Efficient RNS Software Implementation for Elliptic Curve Cryptography*. In *2017 IEEE Eur. Symp. Secur. Priv. Work.*, pp. 86–93 (IEEE, 2017). URL http://ieeexplore.ieee.org/document/7966976/.

[146] J.-C. Bajard, S. Duquesne, and N. Meloni. *Combining Montgomery Ladder for Elliptic Curves defined over Fp and RNS Representation*. In *Research Report 06041* (2006).

[147] M. Ciet, M. Neve, E. Peeters, and J.-J. Quisquater. *Parallel fpga implementation of rsa with residue number systems - can side-channel threats be avoided?* In *2003 46th Midwest Symposium on Circuits and Systems* (2003).

[148] J.-C. Bajard, J. Eynard, and F. Gandino. *Fault Detection in RNS Montgomery Modular Multiplication*. In *IEEE 21st Symp. on Comp. Arithmetic*, pp. 119–126 (IEEE, 2013).

[149] A. P. Fournaris, N. Klaoudatos, N.Sklavos, and C. Koulamas. *Fault and Power Analysis Attack Resistant RNS based Edwards Curve Point Multiplication*. In *Proceedings of the 2nd Workshop on Cryptography and Security in Computing Systems, CS2 at HiPEAC 2015, Amsterdam, Netherlands, January 19-21, 2015*, pp. 43–46 (2015).

[150] A. P. Fournaris, L. Papachristodoulou, L.Batina, and N. Sklavos. *Residue Number System as a side channel and fault injection attack countermeasure in elliptic curve cryptography*. In *2016 International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS)*, pp. 1–4 (2016).

[151] M. Joye and S. Yen. *The montgomery powering ladder*. In B. S. K. Jr., Ç. K. Koç, and C. Paar, eds., *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, vol. 2523 of *Lecture Notes in Computer Science*, pp. 291–302 (Springer, 2002). URL https://doi.org/10.1007/3-540-36400-5_22.

[152] A. P. Fournaris. *RNS_LRA_EC_scalar Multiplier* (2018). https://github.com/afournaris/RNS_LRA_EC_Scalar_Multiplier.

[153] L. Breiman. *Random forests*. Mach. Learn. **45**(1), 5–32 (2001). URL https://doi.org/10.1023/A:1010933404324.

[154] A. L. Blum and P. Langley. *Selection of relevant features and examples in machine learning*. Artif. Intell. **97**(1–2), 245–271 (1997).

[155] L.Batina, J. Hogenboom, and J. J. van Woudenberg. *Getting more from PCA: first results of using principal component analysis for extensive power analysis*. In O. Dunkelman, ed., *Topics in Cryptology - CT-RSA 2012 - The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27 - March 2, 2012. Proceedings*, vol. 7178 of *Lecture Notes in Computer Science*, pp. 383–397 (Springer, 2012).

[156] A.Golder, D. Das, J. Danial, S. Ghosh, S. Sen, and A. Raychowdhury. *Practical approaches toward deep-learning-based cross-device power side-channel attack*. IEEE Transactions on Very Large Scale Integration (VLSI) Systems **27**, 2720 (2019).

[157] F.-X. Standaert and C. Archambeau. *Using subspace-based template attacks to compare and combine power and electromagnetic information leakages*. In E. Oswald and P. Rohatgi, eds., *Cryptographic Hardware and Embedded Systems – CHES 2008*, pp. 411–425 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2008).

[158] I. Jolliffe. *Principal Component Analysis*, pp. 1094–1096 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2011).

[159] P. Belhumeur, J. Hespanha, and D. Kriegman. *Eigenfaces vs. fisherfaces: Recognition using class specific linear projection*. IEEE Trans. Pattern Anal. Mach. Intell. **19**, 711 (1997).

[160] D. L. Swets and J. J. Weng. *Using discriminant eigenfeatures for image retrieval*. IEEE Transactions on Pattern Analysis and Machine Intelligence **18**(8), 831 (1996).

[161] G. H. John, R. Kohavi, and K. Pfleger. *Irrelevant features and the subset selection problem.* In *Proceedings of the Eleventh International Conference on International Conference on Machine Learning*, ICML'94, p. 121–129 (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994).

[162] R. Kohavi and G. John. *Wrappers for feature subset selection.* Artif. Intell. **97**(1–2), 273–324 (1997).

