CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Anomaly detection in IoT devices using side-channel power consumption
Authors	Lightbody, Dominic
Publication date	2026-04-23
Original Citation	Lightbody, D. 2026. Anomaly detection in IoT devices using side-channel power consumption. PhD Thesis, University College Cork.
Type of publication	Doctoral thesis
Rights	© 2026, Dominic Lightbody. - https://creativecommons.org/licenses/by/4.0/
Download date	2026-11-17 10:51:56
Item downloaded from	https://hdl.handle.net/10468/18851

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Ollscoil na hÉireann, Corcaigh

National University of Ireland, Cork



Anomaly Detection in IoT Devices Using Side-Channel Power Consumption

Thesis presented by

DOMINIC LIGHTBODY

[0000-0001-7114-2055]

for the degree of

Doctor of Philosophy

University College Cork

School of Engineering

Department of Electrical and Electronic Engineering

Head of School: Dr. Alan Morrison

Supervisors: Prof. Emanuel Popovici
Dr. Colin C. Murphy
Prof. Andriy Temko

2026

Contents

List of Figures v
List of Tables ix
List of Acronyms xi
List of Publications xv
Competitions xvi
Acknowledgements xvii
ABSTRACT xix

1 Introduction **1**
1.1 Motivation 1
1.2 Aims of the Thesis 3
1.3 Contributions of the Thesis 4
1.4 Thesis Outline 7

2 Security in the IoT: Background **14**
2.1 Internet of Things (IoT) 14
2.2 Growing Cyber Concerns 16
2.3 Technical Aspects of IoT-Specific Attacks 18
2.4 Intrusion Detection Systems (IDS) for the IoT 22
2.4.1 Signature-Based IDS 22
2.4.2 Anomaly-Based IDS 24
2.5 Available Datasets 27
2.5.1 Evolution of IoT Datasets 28
2.6 Alternatives to Network Traffic 29
2.7 Summary 32

3 Host-Based Intrusion Detection for IoT Devices Using Convolutional Neural Networks **33**
3.1 Introduction 33
3.1.1 Architectural Assumptions 35
3.2 Background: Neural Network Foundations 36
3.2.1 Artificial Neural Networks 36
3.2.2 Training Neural Networks: Loss and Optimisation 37
3.2.3 Convolutional Neural Networks 38
3.2.4 Advantages of CNNs for Time-series and Structured Signals 39
3.3 Testbed and Data Creation 40
3.3.1 Normal Behaviour 42
3.3.2 Data Sampling 43
3.3.3 Injection of Synthetic Anomalies 44
3.3.4 Data Processing 47
3.4 CNN Architecture 49
3.4.1 Training—Quantisation-Aware 50
3.5 Overall Results 51
3.6 Further Analysis of Best Models 53

3.6.1 Metric Definition . . . 53
3.6.2 Best Models Comparison . . . 55
3.7 Implementation Results . . . 57
3.7.1 Post-Quantisation Evaluation . . . 58
3.7.2 Model Size and Memory Footprint . . . 58
3.7.3 Runtime Performance and Power Consumption . . . 58
3.8 Summary . . . 59

4 Attacks on IoT—An Analysis of the Signatures of Attacks in Device Power Consumption 62
4.1 Introduction . . . 62
4.1.1 Architectural Assumptions . . . 64
4.2 Testbed Creation . . . 65
4.3 Data Collection . . . 67
4.3.1 Normal Power Behaviour . . . 68
4.4 Attacks . . . 69
4.4.1 Reconnaissance . . . 70
4.4.1.1 Port Scanning . . . 70
4.4.2 Brute-force . . . 73
4.4.2.1 SSH Brute-force . . . 73
4.4.2.2 Correct-Key Event . . . 75
4.4.3 Denial of Service (DoS) . . . 76
4.4.3.1 SYN Flood . . . 77
4.5 Capture the Flag Scenario . . . 79
4.5.1 Reconnaissance . . . 80
4.5.1.1 Ping . . . 80
4.5.1.2 Port Scanning . . . 82
4.5.1.3 Brute-force . . . 82
4.5.2 Infiltration . . . 86
4.5.3 Exploration . . . 87
4.5.4 Exfiltration . . . 89
4.5.5 Withdrawal . . . 90
4.6 Dataset Creation . . . 92
4.7 Discussion . . . 94
4.7.1 Attack Behaviour and Detectability in Power Data . . . 95
4.8 Summary . . . 96

5 Convolutional Autoencoder for Self-supervised Intrusion Detection in IoT Devices 99
5.1 Introduction . . . 99
5.1.1 Architectural Assumptions . . . 100
5.2 Background: Autoencoders and Convolutional Feature Reconstruction . . . 101
5.2.1 Autoencoders . . . 101
5.2.2 Latent Representations and Anomaly Detection . . . 103
5.2.3 Convolutional Autoencoders . . . 103
5.2.4 Dropout and Controlled Reconstruction Capacity . . . 105

5.3 Data Preprocessing . . . 105
5.4 Architecture and Training . . . 108
5.4.1 Architecture . . . 108
5.4.2 Training . . . 110
5.4.2.1 Metric: Area Under the ROC Curve (AUC). . . 112
5.4.2.2 Results . . . 113
5.4.3 Reconstruction Examples . . . 117
5.4.4 MSE Exploration . . . 120
5.5 Dropout Architecture and Training . . . 122
5.5.1 Architecture . . . 122
5.5.2 Training . . . 124
5.5.3 Results . . . 124
5.5.4 Reconstructions with Dropout . . . 125
5.5.5 MSE Exploration With Dropout . . . 130
5.6 Discussions . . . 131
5.6.1 Self-supervised AI . . . 131
5.6.2 Human-in-the-Loop . . . 132
5.6.3 Explainability of the AI . . . 132
5.7 Summary . . . 133

6 Compression Enhanced Masked Autoencoders for Self-Supervised Intrusion Detection in IoT Devices 136
6.1 Introduction . . . 136
6.1.1 Architectural Assumptions . . . 138
6.2 Background: Transformers, Conformers, and Masked Autoencoders . . . 139
6.2.1 Transformers . . . 139
6.2.2 Self-Attention and Multi-Head Attention . . . 141
6.2.3 Tokens and Positional Encoding . . . 142
6.2.4 Layer Normalisation and Feed-Forward Networks . . . 143
6.3 Conformer Architectures . . . 143
6.3.1 Convolutional Module . . . 144
6.4 Masked Autoencoders (MAEs) . . . 145
6.4.1 Architecture and Objective . . . 146
6.5 CompMAE Architecture Design . . . 147
6.5.1 CompMAE Encoder . . . 147
6.5.2 Masking . . . 148
6.5.3 Input and Reconstruction Targets: Training and Inference 149
6.5.4 Self-Supervised Learning . . . 151
6.5.5 Zero-Masking Inference . . . 151
6.5.6 Multi-Headed Self-Attention Module . . . 152
6.5.7 Convolution Module . . . 153
6.5.8 Compression Module . . . 154
6.5.9 Feed-Forward Module . . . 158
6.6 Training . . . 159
6.6.1 Dragon_Pi Dataset . . . 159

6.6.2 Automatic Mixed Precision 159
6.6.3 CONFORMER–MAE vs. COMPMAE Training 160
6.7 Results on Dragon_Pi 161
6.7.1 Masked Reconstruction Performance 164
6.7.2 Reconstruction Performance 167
6.7.2.1 Over-Time Performance 174
6.8 Ablation 177
6.8.1 CONFORMER–MAE versus COMPMAE 177
6.8.2 Conformer Module Depthwise Kernel 177
6.8.3 Compressor Module Depthwise Kernel 179
6.8.4 Compressor Module Attention Heads 180
6.8.5 Compression Strategy Ablation 181
6.8.6 Compressor Component Ablation 182
6.8.7 Residual Dropout 184
6.8.8 Masking Strategy and Ratio 186
6.8.9 Model Depth and Capacity Considerations 187
6.9 Edge Implementation Results: Preliminary Investigation 187
6.9.1 Summary 189

7 Conclusions and Future Work **191**
7.1 Aim 1: Power-Based Intrusion Detection Proof of Concept . . . 191
7.2 Aim 2: Synthetic vs. Real Anomalies 192
7.3 Aim 3: Model Evolution and Innovation 194
7.4 Conclusions and Future Directions 196
7.5 Limitations and Scope of the Study 198
7.5.1 What Has Been Demonstrated 198
7.5.2 What Has Not Been Fully Ascertained 198
7.5.3 Sensitivity to Real-World Variability 199

References **201**

List of Figures

2.1 A generic three-layer IoT architecture illustrating the perception (sensing), network (transport), and application (services) layers, along with representative devices, gateways, edge resources, cloud platforms, and end-users. 15
2.2 High-level taxonomy of IoT attack types examined in this Chapter. 19
2.3 Timeline showing the progression of widely used IDS and IoT security datasets. 28
2.4 Example time series power segment of normal data. Specifically of a burst of Bluetooth communication on the DragonBoard 410c (seen around 72.1s to 73.1s). 30

3.1 Schematic of a simple artificial neural network (ANN) with four input neurons, a single hidden layer of six neurons, and three output neurons. 37
3.2 Illustration of a convolutional neural network (CNN) operating on an image input. As the network deepens, the spatial resolution is progressively reduced through convolutions and pooling before the resulting features are passed to a fully connected classifier. 40
3.3 High-level framework showing the device to be protected and the IDS board monitoring its power usage. 41
3.4 Normal power data sampled from the MAX78000 during six consecutive inferences of the road-sign classification model. Each peak corresponds to combined CPU and NPU activity. 43
3.5 Road-sign CNN inference power traces. (Left) Single inference showing CPU loading (A) and NPU spike (B). (Right) Ten overlaid inferences demonstrating high repeatability. 44
3.6 Road-sign CNN inference traces with injected anomalies. Spike, short-noise, and long-noise disturbances are shown alongside constant and drift anomalies. Red annotations indicate injected regions. 46
3.7 Road-sign CNN inference represented as grayscale images. The normal inference (top-left) is shown alongside injected anomaly types: spike, short-noise, long-noise, and drift. Distortions introduced by each anomaly are visually apparent in the corresponding panels. 48
3.8 Kaggle *Cats vs Dogs* CNN architecture, supplied with the MAX78000. 49

4.1 IoT power data acquisition framework used in this Chapter, illustrating the balance between human-in-the-loop oversight and automated (AI-driven) security analysis. 67
4.2 Normal power signatures for the DragonBoard and Raspberry Pi. Note the strong similarity in signature shape between the two devices despite the amplitude difference. 69

4.3 Each Nmap T-Switch behaviour for DragonBoard versus Raspberry Pi (data represented using the “summer” colormap [119]). 72
4.4 Incorrect Key Events (IKE) across Hydra `t`-switch settings for the DragonBoard and Raspberry Pi (data visualised using the “autumn” colormap [119]). 75
4.5 Correct Key Events (CKE) across all Hydra `t`-switch configurations for the DragonBoard and Raspberry Pi (data visualised using the “autumn” colormap [119]). 77
4.6 Comparison of SYN flood attacks on the DragonBoard and Raspberry Pi (data visualised using the “autumn” colormap [119]). . 79
4.7 Comparison of ping behaviour for the DragonBoard and Raspberry Pi (data visualised using the “winter” colormap [119]). . 81
4.8 Comparison of Nmap behaviour during the CTF between the DragonBoard and Raspberry Pi (data visualised using the “summer” colormap [119]). 83
4.9 Start of the SSH brute-force phase during the CTF, comparing DragonBoard and Raspberry Pi (data visualised using the “autumn” colormap [119]). 84
4.10 Comparison of Correct-Key Events during the CTF for the DragonBoard and Raspberry Pi (data visualised using the “autumn” colormap [119]). 85
4.11 Infiltration during the CTF for the DragonBoard and Raspberry Pi (data visualised using the “hot” colormap [119]). 87
4.12 Comparison of attacker exploration activity on the DragonBoard and Raspberry Pi (data visualised using the “hot” colormap [119]). 88
4.13 Exfiltration during the CTF for the DragonBoard and Raspberry Pi (data visualised using the “gist_gray” colormap [119]). 90
4.14 Withdrawal of the attacker following CTF completion for the DragonBoard and Raspberry Pi (data visualised using the “hot” colormap [119]). 91
4.15 Example DataFrame header from the dataset showing hierarchical annotation structure. 93
4.16 Example of annotation accuracy from the dataset. 94

5.1 Simple AE architecture comprising an encoder, a low-dimensional bottleneck (latent representation), and a decoder. Note the compression of the latent space. 102
5.2 Convolutional autoencoder (CAE) with a convolutional encoder, a fully connected AE bottleneck, and a convolutional decoder. . 104
5.3 Preprocessing pipeline: (*A*) raw data at 48,828 Hz, (*B*) resampled data at 2,048 Hz, and (*C*) MAF-filtered data at 2,048 Hz with MAF size 15. 107
5.4 Convolutional Autoencoder (CAE) architecture used in this Chapter. 109

5.5 Performance of the CAE (1024) model: (*A*) raw AUC without smoothing; (*B*) AUC versus MAF window length; (*C*) ROC curve at optimal MAF 16; and (*D*) event-based anomaly detection rate (GDR) versus false alarms per hour. 116
5.6 Effect of the Moving Average Filter (MAF) on the mean squared error (MSE) signal. MAF smoothing broadens anomaly responses and reduces false positives in normal regions. 116
5.7 Reconstructions using the CAE (1024) model (DragonBoard (left) vs. Raspberry Pi (right)). (*A*,*B*) normal behaviour; (*C*,*D*) `Nmap_T3` anomalies; (*E*,*F*) `Hydra_T1` anomalies; (*G*,*H*) `Hydra_T16` anomalies; (*I*,*J*) infiltration traces. 119
5.8 Exploration of mean squared error (MSE) behaviour in the CAE (1024) model. (*A*) Window-wise MSE across 101 segments (1360–1460) highlighting one low-amplitude anomaly (red) and two normal Bluetooth windows (green). (*B*,*C*) normal Bluetooth reconstructions. (*D*) Anomalous reconstruction (window 1410), showing unexpectedly low error due to partial overfitting. . . . 121
5.9 Revised CAE architecture incorporating dropout regularisation. Dropout is applied after GELU activations in both convolutional and dense layers, with dropout probabilities of 0.1 and 0.5, respectively. 123
5.10 Performance of the dropout-based CAE (1024) model. (*A*) Raw MSE AUC; (*B*) AUC versus MAF window length; (*C*) ROC curve at optimal MAF length; and (*D*) GDR performance showing detection versus false-alarm rate. 126
5.11 Reconstructions using the dropout-regularised CAE (1024) model (DragonBoard (left) vs. Raspberry Pi (right)). (*A*,*B*) normal behaviour; (*C*,*D*) `Nmap_T3` anomalies; (*E*,*F*) `Hydra_T1` anomalies; (*G*,*H*) `Hydra_T16` anomalies; (*I*,*J*) infiltration traces. 129
5.12 MSE analysis of the dropout-based CAE (1024) model. (*A*) Overall MSE sequence; (*B*,*C*) Bluetooth behaviour segments; and (*D*) `Nmap_T3` anomaly. The dropout model produces a distinct separation between normal and anomalous error magnitudes, enhancing robustness. 131

6.1 Overview of a standard encoder–decoder Transformer architecture as presented in [139]. A classification head attached to the decoder outputs can be used for tasks such as sentence classification. 140
6.2 Overview of an encoder-only Transformer architecture composed of stacked self-attention and position-wise feed-forward layers. 141
6.3 Diagram of a standard Conformer block combining macaron-style FFNs, MHSA, and a convolutional module within a residual and LayerNorm framework. 144
6.4 Illustration of an MAE with an asymmetric encoder–decoder design. 146

6.5 Overview of the CompMAE encoder architecture. 148
6.6 Schematic of MHSA within a CompMAE block. Each head models contextual dependencies across masked and unmasked tokens before convolutional refinement. 153
6.7 Conformer-style convolution module used in both Conformer–MAE and CompMAE. 154
6.8 Compression module used in CompMAE. Note the similarities to the convolutional module, albeit inverted in key areas. 155
6.9 Examples of disrupted attention patterns under anomalous inputs within the compressed latent space (attention for a specific token). 157
6.10 Feed-forward module used in CompMAE, comprising a linear expansion, Swish (SiLU) activation, dropout, and linear compression. 158
6.11 Original normal Bluetooth segment used for masked-reconstruction evaluation. 164
6.12 Reconstruction of the same normal Bluetooth segment under varying inference masking ratios (rows: 50%, 75%, 90%). Left: masked inputs. Right: corresponding reconstructions. 166
6.13 Reconstruction of *normal* power behaviour across devices and activities. Left column: Conformer–MAE baseline. Right column: CompMAE. 168
6.14 Reconstruction behaviour of Conformer–MAE (left column) and CompMAE (right column) on selected anomalous DragonBoard events from the Dragon_Pi dataset. 171
6.15 Reconstruction behaviour of Conformer–MAE (left column) and CompMAE (right column) on selected anomalous Raspberry Pi events from the Dragon_Pi dataset. 173
6.16 Reconstruction behaviour of the Conformer–MAE baseline over 200 windows. Normal windows are reconstructed accurately, but the Nmap port-scan anomaly is also reproduced with high fidelity, yielding an MSE trace in which the anomalous response is almost indistinguishable from normal operation (the anomaly lies within the region marked in red, while normal Bluetooth activity appears in the regions highlighted in green). 175
6.17 Reconstruction behaviour of CompMAE across 200 windows. Normal windows are reconstructed with high consistency and consequently low error, as seen in adjacent normal windows A and D (highlighted in green), whereas the Nmap anomaly produces a strong divergence and a large, easily identifiable peak in the MSE trace (annotated by the red arrow). 176

List of Tables

3.1 *Cats vs Dogs* CNN architecture showing specific layer operations and output dimensions. 49
3.2 Final validation accuracy and corresponding training epoch for each trained model. 52
3.3 Generalisation performance of each trained model across all dataset test sets (post-quantisation). 53
3.4 Per-class performance of Comp3 across all test sets. 56
3.5 Per-class performance of AIO across all test sets. 56

4.1 Breakdown of labelled data files by attack type and platform. . 92

5.1 Best-performing epoch for each latent dimension (lower corresponds to greater compression). 114
5.2 Best-performing epoch and performance summary for each latent compression level (with dropout). 125

6.1 Conformer vs. CompMAE hyperparameters. 161
6.2 Performance comparison between the Conformer (CONFORMER–MAE) and CompMAE (COMPMAE) models across various subsets of the DRAGON_PI dataset. Metrics are AUC and detection delay in seconds. 162
6.3 Comparison of Conformer and CompMAE with varying depthwise convolution kernel sizes. 178
6.4 Effect of compressor *internal* depthwise kernel size (k_{int}) with fixed external kernel $k_{\text{ext}} = 17$. 179
6.5 Effect of compressor attention heads H (and resulting per-head dimension $d_{\text{head}} = D/H$, with D=32) on performance. 180
6.6 Comparison of compression strategies within the compressor block. Compression ratio refers to the total reduction applied to the encoder dimension ($D = 128$). 182
6.7 Ablation of compressor components, ordered by CTF-only AUC. 184
6.8 Effect of residual dropout (external kernel size fixed at 17). . . 185
6.9 Effect of masking strategy and ratio on reconstruction-based anomaly detection performance. 186
6.10 Inference performance and power across server and edge devices. For server-class hardware, reported values reflect maximum power spec. 188

I, DOMINIC LIGHTBODY, certify that this thesis is my own work and has not been submitted for another degree at University College Cork or elsewhere. All the assistance received in preparing this thesis and sources have been acknowledged. I have read and understood the regulations of University College Cork concerning plagiarism and intellectual property.

DOMINIC LIGHTBODY

List of Acronyms

6LoWPAN IPv6 over Low-Power Wireless Personal Area Networks.

AE Autoencoder.

AI Artificial Intelligence.

ANNs Artificial Neural Networks.

APIs Application Programming Interfaces.

AUC Area Under the Curve.

BLE Bluetooth Low Energy.

CAE Convolutional Autoencoder.

CART Classification and Regression Trees.

CIA Confidentiality, Integrity, and Availability.

CISA Cybersecurity and Infrastructure Security Agency.

CKE Correct Key Event.

CNNs Convolutional Neural Networks.

CPUs Central Processing Units.

CTF Capture-the-Flag.

CVAE Conditional Variational Autoencoder.

DAEs Denoising Autoencoders.

DC Direct Current.

DDoS Distributed Denial-of-Service.

DES Data Encryption Standard.

DNS Domain Name System.

DoS Denial-of-Service.

DPA Differential Power Analysis.

DRs Detection Rates.

EEG Electroencephalography.

EV Electric Vehicle.

FARs False Alarm Rates.

FastGRNN Fast Gated Recurrent Neural Network.

FCNN Fully Convolutional Neural Network.

FFN Feed-Forward Network.

FN False Negative.

FP False Positive.

FPR False Positive Rate.

GELU Gaussian Error Linear Unit.

GPU Graphics Processing Unit.

GRUs Gated Recurrent Units.

GTSRB German Traffic Sign Recognition Benchmark.

HIDS Host-based Intrusion Detection System.

HSE Health Service Executive.

I/O Input/Output.

ICMP Internet Control Message Protocol.

IDSs Intrusion Detection Systems.

IKEs Incorrect Key Events.

IIoT Industrial Internet of Things.

IoCs Indicators of Compromise.

IoE Internet of Everything.

IoT Internet of Things.

IP Internet Protocol.

IT Information Technology.

JTAG Joint Test Action Group (JTAG) Interface.

KDD Knowledge Discovery and Data Mining.

KNN k-Nearest Neighbours.

LDA Linear Discriminant Analysis.

LOF Local Outlier Factor.

LR Logistic Regression.

LSTM Long Short-Term Memory.

LVQ Learning Vector Quantisation.

MAE Mean Absolute Error.

MEMS Microelectromechanical Systems.

MHA Multi-Head Attention.

MHSA Multi-Head Self-Attention.

MitM Man-in-the-Middle.

MLP Multilayer Perceptron.

MSE Mean Squared Error.

NAS Neural Architecture Search.

NB Naive Bayes.

NCI National College of Ireland.

NCISS National Cyber Incident Scoring System.

NIDS Network-based Intrusion Detection System.

NPU Neural Processing Unit.

NUI National University of Ireland.

NaOH Sodium Hydroxide.

OC-SVM One-Class Support Vector Machine.

PCA Principal Component Analysis.

PII Personally Identifiable Information.

QAT Quantization-Aware Training.

RAM Random Access Memory.

ReLU Rectified Linear Unit.

RF Random Forest.

SCADA Supervisory Control and Data Acquisition.

SGD Stochastic Gradient Descent.

SNORT Snort Network Intrusion Detection System.

SSH Secure Shell.

SoC System-on-Chip.

SVM Support Vector Machine.

SYN Synchronize (TCP SYN flag).

TCP Transmission Control Protocol.

TN True Negative.

TP True Positive.

TPR True Positive Rate.

TU Technological University.

UART Universal Asynchronous Receiver-Transmitter.

UPSs Uninterruptible Power Supplies.

USD United States Dollar.

WEF World Economic Forum.

WSN Wireless Sensor Network.

List of Publications

The majority of this work, specifically Chapters 3 - 6, has been published in peer reviewed conferences and journals:

First Author:

- **Dominic Lightbody**, Duc-Minh Ngo, Andriy Temko, Colin C Murphy, Emanuel Popovici. **Compression Enhanced Masked Autoencoders for IoT Intrusion Detection**. In: 2025 10th International Workshop on Advances in Sensors and Interfaces (IWASI), 1–6. (Chapter 6).
- **Dominic Lightbody**, Duc-Minh Ngo, Andriy Temko, Colin C Murphy, Emanuel Popovici. **Dragon_Pi: IoT Side-Channel Power Data Intrusion Detection Dataset and Unsupervised Convolutional Autoencoder for Intrusion Detection**. In: Future Internet 16 (3), 88. (Chapter 5).
- **Dominic Lightbody**, Duc-Minh Ngo, Andriy Temko, Colin C Murphy, Emanuel Popovici. **Attacks on IoT: side-channel power acquisition framework for intrusion detection**. In: Future Internet 15 (5), 187. (Chapter 4).
- **Dominic Lightbody**, Duc-Minh Ngo, Andriy Temko, Colin C Murphy, Emanuel Popovici. **Host-based intrusion detection system for iot using convolutional neural networks**. In: 2022 33rd Irish Signals and Systems Conference (ISSC), 1-7. (Chapter 3).

Co-author:

- Duc-Minh Ngo, **Dominic Lightbody**, Andriy Temko, Colin C Murphy, Emanuel Popovici. **A Scalable Security Approach in IoT Networks: Smart Contracts and Anomaly-based IDS for Gateways using Hardware Accelerators**. In: IEEE Access 2024.
- Duc-Minh Ngo, **Dominic Lightbody**, Andriy Temko, Colin C Murphy, Emanuel Popovici. **Enhancing Security and Scalability in IoT Networks Applying Blockchain Technology**. In: Cooperative Design, Visualization, and Engineering: 21st International Conference, CDVE 2024, Valencia, Spain. Proceedings (p. 265). Springer Nature.
- Duc-Minh Ngo, **Dominic Lightbody**, Andriy Temko, Cuong Pham-Quoc, Ngoc-Thinh Tran, Colin C Murphy, Emanuel Popovici. **Network Attack Detection on IoT Devices Using 2D-CNN Models**. In: International Conference on Intelligence of Things (pp. 237-247). Cham: Springer Nature Switzerland.
- Duc-Minh Ngo, **Dominic Lightbody**, Andriy Temko, Cuong Pham-Quoc, Ngoc-Thinh Tran, Colin C Murphy, Emanuel Popovici. **HH-NIDS: heterogeneous hardware-based network intrusion detection framework for IoT security**. In: Future Internet, 15(1), 9.

Competitions

- **MAX78000 AI Design Contest**[1]: In 2021, our team placed first in the MAX78000 AI Design Contest, a global competition showcasing innovative applications of AI in embedded systems. My contribution closely followed the work presented in Chapter 3 and published in [1], implementing a lightweight CNN for anomaly detection in the power time series of IoT devices on the MAX78000's on-chip neural processing unit (NPU). Team members: Dominic Lightbody, Duc-Minh Ngo, and Cathal Kelly.
- **Kaggle AI competition**[2]: In 2022, we won a bronze medal in the *Sartorius – Cell Instance Segmentation* competition hosted on Kaggle, a leading platform for AI and data science challenges. The task focused on applying deep learning to accurately segment individual cells in high-resolution microscopic images. We implemented a Swin Transformer-based model to obtain competitive results, demonstrating our ability to design and train advanced architectures for complex computer vision problems. Team members: Dominic Lightbody, Duc-Minh Ngo, Emanuel Popovici, Andriy Temko.
- **Kaggle AI competition**[3]: In 2025, we achieved a bronze medal in the *MAP – Charting Student Math Misunderstandings* Kaggle competition. This challenge focused on leveraging large language models to identify the specific mathematical misconception underlying a student's written explanation of their answer. Team members: Dominic Lightbody, Oksana Semonova, Andriy Temko.
- **Kaggle Expert rank achieved**[4]: Following these two bronze-medal performances, I attained the *Expert* rank on Kaggle. At the time of attaining the Expert rank, this placed me within the top 5,000 Kaggle members out of more than 200,000 users. Kaggle rankings are dynamic and change over time, and maintaining such a position requires continual participation in competitions.
- **Second-place publication of the year**: The work presented in Chapter 4 and published in [2] was awarded joint second place in the *Publication of the Year* competition within the School of Engineering.

[1] `https://www.elektormagazine.com/ai-contest-max78000`

[2] `https://www.kaggle.com/competitions/sartorius-cell-instance-segmentation`

[3] `https://www.kaggle.com/competitions/map-charting-student-math-misunderstandings`

[4] `https://www.kaggle.com/dominiclightbody`

Acknowledgements

First, I would like to thank God for gifting me every strength throughout my PhD journey through His loving Son, Jesus Christ. None of what I am, which is good, is without Him.

I would like to thank my parents and family for their support throughout my PhD. In particular, my mother, Deirdre, for her constant support and for always being there for me beyond the PhD, and my father, Gordon, with whom I spent many Friday afternoons in our favourite pub discussing our shared interest in machine learning with great enthusiasm.

I would like to thank my primary supervisor, Professor Emanuel Popovici. Emanuel created a wonderful environment in which we could work and grow, and many of my best days were spent in his lab. He always brought enthusiastic ideas and maintained a strong focus on enjoyment within the Embedded Systems @ UCC lab. He consistently put his students first and gave me a fantastic opportunity to develop both personally and professionally.

I would also like to thank my secondary supervisors, Colin Murphy and Andriy Temko. Throughout years of PhD meetings, Colin always brought enthusiasm, exceptional editing skills (which I frequently relied on), and, importantly, a caring pastoral approach. The long meetings that Colin, Emanuel, Minh, and I shared—full of new ideas and discussions—would always conclude with Colin checking in on how we were doing.

Andriy brought deep technical expertise to my PhD. I can honestly say I do not personally know anyone with greater ability in the field of machine learning. He provided invaluable advice, mentored me through Kaggle competitions during the PhD, and later welcomed me into his team at Qualcomm Cork.

I would like to thank the Qualcomm GMRD team, and especially GMRD Cork. The final year of my PhD was spent transitioning from an internship to a full-time role, and my colleagues quickly became people I truly value working alongside every day.

I would like to thank Duc-Minh Ngo, my partner throughout the PhD. From long days playing cards in the lab to organising weekly pool sessions and bringing together postgraduates from across the department, it was a real gift to have such a close friend to work alongside during those years—and to continue that journey together at Qualcomm.

I would like to thank all postgraduate students, past and present, in the UCC Electrical Engineering department. There was never a dull moment with so many friends around, and we shared many great times packed into the tiny lunch room playing cards.

I would like to thank my examiners for their time, feedback, and constructive suggestions, which have helped improve this thesis.

I would also like to acknowledge the use of ChatGPT as a supportive tool during the preparation of this thesis. It was used to assist with refining written expression, improving clarity, and checking spelling and grammar. Its use supported the writing process and helped enhance the overall presentation of the work.

ABSTRACT

The Internet of Things (IoT) has transformed modern infrastructure, but its pervasive connectivity has introduced significant security challenges. Many IoT devices lack adequate protection, leaving critical systems vulnerable to compromise. Intrusion detection systems (IDS) for IoT are often focused on network traffic analysis, which can overlook alternative fundamental behavioural signals present within electronic devices, such as device-level side-channel data. One such signal is device power consumption—a ubiquitous, low-level indicator of computation. This thesis investigates side-channel power data as a modality for IoT intrusion detection, progressing from lightweight supervised models to advanced self-supervised architectures trained directly on unlabelled power traces.

The research begins by demonstrating the feasibility of power-based intrusion detection using lightweight Convolutional Neural Networks (CNNs) trained on time-series power traces with synthetically injected anomalies. Implemented on an ultra-low-power microcontroller, this early system shows that IoT power behaviour contains distinct, repeatable structure that can be detected reliably at the edge.

To address the limitations of synthetic anomalies, the thesis develops a comprehensive dataset of real IoT attack power traces using Raspberry Pi and DragonBoard-based testbeds (Dragon_Pi). Multiple attack classes—including reconnaissance, brute force, and denial of service—were executed, alongside a realistic multi-stage "capture-the-flag" scenario. The resulting dataset reveals the rich, multi-phase temporal structure of genuine intrusion behaviour and highlights the value of side-channel power data for forensic analysis.

Building on these insights, the work transitions to self-supervised learning trained exclusively on normal behaviour. A Convolutional Autoencoder (CAE) and a dropout-enhanced variant establish that reconstruction-based anomaly detection can identify unseen attacks while remaining interpretable through reconstruction-error visualisation.

The final contribution introduces CompMAE—a Compression-Enhanced Masked Autoencoder that integrates Conformer-style local–global modelling with a novel compression module tailored for anomaly detection. Trained on real normal power data, CompMAE sharply amplifies reconstruction errors for anomalous inputs, leading to far more detectable anomalies than the CAE

models on the Dragon_Pi dataset. Despite its architectural sophistication, it remains lightweight and compatible with embedded hardware, bridging modern deep learning methods with practical IoT security constraints.

Collectively, this thesis establishes side-channel power monitoring as an under-explored yet viable modality for IoT intrusion detection. It presents a clear methodological progression—from supervised CNNs on synthetic data, to real attack characterisation, to initial self-supervised experiments with CAEs, to modern transformer-based self-supervised models—culminating in a robust, explainable, and edge-deployable IDS framework.

Chapter 1

Introduction

1.1 Motivation

With an ever-expanding global attack surface, recent years have witnessed an unprecedented surge in both the volume and sophistication of cyberattacks. Reports indicate that 2020 saw a 358% increase in malware activity compared to the previous year [3]. This trend persisted well into 2021 and 2022, during which over 236 million ransomware attacks were recorded globally. The World Economic Forum (WEF) estimates that if cybercrime were considered a nation, it would represent the world's third-largest economy, with global damages exceeding USD 8 trillion in 2022 alone [4]. Unfortunately, there is little evidence to suggest that this acceleration will slow in the near future.

Among the most disruptive forms of cybercrime is *ransomware*—malware that encrypts a victim's data and demands payment for its release [5]. Though often viewed as a recent phenomenon, ransomware has long plagued both private and public institutions.

The infamous WannaCry attack of May 2017 [6], which exploited a vulnerability in Microsoft Windows, infected over 300,000 computers across 150 countries and caused billions of dollars in damages [7]. More concerningly, WannaCry marked the start of a new wave of similar ransomware attacks with a distinct change in targeting strategy, deliberately focusing on hospitals, schools, and universities—critical sectors where service disruption can directly endanger human lives.

Recent analyses by the Cybersecurity and Infrastructure Security Agency (CISA)

have highlighted healthcare systems, educational institutions, and utilities as high-priority sectors [8], all of which are increasingly targeted yet remain resource-constrained. In 2022 alone, 25 ransomware attacks on U.S. healthcare systems affected roughly 290 hospitals [9], while in 2021, 67 ransomware incidents disrupted nearly one million students, with estimated economic losses exceeding USD 3.5 billion [10]. Ireland has experienced similar crises: the 2021 cyberattack on the Health Service Executive (HSE) [11] remains the most severe in the nation's history, and subsequent attacks on NUI Galway [12], NCI Dublin [13], and TU Dublin [14] further underscored the vulnerability of public services.

While ransomware exemplifies one threat vector, the broader cyber landscape has been transformed by the exponential growth of the *Internet of Things (IoT)*—a globally interconnected ecosystem of devices, sensors, and networks. By 2025, over 55 billion IoT devices are expected to be deployed worldwide [15], driving innovation across healthcare, industry, transport, and education. However, this proliferation has dramatically expanded the attack surface. Many IoT devices prioritise cost and connectivity over security [16], leaving them highly susceptible to compromise.

The consequences of insecure IoT deployments have been well documented. The Mirai botnet (2016) [17], composed of roughly 380,000 compromised IoT devices, executed one of the largest Distributed Denial-of-Service (DDoS) attacks ever recorded, peaking at 1.5 Tbps. Other incidents have demonstrated the potential for direct physical harm: the 2015 Ukrainian power grid attack [18] disrupted electricity to 225,000 customers, while vulnerabilities in connected medical and automotive systems [19] exposed patients and drivers to life-threatening risks. More recent breaches, such as Verkada's 2021 exposure of 150,000 live security camera feeds and the Florida water treatment facility compromise [20], reveal the growing intersection between cybersecurity and public safety.

Despite extensive research into Intrusion Detection Systems (IDSs), IoT environments remain uniquely challenging to secure. Signature-based IDSs perform well on known threats but struggle to detect zero-day attacks, while anomaly-based approaches can identify novel behaviour yet can suffer from higher false-positive rates [21]. Compounding these issues is the scarcity of diverse, high-quality training data—especially for AI-driven detection systems that rely on representative samples of both normal and anomalous activity.

At the same time, Artificial Intelligence (AI) has become increasingly central to intrusion detection, driving a shift from manual rule-based systems to automated, data-driven defences. Supervised AI models (signature-based) have proven effective for detecting known patterns, but semi-supervised and unsupervised learning approaches—particularly those using self-supervised paradigms—show greater promise for identifying previously unseen anomalies [22]. However, their success is limited by the availability and diversity of relevant data.

One particularly underexplored modality for IoT security is *device power data*. Every device, regardless of function or complexity, produces a measurable power consumption signature that reflects its operational state. When monitored as a side channel, these data provide unique insight into device behaviour—offering a new dimension for intrusion detection. Inspired in part by side-channel analysis techniques such as Simple Power Analysis (SPA) and Differential Power Analysis (DPA) [23], this research explores power consumption not as an attack vector, but as a *defensive signal*. By leveraging the temporal and statistical properties of power traces, this work seeks to establish power as a complementary and resilient basis for detecting anomalies and malicious behaviour in IoT systems.

1.2 Aims of the Thesis

This thesis investigates the feasibility and effectiveness of using side-channel power data as a defensive signal for intrusion detection in IoT systems, with a particular focus on developing lightweight, AI-based IDSs suitable for deployment on edge devices. More broadly, it seeks to promote the integration of power-based monitoring into the wider cybersecurity ecosystem—reframing side-channel analysis from a solely offensive discipline to a defensive one.

The research is structured to evolve progressively across chapters, reflecting both methodological development and conceptual refinement. The thesis's overarching aim is to explore effective, explainable, and edge-feasible intrusion detection via side-channel power monitoring using AI. This aim is realised through the following goals:

- **Initial Proof of Concept:** To evaluate whether useful behavioural features are present in the side-channel power consumption of IoT devices, and whether artificially injected anomalies can be reliably detected us-

ing lightweight supervised AI models implementable on embedded or edge systems—essentially ascertaining the plausibility of using power-consumption data for IoT anomaly detection.

- **Real-World Power Anomalies:** To progress from synthetic anomaly injection to realistic attack scenarios by establishing controlled IoT testbeds for capturing power data under both normal and adversarial conditions. This includes analysing how real anomalous and normal power signatures behave in practice, and assessing the extent to which power data alone can support reliable intrusion detection in realistic IoT environments.
- **Model Evolution and Innovation:** To investigate and refine the learning paradigms suitable for AI-based IDSs in IoT settings—beginning with supervised Convolutional Neural Networks (CNNs) to establish baseline feasibility, then transitioning to self-supervised approaches in response to the limitations of supervised learning for real-world anomalies. Through iterative architectural development, this aim seeks to deliver an advanced, lightweight, edge-deployable self-supervised model for IoT anomaly detection, drawing on modern designs such as the Conformer [24] and Masked Autoencoders (MAE) [25].

These aims are pursued as the thesis progresses.

1.3 Contributions of the Thesis

This thesis makes several contributions to the fields of IoT security, side-channel analysis, and AI-based intrusion detection. Broadly, it promotes the idea that device power consumption can serve not only as an offensive side-channel, but also as a practical *defensive* signal for identifying anomalous and malicious behaviour in resource-constrained IoT systems. These contributions span conceptual framing, dataset creation, edge-deployable intrusion detection, and the development of increasingly capable self-supervised learning architectures. The principal contributions are as follows:

1. **Establishing side-channel power consumption as a viable defensive modality for IoT intrusion detection.**

   A central contribution of this thesis is the demonstration that side-channel power data constitute a rich and informative behavioural signal for intrusion detection in IoT systems. While prior side-channel research has

overwhelmingly focused on offensive applications—such as key extraction or information leakage analysis—this work reframes power monitoring as a *defensive* mechanism for cybersecurity. Through both controlled and realistic experiments, the thesis shows that device power traces contain stable, repeatable structure that can be learned by AI models and exploited to distinguish normal from abnormal behaviour. In doing so, it promotes power monitoring as an underexplored but viable complement to traditional network- and host-based observability in IoT security. This reframing establishes a **device-centric security layer** that is inherent to the hardware itself, rather than relying solely on external network observations, thereby promoting a fundamentally different perspective on IoT intrusion detection.