[163] V. Svetnik, A. Liaw, C. Tong, J. Culberson, R. Sheridan, and B. Feuston. *Random forest: a classification and regression tool for compound classification and qsar modeling*. Journal of Chemical Information and Computer Sciences **43**(6), 1947 (2003).

[164] D. Schinianakis, A. Fournaris, H. Michail, A. Kakarountas, and T. Stouraitis. *An rns implementation of an elliptic curve point multiplier*. Circuits and Systems I: Regular Papers, IEEE Transactions on **56**, 1202 (2009).

[165] *Inspector SCA tool.* URL hhttps://www.riscure.com/security-tools/978inspector-sca/ .Accessed:2017-12-14.

[166] F. Chollet *et al. Keras*. https://keras.io (2015).

[167] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. *Scikit-learn: Machine learning in Python*. Journal of Machine Learning Research **12**, 2825 (2011).

[168] *National Computational Infrastructure Australia*. URL https://nci.org.au/our-services/ supercomputing.

[169] P. Guilherme, L. Imbert, L. Torres, and P. Maurine. *Attacking randomized exponentiations using unsupervised learning* (2014).

[170] C. Andrikos, L. Batina, L. Chmielewski, L. Lerman, V. Mavroudis, K. Papagiannopoulos, G. Perin, G. Rassias, and A. Sonnino. *Location, location, location: Revisiting modeling and exploitation for location-based side channel leakages*. In *Advances in*

Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part III, vol. 11923 of *Lecture Notes in Computer Science*, pp. 285–314 (Springer, 2019). URL https://doi.org/10.1007/978-3-030-34618-8_10.

[171] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer. *Smote: Synthetic minority over-sampling technique*. J. Artif. Intell. Res. (JAIR) **16**, 321 (2002).

[172] G. James, D. W. andT. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning: With Applications in R* (2013).

[173] F. Durvaux, F.-X. Standaert, and N. Veyrat-Charvillon. *How to certify the leakage of a chip?* In *EUROCRYPT*, pp. 459–476 (Springer, 2014). https://www.iacr.org/archive/eurocrypt2014/84410138/84410138.pdf.

[174] S. Mangard and E. O. andT. Popp. *Power analysis attacks: Revealing the secrets of smart cards*, vol. 31 (Springer Science & Business Media, 2008).

[175] M. Medwed and E. Oswald. *Template attacks on ecdsa*. pp. 14–27 (2009).

[176] N. Mukhtar and Y. Kong. *On features suitable for power analysis - filtering the contributing features for symmetric key recovery*. In A. Varol, M. Karabatak, and C. Varol, eds., *6th International Symposium on Digital Forensic and Security*, pp. 265–270 (Institute of Electrical and Electronics Engineers (IEEE), United States, 2018).

[177] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. *Focal loss for dense object detection*. CoRR **abs/1708.02002** (2017). 1708.02002, URL http://arxiv.org/abs/1708.02002.

[178] D. Hankerson, A. J. Menezes, and S. Vanstone. *Guide to Elliptic Curve Cryptography* (Springer-Verlag New York, Inc., 2003).

[179] A. Fournaris. *Fault and Power Analysis Attack Protection Techniques for Standardized*

Public Key Cryptosystems, pp. 93–105 (Springer International Publishing, Cham, 2017).

[180] A. Bauer, E. Jaulmes, E. Prouff, and J. Wild. *Horizontal and vertical side-channel attacks against secure rsa implementations*. In E. Dawson, ed., *Topics in Cryptology, CT-RSA 2013*, vol. 7779 of *LNCS*, pp. 1–17 (Springer Berlin Heidelberg, 2013).

[181] A. Bauer, E. Jaulmes, E. Prouff, and j. Wild. *Horizontal collision correlation attack on elliptic curves*. In T. Lange, K. Lauter, and P. Lison?k, eds., *Selected Areas in Cryptography – SAC 2013*, vol. 8282 of *Lecture Notes in Computer Science*, pp. 553–570 (Springer Berlin Heidelberg, 2014).

[182] M. Joye and S.-M. Yen. *The Montgomery Powering Ladder*. In *CHES '02: Revised Papers from the 4th International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 291–302 (Springer-Verlag, London, UK, 2003).

[183] P. Fouque and F. Valette. *The Doubling Attack Why Upwards Is Better than Downwards*. In C. Walter, C. Koc, and C. Paar, eds., *Cryptogr. Hardw. Embed. Syst. - CHES 2003*, vol. 2779 of *Lecture Notes in Computer Science*, pp. 269–280 (Springer Berlin / Heidelberg).