2. **Providing an edge-feasible proof of concept for power-based intrusion detection using lightweight supervised CNNs.**

   The thesis demonstrates that power-based intrusion detection is not only conceptually plausible, but also practically feasible on constrained embedded hardware. In Chapter 3, lightweight Convolutional Neural Networks (CNNs) were trained on normal and synthetically perturbed power traces and deployed on a MAX78000 microcontroller with an integrated Neural Processing Unit (NPU). The resulting host-based intrusion detection framework achieved strong detection performance while remaining compact and energy-efficient, requiring only *55 KB* of model storage and drawing approximately *15 mW* during inference. This provides concrete evidence that AI-driven intrusion detection based on side-channel power data can operate directly at the edge, without dependence on cloud infrastructure or external packet monitoring.

3. **Creating and publicly releasing a realistic IoT side-channel power dataset of real cyberattacks.**

   A major empirical contribution of this thesis is the design, collection, annotation, and public release of the DRAGON_PI dataset, a benchmark of side-channel power traces captured from realistic IoT devices under both normal and adversarial conditions. Using Raspberry Pi and DragonBoard-based testbeds, the thesis records multiple classes of cyberattack, including reconnaissance, brute-force, denial-of-service, and a multi-stage Capture-the-Flag (CTF) scenario. The dataset addresses a gap in publicly available resources for power-based intrusion detection

and provides a realistic benchmark for future research in this domain.

4. **Characterising the manifestation of cyberattacks in device-level power consumption.**

   Beyond dataset creation, this thesis provides a detailed analysis of how cyberattacks can manifest within side-channel power data. It demonstrates that attack behaviours produce distinct, structured, and often multi-phase signatures, which vary depending on attack type, configuration, and execution context. This contribution offers both practical insight for anomaly detection and a foundation for forensic analysis, enabling the interpretation of device behaviour directly from power traces.

5. **Demonstrating the suitability of self-supervised reconstruction-based learning for power-based anomaly detection.**

   Building on the limitations of supervised learning identified in early work in the thesis, this work shows that self-supervised learning provides a more realistic and scalable approach for intrusion detection in the power domain. In Chapter 5, a Convolutional Autoencoder (CAE) and a dropout-regularised variant (CAE–DROP) were trained exclusively on normal device behaviour and used to detect anomalous activity through reconstruction error. This contribution establishes that models trained only on normal data can generalise to previously unseen attacks while avoiding reliance on scarce and highly variable labelled anomaly data. It also demonstrates an important practical advantage of reconstruction-based methods: their decisions are inherently interpretable, as analysts can compare the input signal, reconstruction, and error profile to understand why a segment was flagged as abnormal. This provides a pathway toward explainable, human-in-the-loop intrusion detection for IoT systems.

6. **Introducing COMPMAE, a novel compression-enhanced masked autoencoder for explainable and edge-compatible IoT anomaly detection.**

   The culminating technical contribution of the thesis is the development of the *Compression-Enhanced Masked Autoencoder* (COMPMAE), a novel self-supervised architecture tailored for power-based intrusion detection in IoT systems. COMPMAE integrates Conformer-style local–global modelling with a bespoke compression mechanism that constrains representa-

tional capacity in order to prioritise salient normal structure and amplify deviations caused by anomalous behaviour. Unlike standard reconstruction models that merely fail to reconstruct attacks, COMPMAE often responds to anomalous inputs with pronounced structural distortion, making abnormal behaviour more detectable both numerically and visually. This architecture therefore contributes not only improved anomaly sensitivity, but also enhanced explainability through what may be described as *reactive reconstruction*. The COMPMAE architecture is explained and ablated in detail, clearly outlining the design process for others. Despite its architectural sophistication, COMPMAE remains lightweight enough for realistic embedded deployment. In doing so, this work bridges a critical gap between modern self-supervised deep learning and the strict computational constraints of IoT devices, demonstrating that advanced architectures can be adapted for practical edge deployment.

Taken together, these contributions define a clear developmental trajectory for the thesis: from demonstrating feasibility with lightweight supervised models, to characterising real attack behaviour through a new public dataset, to advancing toward robust and explainable self-supervised architectures capable of operating under real IoT constraints. Collectively, they translate side-channel power monitoring from a conceptual research direction into a practical, edge-deployable intrusion detection framework for IoT systems.

1.4 Thesis Outline

This thesis is organised into seven Chapters, outlined as follows:

Chapter 1 Introduction

Chapter 2 Overview: *Security in the IoT: Background*.

This Chapter presents background material on security in the IoT, including notable recent attacks and prior research into IDS for the IoT. The review focuses on AI-based IDSs and organises the literature into two main categories: signature-based and anomaly-based approaches. Each category is further subdivided into works that employ traditional machine learning techniques and those that utilise deep learning methods.

Chapter 3 Overview: *Host-Based Intrusion Detection for IoT Devices Using Convolutional Neural Networks*.

This Chapter presents the first major stage of the research, introducing and validating the concept of *power-based intrusion detection* for IoT devices. It demonstrates that a device's side-channel power consumption—an inherently available, modality-independent signal—can be leveraged to identify synthetic abnormal behaviour via machine-learning-based pattern recognition. By training a lightweight CNN on time-series power traces, the study establishes a practical baseline for detecting anomalies in embedded and IoT systems, offering a complementary alternative to traditional packet-based intrusion detection.

The framework adopts a *host-based* IDS (HIDS) architecture, running directly on the protected device. This design reduces detection latency, removes reliance on persistent network connectivity, and enhances privacy by keeping all telemetry local. To support real-time inference on resource-constrained platforms, the approach leverages the MAX78000 microcontroller, which integrates a *Neural Processing Unit (NPU)* for low-power, hardware-accelerated inference. The Chapter also highlights the growing importance of dedicated edge-AI hardware in enabling practical on-device security analytics.

Synthetic anomalies are injected into recorded normal power traces from an IoT testbed to provide a controlled training environment. Experiments show that simple CNN models can learn the temporal and structural characteristics of device power profiles effectively, achieving strong classification performance despite tight computational limits. These results confirm that power-based intrusion detection is both feasible and promising as a foundation of lightweight, decentralised IoT security frameworks.

Overall, this Chapter provides a foundational proof of concept for the thesis, establishing the viability of a power-centric IDS and supplying a benchmark for the more advanced, self-supervised approaches developed in later chapters.

The work presented in this Chapter was peer-reviewed and published in [1].

Chapter 4 Overview: *Attacks on IoT—An Analysis of the Signatures of Attacks in Device Power Consumption*.

This Chapter presents the second major stage of the research, advancing beyond the controlled, synthetic-anomaly experiments of Chapter 3 toward the collection and analysis of *real power-based anomalies*. To achieve this, two re-

alistic IoT testbeds were developed—one based on a Raspberry Pi 3 Model B and the other on a DragonBoard 410c—each designed to simulate practical IoT deployment scenarios. These testbeds were subjected to a range of common cyberattacks, including reconnaissance, brute-force, and Denial-of-Service (DoS) attacks, in order to capture their corresponding side-channel power signatures.

The primary goal of this Chapter is to provide a clear, practical understanding of how cyberattacks can manifest in the time-series power data of IoT devices. Whereas most existing studies focus solely on network traffic, this work highlights that power traces, being fundamental to all computing devices, offer a rich and underexplored signal source for security analysis. Through detailed visual examination, the Chapter demonstrates how distinct attacks leave identifiable power signatures, enabling the distinction between normal and abnormal behaviour without relying on network-level data.

A key contribution of this Chapter lies in the transparency and granularity of its data analysis. Many previous works report model performance without clearly illustrating how attacks appear within the raw data itself. In contrast, this Chapter explicitly visualises the power signatures of each attack, including a comprehensive Capture-the-Flag (CTF) scenario in which multiple techniques are combined to compromise a target device. This detailed breakdown enables intuitive understanding of how different attack types influence device power consumption over time and shows how specific operational events can be pinpointed directly from power traces.

The insights gained from these experiments culminate in the design, curation, and *public release* of the DRAGON_PI dataset—a labelled benchmark of IoT side-channel power data collected under realistic normal and attack conditions. This Chapter documents the dataset's testbed design, acquisition pipeline, annotation methodology, and labelling schema, and formally introduces DRAGON_PI as a resource for the community [2]. The dataset forms the foundation for the deep-learning-based intrusion detection systems developed in subsequent chapters, bridging the gap between conceptual feasibility and operational application.

The work presented in this Chapter was peer-reviewed and published in [26].

Chapter 5 Overview: *Convolutional Autoencoder for Self-supervised Intrusion Detection in IoT Devices*.

This Chapter marks the third major stage of the research, advancing from the descriptive and practical analysis of Chapter 4 toward an AI-driven intrusion detection framework. Building on the DRAGON_PI dataset, it introduces and evaluates a *self-supervised Convolutional Autoencoder (*CAE*)* designed to detect anomalies in IoT power data without relying on labelled attack samples. This represents a critical shift in strategy—from learning explicit labelled attack patterns to learning the underlying structure of normal behaviour and identifying unseen deviations from it.

The CAE architecture combines convolutional encoding and decoding layers encapsulating a traditional Autoencoder (AE) bottleneck, enabling efficient compression and reconstruction of time-series power traces. Trained solely on normal power consumption data, the model learns the intrinsic temporal and local regularities of device behaviour. At inference time, anomalous activity manifests as elevated reconstruction error, making the approach both lightweight and interpretable. Its reconstruction-driven design also provides an unusually strong degree of explainability for an IDS model: by visualising reconstruction discrepancies, practitioners can directly observe why specific segments are flagged as abnormal.

To evaluate generalisability, the CAE is trained jointly on power traces from both the DragonBoard and Raspberry Pi platforms. This multi-device training reflects the practical heterogeneity of modern IoT deployments, where diverse devices operate concurrently under shared workloads. The model's ability to learn cross-platform structure demonstrates the potential for scalable intrusion detection that does not require per-device retraining.

Additionally, a heavily dropout-regularised variant, CAE–DROP, is introduced to improve robustness by injecting stochastic noise during training. This regularisation reduces overfitting on normal data and forces the network to learn more salient representations. As a result, CAE–DROP reconstructs normal traces reliably while failing more strongly on anomalous inputs, producing clearer separation between normal and attack behaviour. Comparative analysis between the two variants shows how increased regularisation can enhance detection performance in some self-supervised settings.

Overall, this Chapter establishes the viability of self-supervised, reconstruction-

based learning for side-channel intrusion detection. It bridges the gap between raw data exploration and the advanced Transformer-based models introduced in Chapter 6, demonstrating that compact, interpretable architectures can achieve strong detection performance on real IoT power data while remaining computationally efficient and suitable for edge deployment.

The work presented in this Chapter was peer-reviewed and published in [2].

Chapter 6 Overview: *Compression-Enhanced Masked Autoencoders for Self-Supervised Intrusion Detection in IoT Devices*.

This Chapter represents the culmination of the research, introducing the most advanced and conceptually mature model developed in the thesis—the COMPRESSION-ENHANCED MASKED AUTOENCODER (COMPMAE). Building upon the foundations established in Chapters 4 and 5, where the DRAGON_PI dataset was introduced and self-supervised convolutional approaches were explored, this Chapter integrates modern Transformer-based learning into the power-domain intrusion detection pipeline. The work reflects a clear progression from lightweight convolutional reconstruction to state-of-the-art self-supervised representation learning tailored for IoT power data.

At the core of this Chapter is the combination of three key architectural components: MAEs, Conformer blocks, and a novel compression module designed to enhance anomaly sensitivity. The baseline CONFORMER–MAE establishes the integration of convolutional and attention mechanisms under a heavy masking regime, while the enhanced COMPMAE introduces a dedicated compression module that amplifies deviations during reconstruction. Using a masking ratio of 75%, the models learn deep, context-aware representations of normal device behaviour, enabling robust reconstruction of normal sequences. The compression module strengthens the model's inherent reaction to anomalous inputs, sharply increasing the detectability of unseen attacks purely through reconstruction error.

A distinguishing feature of this work is its emphasis on anomaly *reactivity*. Unlike the CAEs presented in Chapter 5, COMPMAE not only fails to accurately reconstruct anomalies but actively exaggerates their differences from the input signal, making abnormal behaviour visually and numerically pronounced. This reactive reconstruction mechanism provides valuable insight into what the model perceives as abnormal, enhancing interpretability for security analysts. Despite its architectural sophistication, the model remains lightweight and com-

putationally feasible for deployment on edge hardware—continuing a central design philosophy of this thesis.

Comprehensive experimentation demonstrates that COMPMAE achieves high detection accuracy across multiple IoT devices while maintaining scalability and explainability. An ablation study further dissects the contribution of each architectural component, confirming the importance of the compression mechanism in achieving strong anomaly separation. Together, these findings position COMPMAE as a promising direction for future research in self-supervised and potentially multimodal IoT intrusion detection.

The work presented in this Chapter was peer-reviewed and published in [27].

Chapter 7 Overview: *Conclusion and Future Work*.

This final Chapter synthesises the key outcomes, insights, and contributions of the research presented throughout the thesis. It revisits the original aims outlined in Chapter 1 and evaluates how each was fulfilled through the progressive development of power-based intrusion detection—from early feasibility experiments with lightweight CNNs and synthetic anomalies, through the characterisation of real cyberattacks in the DRAGON_PI dataset, to the advanced self-supervised architectures introduced in later chapters.

Each aim is addressed in turn, summarising the methodologies developed, the results achieved, and the limitations encountered. The discussion highlights the thesis's central trajectory: a shift from supervised learning toward self-supervised reconstruction; a growing emphasis on explainability via reconstruction-based analysis; and a consistent focus on lightweight, edge-feasible AI suitable for real-world IoT deployments. Particular attention is given to the COMPMAE framework introduced in Chapter 6, which represents the culmination of these principles by uniting interpretability, performance, and practical deployability within a single, modern architecture.

The Chapter concludes by outlining key limitations and identifying several promising avenues for future research. These include the development of multimodal intrusion detection systems that combine power data with network or sensor telemetry, the extension of compression-aware Transformer architectures to alternative embedded platforms, and the adoption of continual and federated learning paradigms to enable distributed, privacy-preserving security across heterogeneous IoT networks. Collectively, these directions underscore

the broader significance and extensibility of the work, positioning it as a foundation for future advancements in AI-driven IoT security.

Chapter 2

Security in the IoT: Background

This Chapter provides an in-depth background on security in the Internet of Things (IoT), combining a brief survey of notable attacks from recent years with a review of contemporary Artificial Intelligence (AI)-based intrusion detection approaches. It sets the scene for the rest of the thesis by motivating the need to secure IoT ecosystems, highlighting power data as an important signal, and situating this work within the broader landscape of AI-driven security research.

2.1 Internet of Things (IoT)

The term *Internet of Things* [28] refers to the vast and continually expanding ecosystem of interconnected physical devices that communicate and exchange data over the Internet [29]. These devices range from everyday consumer electronics—such as smart thermostats, refrigerators, wearable sensors, and security cameras—to industrial machinery and medical equipment. More recently, autonomous and electric vehicles equipped with embedded sensors, processors, and communication modules have also entered the mainstream, enabling them to perceive, process, and interact with their environments. One of the core principles of the IoT is extending Internet connectivity beyond traditional computers and smartphones, allowing physical objects to participate in a global, data-driven network [30], thereby blurring the boundary between the cyber and physical worlds.

The origins of the IoT can be traced to the convergence of three technological trends: the miniaturisation of computing hardware, the proliferation of wireless communication technologies, and advances in cloud computing and

data analytics [28, 30]. The concept first emerged in industrial automation and supply-chain management, where network-attached sensors and actuators were deployed for monitoring and control. However, IoT technologies have rapidly expanded into healthcare, transportation, agriculture, energy, education, and beyond [29, 31]. The declining cost of sensors, coupled with increasingly ubiquitous connectivity, has accelerated this widespread adoption. This trend has been further supported by the rise of edge and fog computing, which bring computation closer to devices, reducing latency and bandwidth demands [32, 33].

Modern IoT systems are commonly described using a layered architecture comprising (i) a sensing and actuation layer (perception), (ii) a data transport layer (network), and (iii) an analytics and decision-making layer (application) [34]—a generic overview is provided in Figure 2.1. Data generated by IoT devices are typically transmitted to cloud- or edge-based services for analysis and aggregation, enabling intelligent control and automation across domains such as smart cities, industrial IoT (IIoT), connected healthcare, and intelligent transportation systems [31].

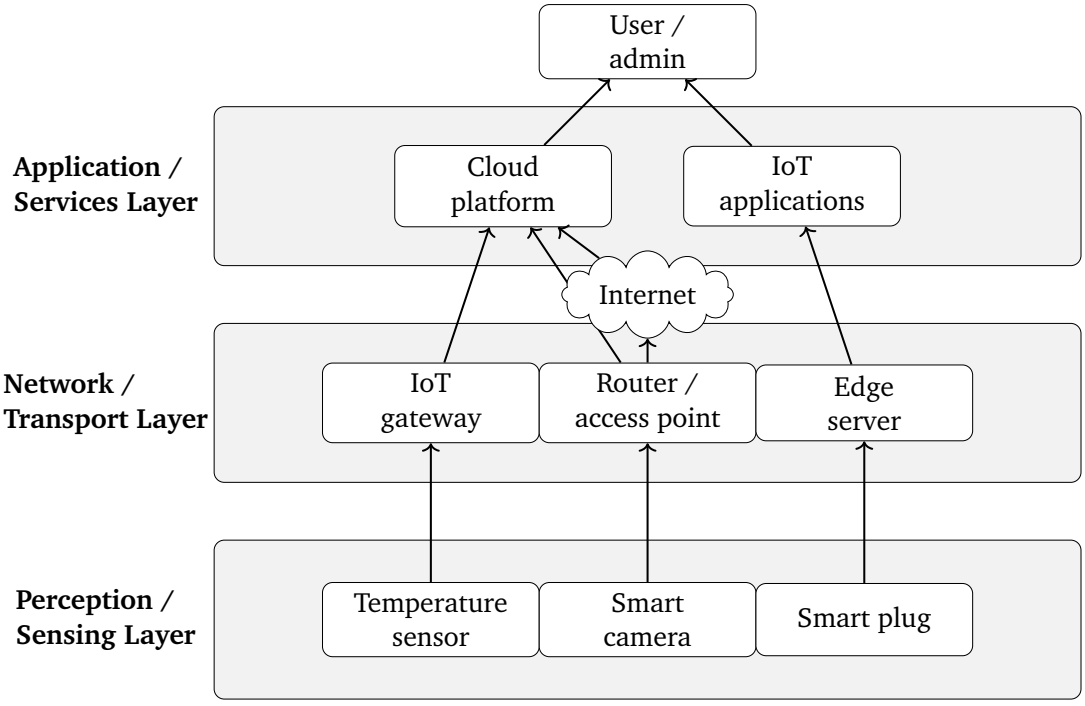


Figure 2.1: A generic three-layer IoT architecture illustrating the perception (sensing), network (transport), and application (services) layers, along with representative devices, gateways, edge resources, cloud platforms, and end-users.

The evolution of IoT has also led to the broader concept of the *Internet of Every-*

thing (IoE), which connects not only devices but also integrates people, data, and processes. In this vision, connectivity extends beyond device–device interactions to encompass human–machine and machine–machine communication, enabling context-aware, adaptive, and increasingly autonomous systems [35].

While the proliferation of IoT technologies has brought significant convenience, efficiency, and innovation, it has also introduced challenges unique to its scale and heterogeneity [36]. The vast number and diversity of IoT devices—often constrained in computational power and lacking robust security mechanisms—have created an unprecedented global attack surface. As IoT systems continue to grow in complexity and ubiquity, ensuring the confidentiality, integrity, and availability (CIA triad) [37] of interconnected devices and their users has become a critical concern. This motivates the research presented in the sections that follow.

2.2 Growing Cyber Concerns

The past decade has witnessed a marked escalation in the frequency, scale, and sophistication of cyberattacks, many of which have had tangible real-world consequences. These incidents have targeted critical infrastructure, healthcare systems, the education sector, and consumer devices alike, demonstrating that no domain is immune. While many of the most widely publicised attacks have primarily affected traditional computing environments—for example, the WannaCry [6] ransomware attack of 2017, which ushered in a new era of large-scale ransomware campaigns—the IoT has also been the target of some of the largest and most impactful cyber incidents in history. A key point to remember is that the IoT is inherently intertwined with the physical world, and in many cases attacks on IoT systems can have direct, personal effects on users.

Mirai Botnet (2016).

The Mirai botnet [17] marked a defining moment in IoT security awareness. Originating in 2016, Mirai infected approximately 380,000 Internet-connected devices—including routers, IP cameras, and digital video recorders—forming a vast botnet that launched Distributed Denial-of-Service (DDoS) attacks of unprecedented scale. In September 2016, a Mirai-based attack on Dyn, a major DNS provider, disrupted access to platforms such as Netflix, Reddit, and Twitter across North America and Europe [38]. Later attacks reached peaks of over

1.1–1.5 Tbps against OVH and 620 Gbps against security journalist Brian Krebs' website. Mirai demonstrated how millions of insecure IoT devices could be weaponised in automated, large-scale attacks.

Ukrainian Power Grid Attack (2015).

The 2015 Ukrainian power grid attack [18] remains one of the most significant examples of a cyberattack producing direct physical consequences. The attackers gained remote access to supervisory control and data acquisition (SCADA) systems, interrupting power delivery to more than 225,000 customers, including hospitals. This coordinated campaign targeted networked control components—such as breakers, serial-to-Ethernet devices, and Uninterruptible Power Supplies (UPSs)—bridging the boundary between traditional infrastructure and modern IoT-enabled systems. It is widely regarded as the first confirmed cyberattack to cause large-scale physical disruption to civilian infrastructure (power outages) [39].

St. Jude Medical Pacemaker Vulnerability (2017).

In 2017, the U.S. Food and Drug Administration (FDA) disclosed a serious vulnerability in pacemakers manufactured by St. Jude Medical [19]. The affected devices communicated wirelessly with external systems for diagnostics and firmware updates. Attackers exploiting this channel could deplete the device's battery, alter pacing parameters, or deliver unsafe electrical shocks to patients. This case emphasised the life-threatening implications of insecure medical IoT devices and prompted renewed regulatory scrutiny of connected healthcare technologies [40].

Jeep Cherokee Hack (2015).

In 2015, researchers demonstrated a remote compromise of the Jeep Cherokee SUV [19]. By exploiting vulnerabilities in the vehicle's infotainment system, attackers gained the ability to manipulate steering, braking, and acceleration remotely. In controlled experiments, researchers caused vehicles to veer off-road, leading to widespread recalls and urgent software updates. This attack served as a turning point in the automotive industry's approach to cybersecurity. More recently (2024), a vulnerability discovered in Kia vehicles [41] allowed unauthorised access to critical vehicle functions thanks to a simple website flaw, reaffirming the persistent risks in connected vehicle ecosystems.

Verkada Breach (2021).

In March 2021, cloud-based security camera provider Verkada suffered a major breach [20], exposing live video feeds from more than 150,000 cameras worldwide. These devices were deployed in hospitals, classrooms, prisons, factories, and corporate offices. Attackers gained access using publicly available administrative credentials, revealing both the fragility of cloud-managed IoT infrastructure and the consequences of poor privilege management. The incident reignited debate surrounding privacy, surveillance, and centralised control in large-scale IoT deployments.

Florida Water Treatment Facility Attack (2021).

In February 2021, attackers compromised a water treatment facility in Oldsmar, Florida. Using remote-access tools on a compromised operator workstation, they accessed SCADA systems and attempted to raise the concentration of sodium hydroxide (NaOH) in the water supply by a factor of 100. Fortunately, the intrusion was detected in time and manually reversed before reaching the public. The attack underscored the vulnerability of critical infrastructure systems that rely on outdated remote-access technologies and minimal authentication controls [42].

2.3 Technical Aspects of IoT-Specific Attacks

While many IoT attacks exploit traditional network vulnerabilities, the heterogeneity, scale, and resource-constrained nature of IoT ecosystems introduce a distinct set of attack vectors that differ significantly from those in conventional IT systems. IoT devices typically operate with limited computational and memory resources, rely on lightweight communication protocols, and often lack robust security mechanisms such as strong authentication, secure boot, or end-to-end encryption.

Moreover, these devices are frequently deployed in close physical proximity to users and embedded within safety- and privacy-critical environments. This tight coupling with the physical world amplifies the consequences of compromise: a single breach can lead to unprecedented infringements on user privacy (for example, smart home webcams exposed directly to the Internet) or manipulation of physical processes resulting in damage to equipment. These characteristics make IoT devices prime targets for attackers seeking to compromise, manipu-

late, or weaponise them at scale, magnifying the security threats of the current Internet [36], including adversaries who profit from selling access to compromised systems. Figure 2.2 outlines the taxonomy of attack types covered in this section. This list is by no means exhaustive, as the field of cybersecurity is continually evolving, driven by determined and adaptive threat actors.

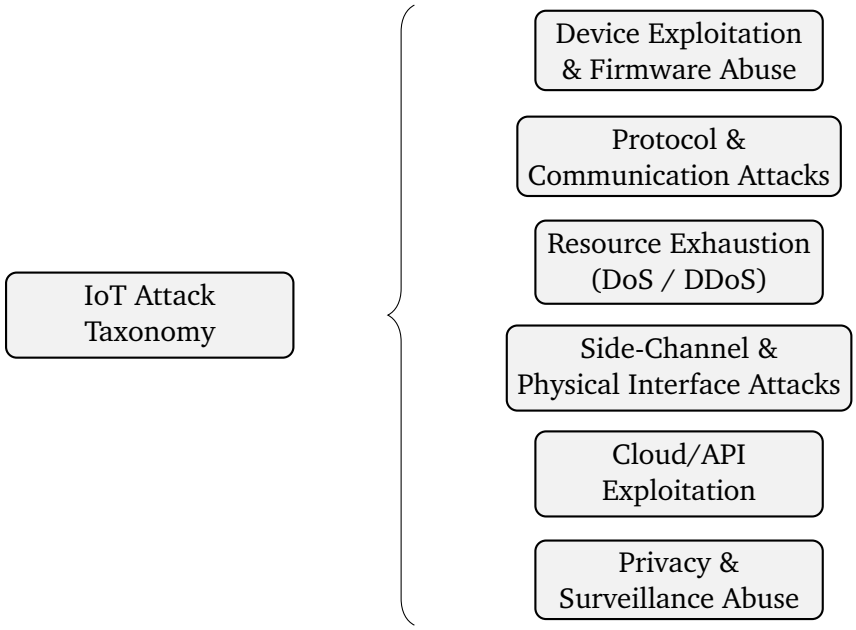


Figure 2.2: High-level taxonomy of IoT attack types examined in this Chapter.

1. Device Exploitation and Firmware Abuse.

One of the most common IoT-specific threats arises from insecure firmware and weak authentication mechanisms. Many devices ship with hard-coded or default credentials that are seldom changed by end users. For instance, it has been reported that just five passwords would grant access to around 10% of all IoT devices available online, and approximately 15% of device owners do not change the default password on newly purchased devices [43]. Attackers exploit such weaknesses to gain unauthorised access, often through remote management interfaces such as Telnet, SSH, or web dashboards. Once compromised, devices can be repurposed as entry points into larger networks or incorporated into botnets.

A prime example of this attack vector is the Mirai botnet (2016), which systematically scanned the Internet for devices using factory-default credentials, compromising hundreds of thousands of cameras and routers. The resulting botnet launched some of the largest DDoS attacks ever recorded [38], illustrating how

seemingly simple authentication oversights—by users or manufacturers—can have catastrophic, global consequences.

2. Protocol and Communication Layer Attacks.

IoT devices commonly employ lightweight protocols such as MQTT, CoAP, Zigbee, and Bluetooth Low Energy (BLE), which often lack robust encryption or authentication. This makes them vulnerable to spoofing, replay, and man-in-the-middle (MitM) attacks [44]. Attackers can intercept or modify control messages between devices and cloud services, injecting false data, disrupting sensor readings, or triggering unintended actuator responses.

In industrial and critical infrastructure environments, such manipulation can have drastic physical consequences, including falsified temperature readings, false alarms, or direct interference with operational processes. The 2015 Ukrainian power grid attack exemplifies this class of vulnerability: remote access to serial-to-Ethernet devices and supervisory control systems was leveraged to disable circuit breakers and disrupt energy delivery [39].

3. Resource Exhaustion and Denial-of-Service (DoS) Attacks.

Due to their constrained hardware, IoT devices are particularly susceptible to resource-exhaustion attacks. DoS and DDoS campaigns can overwhelm limited processing, memory, or network capacity. By flooding communication channels with excessive traffic or malformed packets, attackers can render devices unresponsive or cause repeated reboots.

Many safety- and mission-critical IoT devices are also battery-powered rather than mains-powered. In such cases, sustained communication or computation load can rapidly drain the battery [45], leading to a complete loss of availability and, in some settings, the loss of any ability to manage or update the device remotely.

The St. Jude Medical pacemaker vulnerability is a prominent example of this class of threat: researchers demonstrated that an attacker could exploit the wireless communication channel to trigger repeated, energy-intensive operations, potentially depleting the batteries of health-critical, resource-constrained devices and leading to fatal consequences for patients [40].

4. Side-Channel and Physical Interface Attacks.

Unlike traditional IT systems, IoT devices frequently interact directly with the physical environment, exposing additional attack surfaces through hardware interfaces (e.g., UART, JTAG, SPI) and side-channel leakage (e.g., power, electromagnetic radiation, timing). Attackers can extract cryptographic keys, firmware images, or operational data by measuring power consumption, electromagnetic emissions, or execution-time behaviour [23, 46, 47].

Although these attacks typically require physical proximity or direct access to the device, they can be extremely potent. Classic differential power analysis, for example, can be used to recover cryptographic keys by statistically analysing variations in power consumption during cryptographic operations [23].

5. Cloud and API Exploitation.

Modern IoT devices often depend heavily on cloud platforms for storage, analytics, and remote management, thereby extending the attack surface beyond the device itself to include cloud APIs, databases, and authentication services. Misconfigured access controls, weak authentication, or poor privilege management can lead to large-scale data breaches.

The Verkada breach (2021) [20] is a prime example of this risk: attackers exploited publicly exposed administrative credentials to obtain super-admin access to a cloud-based camera service, revealing live feeds from more than 150,000 devices worldwide. Such incidents underscore the need for secure cloud integration, granular privilege management, and continuous credential auditing for IoT devices that interact with the cloud.

6. Privacy and Surveillance Exploitation.

Many IoT devices—especially those equipped with cameras, microphones, or environmental sensors—continuously capture sensitive personal and contextual data. Weak encryption, insecure transmission protocols, or inadequate access controls can allow adversaries to eavesdrop on, infer, or exfiltrate this information [48].

The Verkada breach again exemplifies this issue, but similar risks apply to consumer-grade smart-home devices, wearable health monitors, and industrial telemetry systems. Such vulnerabilities have serious implications for privacy, surveillance, and data protection in large-scale IoT deployments.

2.4 Intrusion Detection Systems (IDS) for the IoT

An intrusion detection system (IDS) is a security mechanism that monitors system or network activity for signs of malicious behaviour, policy violations, or abnormal patterns indicative of an ongoing or attempted attack [49].

IDSs are broadly categorised into two main families: *signature-based* and *anomaly-based* approaches [50, 51]. This section outlines the operation of each family, reviews representative systems in the research literature, and highlights emerging trends—with particular emphasis on anomaly detection for the IoT.

2.4.1 Signature-Based IDS

Signature-based IDSs detect threats by matching observed activity against a repository of known attack patterns or signatures [51]. In classical security systems, these signatures are typically deterministic (e.g., rule-based patterns as used in systems such as SNORT). In this thesis, however, the term is used in a broader sense to also encompass learned representations of known attack patterns obtained through supervised machine learning, which may capture statistical rather than strictly deterministic relationships. When an event aligns with a stored signature (or learned pattern), the system raises an alert. Because detection relies on detecting known patterns, these systems typically achieve high precision and low false-positive rates for *known* threats. However, their effectiveness diminishes significantly against unseen or zero-day attacks that do not match existing signatures [50]—a particularly serious limitation in IoT ecosystems, which are both heterogeneous and rapidly evolving, and therefore exhibit highly variable behaviours during attacks and between devices.

Classical Machine Learning Approaches.

Historically, signature-based IDSs have been closely associated with *supervised learning* paradigms, where models are trained on labelled datasets containing both normal and malicious samples (e.g., *attack* vs. *normal*). Classical machine-learning algorithms such as *Naïve Bayes (NB)*, *Logistic Regression (LR)*, *Support Vector Machines (SVM)*, *K-Nearest Neighbours (KNN)*, and *Random Forests (RF)* have long been employed to classify network traffic based on these labelled examples [52]. These models are typically computationally efficient, often interpretable, and well-suited to structured network features and static attack signatures. However, they lack the expressive capacity of modern deep learn-

ing approaches, which limits their ability to capture complex, high-dimensional patterns in network traffic.

In [53], KNNs, SVMs, and a hybrid method combining *SNORT* [54] with Naïve Bayes (NB) were evaluated using the KDD99 dataset [55]. In this two-stage approach, traffic initially deemed benign by SNORT was subsequently re-analysed by the NB classifier. This hybrid configuration achieved accuracies of 93% for SVM, 94.4% for KNN, and 96.6% for the proposed SNORT–NB system, highlighting the value of integrating rule-based and statistical learning methods.

Work in [56] released a dataset of diverse modalities and attack types for Internet of Things (IoT) devices (ToN_IoT). Multiple machine-learning algorithms—including LR, Linear Discriminant Analysis (LDA), KNN, Classification and Regression Trees (CART), RF, NB, SVM, and Long Short-Term Memory (LSTM) networks—were benchmarked on ToN_IoT. Performance varied across device data: LSTM achieved 100% accuracy on fridge-sensor data, CART performed best overall (88% binary and 77% multi-class accuracy), while weaker results were observed for Light_Motion and thermostat sensors (approximately 60–66% accuracy).

In [57], a hybrid intrusion detection method (*FA-ML*) for Wireless Sensor Network (WSN)–IoT systems combined SVMs with the Firefly Algorithm [58], with parameters optimised using the Grey Wolf Optimiser. Evaluated on the NSL-KDD [59] dataset, the framework achieved 99.34% accuracy, demonstrating the potential of combining bio-inspired optimisation with supervised machine learning for securing interconnected IoT environments across critical sectors.

Deep Learning Approaches.

In recent years, research has increasingly focused on integrating *deep learning* architectures into signature-based detection frameworks [60]. Neural models such as *Artificial Neural Networks (ANNs)*, *Convolutional Neural Networks (CNNs)*, and, more recently, *Transformer*-based architectures have demonstrated superior representational capacity and generalisation compared with classical machine-learning methods. These networks can automatically learn hierarchical, spatial, and temporal features directly from raw or semi-processed data, allowing them to uncover complex correlations often missed by shallow models and to avoid costly, hand-crafted feature engineering that typically requires substantial domain expertise.

However, they also introduce notable challenges, including high computational demands, extensive data requirements, and reduced interpretability relative to some traditional approaches. A further drawback of deep learning—particularly in supervised settings—is the need for large quantities of high-quality labelled data. This is especially problematic for anomalous or attack-related behaviour, which is, by definition, rare and in some cases (e.g. [61]) impossible to obtain and reliably label in real-world environments.

In [62], a deep recurrent neural network-based IDS (NIDS) was proposed for fog computing environments. Their framework integrated a traffic-processing engine with a recurrent neural classifier trained on a balanced NSL-KDD dataset, achieving 98.27% accuracy in detecting DoS attacks—one of the most pervasive threats in IoT systems.

Similarly, [63] presented a CNN-based IDS framework optimised for IoT environments. Trained on the Bot-IoT [64] and NID [65] datasets, the model achieved accuracies of 92.85% and 99.51%, respectively. The work demonstrated an architecture capable of broad-spectrum traffic analysis across heterogeneous IoT contexts while maintaining network availability and integrity.

In [66], an edge-centric NIDS was proposed using deep recurrent architectures, such as LSTMs and Gated Recurrent Units (GRUs), for DDoS detection in IoT environments. Trained on UNSW-NB15, the models achieved 99% accuracy. When deployed on Raspberry Pi devices, their *FastGRNN* variant proved most suitable, achieving 99.95% accuracy while maintaining minimal inference time and resource overhead.

Finally, [67] designed and implemented a CNN-based architecture for IoT network anomaly detection using the IoT-23 dataset [68]. Both multi-class and binary CNN variants (1D, 2D, and 3D) were evaluated, and transfer learning was applied to derive binary classifiers from multi-class models. Across all architectures, accuracies exceeded 99%, with transfer learning demonstrating strong potential for reusing learned features between related intrusion detection tasks.

2.4.2 Anomaly-Based IDS

In contrast to signature-based approaches, anomaly-based IDSs need to model only the *normal behaviour* of a system or network and raise alerts when statistically significant deviations from this baseline occur [51]. Whereas signature-based systems enumerate known attack patterns, anomaly detectors are ex-

pressly designed to identify *unseen* or *zero-day* threats, thereby avoiding the maintenance burden of large signature repositories. This capability is commonly realised through statistical modelling or machine-learning methods that quantify departures from learned norms [69, 70]–for example, via density-based, distance-based, or reconstruction-based criteria.

Anomaly-based IDSs are naturally aligned with *unsupervised* and *self-supervised* machine-learning paradigms, in which models can be trained using only *normal* data. This preserves the "unknown" nature of attacks during training, mitigating bias toward previously observed threats and improving robustness to distributional shift. It also offers a practical advantage: normal data are abundant and relatively easy to collect, whereas anomalous data are rare and costly to curate. By training exclusively on normal behaviour, scarce anomalous samples can be reserved for validation or testing, supporting better generalisation in real-world deployments.

A well-known trade-off is that anomaly-based systems may exhibit higher false-positive rates and can underperform on *well-characterised* attack signatures, where signature-based approaches typically excel. Consequently, modern deployments increasingly adopt hybrid designs that pair anomaly detection for novel threats with signatures for known ones, balancing sensitivity and precision in heterogeneous IoT environments [50]. This hybrid strategy, however, is only feasible when attack signatures are available in sufficient quantity to train the supervised component—a condition not always met for difficult-to-obtain signals, such as side-channel data.

Classical Machine Learning Approaches.

Traditional anomaly-based IDSs rely on statistical and machine-learning techniques to uncover hidden structure or outliers in data. Commonly used algorithms include *k-means* and *hierarchical* clustering, as well as *Principal Component Analysis (PCA)* [71, 72]. For one-class settings—where only normal data are available—*One-Class SVM (OC-SVM)* [73] is also a well-established choice [22].

In [74], an IDS for IoT networks was presented using a hybrid approach that combines Particle Swarm Optimisation–based [75] gradient descent (PSO-LightGBM) for feature selection with a One-Class SVM for classification. Trained on the UNSW-NB15 dataset, the framework achieved 86.68% accuracy and demonstrated robustness in detecting small-sample attacks such as Shell-

code, Worms, and Backdoor intrusions.

[76] introduced *Passban*, a lightweight, intelligent host-based IDS (HIDS) designed for resource-constrained IoT gateways. Using Local Outlier Factor (LOF) and Isolation Forests trained on each device's normal behaviour, Passban detects diverse malicious activities, including brute-force, SYN-flood, and port-scanning attacks. Its design emphasises scalability, enabling dynamic adaptation to new threats without hardware modification—an attractive property for large-scale IoT deployments.

In a hybrid setting, [77] proposed an anomaly–misuse detection framework that combines supervised and unsupervised Optimum Path Forest models for intrusion detection, targeting both Internet and WSN traffic. Using the NSL-KDD dataset for Internet traffic and simulated 6LoWPAN traffic for WSNs, the framework achieved detection rates (DRs) of 80.95% (anomaly module) and 96.20% (misuse module), with false alarm rates (FARs) of 5.92% and 1.44%, respectively.