[184] S. Yen, L. Ko, S. Moon, and J. Ha. *Relative doubling attack against Montgomery Ladder*. Inf. Secur. Cryptology-ICISC 2005 pp. 117–128 (2006).

[185] S.-M. Yen, W.-C. Lien, S.-J. Moon, and J. Ha. *Power Analysis by Exploiting Chosen Message and Internal Collisions - Vulnerability of Checking Mechanism for RSA-Decryption*. In *Mycrypt*, vol. 3715 of *LNCS*, pp. 183–195 (Springer, 2005).

[186] P. Kocher, J. Jaffe, and B. Jun. *Differential power analysis*. In *in Proc. of CRYPTO '99*, pp. 388–397 (Springer-Verlag, 1999).

[187] F. Amiel, B. Feix, and K. Villegas. *Power analysis for secret recovering and reverse engineering of public key algorithms*. In C. Adams, A. Miri, and M. Wiener, eds.,

Selected Areas in Cryptography, vol. 4876 of *Lecture Notes in Computer Science*, pp. 110–125 (Springer Berlin Heidelberg, 2007).

[188] A. Bogdanov, I. Kizhvatov, and A. Pyshkin. *Algebraic methods in side-channel collision attacks and practical collision detection*. In D. Chowdhury, V. Rijmen, and A. Das, eds., *Progress in Cryptology - INDOCRYPT 2008*, vol. 5365 of *Lecture Notes in Computer Science*, pp. 251–265 (Springer Berlin Heidelberg, 2008).

[189] A. Moradi. *Statistical tools flavor side-channel collision attacks*. In D. Pointcheval and T. Johansson, eds., *Advances in Cryptology EUROCRYPT 2012*, vol. 7237 of *Lecture Notes in Computer Science*, pp. 428–445 (Springer Berlin Heidelberg, 2012).

[190] B. Feix, M. Roussellet, and A. Venelli. *Side-channel analysis on blinded regular scalar multiplications*. In W. Meier and D. Mukhopadhyay, eds., *Progress in Cryptology – INDOCRYPT 2014*, vol. 8885 of *Lecture Notes in Computer Science*, pp. 3–20 (Springer International Publishing, 2014).

[191] C. Clavier, B. Feix, G. Gagnerot, M. Roussellet, and V. Verneuil. *Horizontal correlation analysis on exponentiation*. In M. Soriano, S. Qing, and J. López, eds., *Information and Communications Security*, vol. 6476 of *Lecture Notes in Computer Science*, pp. 46–61 (Springer Berlin Heidelberg, 2010).

[192] C. Whitnall, E. Oswald, and F.-X. Standaert. *The myth of generic DPA... and the magic of learning*. In *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8366 LNCS, pp. 183–205 (Springer Verlag, 2014).

[193] L. Batina, L. Chmielewski, L. Papachristodoulou, P. Schwabe, and M. Tunstall. *Online template attacks*. In *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8885, pp. 21–36 (Springer, Cham, 2014).

[194] L. Papachristodoulou, A. P. Fournaris, K. Papagiannopoulos, and L. Batina. *Practical evaluation of protected residue number system scalar multiplication*. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**(1), 259 (2018).

[195] S. Mangard, E. Oswald, and T. Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security)* (2007). URL http://dl.acm.org/citation.cfm?id=1208234.

[196] J. Fan and I. Verbauwhede. *An updated survey on secure ecc implementations: Attacks, countermeasures and cost*. In D. Naccache, ed., *Cryptography and Security: From Theory to Applications*, vol. 6805 of *Lecture Notes in Computer Science*, pp. 265–282 (Springer Berlin Heidelberg, 2012).

[197] J.-S. Coron. *Resistance against differential power analysis for elliptic curve cryptosystems*. In *Proc. of CHES '99*, pp. 292–302 (Springer-Verlag, London, UK, 1999).

[198] M. Kubat and S. Matwin. *Addressing the curse of imbalanced training sets: One-sided selection*. In *ICML* (1997).

[199] I. M. J. Zhang. *Knn approach to unbalanced data distributions: A case study involving information extraction*. Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Datasets (2003).

[200] K. Fukushima and S. Miyake. *Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition*. In S. Amari and M. Arbib, eds., *Competition and Cooperation in Neural Nets*, pp. 267–285 (Springer Berlin Heidelberg, Berlin, Heidelberg, 1982).