Deep Learning Approaches.

Recent years have witnessed rapid growth in *deep-learning-based* anomaly detection, fuelled by the ability of advanced neural architectures to automatically extract complex features and temporal dependencies directly from raw data. Such approaches have proven particularly effective in IoT security, where traditional feature engineering is often infeasible due to device diversity and data heterogeneity.

In 2018, the *N-BaIoT* model and accompanying dataset [78] were introduced to detect compromised IoT devices infected by the Mirai and BASHLITE botnets. Using deep Autoencoders (AEs) trained solely on benign network traffic, the system achieved a true-positive rate (TPR) of 100% and a false-positive rate (FPR) of only 0.007%, with detection latency between 174–212 ms—demonstrating the feasibility of real-time intrusion detection in IoT environments.

Building upon N-BaIoT [79], proposed a semi-supervised deep AE framework to enhance scalability across multiple IoT devices. Trained exclusively on normal traffic from the N-BaIoT dataset, the model achieved exceptional F1 scores (0.9996–0.9998) across devices, demonstrating strong generalisation from device-specific training to unseen IoT environments.

Further work [80] explored reconstruction-based anomaly detection using AEs for linear time series. A Neural Architecture Search (NAS) procedure was employed to automatically discover optimal model configurations, trained on a custom dataset featuring temperature, luminance, CO_2 concentration, power consumption, and humidity. In the absence of labelled anomalies, synthetic disturbances were injected during testing. The model successfully identified irregular behaviour, highlighting the adaptability of reconstruction-based methods to domain-specific IoT contexts.

An unsupervised deep-learning framework with additional functionality was introduced by [81], who proposed a *Conditional Variational Autoencoder (CVAE)* trained on the NSL-KDD dataset. The model integrated intrusion labels into its decoder, achieving a multi-class classification accuracy of 80.10%. A key innovation lay in its ability to reconstruct incomplete feature sets in NSL-KDD—such as `protocol`, `flag`, and `service` features—with reconstruction accuracies of 99.02%, 92.84%, and 71.42%, respectively. This made it one of the first works to perform both feature recovery and intrusion detection within a CVAE framework, highlighting the potential of probabilistic reconstruction for anomaly detection.

2.5 Available Datasets

As IoT security continues to emerge as a critical and rapidly evolving field, a substantial body of research has focused on developing automated systems—such as IDSs—to safeguard connected devices. As discussed previously, much of this research relies on AI, which has demonstrated considerable success in both IoT-specific and broader cybersecurity contexts.

Before any AI model can be trained, however, sufficient and representative data must first be collected and validated. There are few publicly available datasets [34] and consequently, a major strand of IoT security research has centred on constructing and labelling datasets that capture realistic attack scenarios. To be effective, such datasets must be *realistic*, *relevant*, and of *high quality*, while also being sufficiently large and diverse to enable models to generalise to real-world deployments [82].

The Need for Modern Data

Many contemporary studies still depend on early data sources, with numerous works utilising synthetic data, or even datasets from one or even two decades ago [60, 83]. This imbalance reflects the scarcity of up-to-date, high-fidelity data in IoT cybersecurity research and directly motivates the creation of newer datasets, as well as datasets exploring alternative modalities—such as the DRAGON_PI dataset introduced in this thesis [2].

Furthermore, most existing IoT datasets focus exclusively on *network packet data*, overlooking alternative telemetry sources such as device logs, sensor states, or power consumption. As cyberattacks increasingly target embedded devices at the hardware–software boundary, expanding to such modalities is essential for comprehensive threat detection.

2.5.1 Evolution of IoT Datasets

Figure 2.3 illustrates the long history of cybersecurity research and highlights how outdated some commonly used datasets have become compared to their modern alternatives. Some of the earliest benchmark datasets for intrusion detection originated from the Defense Advanced Research Projects Agency (DARPA) experiments in 1998 [84]. Derived from these experiments, the *Knowledge Discovery and Data Mining (KDD) 1999* dataset [55, 85] became the de facto standard for training and testing machine-learning-based IDSs. Each network connection is represented as a multi-feature record spanning several categories: *basic* (e.g., `duration`), *content* (e.g., `logged_in`), *time-based* (e.g., `srv_count`), and *connection-based* (e.g., `dst_host_count`).

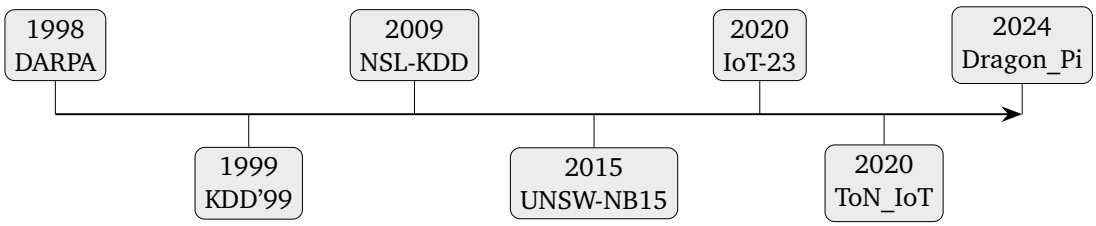


Figure 2.3: Timeline showing the progression of widely used IDS and IoT security datasets.

Despite its historical significance, the KDD dataset exhibits well-known limitations, including redundancy, imbalance, and missing contextual information. During dataset preparation, key identifiers such as timestamps, source/destination IP addresses, and port numbers were removed, causing

numerous flows—particularly ICMP traffic—to appear as duplicates [86]. To address these issues, *NSL-KDD* [59] was released in 2009 as a more balanced and refined alternative, eliminating redundant records and providing improved data diversity.

In the past decade, only a limited number of datasets have been explicitly designed for IoT security research, and those that exist are often constrained to network traffic alone.

To better reflect contemporary network conditions, *UNSW-NB15* [86] was developed to integrate real-world traffic with synthesised attacks across a broader range of intrusion types. The dataset captures modern network-layer behaviours and provides improved labelling granularity, making it one of the most widely adopted modern benchmarks for evaluating AI-driven IDS frameworks.

Another important contribution is the *IoT-23* dataset [68], introduced by the Avast AIC Lab in 2020. IoT-23 contains over 300 million labelled traffic records spanning 23 scenarios containing benign and malicious activity. Owing to its scale and diversity, IoT-23 has become one of the primary benchmarks for evaluating AI-based IDS frameworks tailored to IoT environments.

2.6 Alternatives to Network Traffic

A recurring theme across existing research is the reliance on *network traffic* as the primary signal for detecting malicious activity in IoT systems. Even more contemporary datasets—such as UNSW-NB15 and IoT-23—remain largely network-centric. An exception is the *ToN_IoT* dataset, which introduced additional modalities including operating system logs and IoT telemetry [56]. However, even multimodal datasets such as ToN_IoT still depend heavily on network-flow features, and comparatively few works explore non-network signals as primary detection inputs.

One promising yet comparatively underexplored dimension is the use of *side-channel data*, such as *power* measurements, as a complementary security signal. Side-channel power traces constitute a fundamental and widely available telemetry source, observable on virtually any device regardless of its computational capacity, operating system, or application domain. Despite their potential, studies exploiting power data for defensive purposes remain limited, and

considerably more research is needed to integrate such modalities into mainstream intrusion detection frameworks.

From Side-Channel Attacks to Defensive Monitoring

Early research into power analysis positioned it primarily as an *offensive tool* rather than a defensive one. Kocher et al. [23] introduced Simple Power Analysis (SPA) and Differential Power Analysis (DPA), demonstrating how fine-grained measurements of device power consumption could be used to recover cryptographic keys from tamper-resistant hardware, successfully breaking the Data Encryption Standard (DES) cipher in practice. Their work not only exposed a critical vulnerability in hardware cryptosystems but also laid the foundation for decades of research into side-channel analysis and secure hardware design.

For many readers, this may be the first time encountering a detailed device-level power time series. Figure 2.4 provides a brief illustration of what power data can look like for IoT devices; note the distinctive spikes in power consumption corresponding to bursts of Bluetooth communication.

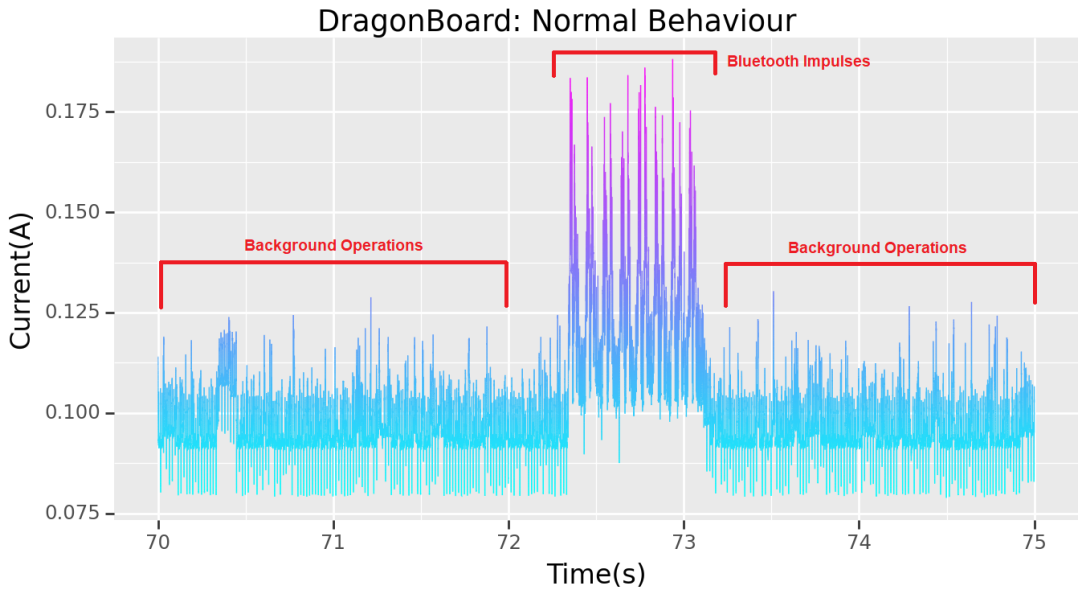


Figure 2.4: Example time series power segment of normal data. Specifically of a burst of Bluetooth communication on the DragonBoard 410c (seen around 72.1s to 73.1s).

More recently, researchers have sought to *repurpose* power data from an attack vector into a defensive one. Power-based intrusion detection leverages the fact

that device activity inherently produces a power signature that is often characteristic and stable during normal operation. These techniques are particularly attractive because they are typically *non-intrusive*—requiring no modifications to existing firmware, operating systems, or network stacks—and can remain effective even when network payloads are encrypted. Crucially, power data provide a direct view of a device's *execution footprint*, offering insight into behaviours that remain invisible to traditional network telemetry.

Despite this potential, works leveraging power data for defensive purposes remain comparatively scarce. WattsUpDoc [87], for example, employs side-channel power measurements of medical devices for real-time malware detection. In experiments, the framework achieved approximately 94% accuracy on previously seen malware and 85% on unseen samples across multiple embedded devices. The method is entirely non-invasive—monitoring external power traces without requiring any software or hardware modification—illustrating the feasibility of power-based intrusion detection in safety-critical settings.

DeepPower [88] extends this concept by applying deep learning to non-intrusive side-channel traces for IoT malware detection. Its two-stage pipeline first filters raw traces to identify suspicious segments before mapping these segments to device activities via fine-grained analysis. The system achieves high detection accuracy while remaining fully deployable, underscoring the scalability of power-domain methods for IoT security.

Although not IoT-specific, Copos et al. [89] demonstrated that power-based telemetry scales effectively even to high-performance-computing (HPC) environments. Using scientific workloads on a large-scale HPC cluster at Lawrence Berkeley National Laboratory, their framework achieved recall and precision rates of 95% and 97%, respectively, in identifying running applications. This result reinforces the broader applicability of power traces as a behavioural signal across diverse computing domains.

Power traces capture low-level *device-internal* behaviour—such as instruction sequences, peripheral activation, and control-flow transitions—that may never manifest on the network, particularly when traffic is sparse, encrypted, or intentionally obfuscated. Nevertheless, practical challenges persist, including sensor placement, sampling-rate selection, environmental noise, scalability, and device heterogeneity, alongside the scarcity of labelled power datasets. Addressing these limitations—and combining power telemetry with traditional network-based monitoring—offers a promising avenue for more robust, attack-agnostic

intrusion detection frameworks in IoT and related embedded environments.

2.7 Summary

This Chapter outlined key IoT security challenges, notable attack types, and existing intrusion detection systems, with a particular focus on signature-based and anomaly-based IDSs and the datasets that underpin them. It highlighted the limitations of traditional, network-centric approaches and the potential of alternative modalities—especially power-based side-channel data—for anomaly detection in resource-constrained IoT environments.

Building on this foundation, the next Chapter presents a lightweight convolutional neural network trained on both real and synthetic power data for IoT intrusion detection. It serves as an initial investigation and proof-of-concept for using deep learning to detect anomalies in power data within the context of this thesis.

Chapter 3

Host-Based Intrusion Detection for IoT Devices Using Convolutional Neural Networks

3.1 Introduction

Building directly on the background established in Chapter 2, this Chapter moves from a high-level survey of IoT security concepts to a proof-of-concept implementation of AI-based detection of anomalies in IoT power data. Here, time-series power traces are used as the primary signal for detecting anomalous behaviour on resource-constrained IoT hardware.

This Chapter presents an initial exploration for this thesis into the concept of Intrusion Detection Systems (IDS) for IoT devices. To this end, a lightweight intrusion detection framework is designed to detect anomalies in the time-series power data of IoT devices using Convolutional Neural Networks (CNNs). In contrast to much of the existing work in IoT security, which focuses on detecting anomalies and attacks using network packets, this Chapter instead examines a data source shared by all electronic devices—their time-series power consumption.

Power data were chosen because they constitute the most fundamental measurement available on any system. Power traces reflect the instruction sequence of the processor core and, depending on sampling configuration, the behaviour of peripheral components with which the core communicates [23]. In systems with limited operational variability, such as embedded or IoT devices, the power

signature should exhibit high replicability, as normal behaviour tends to be regimented with minimal runtime user interaction (for example, environmental sensors transmitting readings at regular intervals). This can allow normal power behaviour to be modelled accurately and used as a trustworthy operational baseline for the device.

For more powerful systems running complex workloads—or those subject to frequent user interaction—normal behaviour can become more difficult to model. This difficulty reflects both the variance in operational patterns relative to the model's learning capacity (with larger models better able to capture such complexity) and the quality and diversity of the data collected.

Another important aspect of the framework proposed in this Chapter is its *host-based* design. Rather than relying on a Network-Based Intrusion Detection System (NIDS) located remotely from the device, a Host-Based IDS (HIDS) operates locally, offering several advantages: reduced detection delay, removal of the need for constant network connectivity, enhanced privacy since data remain on-device, and improved scalability through a "one IDS per device" paradigm.

For this study, a low-cost, low-power microcontroller was required to serve as the IDS board. To remain justifiable in both power and cost, the IDS board should represent only a small fraction of the resources of the device it protects. The MAX78000 was selected as a suitable platform due to its extremely low power consumption, low cost, and integrated Neural Processing Unit (NPU), which enables high-speed, low-power inference at the edge.

Recent advances in hardware have enabled significant acceleration of AI models through technologies such as GPU parallelisation, mixed-precision execution, and the emergence of dedicated NPUs. In the context of edge AI, such acceleration is critical: purely software-based implementations often incur long inference times on low-power edge CPUs. By leveraging NPUs and similar accelerators, devices can exploit the inherently parallel, accelerator-friendly nature of neural networks, achieving substantially faster inference and real-time performance in resource-constrained environments. The deployment of resource-hungry approaches has long been a concern in IoT settings, but this is increasingly mitigated by the wider availability of NPUs and other on-device acceleration technologies.

A baseline normal-behaviour application is first established, and its time-series power data are sampled. Synthetic anomalies are then injected into these data,

and a CNN model is trained to distinguish between normal and anomalous power traces. The methodology proposed here is generic in nature, making it applicable to a wide range of anomaly types and devices, as well as scalable across IoT deployments.

The work presented in this Chapter establishes a proof of concept in this thesis for using side-channel power data as a basis for intrusion detection in IoT devices. By demonstrating that lightweight CNNs can effectively learn and detect anomalies from device power traces in real time, this study validates the feasibility of power-based, host-level IDS deployment. The findings and methods discussed herein have been peer-reviewed and published in [1].

3.1.1 Architectural Assumptions

The experimental framework presented in this Chapter is developed under a set of practical assumptions regarding deployment context and system constraints.

- Intrusion detection is performed **directly on-device** using the MAX78000 microcontroller, reflecting a host-based IDS (HIDS) paradigm.
- Power consumption data are collected locally from the device and processed without reliance on external network telemetry or cloud infrastructure.
- The device is assumed to operate under a controlled and repeatable workload during data collection, enabling the learning of stable baseline behaviour.
- The normal workload consists of continuous execution of a CNN-based road sign detection task, in which input images are repeatedly loaded from memory and processed in a fixed inference loop. Each inference takes approximately 2.5 ms resulting in a consistent and periodic computational pattern. This deterministic execution provides a stable and repeatable baseline of normal device behaviour, suitable for learning via time-series analysis.
- Computational resources are strictly constrained, including limited memory, processing capability, and power budget, necessitating the use of lightweight CNN architectures suitable for embedded inference.
- Due to the lack of network connectivity on the target platform, anomalous behaviour is introduced via **synthetic anomaly injection**, rather than live

attack execution.

- The synthetic anomalies introduced in this Chapter are designed to reflect a broad range of deviations from normal behaviour, both in terms of amplitude and temporal structure. These perturbations are informed by common practices in early-stage intrusion detection research and are intended to simulate generic anomalous patterns in a controlled setting. The goal is not to replicate specific real-world attacks, but rather to evaluate whether the model can distinguish structured normal behaviour from a diverse set of abnormal deviations, thereby establishing feasibility.

These assumptions reflect a constrained but realistic embedded setting and provide a controlled foundation for evaluating the feasibility of power-based intrusion detection.

3.2 Background: Neural Network Foundations

3.2.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are the foundation of modern deep learning models. Inspired by simplified models of biological neurons, ANNs consist of interconnected computational units that transform an input into an output through a series of learned linear and non-linear operations [90, 91].

At their core, each neuron receives input values $\mathbf{x}$, applies a set of learnable weights $\mathbf{w}$ and a bias term b, and passes the result through a non-linear activation function $f(\cdot)$:

$$y = f(\mathbf{w}^\top \mathbf{x} + b). \tag{3.1}$$

By arranging neurons into layers, neural networks can approximate increasingly complex mappings. In a Multilayer Perceptron (MLP), each layer's output serves as the input to the next, forming a hierarchical, deep representation of the data [92]. The expressive power of deep networks arises from this layered composition, together with the use of non-linear activation functions, enabling the modelling of patterns and relationships that are difficult or impossible to capture using linear or rule-based methods. Figure 3.1 illustrates the structure of a simple ANN comprising four input neurons, a six-neuron hidden layer, and three output neurons.

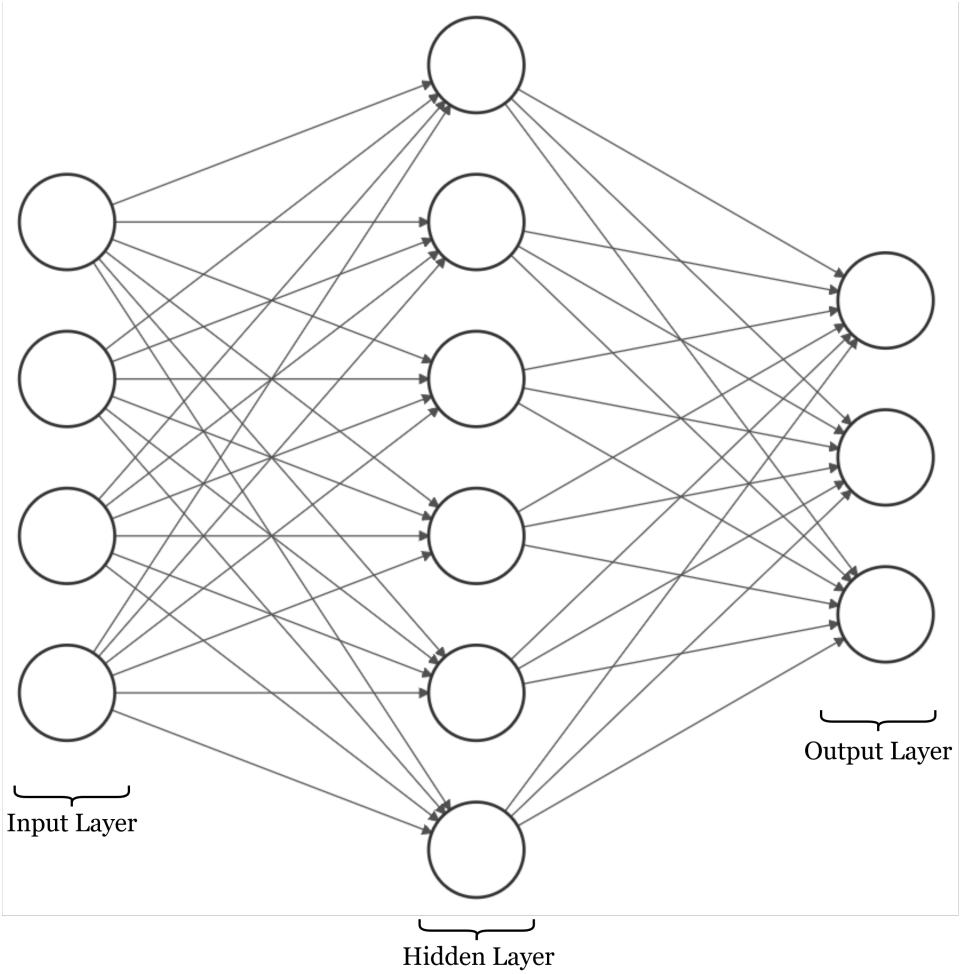


Figure 3.1: Schematic of a simple artificial neural network (ANN) with four input neurons, a single hidden layer of six neurons, and three output neurons.

3.2.2 Training Neural Networks: Loss and Optimisation

Training a neural network involves adjusting its parameters to minimise a chosen loss function $\mathcal{L}(y, \hat{y})$, which measures the discrepancy between the predicted output $\hat{y}$ and the target value y. This process is known as optimisation.

Optimisation is typically performed via Stochastic Gradient Descent (SGD) or one of its modern variants (e.g., Adam, RMSProp), where the parameters are updated iteratively according to:

$$\theta \leftarrow \theta - \eta \frac{\partial \mathcal{L}}{\partial \theta}, \tag{3.2}$$

with the learning rate η controlling the step size.

Gradients are computed efficiently using the backpropagation algorithm [93],

which propagates the loss gradient from the network output $\hat{y}$ backwards through its layers.

The choice of loss function depends on the task: Mean Squared Error (MSE) is common for regression and reconstruction, while cross-entropy loss is standard in classification tasks.

Activation functions also play a central role in learning dynamics. Non-linear functions such as the Rectified Linear Unit (ReLU) enable networks to approximate complex decision boundaries while mitigating issues such as vanishing gradients [94]. Without the complexity added by activation functions, neural networks would reduce to simple linear mappings.

Regularisation strategies help prevent overfitting, particularly when training on small or noisy datasets. Common techniques include dropout [95], weight decay, and early stopping, each encouraging the network to learn stable and generalisable representations rather than fitting too closely to the training data (or even validation or test data in highly biased evaluation scenarios).

The training process for most neural network–based deep learning models follows a broadly similar pattern. The model iterates over the dataset multiple times (each full pass known as an epoch), updating its parameters to minimise a loss that measures the discrepancy between predictions and targets. This paradigm underpins classical architectures such as ANNs and CNNs, as well as more recent Transformer-based models. Although many new learning paradigms have emerged—including masked prediction and causal (autoregressive) modelling—the core notion of a training loop, consisting of forward passes, loss computation, backpropagation, and parameter updates, has remained largely unchanged.

3.2.3 Convolutional Neural Networks

While MLPs treat input features independently, many real-world signals exhibit strong local structure. CNNs were developed to leverage such structure by introducing two key principles: *local connectivity* and *weight sharing* [96, 97]. Instead of learning independent weights for every input dimension, CNNs apply small learnable filters (*kernels*) that slide across the input, detecting local patterns that are often repeated throughout the data [92].

Another key advantage of CNNs over traditional MLP-based ANNs is their ability to automatically extract features directly from raw or minimally processed input

data, whereas MLPs typically require hand-crafted feature vectors. This manual feature design can be costly, time-consuming, and heavily dependent on domain knowledge to obtain sufficiently salient features. The convolutional layers are often paired with an MLP classifier, where the convolutional layers extract the features and the fully connected layers perform supervised classification.

For an input array $\mathbf{X}$ and kernel $\mathbf{K}$, the convolution operation produces an output feature map $\mathbf{Y}$:

$$Y_{i,j} = \sum_{m,n} K_{m,n} \cdot X_{i+m-1,j+n-1}. \tag{3.3}$$

Early layers typically learn low-level features (e.g., edges, bursts, local fluctuations), while deeper layers combine these into higher-level abstractions, allowing increasingly powerful features to be learned [98]. This hierarchical feature learning process allows CNNs to represent complex patterns efficiently while maintaining a relatively low parameter count compared to fully connected networks. This increase in the effective computation area with depth is known as the receptive field of the CNN, which grows as the model deepens.

Dimensionality-reduction techniques such as pooling and strided convolutions further improve efficiency by summarising local activations and reducing spatial resolution [99], as illustrated in Figure 3.2, where the input resolution decreases from 128×128 to 16×16 before the flattening layer of the fully connected classifier. Without dimensionality reduction, computations would become infeasible as deeper layers continue to operate on the original input dimensions.

Normalisation methods, such as BatchNorm [100], additionally stabilise training by controlling the distribution of activations within each layer.

3.2.4 Advantages of CNNs for Time-series and Structured Signals

Although CNNs were originally designed for images, their core properties make them suitable for a wide range of structured data modalities, including time series, sensor streams, and audio. The inherent use of local filters enables the network to capture short-term temporal or spatial correlations, while hierarchical feature extraction allows it to recognise recurring patterns at multiple scales. The more layers, the larger the scale (or receptive field) at which the

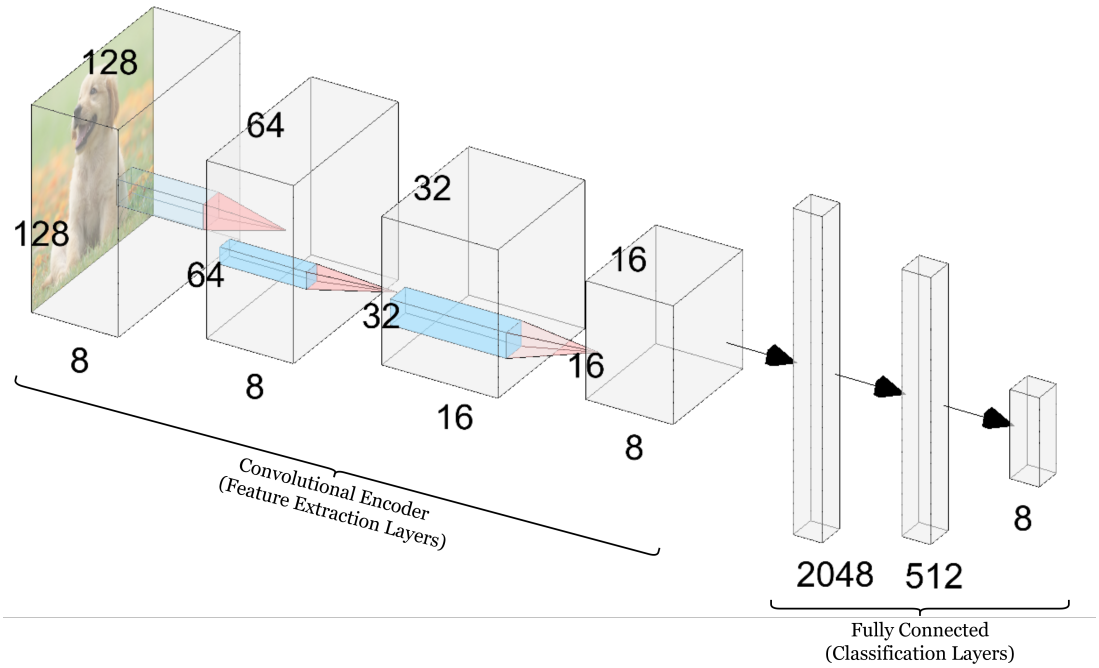


Figure 3.2: Illustration of a convolutional neural network (CNN) operating on an image input. As the network deepens, the spatial resolution is progressively reduced through convolutions and pooling before the resulting features are passed to a fully connected classifier.

model operates on the input.

Furthermore, CNNs achieve significant parameter efficiency through weight sharing, where the same convolutional filters are applied across all spatial locations. This makes them well-suited to deployment in resource-constrained environments. Their ability to learn directly from raw or minimally processed data reduces reliance on hand-crafted feature engineering and improves adaptability to diverse signal characteristics.

These qualities provide the foundation for the approach developed in this Chapter, where CNNs are applied to time-series power data to model and distinguish normal and anomalous device behaviour.

3.3 Testbed and Data Creation

For the work presented in this Chapter, the system was treated as comprising two main components:

1. the device to be protected, and

2. the IDS board responsible for monitoring and protecting it.

Given the heterogeneous nature of modern embedded systems, a more advanced architecture could incorporate a secure core dedicated to monitoring the power consumption, or other telemetry, of other cores or peripheral components within the same device. Such a secure core would need to remain isolated from vulnerabilities affecting the main system. This approach is particularly suitable given the heterogeneous and often distributed nature of modern IoT systems, which may manifest as tightly coupled clusters of devices (e.g., a modern electric vehicle, EV). The simplified framework adopted in this Chapter is illustrated in Figure 3.3.

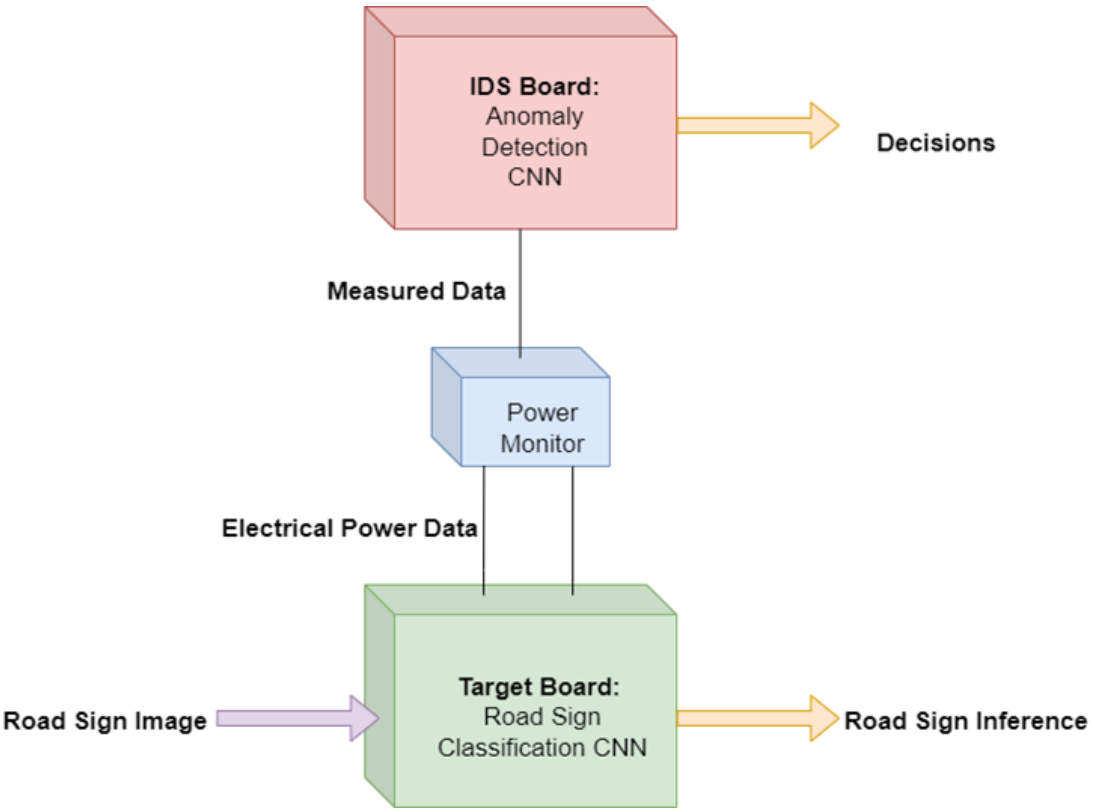


Figure 3.3: High-level framework showing the device to be protected and the IDS board monitoring its power usage.

For convenience during this implementation, the MAX78000 served as both the target and the IDS board, with the two roles operating separately. For real-world deployment, the IDS should consume but a fraction of the power as the target.

The MAX78000 integrates an Arm Cortex-M4, a RISC-V core, and a dedicated NPU, making it well-suited to edge-AI experimentation [101]. The approach in this Chapter is general: the protected device (target) need not be the same

hardware as the IDS device (although this was the path chosen for convenience in this Chapter).

3.3.1 Normal Behaviour

To establish the baseline normal power time-series data, it was first necessary to define the normal functionality of the target board. To utilise the onboard NPU effectively, a road-sign detection CNN trained on the German Traffic Sign Recognition Benchmark (GTSRB) dataset [102] was selected as the representative normal application. This model was configured to perform repeated inferences on different road-sign images in sequence, simulating the type of CNN-based classification commonly found in embedded AI camera systems, such as those deployed in modern or autonomous vehicles.

The CNN was implemented on the MAX78000 using a hybrid configuration: the Arm Cortex-M4 processor handled general software operations (e.g., data loading and control flow), while the NPU executed the convolutional inference tasks. As a result, the recorded power profile reflected the combined activity of both CPU and NPU subsystems, representing a realistic operational workload for an edge-AI application.

Figure 3.4 shows approximately 48,000 data points covering six consecutive inferences, illustrating the highly replicable nature of the normal power traces sampled from the target application. Each peak corresponds to the concurrent activity of the core and NPU during the classification of a single road sign.

A defining characteristic of edge and IoT devices is that their normal behaviour tends to be highly regular, as it is typically prescribed by fixed firmware or manufacturer-defined routines. Although user interaction can introduce variability, most systems can still be decomposed into a small number of fundamental operational patterns.

For example, a modern electric vehicle (EV) may appear complex and unpredictable when viewed as a single system encompassing infotainment, drive control, and other systems such as safety subsystems. However, when each subsystem is considered independently according to its core functionality, its behaviour becomes far more structured and interpretable. This modular view mirrors IoT design principles: each subsystem exhibits regular, repeatable patterns that can be modelled and monitored individually. For smaller IoT devices, this effect is even more pronounced, as their restricted function sets pro-

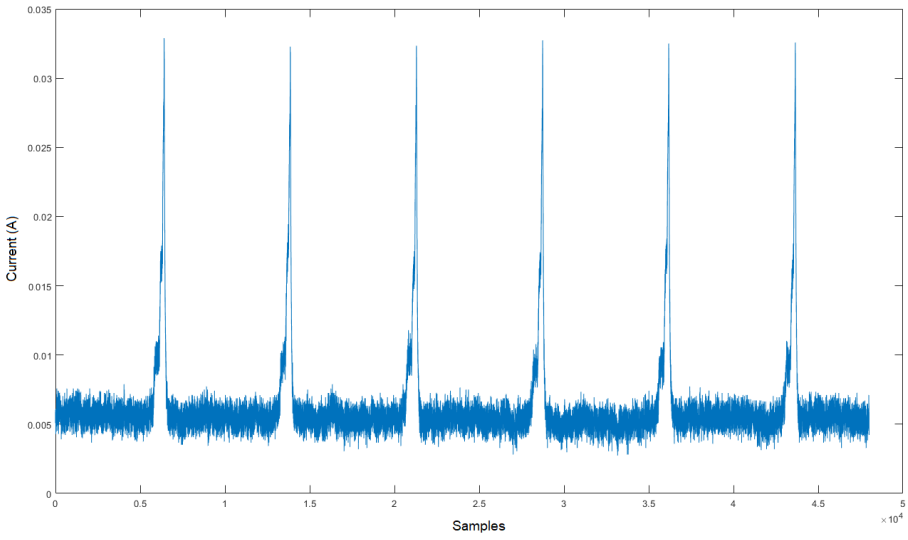


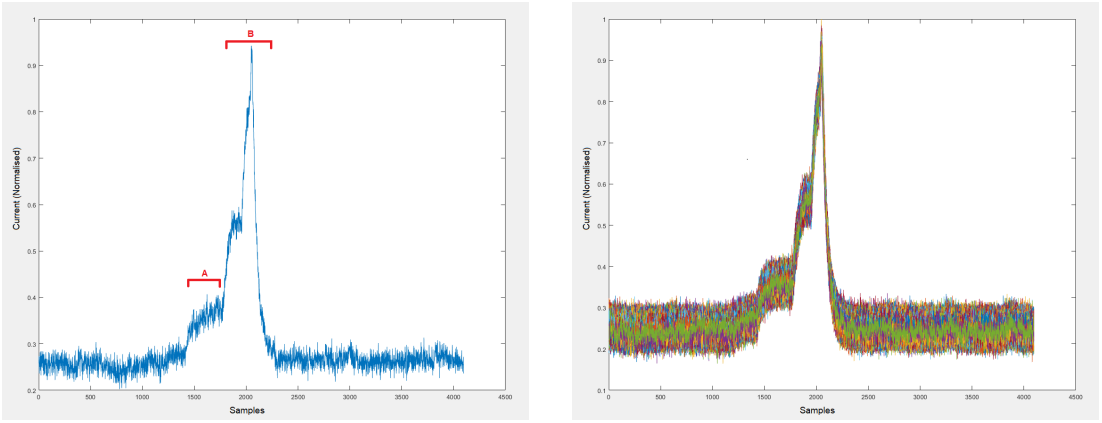
Figure 3.4: Normal power data sampled from the MAX78000 during six consecutive inferences of the road-sign classification model. Each peak corresponds to combined CPU and NPU activity.

duce inherently stable and predictable power signatures, making them highly amenable to modelling via lightweight neural networks.

3.3.2 Data Sampling

A relatively high sampling rate of 250 kHz was selected to capture the high-frequency activity associated with NPU inferences. While such a rate may exceed the practical limits of many IoT systems, it ensures sufficient temporal resolution to observe fine-grained power fluctuations during inference. In real-world deployments, this rate could be reduced or replaced with an adaptive sampling methodology, balancing fidelity with energy and storage constraints.

In total, approximately 3.5×10^8 individual samples were collected from the road-sign CNN performing repeated inferences (as shown in Figure 3.4). Using a fixed-length window of 4,096 samples centred on each inference peak, the complete power signature of each CNN operation was captured while retaining partial overlap between consecutive inferences. Figure 3.5a illustrates a single inference window, while Figure 3.5b overlays ten separate inferences, demonstrating the highly repeatable nature of the normal power behaviour.



(**a**) Single road-sign CNN inference. (**b**) Ten overlaid inferences.

Figure 3.5: Road-sign CNN inference power traces. (Left) Single inference showing CPU loading (A) and NPU spike (B). (Right) Ten overlaid inferences demonstrating high repeatability.

3.3.3 Injection of Synthetic Anomalies

As the MAX78000 features no network connectivity, and performing side-channel attacks was outside the scope of this thesis, *synthetic anomalies* were introduced to emulate abnormal device behaviour in the form of a cyberattack. The use of synthetic anomalies is common in domains such as cybersecurity and medicine, where true anomalies can be rare, unsafe, or costly to collect. Based on the methodology outlined in [103], anomalies were injected directly into the normal power traces of the road-sign CNN.

Five anomaly types were defined to span a broad range of behavioural deviations: *Spike*, *Noise (short duration)*, *Noise (entire signal)*, *Constant*, and *Drift*. These categories were selected to test diverse detection sensitivities, from abrupt transient disruptions to slowly evolving signal changes.

Spike. Spike-type anomalies represent sudden, sharp peaks or drops in the power signal that persist for only a few samples—often a single sample. Such anomalies stand out clearly against the surrounding values due to their abrupt nature, as shown in Figure 3.6a. Common causes include hardware glitches, transient sensor faults, or bit flips during data transmission or storage, all of which can introduce isolated outliers into the signal. Between one and three spikes of varying intensity were randomly added to each normal segment.

Noise (short duration). Short-noise anomalies are characterised by a brief, unexpected increase in signal variance. This type of anomaly often indicates tran-

sient instability within the system and may be caused by factors such as a depleting battery, abrupt environmental changes (e.g., rapid temperature shifts), or the execution of unintended or resource-intensive processes on the device, including malicious code. Figure 3.6b illustrates such an event.

Noise (entire signal). In contrast, the *long-noise* anomaly (referred to later as *Noise 2*) persists throughout the entire duration of the signal. Such sustained irregularity may arise from a systemic issue, such as a faulty or unstable connection between the target device and the power-monitoring interface of the IDS. It is hypothesised that the IDS should be capable of detecting such faults within its own monitoring pathway, as these could otherwise lead to blind spots or false readings. Figure 3.6c shows a signal affected by this persistent form of noise.

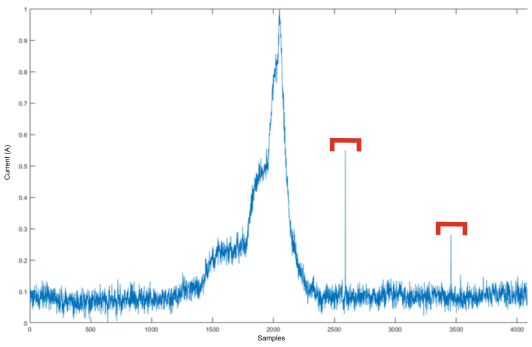
Constant. *Constant*-type anomalies occur when a signal remains fixed at a single value for an extended period, exhibiting no variation where natural fluctuations would normally occur. This behaviour may indicate a hardware fault on the target device—such as a malfunctioning power supply resulting from an attack or system failure—or a defect in the IDS power-monitoring subsystem itself. Figure 3.6d shows an example of this anomaly type.

Drift. *Drift* anomalies manifest as gradual shifts in the signal baseline, either upward or downward relative to its expected level. Such anomalies can result from various factors, including sensor degradation over time; environmental influences such as temperature changes, electromagnetic interference, or humidity; calibration drift; long-term device wear; or unauthorised resource consumption. Figure 3.6e shows a segment of the signal gradually shifting downward over the course of an inference.

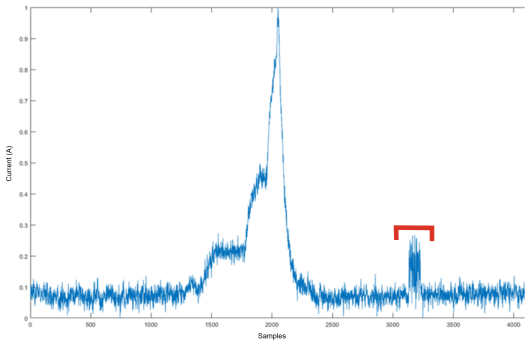
By varying the *intensity* of the inserted anomalies, five datasets of increasing detection difficulty were created, allowing the task to range from simple to challenging (difficulty was assessed subjectively through visual inspection). These datasets were labelled *Complexities 1–5* and are referred to as *Comp1–5* throughout this Chapter, with *Comp5* being the highest difficulty (anomalies closer to normal behaviour and/or less frequent). In addition, an *All-in-One (AIO)* dataset was generated, combining a range of parameter values from the minimum of *Comp1* to the maximum of *Comp5* during anomaly creation. The purpose of this dataset was to evaluate each model's ability to detect a broader spectrum of anomalies and to assess its capacity to generalise across varying levels of difficulty (e.g., spikes of *Comp1* amplitude appearing alongside those

of *Comp5*).

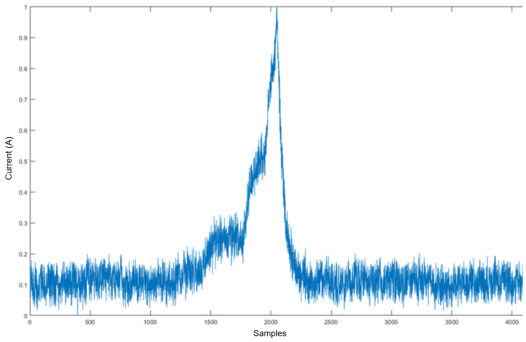
Anomaly complexity was controlled using an integer value from 1 to 5, applied through a simple scaling equation that simultaneously adjusted relevant parameters such as amplitude, duration, and noise intensity.



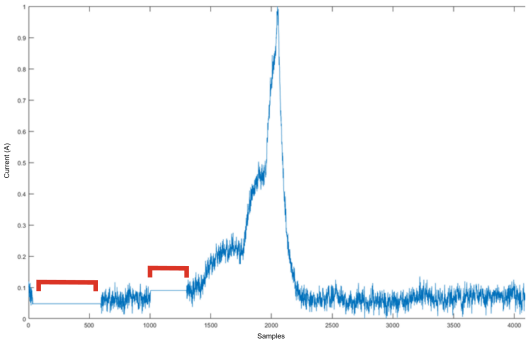
(**a**) Spike anomalies



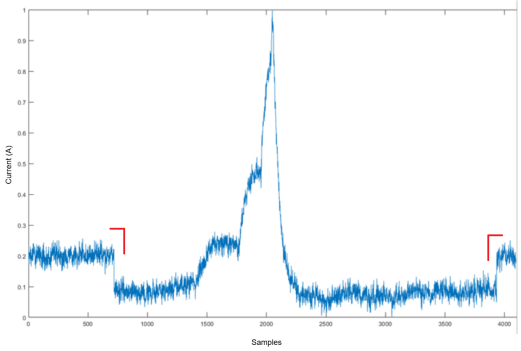
(**b**) Short-noise anomaly



(**c**) Long-noise anomaly



(**d**) Constant anomalies



(**e**) Drift anomaly

Figure 3.6: Road-sign CNN inference traces with injected anomalies. Spike, short-noise, and long-noise disturbances are shown alongside constant and drift anomalies. Red annotations indicate injected regions.

3.3.4 Data Processing

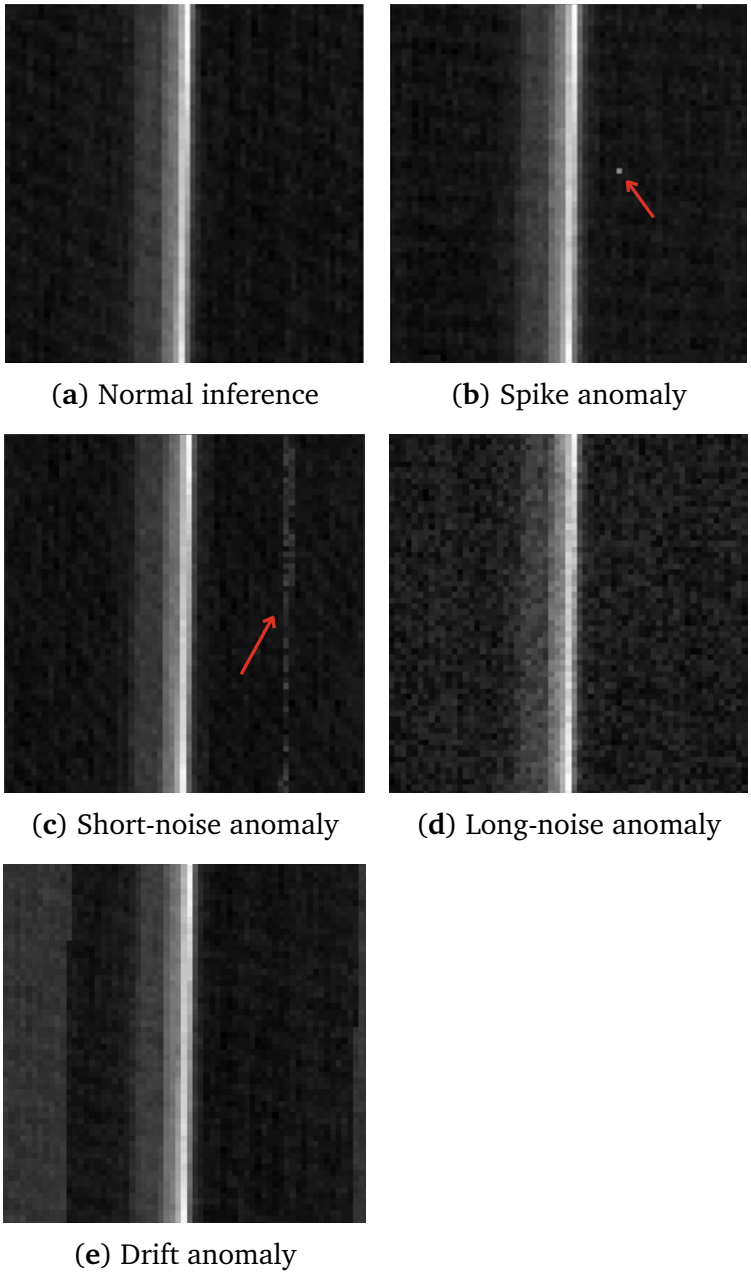
As this Chapter employs a lightweight image-based CNN architecture (without pre-trained weights) provided with the MAX78000, the 1D power time-series data were represented as grayscale images for compatibility—an example of normal data as a grayscale image illustrated in Figure 3.7a. CNNs are particularly effective for image processing [104], with a key advantage being their ability to automatically extract spatial and structural features from the input. This property makes them well-suited to lightweight, time-critical scenarios, as it eliminates the need for computationally expensive feature engineering prior to inference—an important benefit in resource-constrained IoT environments.

Before generating images, the full sequence of approximately 3.5×10^8 samples was divided into fixed-size windows suitable for network input. A window length of 4,096 samples was selected to ensure that each segment captured a complete inference of the normal application, allowing the IDS CNN to observe the full behavioural signature in every instance.

To align the windows precisely with each inference and avoid splitting signatures across multiple segments, a simple peak-finding algorithm was applied to centre each window on the maximum power value of the CNN inference. This approach ensured consistent window alignment while preserving slight overlap in the surrounding background activity.

Each 1D segment of 4,096 samples was then reshaped into a 64×64 grid (row-major order from the top-left sample at index 0 to the bottom-right sample at index 4095). These grids were normalised to grayscale intensity values in the range 0–255, resulting in images in which a distinctive streak marks the inference peak near the centre. This transformation effectively reinterprets the temporal signal as a spatial pattern. Figure 3.7a illustrates the normal power data as an image as mentioned; note the "streak" of the central peak. This streak, a key feature caused by the peak in the middle of the window, is also visible in Figures 3.7b to 3.7e, though with additional distortion introduced by the anomalies present in each case.

Because the base CNN architecture was designed for three-channel RGB input, two additional channels were required. To maintain architectural compatibility, the single grayscale image was concatenated with two zero-valued channels, producing a three-channel tensor. The network's input layer also required dimensions of 128×128; this resizing was performed using the augmentation and



(**a**) Normal inference (**b**) Spike anomaly

(**c**) Short-noise anomaly (**d**) Long-noise anomaly

(**e**) Drift anomaly

Figure 3.7: Road-sign CNN inference represented as grayscale images. The normal inference (top-left) is shown alongside injected anomaly types: spike, short-noise, long-noise, and drift. Distortions introduced by each anomaly are visually apparent in the corresponding panels.

interpolation utilities provided by PyTorch, ensuring consistent image scaling while preserving the original power-pattern structure.

3.4 CNN Architecture

A lightweight CNN was chosen for this task, illustrated in Figure 3.8. A model originally designed for the Kaggle *Cats vs Dogs* dataset was selected. This architecture provides an effective balance between simplicity and representational capacity, with only seven layers and 55,941 parameters, making it well-suited to highly constrained IoT hardware while retaining sufficient depth to model the temporal-spatial patterns present in the power data.

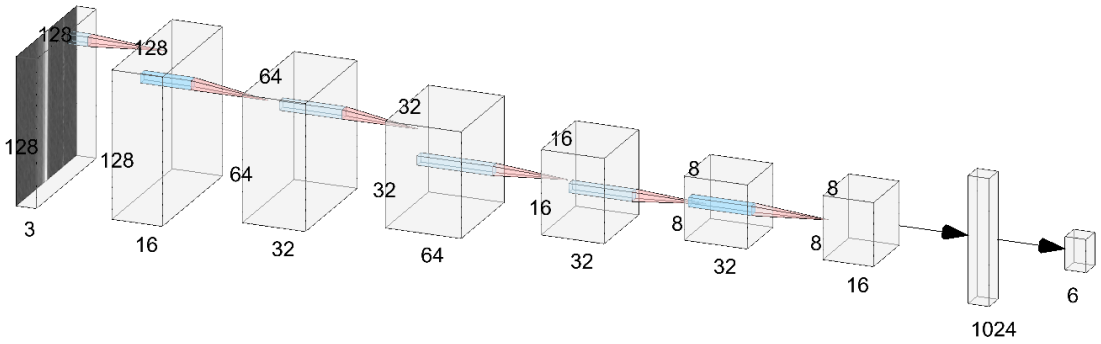


Figure 3.8: Kaggle *Cats vs Dogs* CNN architecture, supplied with the MAX78000.

All pre-trained weights were discarded, and the architecture was trained from scratch for both use cases: the *RoadSign CNN* (providing the baseline normal behaviour) and again for the *IDS CNN* (providing anomaly detection capabilities). In both cases, the CNN contained 55,941 learnable parameters—an exceptionally low count for a modern deep network but appropriate for a super-lightweight, low-power, edge-deployable implementation. The architecture specification can be found in Table 3.1

Table 3.1: *Cats vs Dogs* CNN architecture showing specific layer operations and output dimensions.

Layer	Operation(s)	Output Shape
Input	Input image (HWC format)	$128 \times 128 \times 3$
0	Conv 3×3, stride 1, padding 1; ReLU	$128 \times 128 \times 16$
1	MaxPool 2×2, stride 2; Conv 3×3, stride 1, padding 1; ReLU	$64 \times 64 \times 32$
2	MaxPool 2×2, stride 2; Conv 3×3, stride 1, padding 1; ReLU	$32 \times 32 \times 64$
3	MaxPool 2×2, stride 2; Conv 3×3, stride 1, padding 1; ReLU	$16 \times 16 \times 32$
4	MaxPool 2×2, stride 2; Conv 3×3, stride 1, padding 1; ReLU	$8 \times 8 \times 32$
5	Conv 3×3, stride 1, padding 1; ReLU	$8 \times 8 \times 16$
6	Flatten (1024); Linear (FC) layer to 6 outputs; Softmax	6

To accommodate the six classes in the IDS datasets (Normal and Comp1–5), the final fully connected layer was modified to output six logits instead of the

original two used for binary classification. A softmax activation was applied to convert the logits into probability distributions.

Six separate IDS models were trained—one for each dataset described earlier. Each dataset was divided into non-overlapping training, validation, and test subsets, comprising 33,161, 3,684, and 9,211 images respectively (approximately a 72% / 8% / 20% split). The inclusion of a dedicated validation set during training helped to minimise overfitting and provided a more reliable indication of generalisation performance. This is especially important in supervised learning scenarios.

3.4.1 Training—Quantisation-Aware

All models were trained using Python scripts provided by Maxim Integrated, adapted to support Quantisation-Aware Training (QAT). QAT simulates the effects of low-precision arithmetic (e.g., 8-bit integer operations) during training, allowing the model to learn parameters that are inherently robust to quantisation-induced errors [105]. This approach substantially mitigates the accuracy degradation often observed when models trained in full precision are quantised *post hoc* for embedded deployment. Moreover, the QAT-enabled setup streamlined deployment to the MAX78000 platform.

The training pipeline was implemented in PYTORCH, a widely adopted deep learning framework. All models were trained from scratch using the Adam optimiser [106], selected for its adaptive learning-rate mechanism, which stabilises training on small or noisy datasets—conditions typical of embedded anomaly-detection scenarios. To reduce overfitting and improve generalisation, ℓ_2 regularisation (weight decay) was applied, penalising large weight magnitudes and encouraging simpler, more transferable features—particularly valuable given the limited variability in certain datasets and the relatively low input resolution.

Each model was trained for 150 epochs, except for the *All-in-One (AIO)* model. It was hypothesised that, due to the added complexity of jointly learning across all anomaly intensities, the AIO model may require a longer convergence period; accordingly, it was allowed to train for 200 epochs. Each training epoch consisted of 130 iterations, with validation performed at the end of each epoch. The model checkpoint achieving the highest validation accuracy was retained for final evaluation on the test set, ensuring the best generalisation perfor-

mance.

An initial learning rate of 5×10^{-4} was used for all models. To refine convergence and prevent oscillations, a step-based learning-rate scheduler was employed, applying a decay factor of $\gamma = 0.1$ beginning at epoch 60. This reduced the learning rate to 5×10^{-5}, enabling the optimiser to fine-tune parameters with smaller adjustments as training progressed. A batch size of 256 was used, providing a good balance between training efficiency (via GPU parallelisation) and stable gradient estimation.

Quantisation-Aware Training was a critical component of this workflow. Since the MAX78000 executes inference exclusively using 8-bit integer operations, simulating quantisation effects during backpropagation ensured that the trained models maintained robust performance under real hardware constraints.

Once training was complete, each model was converted into deployable form using Maxim's model-synthesis toolchain. This process involved defining a YAML configuration file specifying the CNN architecture, quantisation settings, and layer parameters. The synthesis tools then automatically generated optimised C++ source code ready for compilation and deployment on the MAX78000.

Training was performed on a consumer-grade laptop equipped with an Intel Core i7-10750H CPU and an NVIDIA GTX 1650 Ti GPU, running CUDA 10.2. Despite the modest hardware resources, training was efficient: approximately 2 hours 10 minutes for the standard datasets and around 2 hours 54 minutes for the AIO dataset with its extended number of epochs. These results underscore the computational efficiency and deployability of the proposed lightweight CNN models.

3.5 Overall Results

The best model for each dataset was selected based on validation accuracy during training. The methodology used to calculate these accuracies is discussed in a later section. The validation results for all models are presented in Table 3.2.

Validation was performed once per epoch and was critical for identifying both *underfitting* (when the model fails to learn adequately) and *overfitting* (when the model becomes overly specialised to the training data). This process en-

Table 3.2: Final validation accuracy and corresponding training epoch for each trained model.

Model	Accuracy (%)	Epoch
Comp1	99.946	149
Comp2	99.837	149
Comp3	99.810	147
Comp4	99.403	140
Comp5	87.188	147
AIO	98.046	186

abled model selection without prematurely evaluating on the test set, thus avoiding bias and preserving the independence of the final evaluation [92]. Adjustments to hyperparameters or training duration were therefore guided solely by validation performance.

Table 3.3 presents the generalisation results of each model, showing accuracy not only on its own test set but also across the test sets of all other complexity levels. As expected, each model achieved its highest accuracy on its corresponding test set, reflecting strong in-distribution performance where the anomalies matched those observed during training.

Among all models, *Comp3* and *AIO* exhibited the strongest cross-dataset generalisation, achieving average accuracies of 90.46% and 90.42%, respectively, across all test sets. This suggests that these networks learned more generalisable representations rather than overfitting to dataset-specific characteristics. This result is intuitively consistent with the dataset design: the AIO dataset encompasses the full range of anomaly intensities (Comp1–5), while Comp3 represents an intermediate level of difficulty, naturally promoting balanced feature learning.

As anticipated, the largest performance decline was observed on the *Comp5* dataset, which contained the most challenging and complex anomalies closest to the normal behaviour. This degradation underscores the increasing difficulty of accurately modelling highly variable or subtle deviations, and highlights the importance of training with diverse, high-complexity data to improve robustness.

It is important to emphasise that all models were trained using QAT, and the results in Table 3.3 therefore reflect their post-quantisation performance—i.e., as deployed on the MAX78000 hardware. While quantisation typically leads to some loss of accuracy when reducing precision from 32-bit to 8-bit, this impact

was minimised through the QAT procedure, demonstrating its effectiveness in preserving model fidelity under deployment constraints.

Table 3.3: Generalisation performance of each trained model across all dataset test sets (post-quantisation).

Model	Comp1 (%)	Comp2 (%)	Comp3 (%)	Comp4 (%)	Comp5 (%)	AIO (%)
Comp1	99.924	98.209	94.333	81.978	27.359	90.338
Comp2	99.609	99.750	95.679	92.216	41.060	90.924
Comp3	99.653	99.283	99.729	96.721	52.166	95.234
Comp4	79.670	93.562	98.046	99.457	53.382	75.334
Comp5	57.279	62.317	64.597	69.862	87.102	57.572
AIO	99.479	97.753	98.382	97.670	50.850	98.393

3.6 Further Analysis of Best Models

3.6.1 Metric Definition

As the models trained on *Comp3* and *AIO* demonstrated the best generalisation performance across all test sets (Table 3.3), they were selected for further analysis using additional classification metrics.

Relying solely on accuracy can be misleading—particularly in imbalanced datasets, which are common in anomaly detection tasks where the anomalous class can be severely underrepresented. Accordingly, *precision*, *recall*, and *F1-score* were also computed to provide a more comprehensive assessment of each model's strengths and weaknesses.

To calculate these metrics, confusion matrices were first generated for each model. A confusion matrix summarises how a classifier's predictions correspond to the true labels, enabling direct comparison of predicted and actual classes. These confusion matrices are shown combined in the next section. Although commonly used in binary classification, the concept can be naturally extended to multi-class problems. From these matrices, the four fundamental quantities—true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN)—were derived.

For multi-class classification, TP, FP, FN, and TN can be defined relative to the class currently under evaluation. When focusing on a single class, it is treated as the "positive" class, while all others are considered "negative." Overall model performance can then be calculated either by averaging per-class metrics (macro-averaging) or by aggregating counts across the entire matrix

(micro-averaging). The latter can introduce interpretive ambiguity, as FP and FN values are summed across classes, potentially masking class-specific weaknesses. One practical approach to resolving this issue is the *one-vs-all* method, where each class is independently evaluated against all others.

In this Chapter, the following definitions are applied:

- **TP** — Correct positive detections for the class under consideration.
- **FP** — Incorrect positive detections, where another class is misclassified as the current class.
- **FN** — Missed detections, where the current class is misclassified as another class.
- **TN** — Correct negative detections, where other classes are correctly identified as not belonging to the current class.

It is important to note that, in this system, TP, FP, FN, and TN are defined consistently across all classes using a one-vs-all approach. However, the *interpretation* of these quantities differs depending on which class is treated as the positive class. When the *normal* class is the positive class, a false negative (FN)—a normal sample misclassified as an Anomaly—represents a *false alarm*. In contrast, a false positive (FP) for the normal class—an Anomalous sample misclassified as normal—corresponds to a *missed anomaly*. For anomaly classes evaluated as the positive class, the interpretation returns to the conventional intrusion-detection view: FP corresponds to false alarms, and FN corresponds to missed detections. Although these interpretations differ between classes, the mathematical definitions and metric formulations remain unchanged.

Using these definitions, the evaluation metrics were computed for each class independently using a one-vs-all view, allowing class-specific behaviour to be analysed for normal and anomalous categories alike.

Precision (P)

$$P = \frac{TP}{TP + FP} \tag{3.4}$$

Precision measures the proportion of correctly predicted positive observations among all predicted positives. In this IDS, a high number of FP for the normal class would allow anomalies to “slip through the net,” as they would be falsely classified as normal behaviour—as in the one-vs-all view for the normal class,

FP corresponds to anomalies misclassified as normal. Given the rarity and importance of anomalies, high precision is therefore essential for a trustworthy IDS.

Recall (R)

$$R = \frac{TP}{TP + FN} \tag{3.5}$$

Recall, also known as the *True Positive Rate (TPR)*, measures the proportion of correctly predicted positives among all actual positives. In this context, a high number of FN for the normal class (i.e., false alarms) would cause the IDS to trigger unnecessary interruptions or warnings. Thus, high recall is required to minimise spurious alerts and maintain operational usability.

F1-Score

$$F1 = 2 \times \frac{P \cdot R}{P + R} \tag{3.6}$$

The F1-score is the harmonic mean of precision and recall, providing a single balanced measure that accounts for both false alarms and missed detections. In this study, it was often difficult to determine the superior model from precision or recall alone; the F1-score therefore served as a more holistic indicator of classification performance.

Accuracy (A)

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.7}$$

Accuracy, used primarily during model validation, expresses the overall proportion of correct predictions—both positive and negative—among all samples. While intuitive, accuracy can be misleading in imbalanced datasets typical of intrusion detection, as it may be dominated by large numbers of true negatives, thereby obscuring poor performance on minority (anomalous) classes.

3.6.2 Best Models Comparison

Tables 3.4 and 3.5 illustrate the overall performance of the best models to be compared. While the AIO model achieved slightly higher overall precision than

Comp3—which would traditionally suggest superior performance—the context of multi-class anomaly detection demands a more nuanced interpretation. In such systems, the *false alarm rate*, representing the frequency with which normal behaviour is incorrectly flagged as anomalous, can be particularly problematic, as too many false positives can inundate responders and result in the IDS being turned off. Because anomalies are inherently rare while normal behaviour dominates, minimising false negatives (FN) for the normal class is essential to reducing false alarms and maintaining system trustworthiness.

Table 3.4: Per-class performance of Comp3 across all test sets.

Class	TP	FP	TN	FN	ACC	PREC	REC	F1
Overall	49850	5416	270914	5416	0.96733	0.90200	0.90200	0.90200
Normal	9198	4145	41910	13	0.92476	0.68935	0.99859	0.81564
Spike	8699	735	45400	432	0.97888	0.92209	0.95269	0.93714
Noise 1	7520	180	45904	1662	0.96667	0.97662	0.81899	0.89089
Noise 2	7814	18	46001	1433	0.97375	0.99770	0.84503	0.91504
Constant	8434	103	45972	757	0.98444	0.98793	0.91764	0.95149
Drift	8185	235	45727	1119	0.97550	0.97209	0.87973	0.92361

Table 3.5: Per-class performance of AIO across all test sets.

Class	TP	FP	TN	FN	ACC	PREC	REC	F1
Overall	49915	5351	270979	5351	0.96773	0.90318	0.90318	0.90318
Normal	9110	2890	43165	101	0.94588	0.75917	0.98903	0.85899
Spike	7973	47	46088	1158	0.97820	0.99414	0.87318	0.92974
Noise 1	7623	1446	44638	1559	0.94563	0.84056	0.83021	0.83535
Noise 2	7694	0	46019	1553	0.97190	1.00000	0.83205	0.90833
Constant	9087	734	45341	104	0.98484	0.92526	0.98868	0.95592
Drift	8428	234	45728	876	0.97992	0.97299	0.90585	0.93822

False Negatives in the Normal Class. As established earlier, false negatives (FN) for the normal class correspond to false alarms in this system. Comparing the two models, Comp3 produced only thirteen FN, whereas AIO yielded 101. This disparity is reflected in their recall values for the normal class:

- **Comp3:** 0.99859
- **AIO:** 0.98903

Although the numerical difference in recall appears small, the roughly tenfold increase in FN for AIO highlights a substantive shortfall in correctly recognising normal behaviour—an effect that could translate into frequent and disruptive false alarms in real-world deployments.

False Positives and Precision. Recall alone, however, does not provide the full

picture. A high number of false positives (FP) indicates that anomalies are being misclassified as normal, which compromises detection reliability. As shown in Tables 3.4 and Table 3.5, Comp3 generated slightly more FP overall than AIO (5416 vs. 5351), a difference reflected in their overall precision scores, with AIO performing marginally better.

Per-Class Precision on the Normal Class. Examining precision for the normal class specifically:

- **AIO:** 0.75917
- **Comp3:** 0.68935

This indicates that AIO was more consistent in avoiding false alarms, distributing its FP errors more evenly across anomaly classes. In contrast, Comp3 concentrated a larger proportion of FP errors within the Normal class itself.

F1-Score Comparison. To balance these trade-offs, the F1-score was calculated as the harmonic mean of precision and recall, penalising imbalances between the two. The results are:

- **AIO:** 0.90318
- **Comp3:** 0.90200

This marginal difference suggests that AIO provides slightly more balanced performance across the precision–recall spectrum. For this application, the F1-score offers a fairer and more informative evaluation than accuracy alone, as it captures both false alarms and missed detections within a unified measure.

From an IoT intrusion detection perspective, these findings indicate that although Comp3 exhibits fewer false alarms, AIO achieves a more general and balanced detection capability across multiple anomaly types. In operational deployments, the choice between these models would depend on the intended tolerance for false alarms versus missed detections—a trade-off central to practical IDS design.

3.7 Implementation Results

The Maxim microcontroller used in this work imposed strict resource constraints, the most notable being its support for 8-bit integer weights on its NPU. To satisfy this requirement, all models were trained using QAT. This ensured

that, upon completion of training, the models could be directly converted to efficient 8-bit representations with minimal degradation in accuracy.

3.7.1 Post-Quantisation Evaluation

As outlined earlier, all reported performance metrics were obtained *after quantisation*, meaning they reflect the model's behaviour exactly as it would appear when deployed on the IDS board. No floating-point post-processing or non-quantised inference was used at any stage. This evaluation approach provides a true measure of real-world performance and ensures reproducibility on the target hardware.

3.7.2 Model Size and Memory Footprint

Each model comprised seven parameterised layers, totalling 55,941 trainable parameters. Because the weights were stored as 8-bit integers, the deployed model on the MAX78000 occupied the following memory resources:

- **55,941 bytes** used from a total of 442,368 bytes available for weight storage ($\approx 13\%$)
- **6 bytes** used from a total of 2,048 bytes available for bias memory ($\approx 0.003\%$)

This level of compactness highlights the exceptional efficiency of the model, particularly in terms of parameter density and memory utilisation—an essential property for deployment in low-power, memory-constrained IoT devices.

3.7.3 Runtime Performance and Power Consumption

To assess runtime performance, the deployed IDS model was tested on the MAX78000 by executing continuous inference cycles on a live stream of test samples. Each inference completed in approximately 2.556 ms, with an average power consumption of only 15 mW during operation.

The temporal span of each input window can be calculated as

$$T = \frac{\text{Number of samples}}{\text{Sampling rate}}.$$

For a window of 4096 samples at a sampling rate of $250{,}000$ Hz,

$$T = \frac{4096}{250{,}000}\ \text{s} \approx 0.016384\ \text{s} = 16.384\ \text{ms}.$$

Thus, each window represents approximately 16.4 ms of power data. As inference requires only 2.556 ms, the system comfortably achieves faster-than-real-time processing, operating well within both the timing and energy budgets of embedded environments.

These results demonstrate the successful deployment of a lightweight CNN-based intrusion detection system capable of executing entirely on embedded hardware, without reliance on external compute resources. The framework balances strong classification accuracy with extremely low memory usage and power consumption, affirming its suitability for practical, host-based IoT intrusion detection and broader edge-computing scenarios.

3.8 Summary

This Chapter has presented a host-based anomaly detection system for an IoT platform, leveraging a lightweight Convolutional Neural Network (CNN) accelerated on the Neural Processing Unit (NPU) of an edge device. The system monitors the power-consumption signature of a target device and detects deviations from expected behaviour that may indicate either malicious activity or operational faults.

The CNN was trained on a combination of normal power behaviour and synthetically injected anomalies, resulting in two standout models, Comp3 and AIO, which achieved F1-scores of 0.90200 and 0.90318, respectively. These results demonstrate high classification accuracy while maintaining a balanced trade-off between false positives and false negatives in a multi-class anomaly detection context.

A key contribution of this Chapter lies in the generality and portability of the proposed framework. It is adaptable to a wide range of embedded systems—from ultra-low-power IoT nodes to high-power industrial controllers. The framework's resource footprint is exceptionally small, requiring:

- only *55 KB* of weight storage, and
- approximately *15 mW* of average power consumption during inference,

while still achieving faster-than-real-time inference (2.5 ms per inference vs. 16 ms per window), leaving sufficient time budget for other on-device operations.

These characteristics confirm the feasibility of deploying power-based intrusion detection entirely on constrained hardware—addressing the first major aim of this thesis: establishing the practicality of side-channel power data as a defensive signal for IoT intrusion detection. Furthermore, although the current implementation targets a neural network accelerator (MAX78000), the same architecture can be adapted to generic microcontrollers, reinforcing its potential for scalable, low-cost deployment across heterogeneous IoT ecosystems.

As the first exploration of this topic within the thesis, several limitations were identified that directly inform subsequent research directions and connect to the broader aims outlined in Chapter 1.

1. Synthetic versus real anomalies. While synthetic anomalies are widely used in early-stage IDS research, meaningful progress requires extending to real adversarial behaviour. The MAX78000's lack of network connectivity prevented the execution of live attack experiments, constraining evaluation to artificially injected anomalies. This limitation directly motivates the second thesis aim—to *to progress from synthetic anomaly injection to realistic attack scenarios*—and underlines the need for future work using connected devices and controlled IoT testbeds capable of capturing genuine power traces under realistic attack conditions.

2. Learning paradigm. The CNN models presented in this Chapter were trained in a fully supervised manner using labelled normal and anomalous samples. However, because real-world anomalies are inherently unpredictable and often unavailable during training, such a paradigm may fail to generalise to operational environments. This observation aligns with the third thesis aim—to *investigate alternative learning paradigms*—and motivates the transition in Chapters 5–6 to self-supervised approaches that model normal behaviour without reliance on labelled attack data.

3. Dataset bias and anomaly injection. The anomaly-injection process involved cloning normal segments before applying artificial perturbations, introducing a potential bias whereby paired normal and anomalous versions of identical samples appeared during training. Although this may explain the strong in-distribution performance observed, it may also limit generalisation to com-

plex or previously unseen anomalies. This finding informs refinements to evaluation methodology and dataset splitting in later chapters.

4. Data representation. Transforming 1D power traces into 2D grayscale images—while compatible with the CNN architecture employed—may sacrifice fine-grained temporal information intrinsic to side-channel signals. To address this, later chapters explore more advanced architectures capable of directly processing raw 1D time-series data, including those inspired by modern designs such as the Conformer and masked autoencoding approaches.

In summary, this Chapter fulfils the *first aim of the thesis* by demonstrating that side-channel power data can be effectively leveraged for intrusion detection using lightweight CNNs suitable for embedded deployment. It establishes a baseline methodology from which the more advanced self-supervised models of Chapters 5 and 6 evolve, while also highlighting the need for real attack datasets and more expressive learning paradigms to overcome the limitations of synthetic anomalies and supervised training.

Chapter 4

Attacks on IoT—An Analysis of the Signatures of Attacks in Device Power Consumption

4.1 Introduction

Chapter 3 demonstrated the feasibility of using side-channel power data for intrusion detection through lightweight, supervised Convolutional Neural Network (CNN) models deployed on embedded hardware. While this validated the concept of power-based IDSs for IoT, it also highlighted some key limitations—most notably the reliance on synthetic anomalies and the absence of real attack data.

This Chapter addresses these limitations by constructing two realistic IoT testbeds, based on Raspberry Pi and DragonBoard platforms, to capture genuine power traces under normal operation and controlled cyberattack conditions. Using this setup, the study investigates whether real-world attack behaviour can be reliably characterised from power signatures alone, or whether certain attacks cannot be detected or recognised in power data at all. This transition from synthetic to real anomalies marks a critical step towards grounding the research in realistic IoT conditions and advancing the thesis's second aim.

A key limitation identified in Chapter 3 was the reliance on synthetic anomalies and the constrained representation of normal behaviour. While synthetic anomalies can be a useful proxy in many domains and are used regularly [107],

they rarely capture the diversity and noise of real-world conditions [82]. The limited availability of genuine attack examples reflected in power data, therefore, suggested that far greater insight could be gained by capturing live power traces from actual IoT intrusions.

This Chapter seeks to extend and promote the use of non-intrusive, side-channel power data in cybersecurity, not as an attack vector, but rather as one for monitoring and defence. Power consumption—an inherent property of all computing devices—reflects the underlying operations and workloads active on a system [23, 89]. As discussed throughout the thesis thus far, every electronic device, regardless of scale or power level, consumes energy in a manner that correlates with its internal behaviour. Consequently, substantial effort is often directed towards obscuring or randomising power characteristics in critical System-on-Chip (SoC) components to mitigate side-channel leakage.

Here, side-channel power data are leveraged defensively: by monitoring the device's power behaviour, one can infer and analyse its baseline normal state and identify deviations caused by cyberattacks. This perspective enables a broader, system-level approach to security—one that considers attacks from alternative angles rather than focusing solely on network-layer data. By examining side-channel power traces, a more complete understanding of attack dynamics and defensive responses can be achieved.

A common shortcoming in prior works on power-based intrusion detection is that many studies report only on model performance, without showing or analysing the underlying power data itself. As a result, the reader gains little understanding of the attacks or their observable effects. This Chapter directly addresses that gap by visualising and explaining how real attacks can manifest in the power domain.

To this end, two realistic IoT testbeds were developed, each built around a different embedded platform (Raspberry Pi and DragonBoard 410c). Baseline normal behaviour for both devices was first established and recorded as time-series power traces. These traces were analysed to identify characteristic features of normal operation and to compare any cross-device similarities. The same testbeds were then subjected to a range of common cyberattacks—specifically reconnaissance, Denial-of-Service (DoS), and brute-force attacks—allowing the corresponding anomalous power behaviours to be studied in detail.

Finally, to simulate multi-stage intrusion, comprehensive *Capture-the-Flag*

(CTF) scenarios were conducted. In these exercises, multiple attack types were executed sequentially to compromise the testbed, explore its environment, and exfiltrate a concealed file.

By analysing both individual attacks and the combined CTF scenarios, this Chapter demonstrates that side-channel power traces can reveal distinct and interpretable signatures of malicious activity. Such visibility can enable security teams to reverse-engineer and diagnose attack behaviour from observed power data, enhancing post-incident understanding and forensics.

This work, therefore, represents a major progression from the foundational experiments in Chapter 3, transitioning from synthetic anomaly generation to the capture and practical analysis of real attacks. Through the execution of controlled cyberattacks, the research provides one of the first comprehensive analyses of how these behaviours appear in side-channel power data.

Accordingly, this Chapter serves as the foundation for the data-driven and AI-based detection frameworks developed in later chapters—culminating in the self-supervised Transformer-based models of Chapter 6.

4.1.1 Architectural Assumptions

The experimental design in this Chapter is based on a set of assumptions regarding device operation, data collection, and attack execution within controlled IoT environments.

- IoT devices (Raspberry Pi and DragonBoard) are instrumented to enable **direct measurement of power consumption** during operation using a power analyser.
- Devices are deployed within controlled testbed environments, allowing repeatable execution of both normal workloads and adversarial scenarios.
- Cyberattacks are executed locally or within the same network environment, ensuring that their effects are observable within the device's power consumption.
- The monitored workloads are designed to approximate typical device operation as a simple Bluetooth sensor hub. In particular, normal behaviour consists of a periodic script that parses Bluetooth data at fixed intervals (every 5 seconds), combined with background operating system activity (e.g., scheduling, I/O handling, and system services). This setup provides

a stable and repeatable baseline of normal behaviour, while acknowledging that it does not capture the full diversity of real-world application workloads.