[201] A. Moschos, A. Fournaris, and O. Koufopavlou. *A flexible leakage trace collection setup for arbitrary cryptographic ip cores*. In *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)* (IEEE, 2018).

[202] D. Kim, J. Kim, and J.Kim. *Elastic exponential linear units for convolutional neural networks*. Neurocomputing (2020). URL http://www.sciencedirect.com/science/article/pii/S0925231220304240.

[203] S. Picek, A. Heuser, and S. Guilley. *Profiling side-channel analysis in the restricted attacker framework*. IACR Cryptol. ePrint Arch. **2019**, 168 (2019).

[204] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. *Generative adversarial nets*. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, p. 2672–2680 (MIT Press, Cambridge, MA, USA, 2014).

[205] P. Wang, P. Chen, Z.Luo, G.Dong, M. Zheng, N. Yu, and H. Hu. *Enhancing the performance of practical profiling side-channel attacks using conditional generative adversarial networks* (2020). 2007.05285.

[206] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. *Improved techniques for training gans*. CoRR **abs/1606.03498** (2016). 1606.03498, URL http://arxiv.org/abs/1606.03498.

[207] C. Hsu, C. Lin, W. Su, and G.Cheung. *Sigan: Siamese generative adversarial network for identity-preserving face hallucination*. CoRR **abs/1807.08370** (2018). 1807.08370, URL http://arxiv.org/abs/1807.08370.

[208] A.Radford, L. Metz, and S. Chintala. *Unsupervised representation learning with deep convolutional generative adversarial networks*. In Y. Bengio and Y. LeCun, eds., *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings* (2016). URL http://arxiv.org/abs/1511.06434.

[209] M. Mirza and S. Osindero. *Conditional generative adversarial nets*. CoRR **abs/1411.1784** (2014). 1411.1784, URL http://arxiv.org/abs/1411.1784.

[210] X. Chen, Y. D. R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. *Infogan: Interpretable representation learning by information maximizing generative adversarial nets*. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, eds., *Advances in Neural Information Processing Systems 29*, pp. 2172–2180 (Curran Associates, Inc., 2016). URL http://papers.nips.cc/paper/6399-infogan-interpretable-representation-learning-by-information-maximizing-generative-adve pdf.

[211] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas. *Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks*. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5908–5916 (2017).

[212] G. Koch, R. Zemel, and R. Salakhutdinov. *Siamese neural networks for one-shot image recognition* (2015).

[213] D. Chicco. *Siamese Neural Networks: An Overview*, pp. 73–94 (Springer US, New York, NY, 2021).

[214] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. *Matching networks for one shot learning*. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, p. 3637–3645 (Curran Associates Inc., Red Hook, NY, USA, 2016).

[215] C. Wu, L. Herranz, X. Liu, Y. Wang, J. Weijer, and B. Raducanu. *Memory replay gans: learning to generate images from new categories without forgetting* (2018).

[216] G. Goodwill, B. Jun, J. Jaffe, and P. Rohatgi. *A testing methodology for side channel resistance* (2011).

[217] A. Moradi, B. Richter, T. Schneider, and F.-X. Standaert. *Leakage detection with the x2-test*. IACR Transactions on Cryptographic Hardware and Embedded Systems **2018**(1), 209 (2018). URL https://tches.iacr.org/index.php/TCHES/article/view/838.

[218] F. Wegener, T. Moos, and A.Moradi. *Dl-la: Deep learning leakage assessment: A modern roadmap for sca evaluations*. IACR Cryptol. ePrint Arch. **2019**, 505 (2019).

[219] T. Schneider and A. Moradi. *Leakage assessment methodology*. Journal of Cryptographic Engineering **6**, 85 (2016).

[220] (2011). URL https://csrc.nist.gov/Events/2011/Non-Invasive-Attack-Testing-Workshop.

[221] Li Fei-Fei, R. Fergus, and P. Perona. *One-shot learning of object categories*. IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(4), 594 (2006).

[222] S. Dey, A. Dutta, J. Toledo, S. Ghosh, J. Lladós, and U. Pal. *Signet: Convolutional siamese network for writer independent offline signature verification* (2017).

[223] F. Schroff, D. Kalenichenko, and J. Philbin. *Facenet: A unified embedding for face recognition and clustering*. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–823 (2015).