- Power traces are assumed to capture sufficient temporal resolution to reflect underlying computational activity and attack-induced deviations.
- The cyberattacks selected in this Chapter are chosen to represent a range of commonly encountered IoT threat behaviours, including reconnaissance, brute-force, and denial-of-service scenarios. These attacks span different stages of the attack lifecycle and exhibit varying computational and temporal characteristics, making them suitable for analysing how diverse adversarial behaviours manifest in device power consumption. The selection is not exhaustive, but is intended to provide representative and practically relevant attack patterns for evaluating the detectability of anomalies within the power domain.

These assumptions enable systematic analysis of how real cyberattacks manifest in device-level power behaviour, forming the basis for subsequent modelling approaches.

4.2 Testbed Creation

For the experiments in this Chapter, two devices were selected to act interchangeably as IoT hubs in the testbed: the Raspberry Pi 3 Model B [108] and the DragonBoard 410c [109]. Unlike the MAX78000 featured in Chapter 3, both devices provide extensive connectivity options, including Wi-Fi and Bluetooth, providing a more realistic platform for evaluating attack behaviour.

The DragonBoard 410c, manufactured by Arrow Electronics and based on the Qualcomm Snapdragon 410 processor [110], offers considerable processing capability alongside multiple wireless interfaces. The Snapdragon 410 was a highly popular SoC in the mid-2010s, with Qualcomm shipping over 200 million units [111]. Consequently, any side-channel power signatures observed for this processor type may hold relevance for a broad class of embedded and mobile devices that share similar architectures.

The Raspberry Pi, by contrast, is one of the most widely adopted single-board computers, with over 40 million units sold since its initial release [112]. Like the DragonBoard, it offers the necessary processing and networking capabilities

to serve as a representative IoT test platform. In comparison to the more constrained MAX78000 used in Chapter 3, these two boards support a wider and more realistic set of common attack scenarios, thereby improving the validity of the experiments.

Both the DragonBoard and the Raspberry Pi feature quad-core 1.2 GHz, 64-bit CPUs and approximately 1.2 GB of RAM. In the DragonBoard, the CPU forms part of the Snapdragon 410 SoC, which integrates the Qualcomm APQ8016E application processor and an Adreno 306 GPU. Despite differing manufacturers, release periods, and intended use cases, the two platforms exhibit strikingly similar specifications, allowing for cross-device comparison under common attack types.

These hardware choices enable a broader investigation into whether attack signatures observed in power data are generalisable across heterogeneous IoT systems or remain device-specific. While identical signatures cannot be expected due to some architectural variations, any consistent patterns could significantly aid the development of more generalisable intrusion detection frameworks for IoT networks comprising diverse hardware. Given the combined estimated deployment of around 245 million units between the DragonBoard 410c and Raspberry Pi families, the devices studied here represent a substantial proportion of real-world IoT hardware.

To maximise explainability while retaining realism, each testbed was kept deliberately simple, with minimal functionality and software installed. The Raspberry Pi ran Pi OS [113], which includes essential utilities such as Bluetooth support by default. The DragonBoard, operated on Linaro Debian [114]—a Debian-based distribution optimised for the Snapdragon 410—and required manual installation of these features.

Both boards were connected to the internet via Wi-Fi, with the Secure Shell (SSH) protocol enabled to widen the attack surface (creating an opening for the attacker). SSH [115] allows remote login, data transfer, and command execution—capabilities commonly enabled on IoT devices for maintenance or monitoring. However, recent studies indicate that approximately 15% of IoT device owners do not change default credentials, meaning that as few as five passwords can grant access to around 10% of all IoT devices globally [43]. With an estimated 55 billion IoT devices expected by 2025, this could leave over 5 billion devices vulnerable [15].

Bluetooth connectivity was also enabled on both testbeds to communicate with a TI CC2650STK SensorTag [116], equipped with ten low-power MEMS sensors. A simple Python script [117] running locally on each device requested sensor readings every five seconds via Bluetooth, parsing the received packets in real time.

Both boards were powered by, and connected to, an Agilent Technologies N6705A DC Power Analyser [118]. This device supplied stable DC power while simultaneously capturing high-resolution current measurements, allowing non-intrusive observation of each board's side-channel power behaviour without distorting the signal. Figure 4.1 illustrates the generic data acquisition framework used for IoT power monitoring in this study.

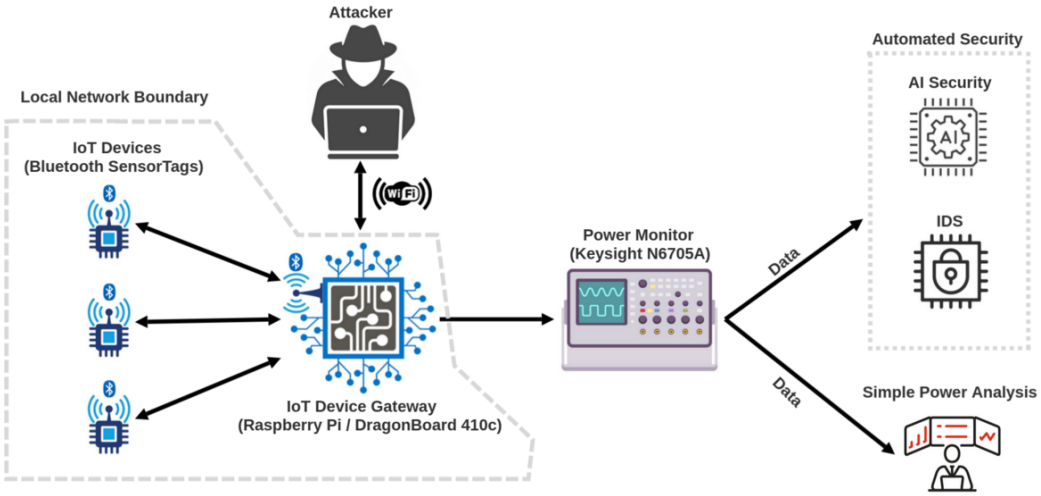


Figure 4.1: IoT power data acquisition framework used in this Chapter, illustrating the balance between human-in-the-loop oversight and automated (AI-driven) security analysis.

4.3 Data Collection

Using the Agilent N6705A, the testbed power was both supplied and monitored. Through the data-logging function of the DC Power Analyser, the current drawn by each device was recorded at approximately 50 kHz (one sample every 0.02048 ms). While the N6705A is capable of significantly higher sampling rates, 50 kHz represents the upper limit of its data-logging mode. This setting was chosen to enable longer capture durations, which are essential for obtaining a wide and representative sample of device operations. This relatively high sampling rate was selected because, until the signatures were actually analysed,

it was unknown how far the sampling rate could be reduced before the resolution of the signatures was compromised beyond recognition.

The data presented in this Chapter are shown exactly as captured, without any pre-processing. Although several pre-processing methods were explored during the development of this work, the decision was made to present the raw measurements as observed by the power analyser. This approach prioritises transparency and allows readers to directly interpret the inherent characteristics of the recorded side-channel signals.

It should be noted that many other power and current monitoring options exist, and using high-end laboratory equipment such as the N6705A is not a strict requirement for reproducing this type of analysis. As demonstrated by [89], inexpensive and non-intrusive sensors—such as plug-level power monitors—can also yield useful insights into device power behaviour in other domains, including High-Performance Computing (HPC) systems. However, cheaper sensors tend to lack the high sampling rates obtained in this initial test.

4.3.1 Normal Power Behaviour

While the components of the testbed have already been introduced, it is useful to clarify how they form the testbed's normal baseline behaviour. As discussed in Chapter 3, IoT devices typically exhibit simple, repetitive workloads: they perform a fixed task at regular intervals, looping continuously.

In this testbed, each device was connected to a single Bluetooth SensorTag. A Python script running locally on the testbed requested sensor readings every five seconds via Bluetooth and parsed the returned packets to display the measured values. This behaviour reflects a realistic, continuous IoT use case—mirroring the simple, periodic functionality common to many consumer and industrial sensor systems.

Figure 4.2 shows three representative five-second segments of normal side-channel current for each device. These time windows (0–5 s, 35–40 s, and 70–75 s) were selected to illustrate typical patterns observed in the dataset. The comparison highlights clear similarities in the normal power signatures of both devices, despite the Raspberry Pi drawing roughly twice the current of the DragonBoard (0.4 A versus 0.2 A). The characteristic spike patterns corresponding to Bluetooth transmissions exhibit similar temporal structure, with both boards completing each communication cycle within the same timeframe.

The Raspberry Pi's power profile, however, is noticeably less smooth than that of the DragonBoard, exhibiting background current fluctuations likely caused by additional operating system processes. Such variations may obscure minor attack signatures by masking them with low-amplitude noise. This will be revisited later in the analysis.

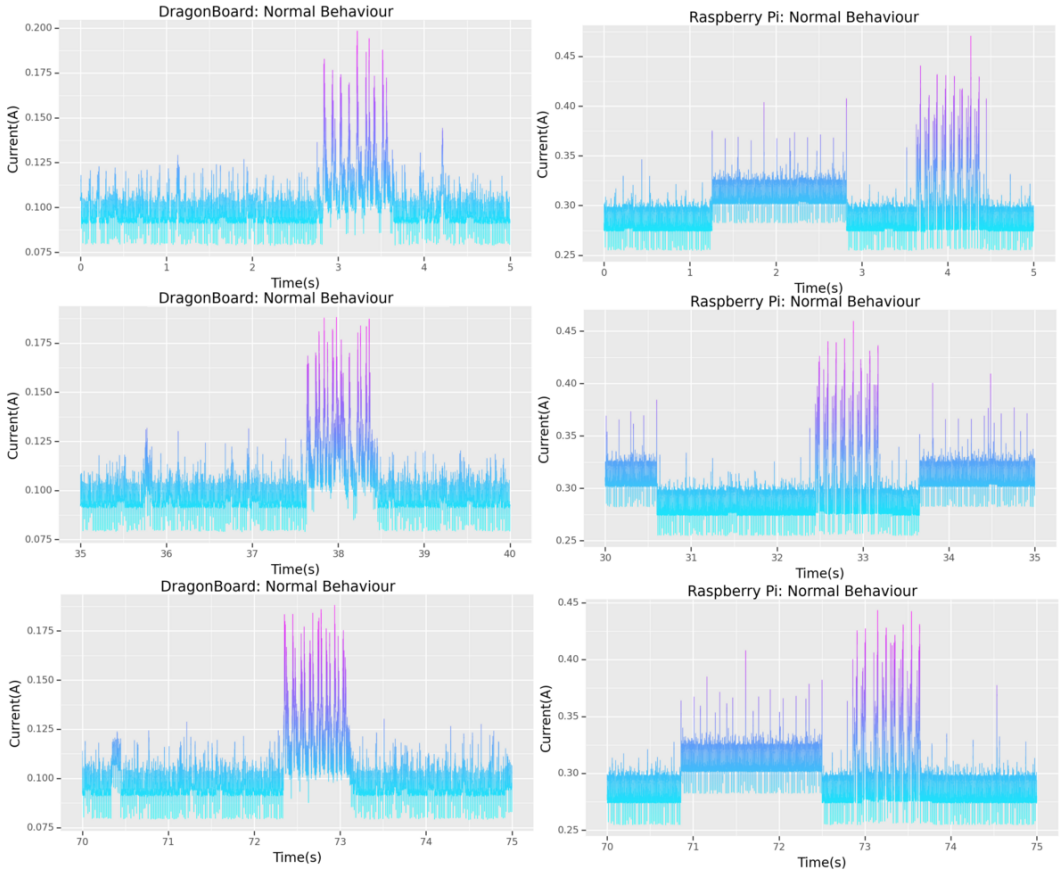


Figure 4.2: Normal power signatures for the DragonBoard and Raspberry Pi. Note the strong similarity in signature shape between the two devices despite the amplitude difference.

In Figure 4.2, as in subsequent figures, side-channel power data is visualised using colormaps of increasing intensity [119]. The scale is based on the CISA National Cyber Incident Scoring System (NCISS) [120], where blue represents baseline conditions and black indicates an emergency or critical state.

4.4 Attacks

In contrast to Chapter 3, the attacks applied in this Chapter are real and commonly observed rather than synthetic. The testbed was subjected to three broad

classes of attack, each corresponding to one of the core principles of the CIA triad [37]: confidentiality, integrity, and availability. These categories were represented, respectively, by reconnaissance, brute-force, and DoS attacks.

4.4.1 Reconnaissance

Reconnaissance attacks primarily threaten *confidentiality* by attempting to gather information about or from a target system or network. Such information can include open services and ports, software versions, user accounts, and peer connections. Even seemingly benign details can be exploited to develop more targeted attacks.

In this work, the testbed was subjected to a classic *port-scanning* attack—a reconnaissance technique used to map a system's network surface and identify potential exploitable entry points.

4.4.1.1 Port Scanning

Port scanning is a well-established reconnaissance technique employed by both attackers and defenders to evaluate system exposure. By sending structured packets to a range of ports and analysing the responses, a scanner can determine which ports are open, closed, or filtered, and infer what services or software are listening on those ports. This information is often the first step in a "cyber kill chain" and is crucial for identifying outdated or unpatched services that may contain known vulnerabilities.

In this experiment, the open-source tool *Nmap* [121] was used to conduct the attack. Nmap is one of the most widely adopted network scanning tools and is pre-installed on many penetration-testing distributions, such as *Kali Linux* [122]. Nmap sends probing packets, analyses responses, and provides detailed host and service discovery. This study focused on the tool's timing and performance profiles, which are categorised as follows:

- **Insane (T5)**—ultra-fast scanning for extremely high-performance networks.
- **Aggressive (T4)**—fast scanning for reliable, high-bandwidth networks.
- **Normal (T3)**—default scan timing.
- **Polite (T2)**—reduced bandwidth and lower load on target systems.

- **Sneaky (T1)**—evasion-focused scanning with substantial delays between packets.
- **Paranoid (T0)**—extreme evasion mode with very long delays.

Figure 4.3 compares the side-channel current signatures observed across these timing settings for both devices. Clear similarities are seen in the higher-speed configurations (T3–T5), which share a common signature shape, differing mainly in amplitude. The Raspberry Pi exhibited higher current magnitudes (around 0.6 A) than the DragonBoard (around 0.2 A), reflecting its greater baseline power consumption.

Interestingly, despite their aggressive naming, the "Insane" (T5) and "Aggressive" (T4) modes did not produce markedly stronger current signatures than the default (T3). This is likely because both devices are resource-constrained, IoT-class systems unable to sustain extremely high packet rates. Consequently, the power signatures plateaued, producing patterns more consistent with moderate scan intensity. On more capable systems or faster networks, these modes would likely exhibit greater differentiation.

The lower-speed and IDS-evasive configurations (T2–T0) reveal a contrasting pattern. These settings increase delays between probes to minimise detection by intrusion detection systems. As shown in the lower half of Figure 4.3, this temporal spreading flattens the distinctive "Nmap spike" visible in T3–T5. For the DragonBoard, the "Polite" (T2) mode caused a persistent but subtle elevation in current, visible as a gentle perturbation in the background signal. The Raspberry Pi exhibited a smaller increase (from approximately 0.25 A to 0.30 A) across the duration of the scan. In both cases, the disturbance persisted for the entire attack period. For the DragonBoard, a short duration spike similar in amplitude to Bluetooth data was also witnessed.

At the extreme end, the IDS-evasion modes (T1 and T0) were effectively undetectable through visual inspection of the side-channel traces. Their low activity levels and long inter-probe delays produce minimal deviation from baseline operation, resulting in no sustained or recognisable signature for a time-series classifier to learn. In principle, highly sensitive statistical or frequency-domain methods—such as Simple Power Analysis (SPA) or Differential Power Analysis (DPA) [23, 123]—could reveal subtle, structured perturbations. Although these techniques are traditionally applied in cryptanalytic contexts, their ability to identify weak but consistent patterns suggests potential applicability to side-

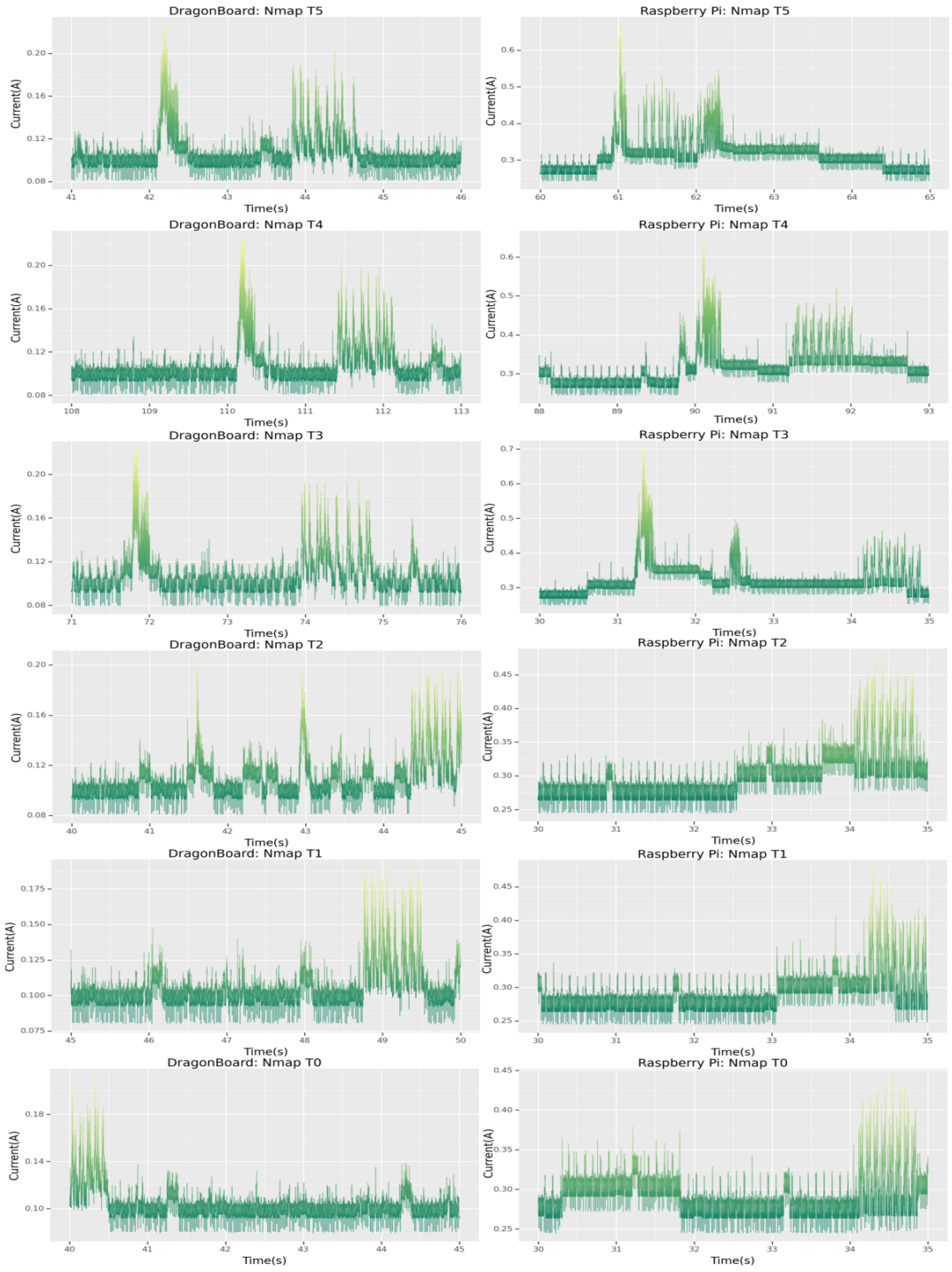


Figure 4.3: Each Nmap T-Switch behaviour for DragonBoard versus Raspberry Pi (data represented using the "summer" colormap [119]).

channel-based intrusion detection.

Thus, for the purposes of this study, modes T1 and T0 were excluded from quantitative evaluation. Their extremely slow probing behaviour (with T0 emitting a packet roughly once every five minutes) does not generate sufficient tempo-

ral density to form a meaningful power signature within the capture windows used. Consequently, while theoretically detectable under different methodologies or much longer sampling durations, T1 and T0 fall outside the practical scope of the detection approach examined in this Chapter. The entries for T1 and T0 in Figure 4.3 serve to show the lack of a present signature, rather than a specific signature in the window.

4.4.2 Brute-force

Brute-force attacks, like reconnaissance attacks, primarily compromise the confidentiality of the target. Unlike reconnaissance, which focuses on gathering information, brute-force attacks exploit intelligence obtained during earlier phases to gain unauthorised access. In essence, brute-force methods aim to overwhelm authentication mechanisms by systematically trying credentials until success is achieved. A common example is password cracking, where an attacker repeatedly attempts combinations of usernames and passwords until the correct one is found. In this category, an SSH brute-force attack was deployed against the testbed.

4.4.2.1 SSH Brute-force

In the scenario designed for this testbed, the prior port-scan reconnaissance revealed that the SSH service was active on the target. Equipped with this knowledge, an attacker may attempt to brute-force the SSH login credentials.

This form of brute-force is essentially a trial-and-error process in which the attacker tries numerous passwords from a known list—such as the widely used "RockYou" dataset [124]—against a known or guessed username. If the username is not known, the attack can iterate through two separate lists simultaneously: one containing candidate usernames and another containing passwords. Such credential lists are easily accessible online and often come pre-installed on penetration-testing distributions like *Kali Linux*.

Once the attacker successfully guesses the correct password, they can gain remote access to the target via SSH. If the compromised account has administrative or "sudo" privileges, the consequences can be severe: unrestricted file access, system configuration changes, and lateral movement across other networked devices that share the same credentials. However, even if only a standard user account is breached, privilege-escalation attacks may still grant root-

level access by exploiting known software vulnerabilities. Online repositories such as *GTFOBins* [125] catalogue numerous privilege-escalation techniques leveraging misconfigured binaries and outdated packages commonly installed on out-of-date systems.

The parallelised network login cracker *Hydra* [126] was used to conduct the SSH brute-force attack. Hydra allows multiple parallel connection attempts, with the parameter "`t`" controlling the number of concurrent threads per target. Higher "`t`" values correspond to greater parallelism and, therefore, increased attack intensity. All other settings were left at their defaults. A subset of the RockYou wordlist was used, with the testbed password either included (to simulate successful compromise) or excluded (to simulate ongoing attack attempts).

Figure 4.4 illustrates how the side-channel current signature scales with the "`t`" parameter. As expected, higher levels of concurrency lead to greater current amplitudes and longer-duration signatures. At low "`t`" values, the signature remains comparable to normal Bluetooth activity, but with multiple parallel connections the brute-force signature becomes dominant, clearly distinguishable from normal behaviour on both devices.

It would be unlikely for a real attacker to use a setting as low as "`t=1`", as this would dramatically increase attack duration unless the potential credential set was extremely small. The signatures observed for higher "`t`" values share consistent temporal and morphological traits across both devices. At the same time, the signatures change drastically when comparing one "`t`" value to another (e.g., "`t=4`" vs. "`t=16`"). This indicates that while the signatures for a given "`t`" value can be consistent across devices, they can differ significantly between "`t`" values, even on the same device.

The more powerful Raspberry Pi sustains higher "`t`" settings for longer periods than the DragonBoard (notably "`t=16`" and "`t=32`" in Figure 4.4). This likely reflects the DragonBoard's lower-power design compared with the Raspberry Pi's performance-oriented architecture. Hydra's own documentation warns that settings above "`t=4`" may be unreliable, as many systems restrict the number of simultaneous SSH connections or fail to handle excessive parallel requests due to limited resources.

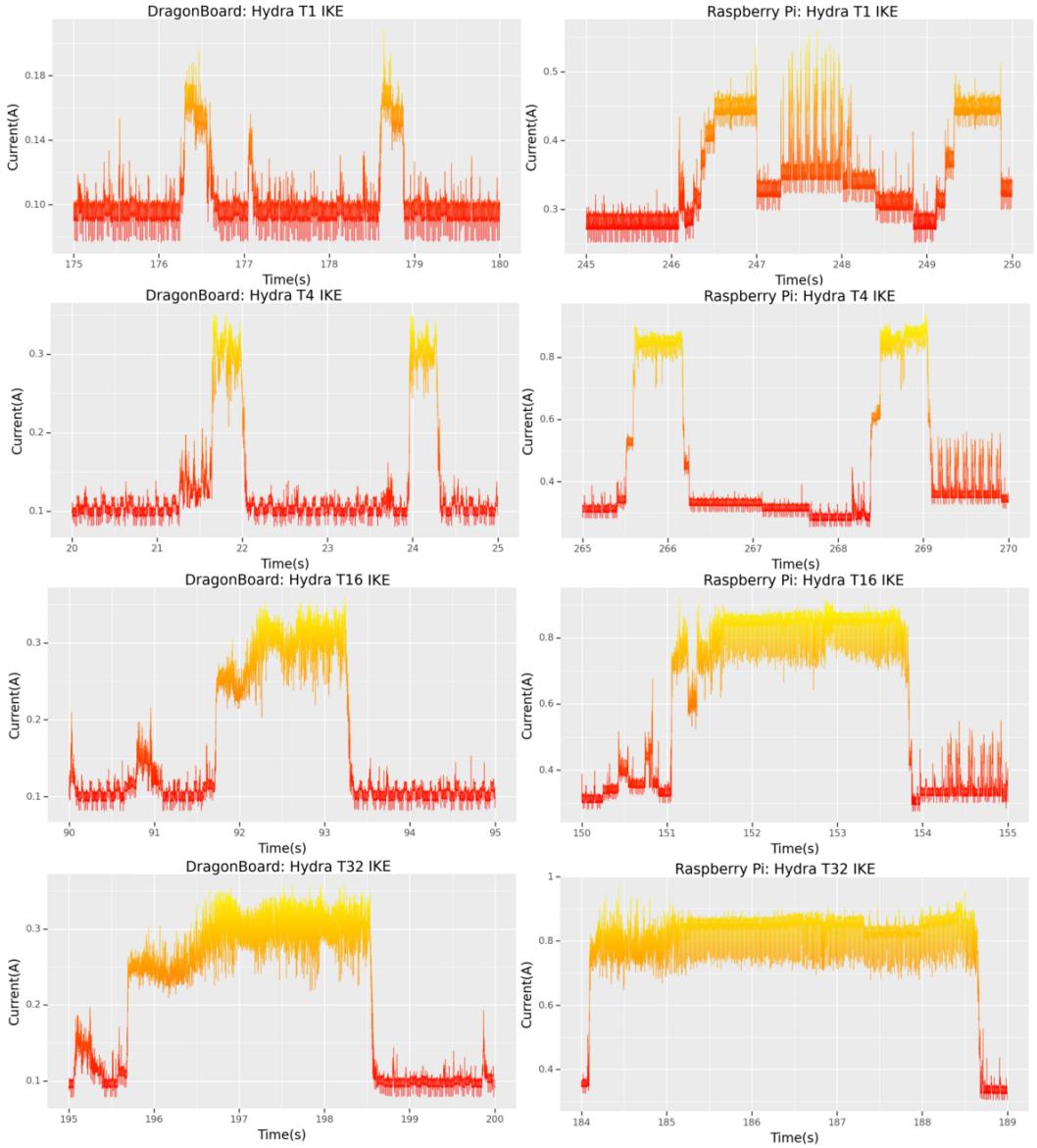


Figure 4.4: Incorrect Key Events (IKE) across Hydra `t`-switch settings for the DragonBoard and Raspberry Pi (data visualised using the "autumn" colormap [119]).

4.4.2.2 Correct-Key Event

During analysis of the Hydra-generated side-channel signatures, a distinct pattern was identified whenever the correct password was attempted. This phenomenon will be referred to as the *Correct Key Event (CKE)* throughout the remainder of the Chapter. Conversely, instances in which no valid password was found are denoted as *Incorrect Key Events (IKEs)*.

Interestingly, while observing the live traces on the power analyser, the moment of successful password entry could often be identified shortly *before* Hydra con-

firmed success on screen. This observation closely parallels the principles of SPA and DPA, where small fluctuations in device power consumption are used to infer key-dependent operations.

Figure 4.5 illustrates the CKEs observed across all Hydra settings. In the example shown for "`t=1`", the final successful attempt appears only milliseconds before the CKE spike on each device. Notably, regardless of the concurrency level (the "`t`" setting), both the Raspberry Pi and DragonBoard exhibited nearly identical CKE signatures. This suggests that the event corresponds to the precise instant Hydra authenticates using the correct SSH password and the system transitions from failed login attempts to a valid authenticated session.

The implications of detecting the CKE are significant for digital forensics and incident response. The ability to pinpoint the precise moment a credential was compromised enables investigators to reconstruct the attack timeline with high temporal accuracy. This level of precision narrows the window of uncertainty in post-incident reporting and can inform defensive countermeasures. For example, if the password was cracked unusually quickly, it would indicate weak credentials and highlight the need for stronger password policies or rate-limiting mechanisms.

Attackers seeking to obscure their activity may configure Hydra to continue attempting passwords even after a successful login, artificially extending the visible duration of the attack. This could mislead analysts into assuming that the attack ended later than the actual moment of compromise, while the intruder remains active. However, by reproducing the attack and cross-referencing the known CKE signature, forensic teams can determine, accurate within milliseconds, the exact moment of compromise. Such side-channel correlation provides a powerful tool for post-event reconstruction and demonstrates how physical-layer signals can enhance situational awareness in IoT security investigations forensically.

4.4.3 Denial of Service (DoS)

DoS attacks seek to compromise the *availability* of a target service by overwhelming it with excessive traffic or by exploiting implementation weaknesses to exhaust system resources. Such attacks are particularly dangerous for IoT devices, which often lack the computational capacity or network bandwidth to withstand sustained load. In this category, a SYN flood attack was deployed

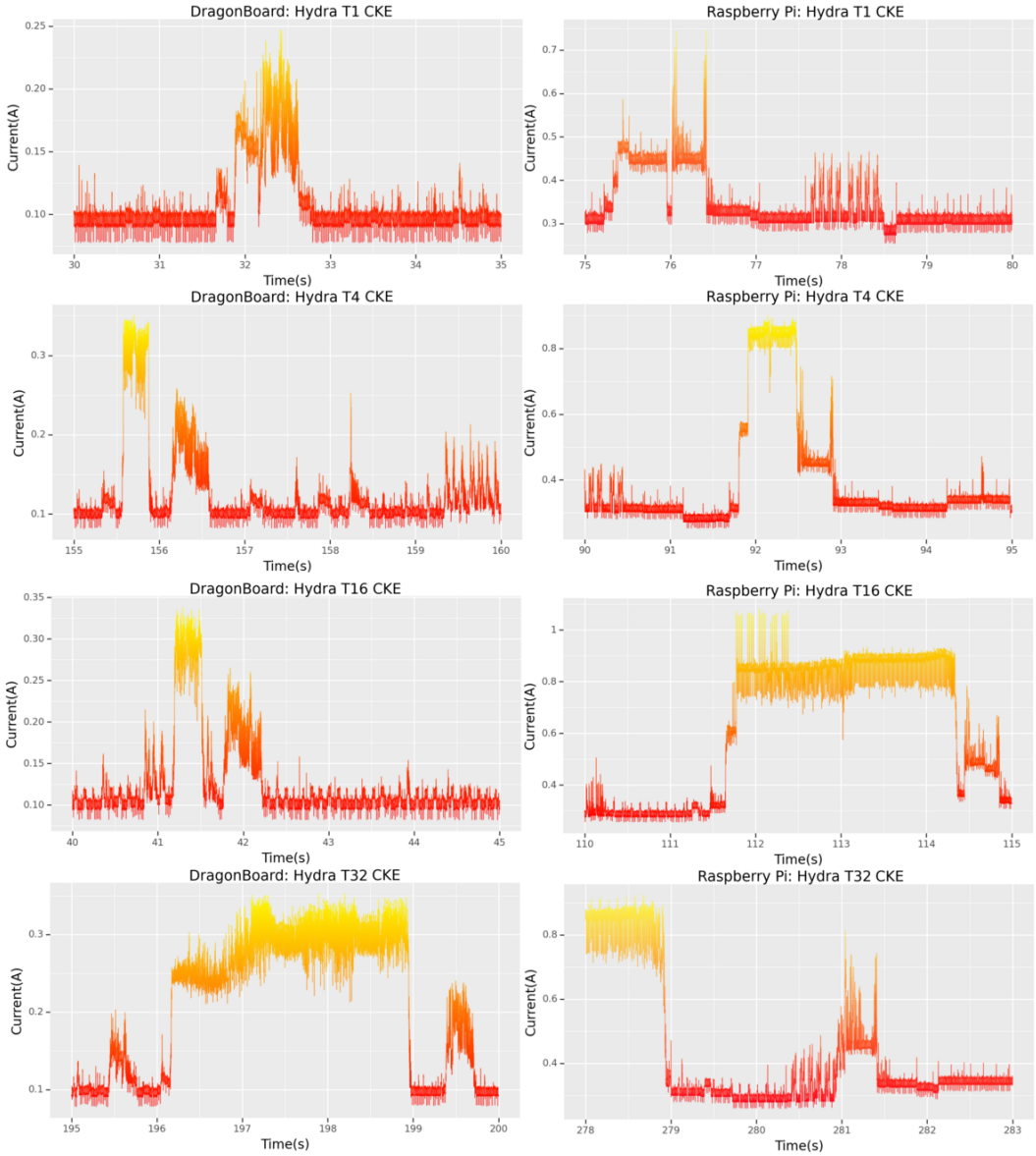


Figure 4.5: Correct Key Events (CKE) across all Hydra `t`-switch configurations for the DragonBoard and Raspberry Pi (data visualised using the "autumn" colormap [119]).

against the testbed to observe its effect on device-side power behaviour.

4.4.3.1 SYN Flood

Also known as a *half-open* attack, a SYN flood overwhelms a target by sending a large number of TCP connection requests (SYN packets) without completing the three-way handshake. Each unacknowledged request forces the target to maintain a half-open connection, consuming memory and CPU resources while requiring minimal effort and resources from the attacker. As the backlog of

pending connections grows, legitimate users experience delays or complete service denial.

This type of attack is one of the most efficient and is a widely observed form of DoS due to its simplicity and low resource requirements. It has been documented extensively in both traditional IT and IoT contexts. It is therefore a logical choice for this study's resource-constrained IoT testbed.

The attack was carried out using the *Hping3* tool [127], a versatile packet-crafting utility often employed for both network auditing and penetration testing. Hping3 was configured to continuously send TCP SYN packets to the target device until the backlog queue became saturated.

Figure 4.6 illustrates the first ten seconds of the SYN flood on both devices. In each case, normal device behaviour is clearly disrupted. The Raspberry Pi sustained a higher volume of SYN requests than the DragonBoard, resulting in a more pronounced current signature. However, its peak current draw of approximately 0.7 A did not exceed the amplitudes observed during the SSH brute-force experiments. This plateau likely reflects a saturation point where the Pi's available resources were exhausted, limiting its ability to process additional requests.

For the DragonBoard, the overall current draw remained lower, reflecting its low-power design and limited connection-handling capacity. Nonetheless, the attack caused complete disruption of normal activity, with sustained periods of elevated current. From a security perspective, this behaviour has two important implications.

First, it confirms that DoS activity produces a distinct, measurable power signature that can be used for detection even in low-layer monitoring scenarios. Second, it highlights the potential of energy-depletion attacks [45], where repeated DoS activity can deplete battery-powered IoT devices, forcing them offline. These findings illustrate how side-channel power monitoring can expose not only logical but also physical impacts of availability-based attacks in real time.

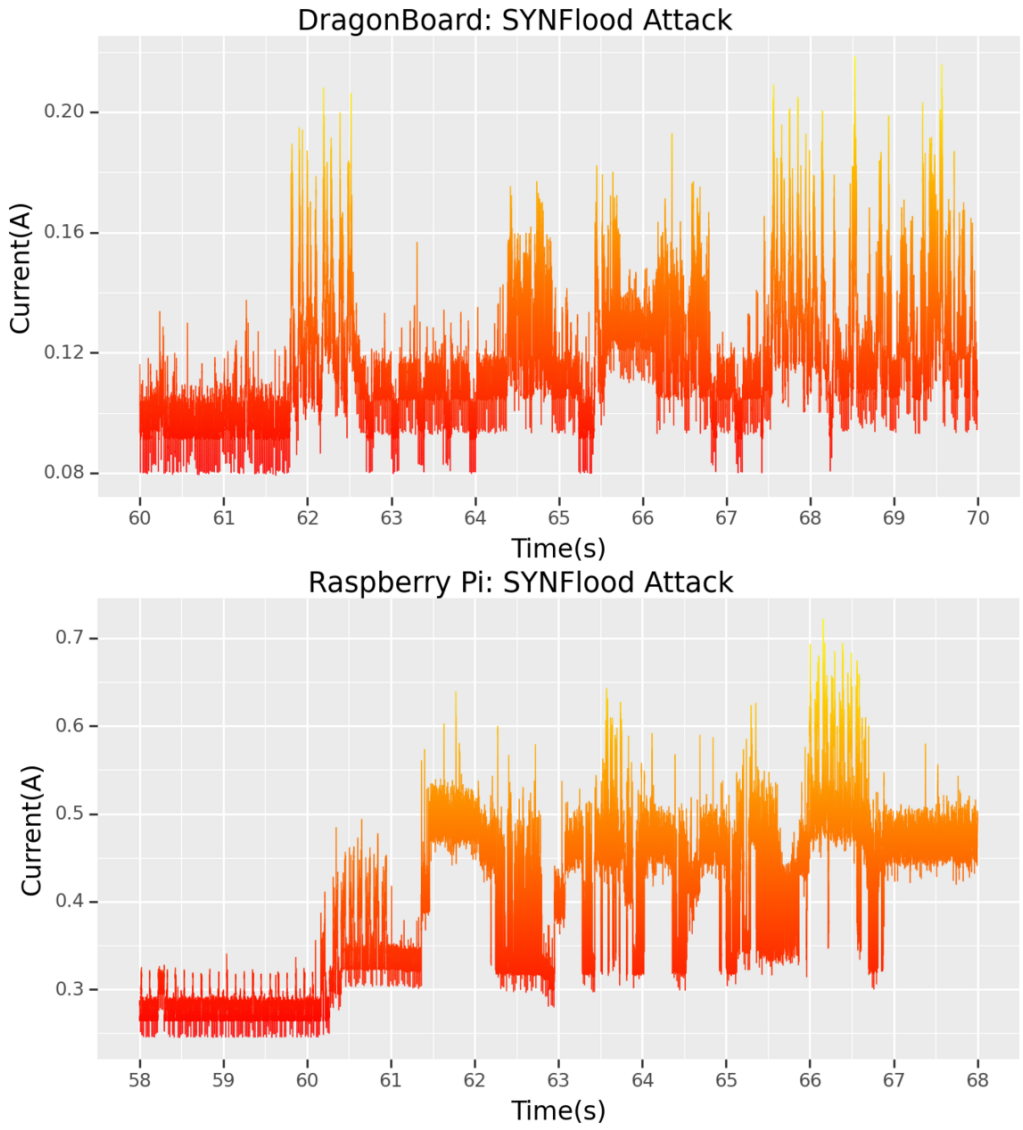


Figure 4.6: Comparison of SYN flood attacks on the DragonBoard and Raspberry Pi (data visualised using the "autumn" colormap [119]).

4.5 Capture the Flag Scenario

To unify and contextualise the work on this testbed, a CTF scenario was executed on both devices. In cybersecurity, CTFs are structured exercises in which a hidden file, or "flag", must be located and exfiltrated by an attacker through a series of exploits. Such scenarios are widely used in both research and education to simulate real-world intrusions and assess the effectiveness of defence mechanisms [128].

In this study, a concealed file was hidden on the testbed and subsequently retrieved through a chained sequence of attacks combining the techniques dis-

cussed in Section 4.4. The end-to-end process—from initial reconnaissance to successful data exfiltration—was executed without interruption. This design mirrors how a minimally secured IoT device might be compromised in the field through a progressive, multi-stage intrusion.

A key contribution of this experiment is that the entire compromise process can be represented visually through side-channel power traces in a single end-to-end time-series capture. By analysing these traces, it becomes possible to identify, segment, and label the different phases of the attack in real time. The CTF scenario is decomposed into six principal stages: *reconnaissance*, *brute-force*, *infiltration*, *exploration*, *exfiltration*, and *withdrawal*.

During the CTF analysis, the recorded data were cross-referenced with the findings from Section 4.4. Using this methodology, it demonstrates that the sequence of attack events may be reconstructed purely from side-channel power signatures, without prior knowledge of the actual order of execution. This approach demonstrates how forensic analysts could employ side-channel analysis to retrace attacker actions after a breach. By reproducing common attacks and comparing their resulting signatures to those recorded during an incident, analysts can establish a detailed reconstruction of the intrusion timeline, similar to other uses of side-channel data forensically [129].

4.5.1 Reconnaissance

The CTF begins with the reconnaissance phase, targeting the confidentiality of the testbed. As discussed in Section 4.4, reconnaissance is the essential preparatory stage for any attack, enabling adversaries to gather key intelligence about their target system. Information such as open ports, operating systems, or service availability can be critical for subsequent intrusion attempts.

4.5.1.1 Ping

Before commencing a detailed scan, an attacker typically issues a *ping* command to verify that the target device is reachable. Ping (Packet Internet or Inter-Network Groper) [130] is one of the oldest and most ubiquitous network utilities, used to test connectivity by sending Internet Control Message Protocol (ICMP) `echo_request` packets and awaiting corresponding replies. The process functions analogously to radar: a packet is sent, and a response indicates that the target is live.

Although a ping is not an attack in itself, it serves as a useful reconnaissance signal. In secure networks, repeated unsolicited ping requests may trigger alerts, as they often precede more invasive scans. However, in IoT environments—especially among consumer-grade devices—such protective measures are uncommon. Figure 4.7 illustrates the first five seconds of ping activity during the CTF.

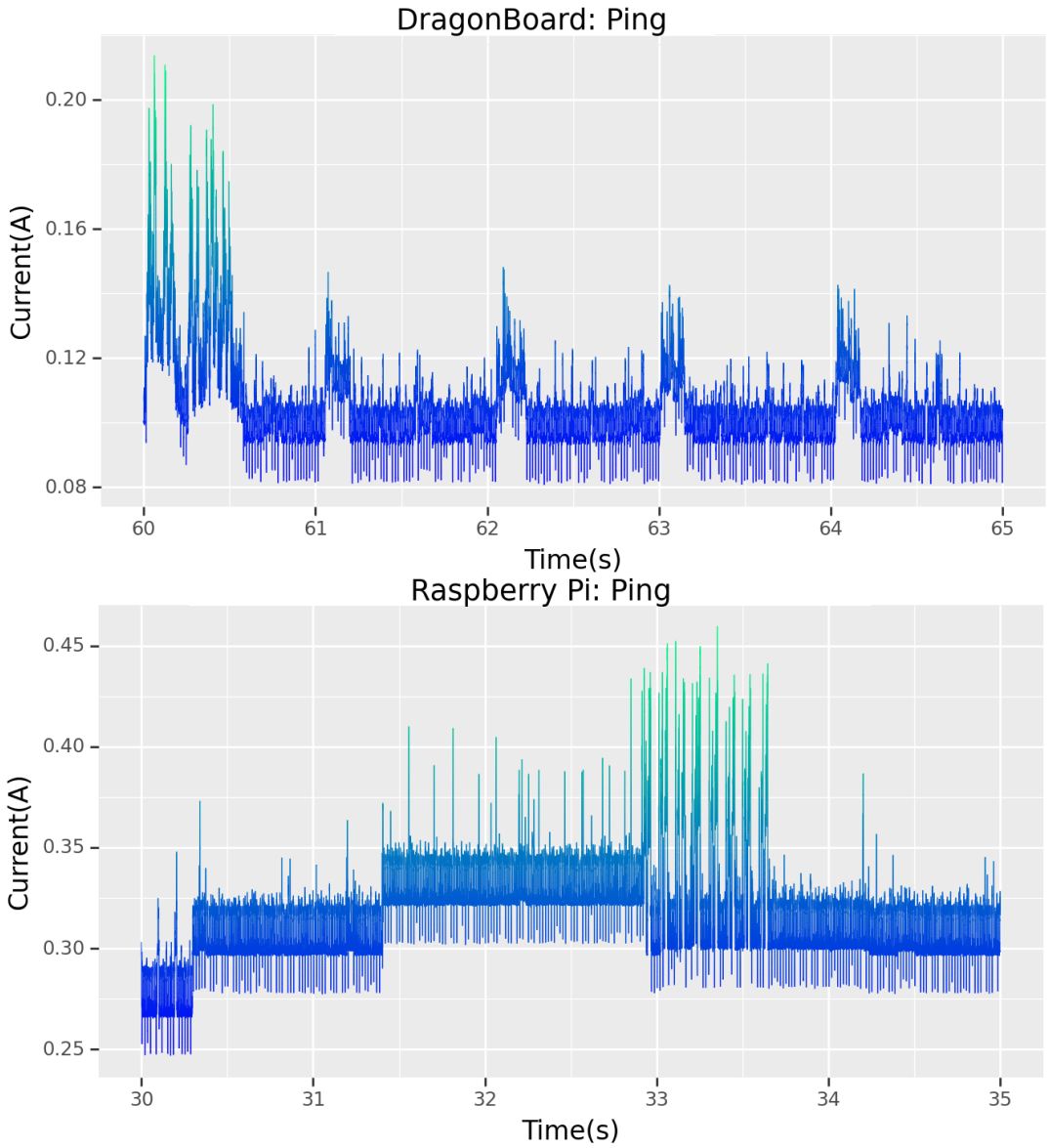


Figure 4.7: Comparison of ping behaviour for the DragonBoard and Raspberry Pi (data visualised using the "winter" colormap [119]).

On the DragonBoard, small but distinct power signatures corresponding to ping packets can be observed following normal Bluetooth activity. These signatures persist for as long as the pings continue. Although minimal in amplitude, their

repetitive structure can make them detectable over longer observation periods, possibly even using simple power analysis techniques.

For the Raspberry Pi, the ping-induced perturbations are subtler: no distinct spikes are visible in Figure 4.7, yet the overall current draw increases from approximately 0.25 A to 0.30 A during the ping sequence. Using statistical techniques such as segment-wise power averaging or temporal differencing, these perturbations remain identifiable. Such analysis underscores the sensitivity of side-channel power monitoring—even simple network probes can be reflected in device-level energy consumption.

4.5.1.2 Port Scanning

As in the earlier experiments, the tool *Nmap* was employed to enumerate open ports and identify active services on the target device [121]. Port scanning remains one of the most widely used reconnaissance techniques, forming a crucial stage of the "cyber kill chain". By analysing the resulting side-channel power traces and cross-referencing them with the baseline signatures obtained in Section 4.4.1.1, one can identify the distinctive power behaviour associated with this activity.

Figure 4.8 demonstrates the similarity between the CTF traces and the reference signatures captured during the standalone Nmap experiments. For both the Raspberry Pi and the DragonBoard, the behaviour closely matches the signatures corresponding to the T3 configuration (see Figure 4.3), which was the mode used during the CTF. This consistency reinforces the reliability of side-channel power traces as repeatable indicators of specific network behaviours.

Such signature-matching methodologies can be highly valuable for digital forensics and intrusion analysis. By systematically subjecting a system to controlled attack scenarios and cataloguing their associated power signatures, analysts can later match observed anomalies from a breach against known profiles. This reverse-correlation process allows investigators to infer which operations or tools were likely used by an attacker, even in cases where conventional network logs have been deleted or tampered with.

4.5.1.3 Brute-force

The reconnaissance stage revealed that the SSH service was active on the target, with port 22 open for inbound connections. This discovery guided the attacker's

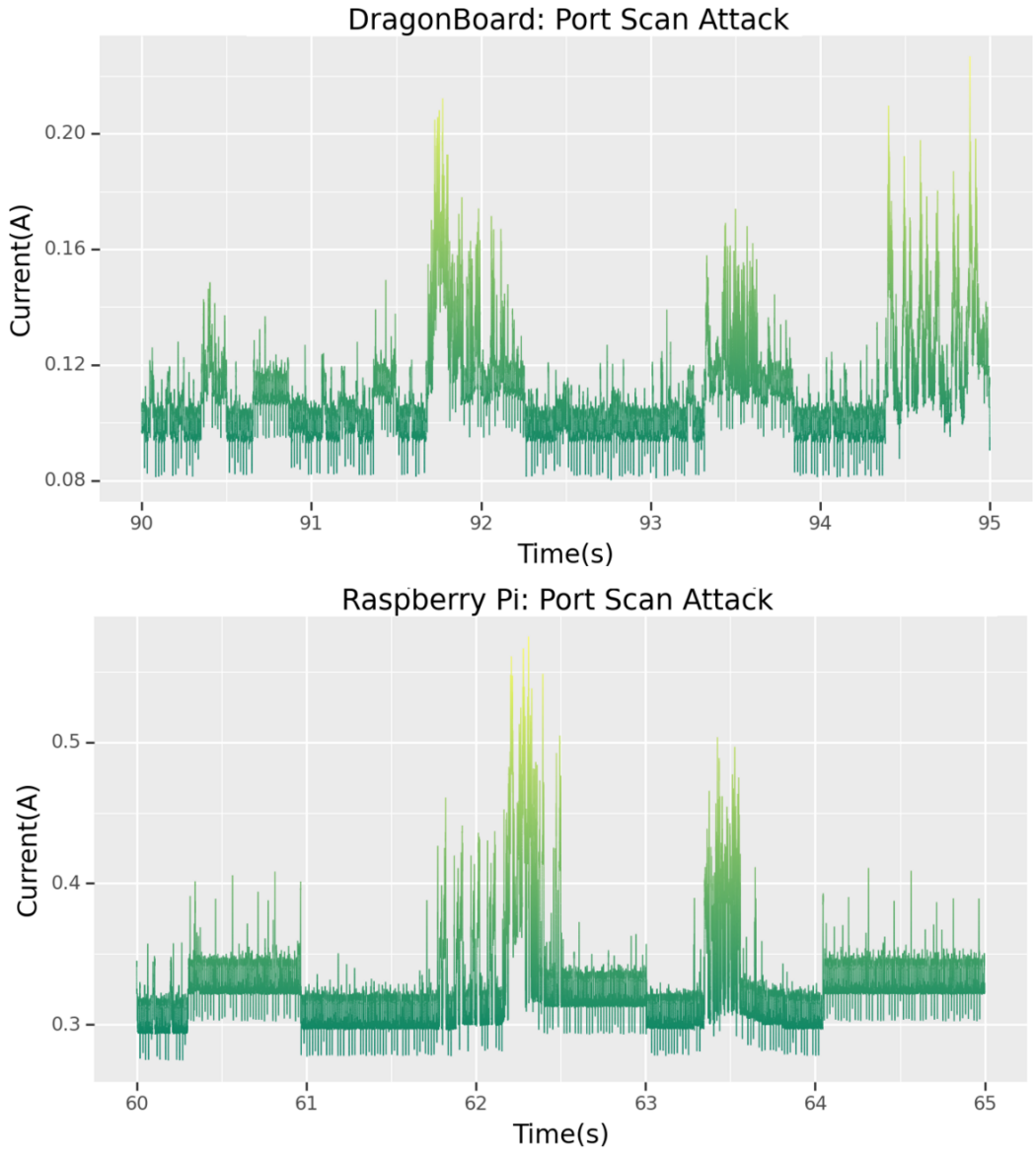


Figure 4.8: Comparison of Nmap behaviour during the CTF between the DragonBoard and Raspberry Pi (data visualised using the "summer" colormap [119]).

next step—an SSH brute-force attack designed to compromise credentials and gain shell access. The corresponding power data from the CTF can be cross-referenced with the results presented in Section 4.4.2 to confirm that such an attack took place.

Interestingly, Figure 4.9 reveals distinct differences between the two platforms. The DragonBoard exhibits a substantially larger current amplitude at the start of the attack compared with the Raspberry Pi. By comparing these traces to those in Figure 4.4, one can infer that the DragonBoard was attacked using *Hy-*

dra with the "`t=16`" setting, whereas the Raspberry Pi was targeted with "`t=4`". This aligns with the respective performance characteristics of each platform, as the DragonBoard's lower-power architecture saturates more quickly under high concurrency.

Such forensic reconstruction exemplifies the practical value of side-channel data in post-incident analysis. By iteratively replicating attacks with varying configurations, investigators can identify which tool parameters most closely reproduce the observed signatures.

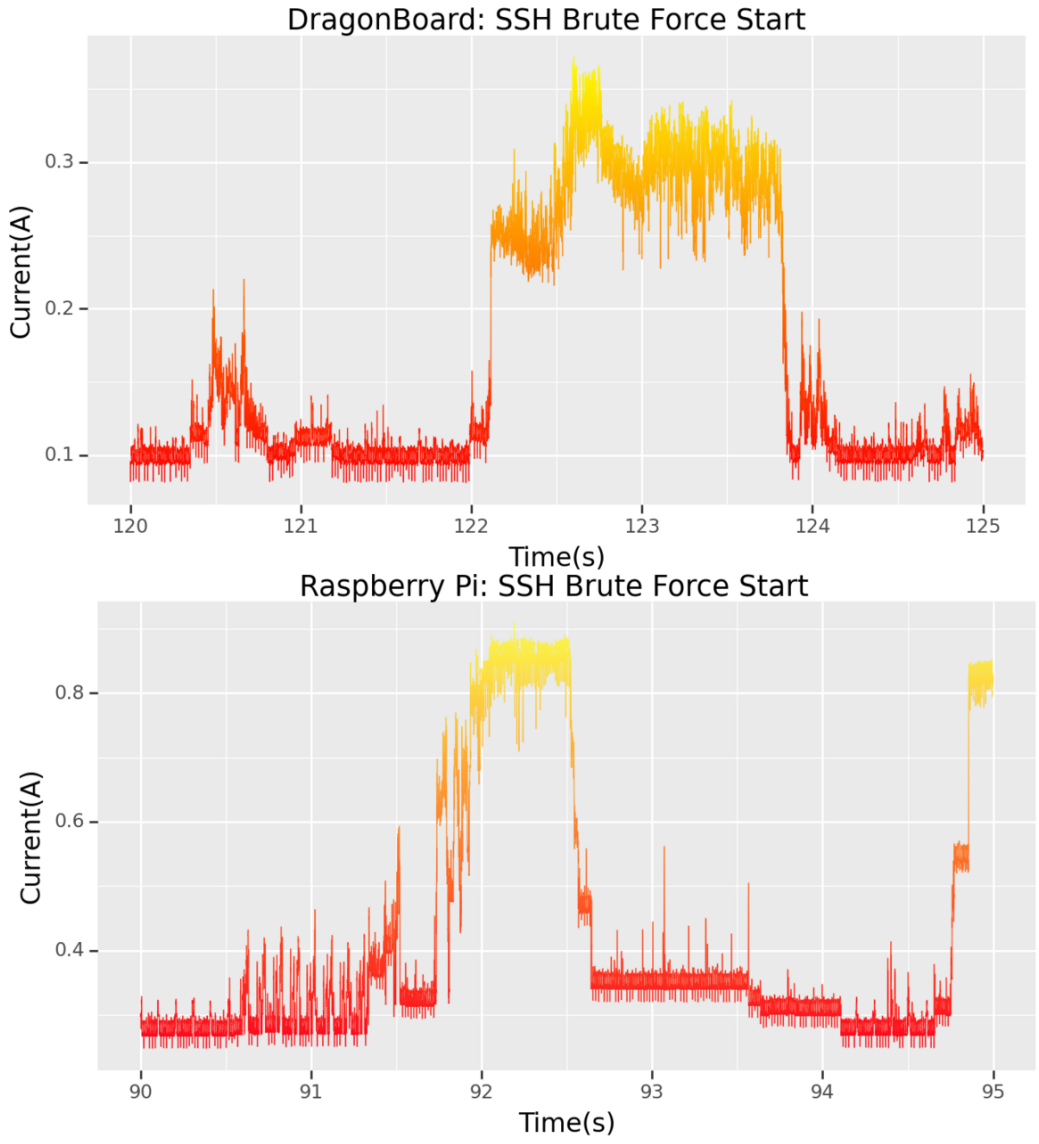


Figure 4.9: Start of the SSH brute-force phase during the CTF, comparing DragonBoard and Raspberry Pi (data visualised using the "autumn" colormap [119]).

In both devices, the CKE can be observed in Figure 4.10, matching the distinctive signature previously identified in Section 4.4.2. For forensic teams, the presence of a CKE can enable millisecond-level determination of when the password was successfully cracked, providing a precise temporal marker for the moment of compromise. This level of granularity can significantly enhance post-breach reconstruction and supports the creation of detailed incident timelines.

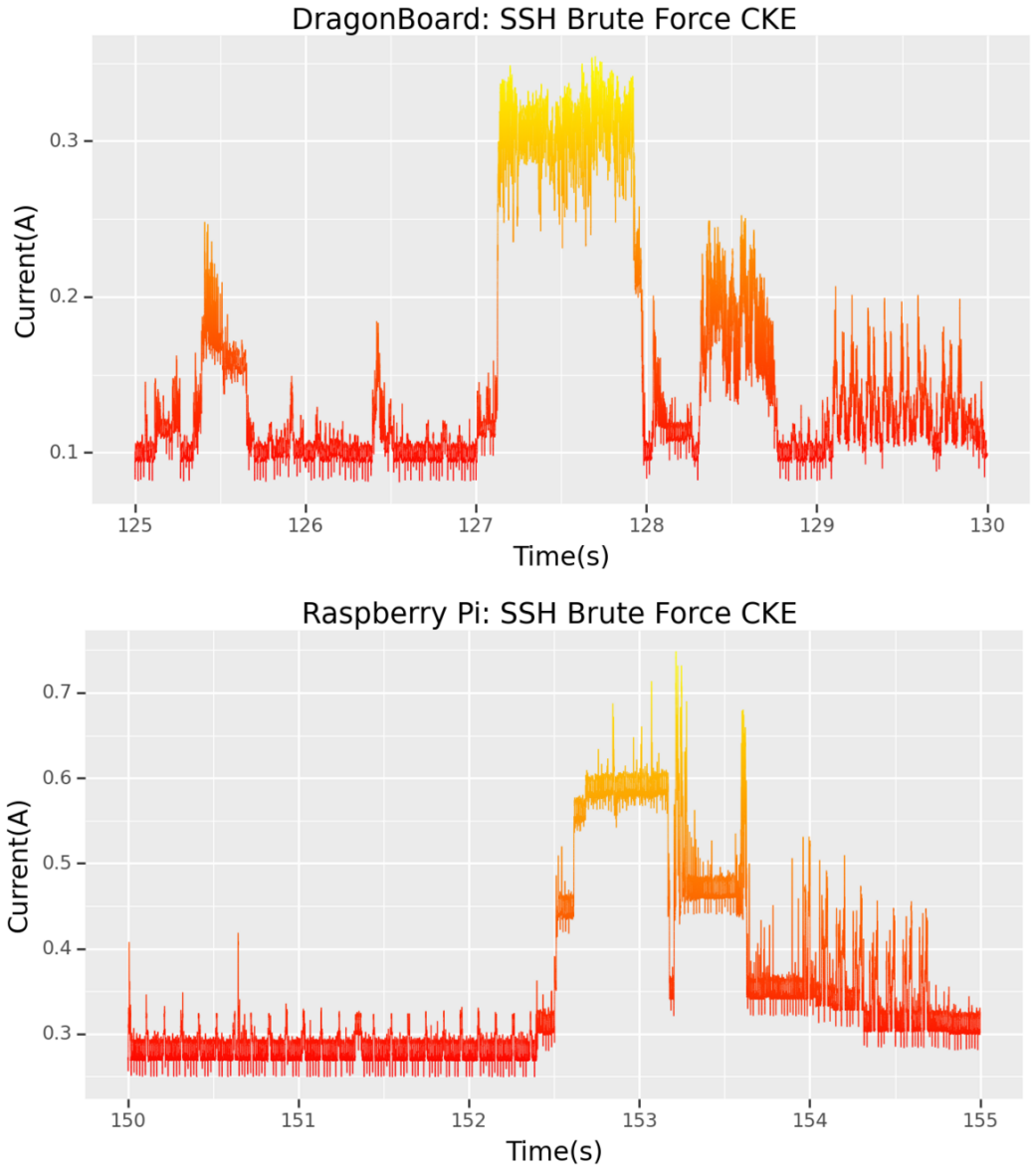


Figure 4.10: Comparison of Correct-Key Events during the CTF for the DragonBoard and Raspberry Pi (data visualised using the "autumn" colormap [119]).

4.5.2 Infiltration

After the SSH password is cracked, the attacker establishes a remote session with the target device via SSH, marking the transition from reconnaissance to full infiltration. This represents a critical escalation in the attack chain: once authenticated access is achieved, the adversary can interact directly with the target operating system.

With access granted, the attacker's capabilities depend on the privileges of the compromised account. If the account possesses administrative or "sudo" permissions, the attacker effectively gains unrestricted control, enabling modification of system files, installation of malicious software, and lateral movement to other networked systems. Even if initial access is limited to a standard user account, privilege-escalation attacks—exploiting outdated kernels, misconfigurations, or vulnerable binaries—can elevate privileges to root level. In this testbed, however, the attacker had already compromised the administrator account; thus, no additional escalation was required.

The infiltration signature shown in Figure 4.11 is an exact match to the SSH brute-force "`t=1`" CKE signature for each board. As demonstrated in Figure 4.5, the "`t=1`" signature consists of a final failed password attempt followed by a distinct spike corresponding to a successful login—the CKE. This one-to-one correspondence offers valuable insight into the operational flow of the password-cracking tool used in the attack.

While examining the Hydra source code could provide full transparency into its behaviour, the ability to deduce such functional patterns through side-channel analysis alone—without access to the tool's internals—is of high forensic value. Such inference aligns with the broader practice of "black-box" behavioural forensics, wherein the properties and intent of malicious tools are inferred from external artefacts or physical observables. This non-intrusive diagnostic approach may prove invaluable in IoT contexts where direct code access is infeasible or restricted.

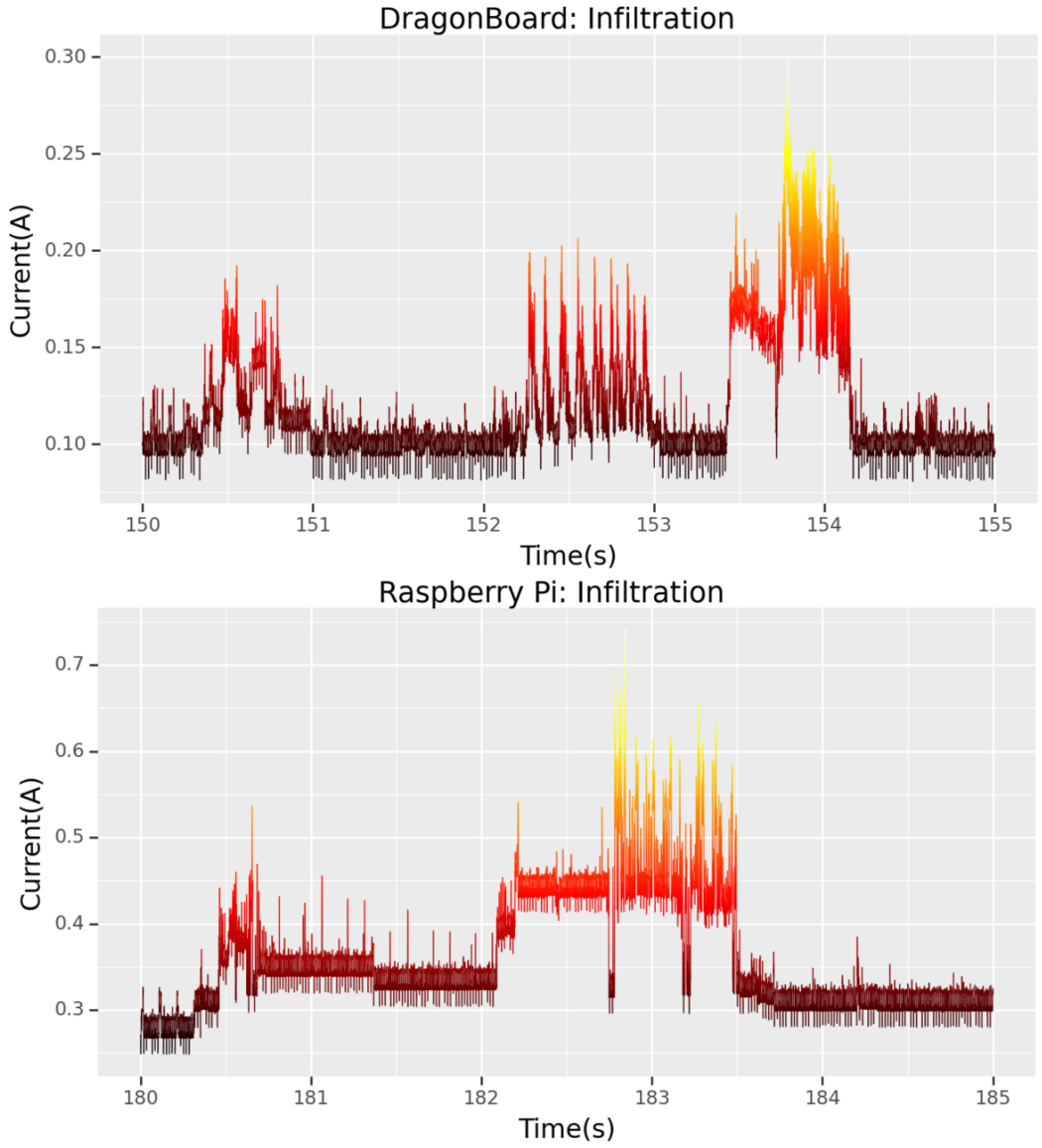


Figure 4.11: Infiltration during the CTF for the DragonBoard and Raspberry Pi (data visualised using the "hot" colormap [119]).

4.5.3 Exploration

Once the attacker has established access to the target device, the next logical step is system exploration—enumerating directories, files, and processes to identify assets of value or exploitable weaknesses. This phase corresponds to the post-compromise "Discovery" and "Collection" stages in the MITRE ATT&CK framework [131], where attackers attempt to learn about the system environment, verify privileges, and locate sensitive data.

Exploration activity typically generates considerable deviations from baseline behaviour, as system commands and file accesses induce bursts of I/O and CPU

usage. Figure 4.12 illustrates such disturbances on both devices. For the Raspberry Pi, the attacker executes the `cat` utility to read file contents, producing sharp, transient spikes in current. On the DragonBoard, the attacker navigates the file system using directory listing commands, generating lower-frequency but sustained perturbations.

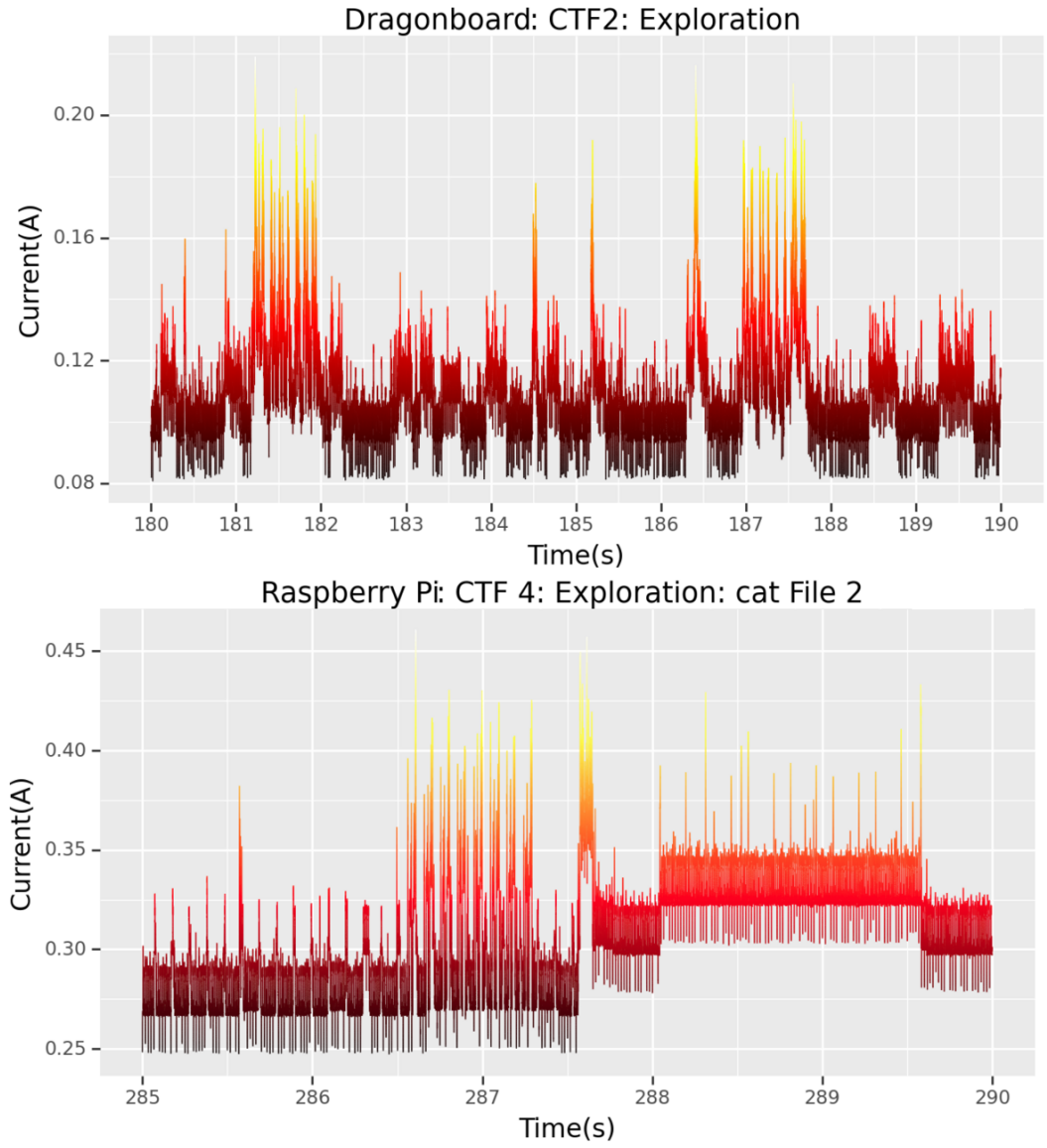


Figure 4.12: Comparison of attacker exploration activity on the DragonBoard and Raspberry Pi (data visualised using the "hot" colormap [119]).

These distinctive signatures underscore the diagnostic capability of side-channel power traces. Even in the absence of direct logging or packet capture, analysts can correlate such traces with known behavioural patterns to infer what operations occurred during system exploration. This capability is particularly valuable for constrained IoT environments, where resource limitations or

design choices may preclude traditional host-based monitoring.

4.5.4 Exfiltration

The most critical stage of this CTF scenario is the *exfiltration* of confidential data from the device. Exfiltration refers to the unauthorised transfer of information from a system, representing a complete breach of confidentiality. In practice, this stage often signifies the successful completion of an intrusion campaign. Within the context of this testbed, it corresponds to the attacker retrieving the hidden file or "flag".

In a real-world setting, such a flag could represent Personally Identifiable Information (PII), authentication data such as username–password files, or device telemetry revealing details of the broader IoT network topology. Regardless of the specific content, unauthorised data exfiltration must be regarded as a critical event, representing the highest severity level in the confidentiality domain.

Figure 4.13 shows that the tool used to copy the file over SSH generates a compound signature comprising a login phase, a sustained data-transfer segment, and a logout phase (the latter signature examined in the following section). By identifying these elements, a forensic analyst could infer not only that a file was transferred but possibly its approximate size, based on the duration and amplitude of the transfer segment. Such an approach parallels traditional network-forensics techniques, which estimate data volumes from timing and packet metadata.

In this instance, the distinctive pattern of activity between the login and logout signatures indicates that the `scp` utility [132] (Secure Copy Protocol), which transfers files over SSH using the same encryption and authentication mechanisms, was employed. This inference is made solely from side-channel power traces, demonstrating how such data can be used to identify specific attack tools and operations even in the absence of network-layer visibility.

The ability to detect and interpret exfiltration behaviour from side-channel data is especially relevant in IoT environments, where traditional intrusion detection systems may not be deployed due to resource limitations, requiring alternative options.

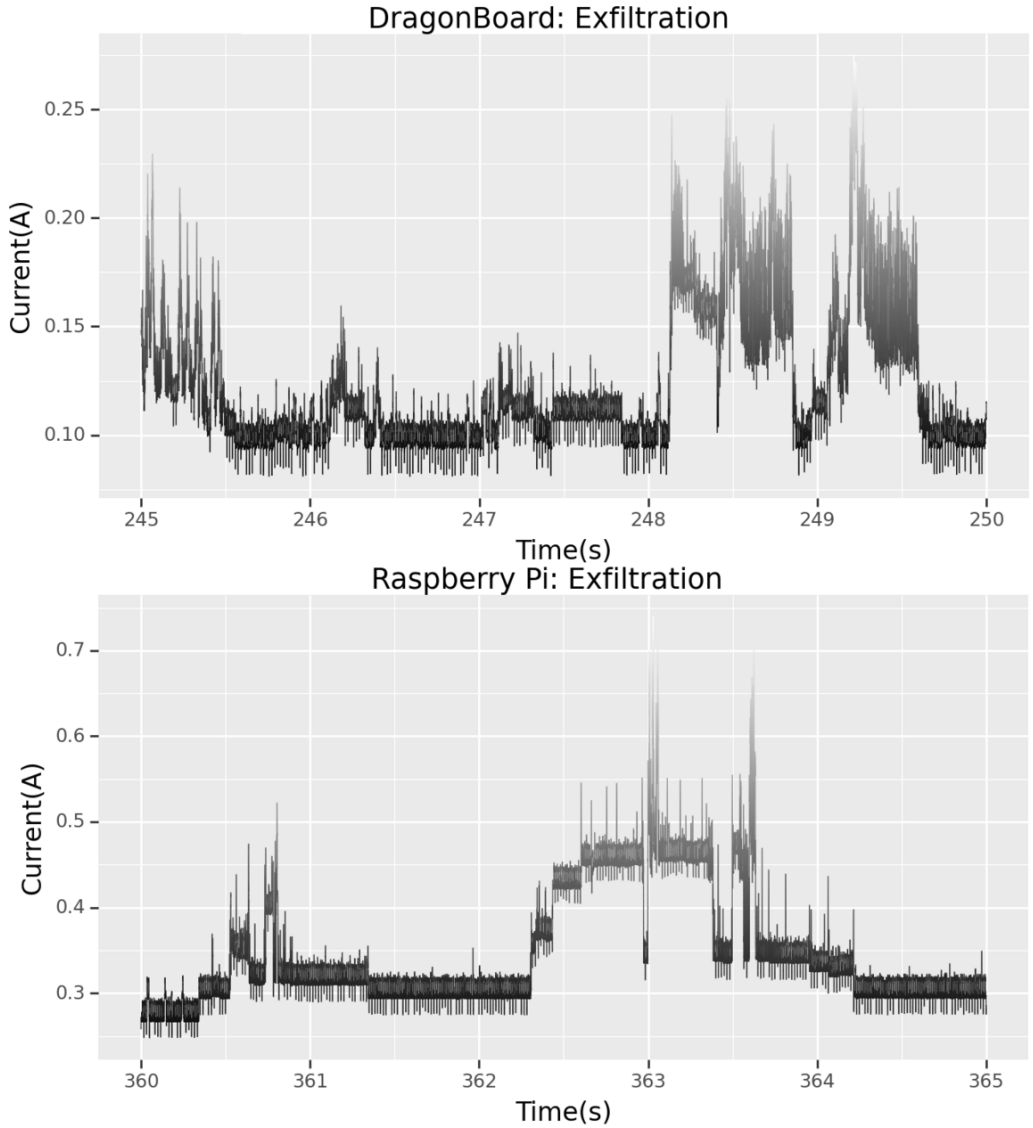


Figure 4.13: Exfiltration during the CTF for the DragonBoard and Raspberry Pi (data visualised using the "gist_gray" colormap [119]).

4.5.5 Withdrawal

Following the completion of the attacker's objectives, the final stage of the intrusion is *withdrawal*. This phase involves the termination of active sessions and, often, attempts to erase traces of the intrusion. Remaining connected would increase the likelihood of detection, while the deletion or modification of system logs represents an attack on the *integrity* of the device, potentially impeding post-incident investigations.

As shown in Figure 4.14, and with reference to the exfiltration signature, the withdrawal process can be understood as a distinct logout event immediately

following the transfer activity. Together, the login, transfer, and logout signatures form a complete behavioural fingerprint of data exfiltration over SSH. The close temporal and structural correspondence of these events provides a forensic timeline that can be used to reconstruct the attacker's actions with high confidence.

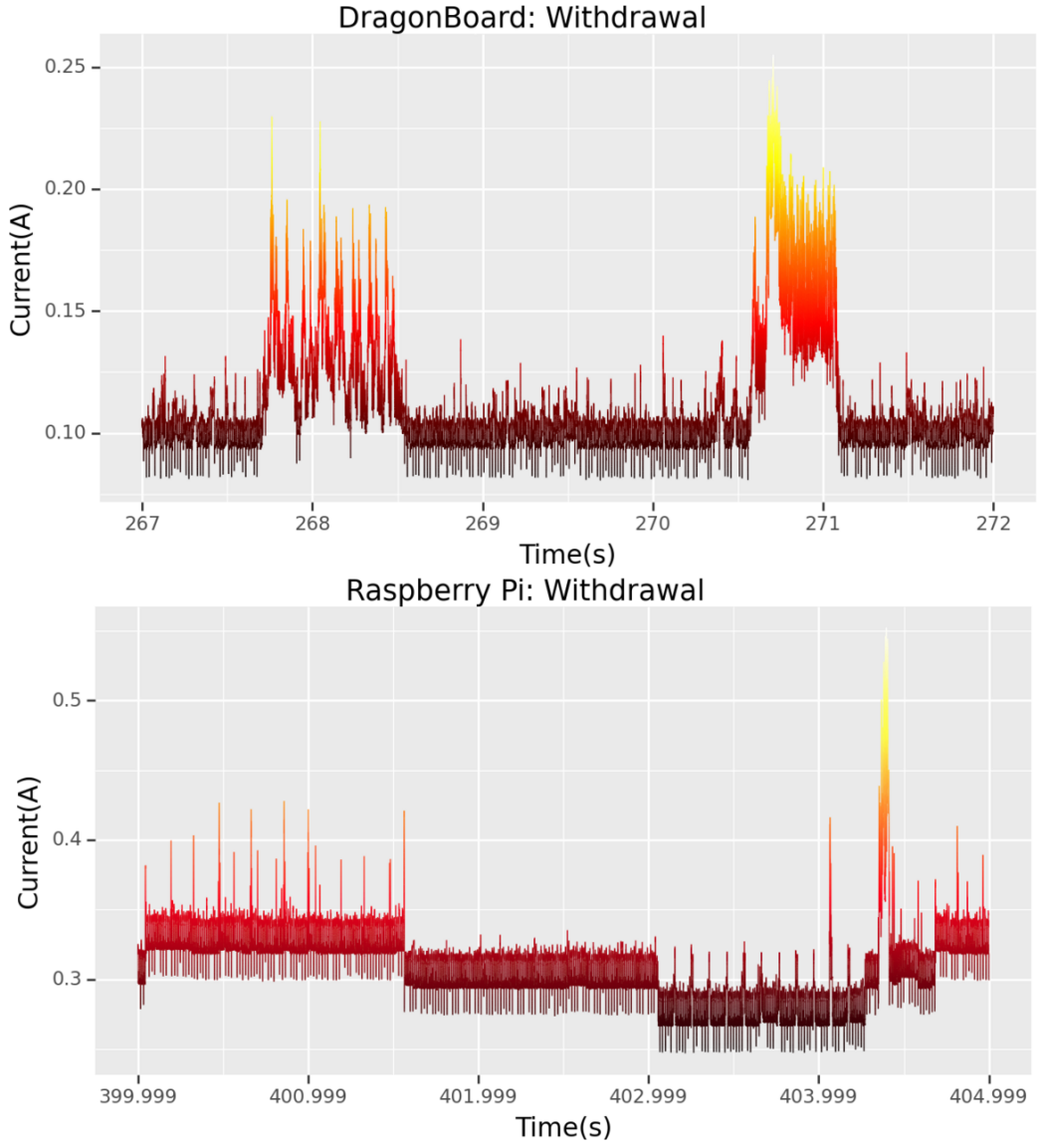


Figure 4.14: Withdrawal of the attacker following CTF completion for the DragonBoard and Raspberry Pi (data visualised using the "hot" colormap [119]).

In a handful of the CTF attack runs in the Dragon_Pi dataset, rather than withdraw from the target covertly, the attacker also uploads a simple battery-draining script to attack the availability of the device.

4.6 Dataset Creation

To further contribute to the field, the data studied in this Chapter were annotated and released as an open-access dataset, as an initial exploration of this area. The resulting DRAGON_PI dataset enables the evaluation of side-channel-based intrusion detection methods in realistic IoT settings. It comprises five normal files and sixty attack files, each containing linear time-series power consumption data from the testbed under either normal or attack conditions. All data were sampled at a rate of approximately 50 kHz, the maximum achievable by the Agilent N6705A power analyser with the chosen configuration, as mentioned earlier in this Chapter.

The specific normal data files were collected in strictly controlled conditions to ensure the complete absence of anomalies. This is crucial for enabling self-supervised or unsupervised approaches, which rely on a clean training distribution. These five normal files can therefore serve as the training class, while the remaining sixty attack files act as unseen test or validation sets. Each attack file contains an approximately balanced mixture of 50% normal and 50% anomalous activity, facilitating unbiased model evaluation (this may vary for some files, such as the DoS attacks).

Table 4.1 summarises the dataset's composition, categorised by anomaly type, collection tool, and device platform, while Figure 4.15 illustrates the hierarchical annotation structure in a loaded `pandas` DataFrame.

Table 4.1: Breakdown of labelled data files by attack type and platform.

Label Type	**Specific Label**	**# Files DragonBoard**	**# Files Raspberry Pi**
Normal	Normal	3	2
Port Scan Attack	Nmap_T5	2	1
	Nmap_T4	1	1
	Nmap_T3	1	1
	Nmap_T2	1	1
SSH Brute-force	Hydra_T32	4	2
	Hydra_T16	16	2
	Hydra_T4	8	2
	Hydra_T1	5	2
SYNFlood DoS	SYNFlood DoS	1	1
Capture the Flag	Misc Attacks	3	5

Each file contains three levels of annotation specificity:

1. `anno_string`—a binary classification (normal or anomaly);

2. `anno_type`—the general attack type (e.g., port scan, brute-force, DoS);
3. `anno_specific`—the precise attack instance, detailing the tool and configuration used to generate the signature.

Time	Current	anno_string	anno_type	anno_specific
28.799877	0.299654	Normal	Normal	Normal
28.799898	0.298771	Normal	Normal	Normal
28.799918	0.298418	Anomaly	SSH_Bruteforce	Hydra_T16
28.799939	0.298683	Anomaly	SSH_Bruteforce	Hydra_T16
28.799959	0.299360	Anomaly	SSH_Bruteforce	Hydra_T16

Figure 4.15: Example DataFrame header from the dataset showing hierarchical annotation structure.

Every file is accompanied by a legend file detailing the expected signatures and their corresponding indices, enabling users to locate specific anomalies efficiently. In most cases, the attack files maintain approximately equal durations of normal and anomalous segments. For instance, in a five-minute capture, around 2.5 minutes may contain anomalous power signatures, interleaved with 2.5 minutes of normal behaviour.

The dataset's annotations are particularly fine-grained: each anomalous instance is marked as precisely as possible within the time series. Figure 4.16 illustrates this annotation precision, showing a 1 s window containing a port-scan event lasting approximately 0.6 s. Such high-resolution labelling facilitates micro-level evaluation of anomaly detection systems and supports models capable of localising attack onsets.

As discussed earlier, special CTF scenarios were also carried out. These provide realistic, temporally extended sequences that combine multiple attack types—ping sweeps, port scans, brute-force logins, remote exploration, file exfiltration, and withdrawal. In contrast to the isolated attacks, these scenarios emulate a full adversarial workflow and are ideal for testing a model's ability to generalise to compound or sequential threats. Hence, in Chapter 6, the evaluation flow is refined to treat the CTF files with specific importance.

Some CTF scenarios additionally simulate post-exfiltration or availability attacks, such as in-place file modification or malicious payload execution designed to stress memory and deplete the target's battery enhancing dataset diversity and realism.

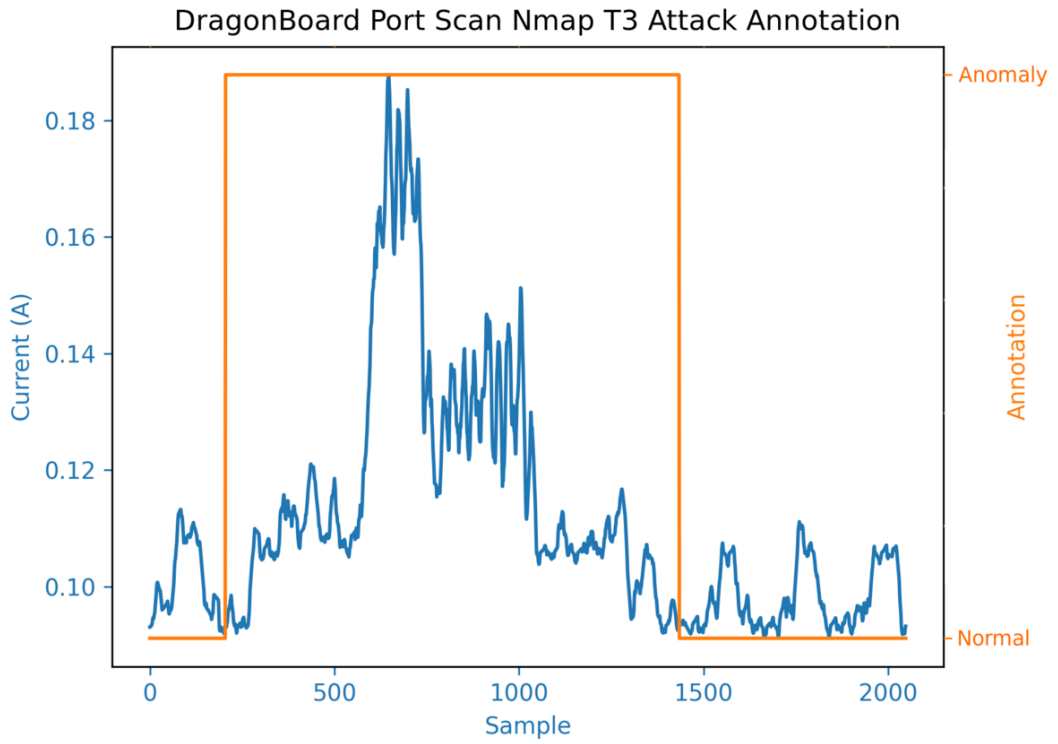


Figure 4.16: Example of annotation accuracy from the dataset.

4.7 Discussion

From a forensic and educational standpoint, studying and visualising the side-channel signatures of these attacks offers significant value. However, this Chapter discourages the use of signature-based *machine learning* methodologies on side-channel power data for one key reason: the availability and variability of attack signatures. To train a signature-based model to classify segments as normal or anomalous, a large and representative labelled dataset is required for each attack type. A key takeaway from this Chapter is that while attacks can exhibit similar power signatures for a specific tool *setting* across devices, substantial variation arises from tool choice, configuration, and network conditions, all of which can drastically affect the signature of the same attack, even on the same device. As a result, developing a generalisable, fit-for-purpose classifier trained on side-channel signatures alone would require datasets that are impractically large and diverse.

Anomaly-detection-based approaches, by contrast, offer a more practical and scalable path forward. Whereas signature-based systems require labelled examples of both normal and attack behaviour (which can be costly and time-

consuming to collect and curate), anomaly-based systems rely primarily on a well-defined normal baseline. These approaches are often *self-supervised*—trained using only normal data—or fully *unsupervised*, clustering observations based on deviations from expected behaviour. This Chapter demonstrates that a self-supervised approach, leveraging the natural abundance of normal data, should be the most effective means of constructing automated detectors for side-channel power anomalies in IoT devices.

While this Chapter discourages direct use of side-channel attack signatures for supervised machine learning classification, it strongly encourages their continued *practical study* - especially to create evaluation data for supervised and unsupervised approaches. Understanding how cyberattacks manifest in device-level power behaviour is invaluable. Such insights can guide the design of resilient IoT hardware and inform training and education within the cybersecurity community. Without a holistic understanding of how attacks appear across multiple observability layers—including power, timing, and packet traces—true defence in depth cannot be achieved. Moreover, this knowledge assists forensic teams in reconstructing post-incident timelines, enabling the development of more effective countermeasures.

It is also important to consider the practicality of deployment. Machine learning models are not suitable for all contexts, particularly in constrained IoT environments. While devices such as the Raspberry Pi can feasibly host on-device intrusion detection, many embedded systems lack the computational or power resources to do so. Consequently, in some constrained settings IDS inference may need to occur at the gateway or cloud level, with the IoT device transmitting its monitored side-channel data upstream.

4.7.1 Attack Behaviour and Detectability in Power Data

While the preceding analysis has demonstrated how specific cyberattacks manifest in device power consumption, it is important to contextualise these findings within broader models of attacker behaviour. This section provides a more comprehensive discussion of attack patterns, behaviours, and their detectability within the power domain, addressing the range of adversarial actions IoT devices may encounter.

Cyberattacks are rarely single, isolated events; rather, they typically unfold as multi-stage processes. Frameworks such as the cyber kill chain describe these

stages as including reconnaissance, initial access, exploitation, persistence, and execution of objectives. Each of these stages may involve distinct tactics, techniques, and procedures (TTPs), which can vary significantly depending on the attacker's goals, tools, and level of sophistication.

From the perspective of side-channel power monitoring, not all stages of an attack are equally observable. Activities that involve sustained or intensive computation on the target device—such as brute-force attempts, denial-of-service attacks, or payload execution—tend to produce clear and measurable deviations in power consumption. These behaviours often manifest as high-amplitude or temporally structured signatures, making them well-suited to detection using the techniques explored in this thesis.

In contrast, other stages of the attack lifecycle may be less readily detectable through power analysis alone. For example, low-intensity reconnaissance activities, intermittent probing, or attacks that primarily target network communication without significantly altering device computation may produce only subtle or negligible changes in power consumption. Similarly, highly optimised or stealthy attack implementations may be designed to minimise their observable footprint, reducing their detectability within the power domain.

These observations highlight that power-based intrusion detection is most effective when targeting behaviours that directly impact device computation, rather than all possible attack stages. Consequently, the approach should be viewed as complementary to existing network- or host-based monitoring techniques, forming part of a broader, multi-layered defence strategy.

In summary, while side-channel power data provide valuable insight into many forms of adversarial behaviour, they do not capture the full spectrum of attack activity. Understanding which attack stages and TTPs are more or less visible within this modality is essential for interpreting detection results and for designing robust IoT security systems.

4.8 Summary

This Chapter investigated an often-overlooked but fundamental source of information available on every device: *side-channel power data*. The Chapter highlighted the existence of some common attack signatures across different devices, while also showing how these signatures can mutate with simple chan-

ges in attack settings. It further outlined key considerations for developing machine-learning methodologies on this modality, and identified the *Correct Key Event* (CKE)—a clear, repeatable power signature at the instant a password is successfully cracked—providing a millisecond-accurate marker of compromise. Overall, the results show that power data are an effective, non-intrusive defensive signal and should be used as a complementary source alongside existing packet- and host-based approaches.

This Chapter represents a first step using real attack data. As in many domains, accurate annotation is challenging; some degree of label noise is inevitable. Long-duration attacks (e.g., SSH brute-force) exhibit bursts of activity interleaved with quieter intervals. During these quieter periods, power traces may appear normal even though the attack is still ongoing. In such cases, the entire interval was labelled as anomalous—favouring inclusivity over potential false negatives—although this choice may increase false positives.

A second limitation is dataset scale: the corpus currently totals roughly one hour of data. Although a range of attack types and tool parameterisations were captured, broader coverage would improve statistical power and model robustness. Expanding the dataset is non-trivial due to the cost and time required for careful, fine-grained annotation.

A third limitation is the under-representation of some attack types. Specifically, the DoS attacks feature only 2 attack files total. This decision was made as the DoS data features no real specific signature of attack to learn from, rather just a large envelope of power under attack—which was entirely expected. In one of the attack files, the DoS seems to die midway. These data were kept as it is still a real attack scenario and may still contain important information with regards to this attack and how it disturbed normal behaviour.

In summary, this Chapter fulfils the *second aim of the thesis* by transitioning from synthetic data to real-world attacks on IoT devices, thereby greatly advancing the practical relevance of the research. The analysis demonstrated that while there is some signature commonality across device, small changes in attack configuration (tool setting) can lead to large variations in power signatures—a finding that directly informs the model design choices in Chapter 5. To preserve the anomalous nature of attack data and avoid biasing models toward known threats, the remainder of this thesis therefore explores *self-supervised* approaches in which models learn only from normal behaviour. This also marks the beginning of fulfilling the *third aim of the thesis*, evolving the work from

supervised approaches that require attack data during training, as in Chapter 3, to the self-supervised approaches developed in chapters 5 and 6.

Chapter 5

Convolutional Autoencoder for Self-supervised Intrusion Detection in IoT Devices

5.1 Introduction

Chapter 4 established that real attack power signatures can be far more complex and variable than their synthetic counterparts, revealing that supervised, signature-based learning may be poorly suited to the unpredictability of real-world IoT threats. Although high accuracies may be achievable during controlled laboratory testing, performance would likely fail to generalise to deployment environments, where unseen, or novel, attack types and parameter configurations dominate. These findings underscored the need for models capable of learning *normal* device behaviour without requiring labelled attack data.

Accordingly, this Chapter begins an initial investigation into self-supervised architectures—starting with Convolutional Autoencoders (CAEs) trained exclusively on normal data—to evaluate how well such approaches generalise to unseen anomalies.

Normal traces are abundant and easily obtainable, even in field-deployed devices. By learning this stable and reliable signal, anomalies can be detected as deviations from expected behaviour [69]—a principle widely applied in modern self-supervised and unsupervised anomaly detection.

The self-supervised CAE presented in this Chapter is trained exclusively on the normal data of both the DragonBoard and the Raspberry Pi simultaneously, thereby assessing the feasibility of a multi-device, multi-component monitoring IDS—a more cost-effective and scalable alternative to training separate models per device. This setup reflects real-world deployment contexts such as industrial control systems or electric vehicles, where multiple subsystems operate collaboratively.

Another advantage of this reconstruction-based approach lies in its inherent explainability, a growing research priority in AI [133, 134], particularly in safety-critical and security-focused applications. Because the CAE reconstructs its input, detections can be interpreted visually by comparing input and output signals. Such transparency is essential in cybersecurity, where analysts must trust and understand model outputs before acting upon them.

By focusing on self-supervised learning using only normal power data, this Chapter demonstrates how robust, device-agnostic anomaly detection can be achieved without reliance on labelled attack traces during training. This is especially valuable in settings where anomalous data—or reliable labels—are costly or impossible to obtain [22, 61]. The insights gained here establish a methodological bridge between the supervised proof-of-concept investigations in Chapter 3, the dataset-driven analysis of Chapter 4, and the advanced Transformer-based architectures explored in Chapter 6.

5.1.1 Architectural Assumptions

The self-supervised framework developed in this Chapter operates under a set of assumptions regarding data availability, deployment, and learning conditions.

- Model training is conducted using **normal power consumption data only**, reflecting the practical constraint that labelled attack data are scarce and highly variable.
- A clean separation is maintained between training and evaluation data, with held-out normal and anomalous traces used exclusively for testing.
- Power traces are assumed to exhibit sufficiently stable and repeatable structure under normal operation to enable effective reconstruction-based learning.
- **Deployment configuration.** Inference may be performed either directly

on-device or on an adjacent system (e.g., a gateway or external processor), depending on available computational resources. However, in the case of this Chapter it was not performed on-device.

- **Data transmission (adjacent inference).** Where inference is performed on an adjacent device, it is assumed that power data are transmitted as time-series segments at a fixed sampling rate, either continuously or in short windows. This transmission is assumed to occur over a lightweight communication interface (e.g., serial, SPI, or network-based transfer), with minimal preprocessing performed on the source device.
- **Application workload assumptions.** The underlying device workloads are assumed to be representative of typical operation, exhibiting sufficiently consistent behaviour to allow the model to learn a stable baseline. It is acknowledged that real-world variability may extend beyond the observed dataset.
- **Computational constraints.** The target deployment platforms are assumed to be resource-constrained IoT devices, with limited memory, compute, and power budgets. As such, all models are designed to remain lightweight and suitable for real-time or near-real-time inference within these constraints.

These assumptions support the transition from supervised classification to self-supervised anomaly detection, aligning the learning paradigm with the constraints of real-world IoT environments.

5.2 Background: Autoencoders and Convolutional Feature Reconstruction

5.2.1 Autoencoders

Autoencoders (AEs) are a class of neural networks trained to reconstruct their input from a compressed internal representation [135]. Unlike supervised learning approaches, which require labelled data, AEs operate in a self-supervised setting (and can sometimes be viewed as unsupervised): the model maps an input $\mathbf{x}$ to a compact latent vector $\mathbf{z}$ and then reconstructs an approximation $\hat{\mathbf{x}}$ of the original input. This compression mechanism enables the network to capture the dominant structure and correlations in the

underlying data distribution by forcing the model to reconstruct the input from a greatly reduced feature space. For the model to accurately reconstruct the input, the compact latent vector $\mathbf{z}$ must learn to discard irrelevant information and retain only the most salient features. A simple AE illustration can be found in Figure 5.1.

In this way, AEs can be used to "pre-train" on a dataset to capture deep representations. Pre-training is the process of self-supervised learning on a dataset before "fine-tuning" on the same or even a different dataset for supervised tasks.

A common approach is to discard the decoder after pre-training and add a fully connected MLP classifier for fine-tuning, moving from self-supervised pre-training back to supervised learning while leveraging the deep features obtained during the pre-training stage.

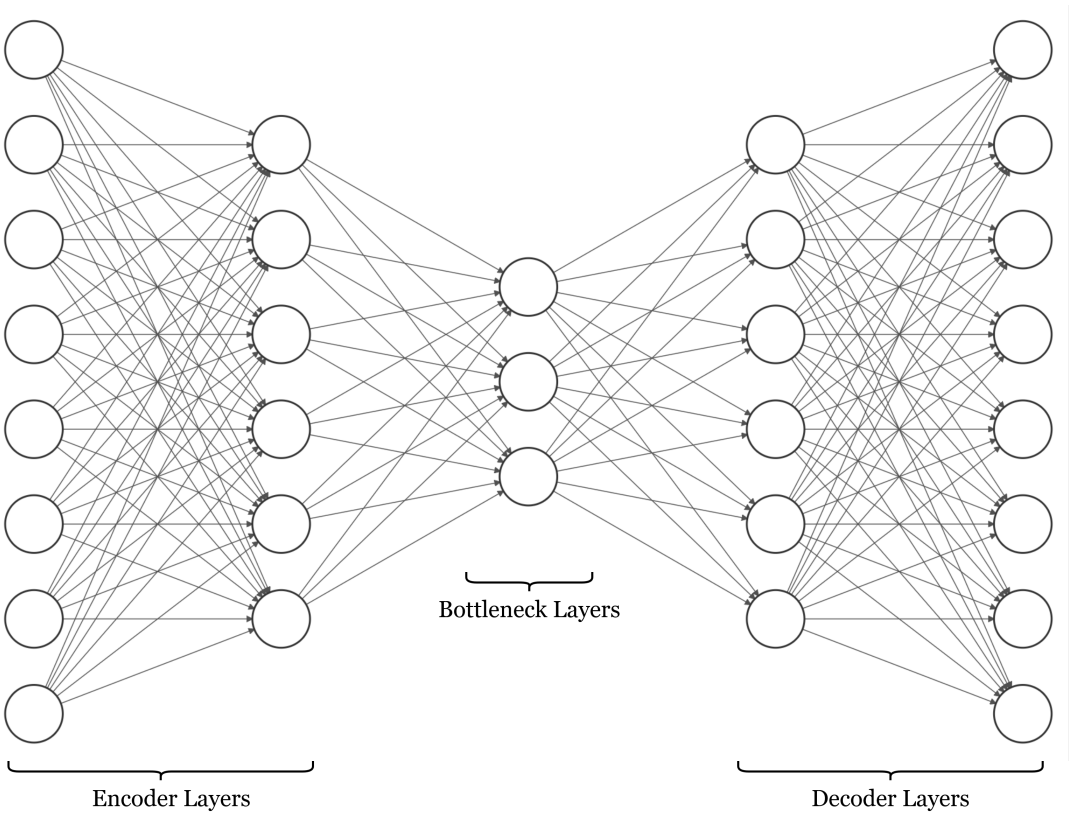


Figure 5.1: Simple AE architecture comprising an encoder, a low-dimensional bottleneck (latent representation), and a decoder. Note the compression of the latent space.

Formally, the encoder–decoder process is given by:

$$\mathbf{z} = f_\theta(\mathbf{x}), \qquad \hat{\mathbf{x}} = g_\phi(\mathbf{z}), \tag{5.1}$$

where f_θ compresses the input into a latent feature representation and g_ϕ reconstructs from that representation. Training typically involves minimising a reconstruction loss such as the mean squared error:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \hat{x}_i\right)^2,$$

5.2.2 Latent Representations and Anomaly Detection

When trained exclusively on *normal* operating data (a self-supervised learning paradigm), an AE learns a latent representation that captures regular system behaviour. Inputs that deviate from this learned structure—such as those produced under anomalous or attack conditions—typically yield elevated reconstruction errors. This property makes AEs well-suited to anomaly detection tasks, especially in settings where labelled attack data is limited or unavailable.

However, standard AEs can suffer from *over-generalisation*, where the model becomes expressive enough to reconstruct anomalous patterns with low error, thereby failing to detect anomalies. This is a particular concern in IoT security contexts, where anomalies may be subtle and legitimate behaviour inherently variable. Regularisation techniques such as dropout [95] or further restricting the latent dimensionality are therefore essential to prevent the network from learning a near-identity mapping and to ensure the model is unable to reconstruct unseen anomalous data.

5.2.3 Convolutional Autoencoders

Convolutional Autoencoders (CAEs) extend traditional AEs by integrating convolutional layers into both the encoder and decoder, allowing the model to exploit local temporal or spatial structure [136]. This is particularly effective for waveform-like power consumption data, where fine local resolution can be essential for signal understanding.

Convolutional encoders learn hierarchical feature maps—analogous to CNNs—while transposed convolutions or upsampling layers in the decoder reconstruct these features back into the original signal domain.

In Figure 5.2 a simple depiction of a CAE with a traditional AE bottleneck is shown. Note the convolutional encoder for strong local feature extraction, the fully connected AE bottleneck for further reduction of the feature space, and

the convolutional decoder to reconstruct the input while preserving as much local structure as possible.

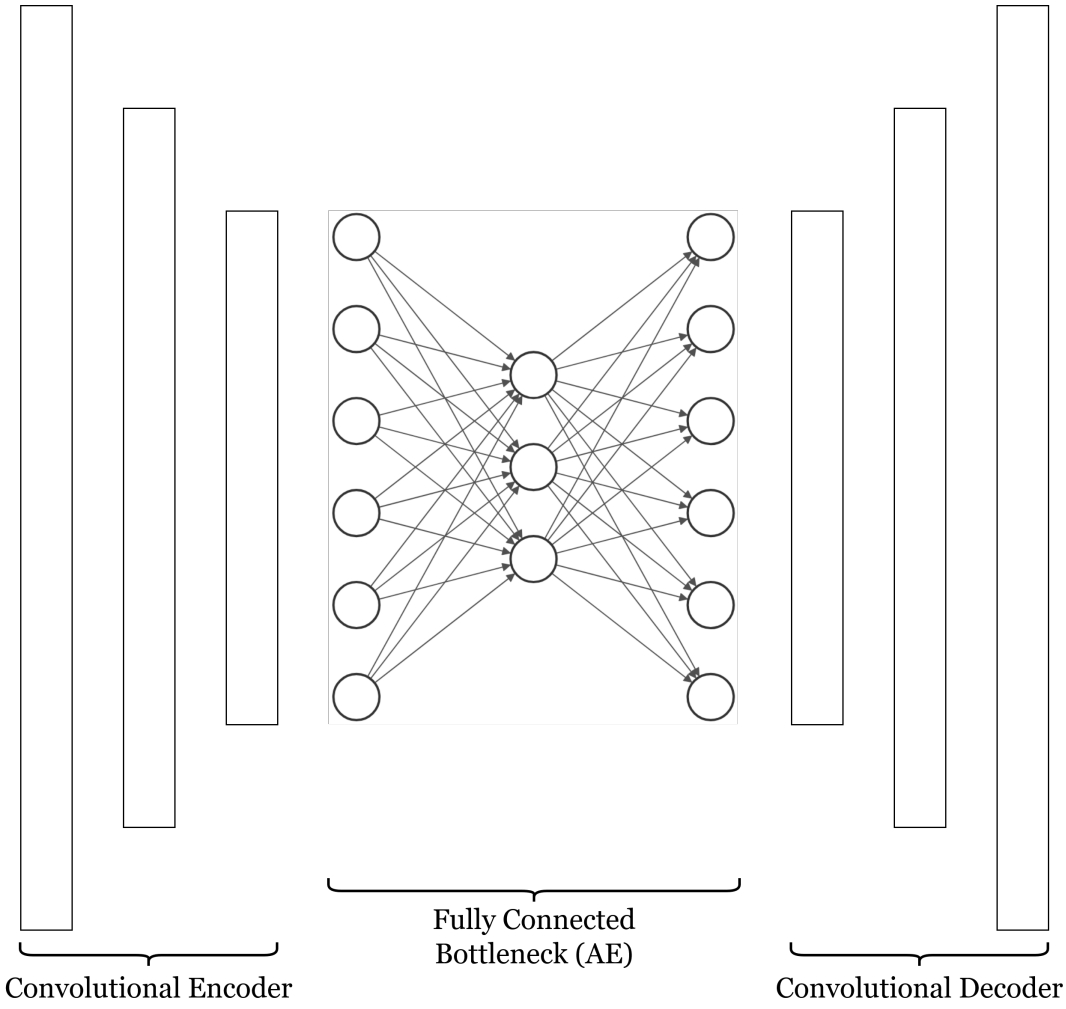


Figure 5.2: Convolutional autoencoder (CAE) with a convolutional encoder, a fully connected AE bottleneck, and a convolutional decoder.

By using weight sharing and local receptive fields, CAEs achieve efficient representation learning while maintaining a compact parameter count. This makes them suitable for deployment on resource-constrained edge hardware such as embedded IoT platforms.

The traditional AE bottleneck in the middle could feasibly be replaced with additional convolutional layers to further reduce spatial dimensions, although this was not the approach taken in this thesis.

5.2.4 Dropout and Controlled Reconstruction Capacity

In anomaly detection, the objective is not only to reconstruct normal data well, but also to ensure that *anomalous* inputs are reconstructed poorly—with the best case scenario being perfect reconstruction of normal data, and complete inability to reconstruct the abnormal. In this Chapter, dropout is used extensively in the dropout-enhanced CAE architecture to prevent the network from learning shortcut mappings and to discourage overly precise reconstructions of rare or abnormal patterns. This encourages the AE to form a robust, generalised latent representation of normal device behaviour, improving the separation between normal and anomalous reconstruction error distributions.

This controlled reduction in reconstruction capacity is a central design choice of the model developed in this Chapter, enabling reliable detection of anomalous behaviour from time-series power consumption data.

5.3 Data Preprocessing

In Chapter 4, no preprocessing was required, as that work was primarily exploratory with little initially known about the data. For effective model training, however, modest preprocessing is necessary to enhance learning efficiency and robustness.

The raw data were sampled at 48,828 Hz, the maximum rate of the power analyser used. This is a relatively high sampling rate and typically demands expensive instrumentation. However, in the course of this Chapter, it is shown that excellent model performance can be achieved at a much lower rate and, consequently, with far more affordable hardware.

Before any preprocessing, the input window length for real-time operation was defined—specifically, the number of seconds of data processed per inference. This window length (epoch) was selected by examining the normal training data and determining the shortest duration that captured sufficient behavioural information. For DRAGON_PI, a “sufficient” segment is one that encapsulates the key Bluetooth transmission signature within a single window, as this represents the distinguishing feature of normal behaviour.

Providing enough temporal context is critical: windows that are too short risk excluding salient features, leaving the model with too little structure to learn from. For instance, a 100 ms epoch may not contain adequate variability for

meaningful feature extraction. Conversely, overly long windows increase latency and memory requirements without necessarily improving generalisation.

For this work, the epoch length was set to one second. As shown in Figure 5.3, the normal Bluetooth signature fits cleanly within this window. Although shorter windows (e.g., 0.5 s) could possibly be used, caution is advised against unnecessary reduction. With deep models such as the CAE employed here, shorter inputs increase the risk of overfitting: the model observes less contextual diversity and may learn overly simplistic features. A model that learns only coarse local features may reconstruct both normal and anomalous data equally well—an undesirable outcome. The goal is for the CAE to reconstruct normal data accurately while failing on anomalous data.

Once the one-second input length was established, the sampling rate was reduced using the `scipy.resample` function to 2,048 Hz. A one-second window at the raw rate contains 48,828 samples—too large for efficient training and memory-constrained during GPU data loading. Reducing the rate to 2,048 Hz dramatically decreases the number of samples per window while remaining compatible with inexpensive, off-the-shelf measurement hardware, eliminating the need for high-end power analysers.

This 2,048 Hz rate strikes a practical balance between minimising data size and preserving signal fidelity. Lower rates such as 1,024 Hz or 512 Hz resulted in visibly degraded traces, risking the loss of subtle but important features. As illustrated in Figure 5.3, the 2,048 Hz signal remains visually indistinguishable from the 48,828 Hz trace for the structures of interest.

After resampling, residual high-frequency noise remained—visible in Figures 5.3A and 5.3B as small fluctuations about the baseline. Based on the analysis in Chapter 4, these fluctuations are attributable to measurement noise (likely analyser-induced) and can be safely removed. A Moving-Average-Filter (MAF) was therefore applied. A 15-sample kernel was chosen as the largest window that effectively reduced noise without distorting the underlying signal. The effect is shown in Figure 5.3, where background noise is substantially reduced.

Finally, the data were segmented into equal-length windows for model input. With a one-second duration, each segment contains 2,048 samples. Given the limited quantity of data for training and evaluation, a sliding window with 75% overlap was used, allowing the model to observe features from slightly varied

temporal perspectives and introducing modest variability between samples.

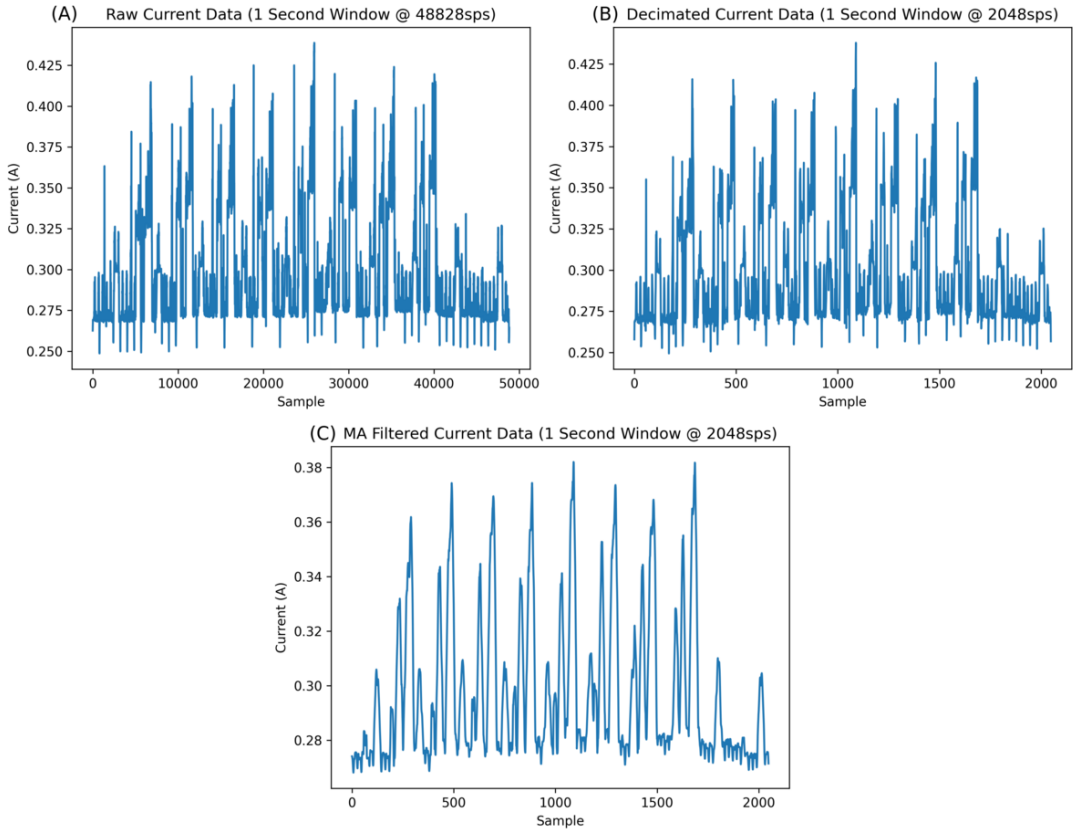


Figure 5.3: Preprocessing pipeline: (*A*) raw data at 48,828 Hz, (*B*) resampled data at 2,048 Hz, and (*C*) MAF-filtered data at 2,048 Hz with MAF size 15.

Importantly, the strong results achieved at this reduced sampling rate demonstrate that excellent detection performance can still feasibly be obtained using affordable, low-specification measurement equipment. This makes the proposed framework considerably more practical and scalable for real-world IoT security applications than if a large, powerful, and expensive power analyser were required.

Normalisation was omitted as a preprocessing step because the normal data already resided in a naturally bounded and numerically stable range (approximately 0.1–0.5 within $[0, 1]$), so additional scaling was not required for training stability. Moreover, absolute amplitude is itself an informative feature: higher-amplitude excursions correspond to anomalous behaviour. Applying normalisation (especially on a per-trace or per-device basis) would tend to rescale these excursions, effectively altering the anomaly contrast before the samples even enter the model. Finally, normalisation would suppress the natural baseline differences present in the data (for example, the Pi traces operate at a consistently

higher baseline amplitude), which we explicitly wish the model to exploit.

5.4 Architecture and Training

5.4.1 Architecture

Chapter 4 highlighted the dynamic nature of attack signatures, which can vary significantly between attacks and even within the same attack when tool parameters change. Consequently, reliance on supervised or signature-based detection for IoT power traces was discouraged.

Supervised learning (or signature-based classification) requires a fully labelled dataset containing both normal and anomalous samples. In contrast, a reconstruction-based *self-supervised* approach dispenses with anomaly labels for training: the model is trained only on held-out normal data and evaluated on unseen data containing both normal and anomalous segments. This approach mitigates the variability challenges of real-world attack signatures while leveraging the abundance of normal data, which is straightforward to collect in practice.

A CAE style architecture was selected to realise this self-supervised objective. The CAE comprises an encoder and a decoder constructed from 1-D convolutional layers tailored for time-series feature extraction. A latent (bottleneck) layer compresses the learned representation further, capturing the most salient features of normal operation. The decoder then attempts to reconstruct the original input from this compressed code. Model performance is measured using mean squared error (MSE) between the reconstructed and original inputs. Because MSE penalises large deviations quadratically, it naturally accentuates reconstruction errors in anomalous samples, producing strong separation between normal and abnormal behaviour during testing. The objective during training is to minimise MSE on normal data; at inference, the model should reconstruct unseen normal traces accurately but yield higher error on anomalous traces that deviate from learned patterns due to poor reconstruction.

Downsampling in the encoder is achieved using strided 1-D convolutions (stride $= 2$) rather than convolution followed by max pooling. This simplifies design of the decoder: transposed convolutions with matching stride can directly reverse the encoder's downsampling. The use of strided convolutions throughout also avoids the interaction issues that can arise when combining pooling layers with

dropout and convolution [137].

As mentioned, the model input is a one-dimensional current segment of length 2,048 samples; the output is its reconstruction. The desired behaviour is high-fidelity reconstruction for unseen normal inputs and poor reconstruction (i.e., higher reconstruction error) for unseen anomalous inputs.

The encoder comprises three convolutional *banks* of layers (with 3, 2, and 2 layers respectively). In each bank, the final convolution uses a stride of 2, halving the temporal length. With three such downsampling operations, the sequence length follows:

$$2048 \to 1024 \to 512 \to 256.$$

At the end of the encoder, the feature map is flattened before entering a dense bottleneck layer. In the reference configuration, the encoder's final tensor has a temporal length of 256 and 32 channels, yielding a flattened size of $256 \times 32 = 8{,}192$. All of these design choices are illustrated in Figure 5.4.

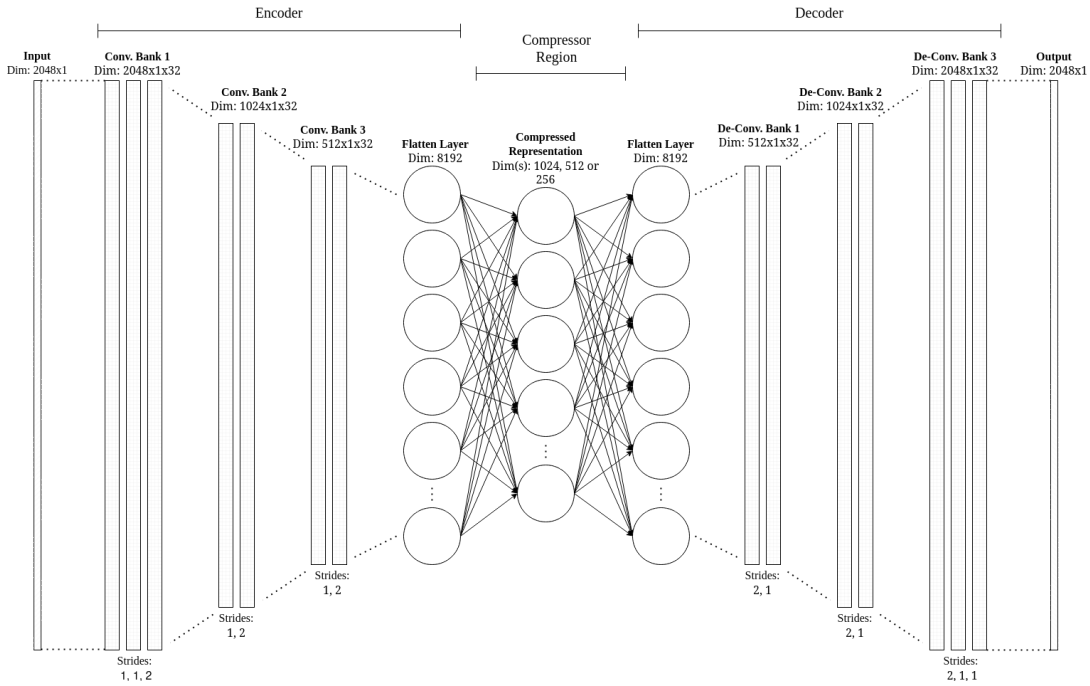


Figure 5.4: Convolutional Autoencoder (CAE) architecture used in this Chapter.

Within the dense compression stage, this 8,192-dimensional vector is further reduced to one of three latent dimensions: 1,024, 512, or 256, corresponding to compression ratios of $8\times$, $16\times$, and $32\times$, respectively. Larger compression ratios impose stronger representational constraints, making reconstruction more

challenging, as there is less information available for the decoder to reconstruct.

The decoder mirrors the encoder, employing transposed 1-D convolutions with a stride of 2 to upsample back to the original sequence length. Unless otherwise stated, each decoder layer uses 32 filters with kernel size 3. Matching kernel sizes and filter counts across the encoder and decoder ensures structural symmetry and stable reconstruction dynamics. Gaussian Error Linear Unit (GELU) activations [138] were used after both convolutional and dense layers.

5.4.2 Training

As with any machine learning system, the train–test split is critical. In this work, training is conducted *exclusively* on one class: the normal class, drawn from *clean-room* normal data.

Here, *clean-room normal* denotes that the normal training set has *no overlap* with the test data—not even as a subset—ensuring that the test set remains entirely novel. Although roughly 50% of the test data also contains normal behaviour, that normality is interspersed with or adjacent to attacks. To preserve the "unseen" nature of test-time normal behaviour, the clean-room normal set is used strictly for training and kept separate from all test files. Throughout this Chapter, this dataset is referred to as the *held-out* normal data.

The working hypothesis is that the model learns the salient temporal and statistical structure of normal behaviour from the held-out set and consequently reconstructs unseen normal inputs accurately, while failing to reconstruct anomalous inputs. The resulting elevation in reconstruction error enables detection without requiring any anomaly labels during training.

Importantly, the model was trained on held-out normal data from *both* devices at once, allowing one unified model to operate across multiple device types. Chapter 4 demonstrated the strong similarity of both normal and attack-related power signatures between the DragonBoard and Raspberry Pi, supporting the feasibility of this cross-device training configuration. Such generalisation is valuable for scalable IDS deployment across heterogeneous IoT setups, or for an IDS employed to protect various heterogeneous components within one system, for example in an advanced EV.

In this Chapter, no explicit validation split was used. Instead, model checkpoints were saved at each epoch and evaluated post hoc. Although this approach risks some degree of bias toward the test distribution, several mitigating factors ap-

ply: (i) the test set is substantially larger and more diverse than the training subset, (ii) the model is evaluated against numerous unseen anomalies and normal behaviours, providing a realistic approximation of deployment conditions, and (iii) the chosen performance metric, Area Under the Curve (AUC), reflects overall discriminative ability rather than performance at a single optimised threshold. This training–evaluation protocol was therefore adopted primarily to analyse the sensitivity of reconstruction error over training progress, rather than to fine-tune hyperparameters on the test data. A more conventional training/validation split with a hand-crafted validation set is introduced later in Chapter 6 for more rigorous model selection and comparison.

The architecture was trained with multiple latent compression ratios to examine how compression influences anomalous reconstruction. Excessive compression can degrade the reconstruction of normal inputs, inflating false positives; insufficient compression can allow anomalous inputs to reconstruct too easily, reducing detectability. The optimal balance lies where normal behaviour is well reconstructed but anomalies produce clearly elevated reconstruction error.

The training objective minimises the MSE between the input and its reconstruction:

$$\mathrm{MSE} \;=\; \frac{1}{n}\sum_{i=1}^{n}\big(x_i - \hat{x}_i\big)^2,$$

where x_i represents the original input samples and $\hat{x}_i$ their corresponding reconstructed values. Minimising this loss encourages high-fidelity reconstruction of normal data while revealing anomalies through larger reconstruction deviations during inference.

Model development and training were conducted in Python using PyTorch. A fixed learning rate of 5×10^{-5}, a batch size of 64, and the Adam optimiser were used throughout. No learning-rate scheduling or early stopping was applied for simplicity and reproducibility.

Primarily for training, MSE serves as the evaluation metric, computed per 2,048-sample input–output window. For testing, MSE is still used, but the overall performance is given as AUC (discussed next). Ideally, unseen normal segments yield low reconstruction error, whereas anomalous segments exhibit high error, forming the basis for anomaly discrimination. The relationship between reconstruction error and anomaly detectability will be examined in the results section, alongside qualitative reconstructions and window-wise MSE visualisations.

5.4.2.1 Metric: Area Under the ROC Curve (AUC).

While MSE is used during training to minimise reconstruction loss on the input, its use for anomaly detection requires manually selecting a decision threshold: reconstructions with MSE higher than this threshold are counted as anomalous. Selecting such a threshold manually can be difficult and may limit the generalisation ability of the model.

Thus, the Area Under the Receiver Operating Characteristic Curve (AUC-ROC) provides a threshold-independent measure of how well a model separates two classes, and is widely used in anomaly detection. The ROC curve plots the *True Positive Rate* (TPR) against the *False Positive Rate* (FPR) as the decision threshold is varied. For a given threshold τ on an anomaly score (here, the MSE reconstruction error), these quantities are defined as

$$\mathrm{TPR}(\tau) = \frac{\mathrm{TP}(\tau)}{\mathrm{TP}(\tau) + \mathrm{FN}(\tau)}, \qquad \mathrm{FPR}(\tau) = \frac{\mathrm{FP}(\tau)}{\mathrm{FP}(\tau) + \mathrm{TN}(\tau)}. \tag{5.2}$$

The ROC curve is then the set of points $\Big(\mathrm{FPR}(\tau), \mathrm{TPR}(\tau)\Big)$ obtained as τ ranges over all possible values. The AUC is defined as the area under this curve, which can be written as

$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}\Big(\mathrm{FPR}^{-1}(x)\Big)\,\mathrm{d}x, \tag{5.3}$$

and in practice is approximated numerically, for example using the trapezoidal rule over the sorted FPR–TPR pairs.

In the reconstruction-based models presented in Chapters 5 and 6, each sample is assigned an anomaly score given by its MSE,

$$s(\mathbf{x}) = \frac{1}{d}\sum_{i=1}^{d}\Big(x_i - \hat{x}_i\Big)^2, \tag{5.4}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the input and $\hat{\mathbf{x}}$ is its reconstruction. Anomalies are expected to produce higher scores $s(\mathbf{x})$ than normal samples—this is the same equation for MSE during training, rephrased for anomalous scoring. Because the absolute scale of $s(\mathbf{x})$ can vary across datasets, models, and training runs, AUC is particularly appropriate: it evaluates how well anomalous samples are ranked above normal ones *without* committing to a specific threshold on the MSE (instead computing performance over a wide range of thresholds). This is especially important for the experiments in this Chapter and Chapter 6, where changing the detection threshold on the MSE can greatly affect the reported results. Thus,

the AUC provides a measure of the overall performance of the system without the need for threshold tuning on the test set, which in itself can be a biasing factor. For this reason, AUC was adopted as a primary detection metric alongside threshold-based measures in this work.

Alongside AUC, this work also evaluates the Good Detection versus False Alarm Rate. This is a more application-oriented metric that assesses how many anomalous *instances* (i.e., distinct anomalies, not necessarily all windows labelled as anomalous) are detected versus the number of false alarms at a given threshold. This supporting metric does rely on the selected operating threshold derived from the AUC analysis, but its primary role is to characterise false alarms, which can significantly hamper IDS usability in the field.

5.4.2.2 Results

As outlined in Section 5.4.2, the CAE model described in this Chapter was trained on held-out normal data, progressively increasing compression within the latent bottleneck (i.e., reducing the dimensionality of the central representation). The chosen compression ratios were $8\times$ (latent size 1,024), $16\times$ (512), and $32\times$ (256).

The best-performing epoch for each compression ratio was identified post-training and evaluated on the full unseen test set, comprising 66,306 non-overlapping one-second segments. The test set remained strictly unseen during data collection, emulating a realistic deployment scenario: training and test data were obtained independently, with the normal training data collected first under controlled conditions and the test set introduced later. This clean-room separation prevents leakage at the data-collection stage.

However, selecting the best epoch per compression ratio based on test performance does introduce a mild optimistic bias, since the test set indirectly informs model selection. In this Chapter, this protocol is adopted as an initial exploratory study to characterise and explore the effect of compression ratio on reconstruction behaviour. All reported results in this Chapter should therefore be interpreted as exploratory upper bounds rather than completely unbiased estimates of deployment performance. A stricter training/validation/test procedure with a dedicated validation set is introduced in Chapter 6 for more rigorous model selection and comparison.

To suppress noise in the MSE reconstructions over windows in sequence, the

MSE output for each test segment was post-processed using a Moving Average Filter (MAF). The MAF broadens the model's inference response, smoothing short-term fluctuations in reconstruction error during normal operation. Its influence on performance is summarised in Table 5.1. All reported performance results in this section refer to MAF-smoothed MSE values.

Table 5.1: Best-performing epoch for each latent dimension (lower corresponds to greater compression).

Latent Dim.	Best Epoch	RAW MSE AUC	MAF MSE AUC	MAF Window Length
1024	250	0.778	0.876	16
512	500	0.768	0.874	16
256	4750	0.772	0.868	16

Similar to AUC thresholds, a wide range of MAF window lengths were explored to determine a suitable configuration across the unseen test set, as shown in Figure 5.5. Although tuning smoothing parameters using test data can be contentious, this work reports a spectrum of performance values across different MAF window lengths, illustrating the trade-off between responsiveness and robustness. Given that the test set is substantially larger and more representative of real deployment conditions than the training subset, the selected MAF window length should be interpreted as a pragmatic, deployment-oriented choice rather than an optimally tuned hyperparameter. A window length of 16 (corresponding to approximately 4.75 s of power data) yielded the highest overall AUC.

The motivation for applying the MAF arises from annotation uncertainty in the dataset. Although great care was taken to label each anomaly precisely, human annotation inevitably introduces temporal imprecision, particularly in marking the end of an attack or, in some cases, within a large annotation. For example, during SSH brute-force attacks, quieter intervals may still constitute part of the attack activity. To account for this, slightly longer anomaly labels were sometimes used, and smoothing the MSE response aligns the model output more closely with these extended boundaries. This approach reduces temporal misalignment between model predictions and ground truth, improving recall.

However, in Chapter 6 this post-processing step was omitted. Although it can increase metric performance, it primarily helps match the model output to the specific annotation boundaries, rather than improving the intrinsic discriminative capability of the model. In Chapter 6 the focus is instead on intrinsic model

performance and the contribution of each architectural component. As such, the overall reported performance does not need to be maximised, but rather to be representative of the underlying model behaviour—hence the lower scores reported in Chapter 6.

Among all configurations, the best overall performance occurred with a latent compression of $8\times$ (1,024 dimensions), representing a balanced trade-off between information retention and reconstruction difficulty. Higher compression ratios reduced performance slightly and required significantly longer training to converge. This behaviour is expected, as greater compression increases reconstruction difficulty and the number of epochs required for stable optimisation. The most extreme case—$32\times$ compression (latent size 256)—achieved its best epoch only after approximately 4,750 epochs (see Table 5.1).

Figure 5.5 summarises the main results for the best-performing CAE configuration (latent size 1,024). The unfiltered MSE achieved an AUC of approximately 0.778 (Figure 5.5A), confirming that the model performs well above random chance (AUC 0.5). Applying MAF smoothing significantly improved detection performance (Figure 5.5B), with a window length of 16 yielding a final AUC of 0.876 (Figure 5.5C). This improvement reflects how temporal smoothing reconciles minor misalignments between true anomaly intervals and reconstruction-error spikes.

Event-based performance is depicted in Figure 5.5D. This analysis evaluates the proportion of distinct anomalies correctly detected (rather than per-window performance) as a function of the allowable false alarm rate, expressed as false detections per hour. The results show that the model detects up to 50% of anomalies with no false detections, 70% with fewer than ten false detections per hour, and approximately 80% with around 50 false detections per hour. The false alarm rate is a useful operational metric, as excessive false positives can severely disrupt system availability or lead to operator fatigue.

The qualitative effect of the MAF on the MSE is illustrated in Figure 5.6. Each data point represents the MSE for a single 2,048-sample input window. As shown, the MAF broadens the anomaly response, improving recall for temporally extended events while suppressing transient spikes in normal activity. This smoothing reduces false positives arising from short-lived fluctuations that temporarily exceed the anomaly threshold.

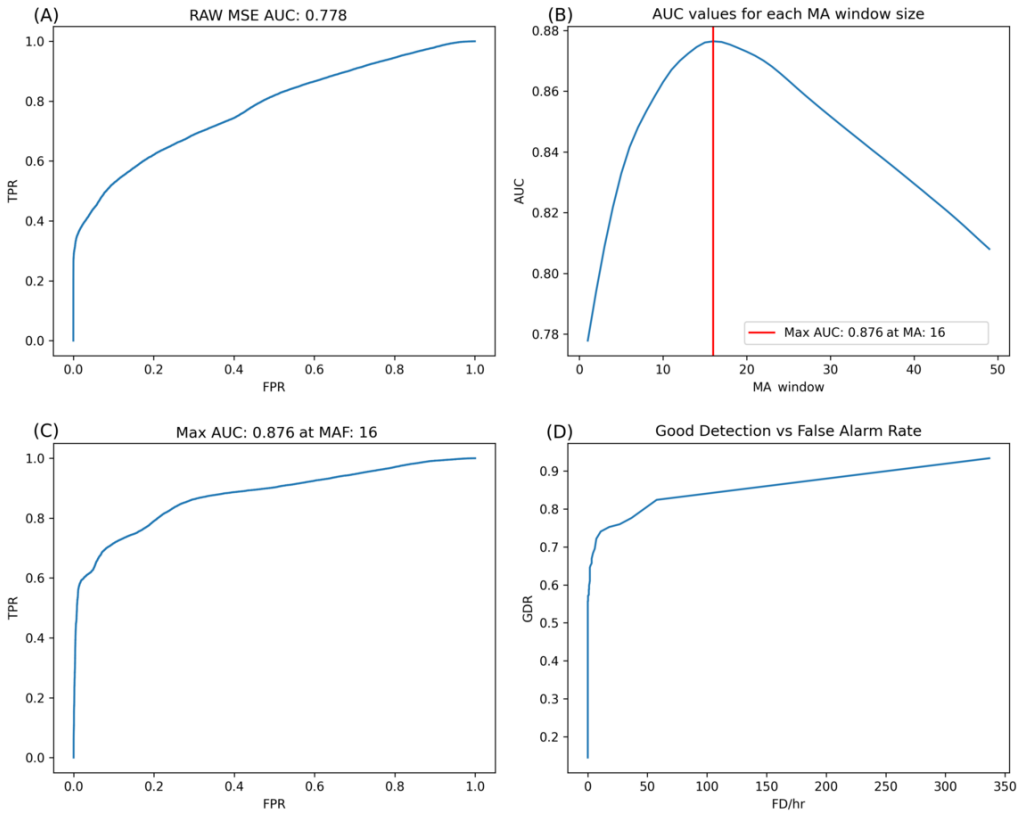


Figure 5.5: Performance of the CAE (1024) model: (*A*) raw AUC without smoothing; (*B*) AUC versus MAF window length; (*C*) ROC curve at optimal MAF 16; and (*D*) event-based anomaly detection rate (GDR) versus false alarms per hour.

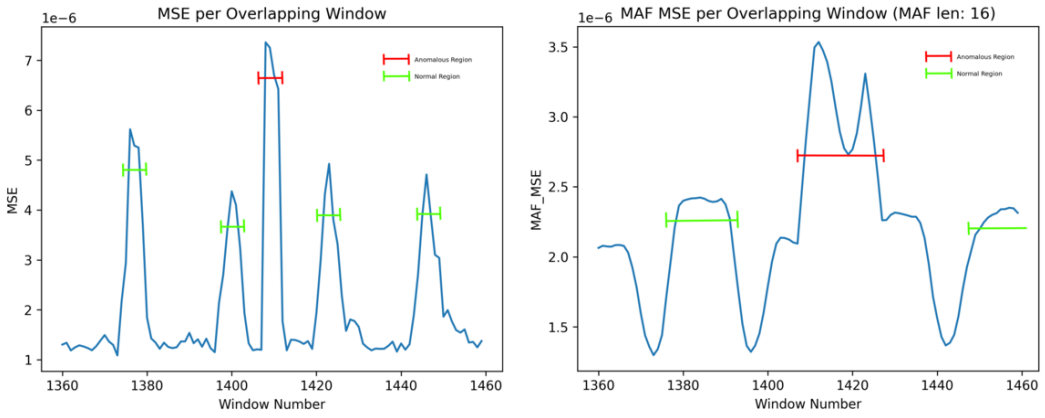


Figure 5.6: Effect of the Moving Average Filter (MAF) on the mean squared error (MSE) signal. MAF smoothing broadens anomaly responses and reduces false positives in normal regions.

5.4.3 Reconstruction Examples

Explainability is an essential component of this exploration into self-supervised learning, ensuring that the model's behaviour can be understood, justified, and verified. In reconstruction-based approaches such as the CAE, explainability arises naturally: the model produces a tangible, human-interpretable output that can be directly compared with its input. This section explores the reconstruction behaviour of the model across normal and anomalous test samples to better understand how it discriminates between them.

As described previously, the model input is a one-second segment of current data comprising 2,048 samples, and the output is the model's reconstruction of this segment. Classification is based on how well the input is reconstructed—the lower the reconstruction error, the more likely the segment represents normal behaviour. Reconstruction quality is quantified using the MSE, which also serves as the training loss. Although alternative metrics exist, MSE remains a widely adopted and interpretable choice for reconstruction-based anomaly detection.

Crucially, this model produces no probabilistic or categorical output; it yields only the reconstructed waveform and the corresponding MSE for each input window. This characteristic greatly enhances interpretability, as analysts can visually inspect where and why reconstruction diverges from the input. In this sense, the model functions similarly to a Fully Convolutional Neural Network (FCNN), enabling localisation of the specific regions within a trace that contribute most to anomaly decisions.

Figure 5.7 presents representative reconstruction examples for both devices, demonstrating the model's behaviour across normal and anomalous conditions. Recall that a single model was trained on held-out normal data from both the DragonBoard and Raspberry Pi, verifying the feasibility of a cross-device approach. As discussed in Chapter 4, both devices exhibit highly similar temporal patterns in their normal power signatures despite differing absolute current amplitudes, enabling a unified model to generalise effectively across them.

Figures 5.7A,B show reconstructions of unseen normal data from the DragonBoard and Raspberry Pi, respectively. The model reproduces both signals with high fidelity, confirming that it has successfully internalised the normal operational patterns of each device. The normal maximum amplitudes range from approximately 0.20 A for the DragonBoard to 0.45 A for the Raspberry Pi.

Despite this amplitude difference, the model reconstructs both accurately, indicating that it primarily learns waveform shape and temporal structure rather than absolute magnitude.

Figures 5.7C,D display reconstructions of a port-scan attack (Nmap_T3). For the DragonBoard (Figure 5.7C), the anomaly amplitude is low—comparable to the device's Bluetooth signature—yet the model still struggles to reproduce the signal's morphology. This suggests that classification is driven primarily by waveform structure rather than amplitude. For the Raspberry Pi (Figure 5.7D), the anomaly's current level (~ 0.45–0.5 A) exceeds the training distribution, and reconstruction failure is seemingly dominated by amplitude.

Figure 5.7E shows a low-amplitude SSH brute-force attempt (Hydra_T1) on the DragonBoard. Since its amplitude closely matches normal Bluetooth activity, reconstruction difficulty arises mainly from the unfamiliar waveform shape. Conversely, in Figure 5.7F (Hydra_T1 on Raspberry Pi), both amplitude and waveform irregularity can be seen contributing to the anomaly signal.

Hydra_T16 attacks, shown in Figures 5.7G,H, demonstrate more pronounced anomalies. For the DragonBoard (Figure 5.7G), amplitude roughly doubles relative to typical operation but remains within the learned dynamic range of the overall dataset; reconstruction failure thus arises primarily from unfamiliar structural variation. For the Raspberry Pi (Figure 5.7H), amplitude far exceeds any normal behaviour, leading to near-complete reconstruction failure.

Figures 5.7I,J illustrate infiltration activity, where commands were issued remotely via SSH to the compromised device. In Figure 5.7I (DragonBoard), the anomaly exhibits a lower-than-normal amplitude but includes a distinctive high-frequency component, resulting in reconstruction errors across both shape and spectral content. In Figure 5.7J (Raspberry Pi), the elevated amplitude contributes more strongly to the reconstruction error, although waveform irregularity remains a secondary factor.

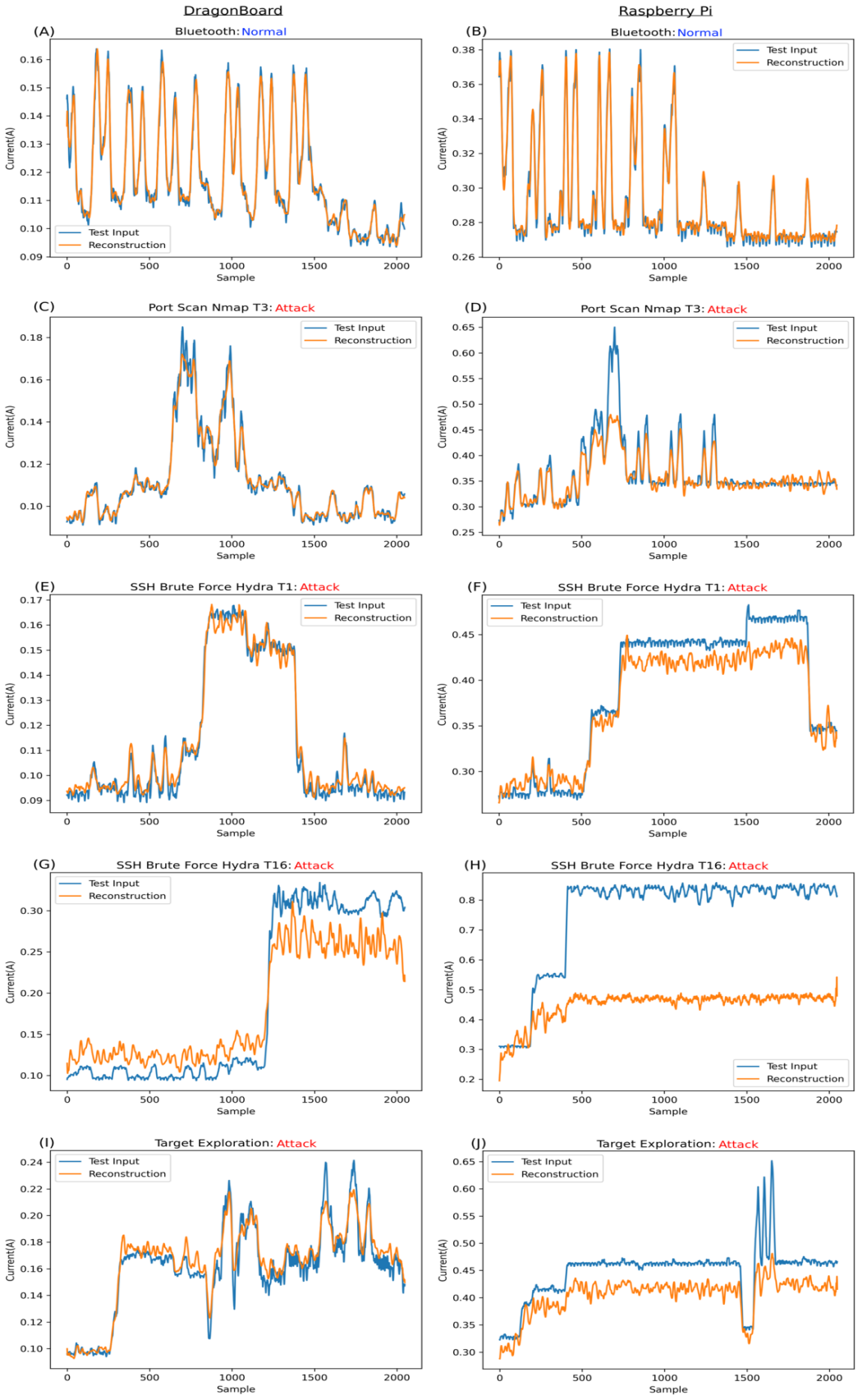


Figure 5.7: Reconstructions using the CAE (1024) model (DragonBoard (left) vs. Raspberry Pi (right)). (*A*,*B*) normal behaviour; (*C*,*D*) `Nmap_T3` anomalies; (*E*,*F*) `Hydra_T1` anomalies; (*G*,*H*) `Hydra_T16` anomalies; (*I*,*J*) infiltration traces.

Across these examples, several key insights emerge. First, the model achieves its intended behaviour: it reconstructs unseen normal data with low error while producing elevated error for anomalous patterns. Second, the model is sensitive to both amplitude and waveform structure. When anomalies fall within the normal amplitude range, detection relies primarily on morphological discrepancies; when amplitudes exceed training bounds, reconstruction failure is dominated by magnitude. Finally, the relatively close reconstruction of some low-amplitude anomalies suggests overfitting, where the model may have learned to reproduce features too precisely even in unfamiliar contexts. This behaviour is examined further in Section 5.4.4, which analyses reconstruction error distributions and temporal error maps in detail.

5.4.4 MSE Exploration

Rather than study only isolated reconstructions, it was decided to inspect the model's performance over time, as it would be in deployment. Figure 5.8A presents the MSE across 101 consecutive segments (indices 1360–1460). Each point corresponds to the MSE of a full 2,048-sample reconstruction window, computed between the reconstructed and original signals.

Within the normal data, periodic bursts of Bluetooth activity occur approximately every five seconds, producing expected increases in reconstruction error. These peaks represent legitimate, benign fluctuations and should remain consistently lower than the pronounced error spikes produced by true anomalies.

Closer inspection of Figure 5.8 reveals a notable concern. Despite the model's reasonable quantitative performance, the MSE of certain anomalous windows (e.g., window 1410, highlighted in red) lies uncomfortably close to that of normal Bluetooth activity (e.g., windows 1400 and 1423, highlighted in green). Detailed examination of these windows shows that the model reconstructs normal Bluetooth segments faithfully (Figures 5.8B,C), while also reconstructing the anomaly in window 1410 surprisingly well (Figure 5.8D).

This behaviour reflects a well-documented limitation of AEs: when under-regularised or insufficiently compressed, they tend toward near-identity mapping, reproducing almost any input regardless of class. In such cases, the latent space fails to act as a meaningful information bottleneck, and anomalies no longer yield distinct reconstruction errors.

While the model demonstrates functional anomaly detection, anomalies with

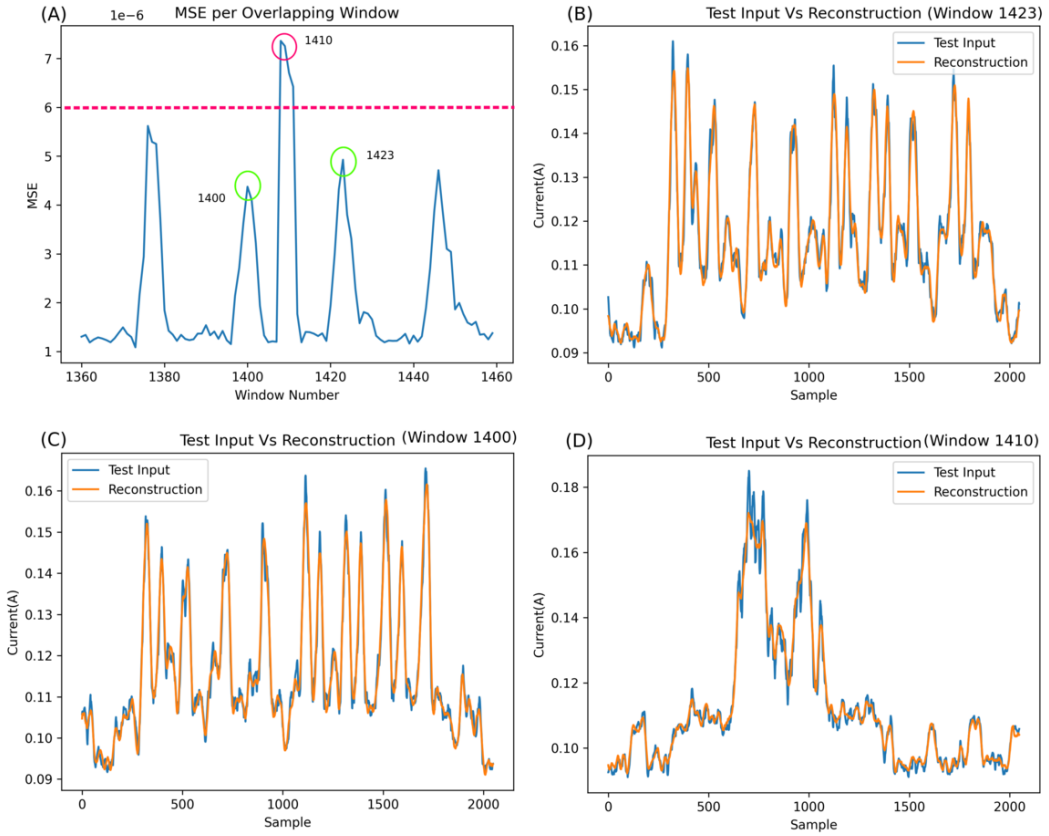


Figure 5.8: Exploration of mean squared error (MSE) behaviour in the CAE (1024) model. (*A*) Window-wise MSE across 101 segments (1360–1460) highlighting one low-amplitude anomaly (red) and two normal Bluetooth windows (green). (*B*,*C*) normal Bluetooth reconstructions. (*D*) Anomalous reconstruction (window 1410), showing unexpectedly low error due to partial overfitting.

low amplitude—such as the one in Figure 5.8—produce MSE values that are too close to those of normal segments. The anomaly's MSE ($\approx 7 \times 10^{-6}$) sits only marginally above the Bluetooth baseline, leaving insufficient margin for reliable classification. A simple arbitrary threshold at 6×10^{-6}, illustrated in Figure 5.8A, highlights this narrow separation: the anomalous segment would barely exceed the decision boundary.

This observation underscores an important limitation of the current configuration. Although overall metrics such as AUC may appear satisfactory, the per-segment MSE distribution reveals a lack of discriminative gap between normal and anomalous reconstructions. Robust anomaly detection requires not only high metric performance but also consistent per-instance separability.

In practical terms, a more trustworthy model should exhibit clear divergence in reconstruction error whenever it encounters unseen or structurally

unfamiliar data. Strengthening this separation likely requires architectural refinements—either increased regularisation, more aggressive latent compression, or improved activation nonlinearity—to ensure that the model *struggles* appropriately with anomalous inputs. Even if headline metrics change only marginally, such enhancements would yield a detector that is both more reliable and more interpretable in deployment.

5.5 Dropout Architecture and Training

5.5.1 Architecture

As demonstrated in Section 5.4.4, the baseline CAE exhibited mild overfitting, reconstructing some anomalous inputs with unrealistically low error. Overfitting is common in AE-based models, particularly when training data are limited or structurally simple. Several strategies exist to mitigate this behaviour, including reducing model capacity, increasing compression, or applying explicit regularisation. However, as witnessed already in Section 5.4.2, stronger compression did not yield consistent improvements. Consequently, this Chapter introduces heavy *dropout* [95] as a targeted regularisation technique to enhance robustness without altering the model's overall architecture, as shown in Figure 5.9.

Dropout operates by randomly deactivating neurons during training with a predefined probability. This stochastic omission injects controlled noise into the learning process, preventing co-adaptation among neurons and forcing individual units to learn more general, independent features. In essence, dropout discourages the formation of low-resistance "information pathways," where a subset of neurons dominates reconstruction by mapping training data. By intermittently removing neuron activations, dropout compels the network to distribute representational responsibility more evenly across layers. This approach is especially effective in AEs, where overfitting often manifests as identity mapping—the ability to reproduce any input regardless of whether it belongs to the normal distribution.

Conceptually, introducing noise to AEs is not new. Denoising Autoencoders (DAEs), for example, explicitly corrupt the input data during training, encouraging the network to reconstruct the original, uncorrupted signal. The present approach adapts this principle to some degree: rather than corrupting the

input itself, dropout introduces stochastic corruption *within* the network's latent representation. This compels the CAE to learn around the induced noise, promoting stronger feature abstraction and reducing its tendency to memorise fine-grained details of the training data. In anomaly detection, this is advantageous—the model learns to reconstruct normal behaviour even when presented with partially corrupted internal activations, while failing to reconstruct genuinely anomalous inputs that deviate from learned norms.

Comparing dropout-based reconstructions with the original input reveals a crucial effect: anomalies yield consistently higher reconstruction errors, thereby improving discriminability between normal and abnormal segments. In this way, dropout enhances robustness and trustworthiness by forcing the model to generalise rather than simply fit the input.

Although dropout was the chosen strategy here, similar effects might be achieved through direct input corruption, similar to the approach taken in Chapter 6 where input masking is employed. Such an approach could also augment training data by producing multiple stochastic variants of each sample—particularly useful for small datasets.

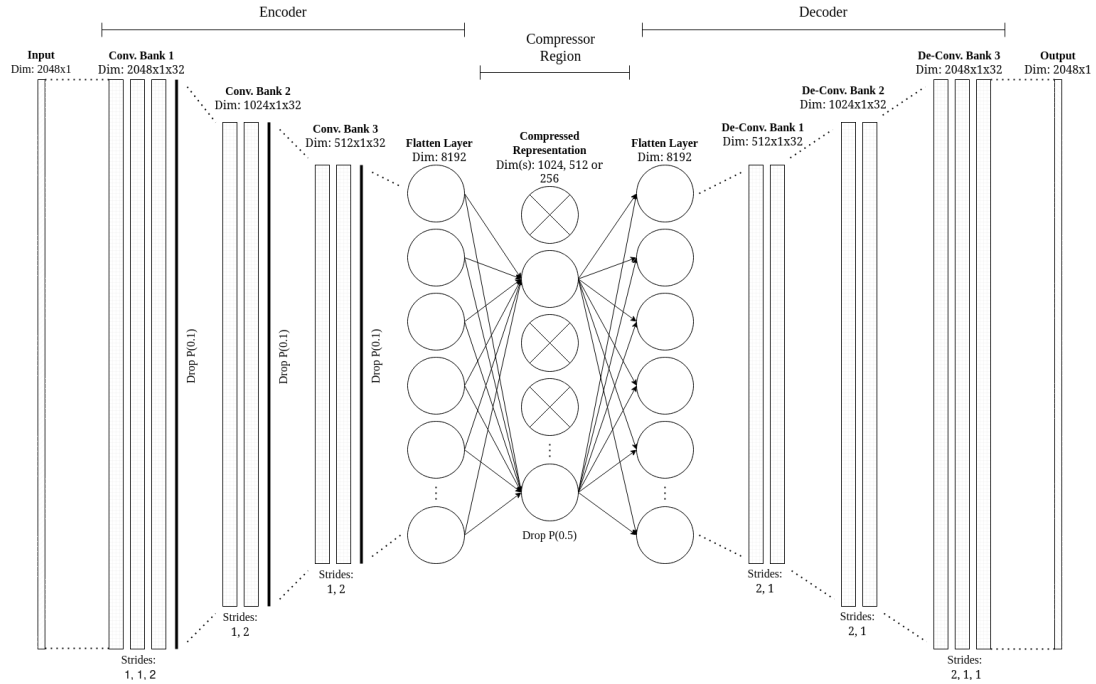


Figure 5.9: Revised CAE architecture incorporating dropout regularisation. Dropout is applied after GELU activations in both convolutional and dense layers, with dropout probabilities of 0.1 and 0.5, respectively.

It is important to note that dropout operates only during training; during infer-

ence, all neurons remain active and the full model is used. In this work, dropout layers were applied after GELU activations throughout the network. To further diversify the noise source, dropout was also applied to convolutional layers—at a lower probability than in dense layers [137]. Specifically, a dropout rate of 0.1 was used for convolutional filters, while the dense latent layer employed a rate of 0.5. These probabilities provided a balanced trade-off between sufficient regularisation and stable training convergence. Figure 5.9 illustrates the revised CAE architecture with dropout applied.

5.5.2 Training

Aside from the inclusion of dropout layers, the training process followed the same procedure described in Section 5.4.2. The introduction of dropout, however, had a pronounced effect on convergence behaviour. Because dropout injects stochastic noise into learning, the model required substantially more epochs to stabilise and reach optimal performance.

For example, for the model with latent size 1,024, the best epoch without dropout occurred at approximately epoch 250. With dropout enabled, peak performance was achieved much later—around epochs 1,500 and 2,250 (both exhibiting nearly identical metrics). To ensure consistency across all subsequent figures and comparisons, epoch 2,250 is used as the representative model for the 1,024–latent configuration with dropout.

This extended training duration is consistent with expectations: dropout increases the stochasticity of gradient updates, slowing convergence but promoting generalisation [92].

5.5.3 Results

Incorporating dropout into the CAE led to a modest yet meaningful performance improvement in the moving-average-filtered (MAF) metric, increasing from an AUC of 0.876 in the baseline model to 0.890 in the dropout-regularised variant. This improvement, though incremental, indicates enhanced robustness to unseen anomalies. A slight reduction in raw (unfiltered) AUC was observed—from 0.778 to approximately 0.766—which is expected given dropout's regularising influence: it introduces reconstruction noise that slightly reduces fine-grained fidelity but improves discrimination after smoothing. The detailed results for each compression level are summarised in Table 5.2. With-

out post-processing, however, the trade-off between fidelity and separability would need to be reconsidered.

Table 5.2: Best-performing epoch and performance summary for each latent compression level (with dropout).

Compression	Best Epoch	Raw MSE AUC	MAF MSE AUC	MAF Window Length
1,024	2,250	**0.766**	**0.890**	17
1,024	1,500	0.764	0.890	17
512	2,000	0.759	0.886	18
256	3,250	0.752	0.884	17

As in the baseline model, performance gradually decreased as latent compression increased. The 1,024-latent configuration remained the best-performing architecture, offering the optimal balance between representational capacity and generalisation. All subsequent figures in this section therefore refer to this configuration.

Beyond aggregate AUC improvements, dropout yielded clear benefits in the event-level General Detection Rate (GDR). As shown in Figure 5.10, the 1,024-dropout model achieved approximately 65% correct anomaly detection at zero false positives per hour, 77% at fewer than ten false positives, and 85% at fewer than fifty false positives per hour. These results demonstrate a more favourable trade-off between sensitivity and stability, suggesting that dropout not only improves global detection accuracy but also reduces the volatility of reconstruction errors over time.

Importantly, as the metric-driven results are largely similar to the baseline, the focus now shifts from metrics to *behavioural robustness*: whether dropout increases the separation between the MSE of normal and anomalous segments. This separation—the effective gap in reconstruction error—is the true measure of the model's resilience and will be examined in detail in the next section.

5.5.4 Reconstructions with Dropout

Figure 5.11 presents the reconstructions of the same normal and anomalous behaviours shown previously in Figure 5.7, now using the dropout-regularised model.

In Figures 5.11A,B, the model continues to reconstruct normal behaviour effectively, though with a higher overall reconstruction error compared with the baseline (non-dropout) variant. This increase in baseline error is expected and

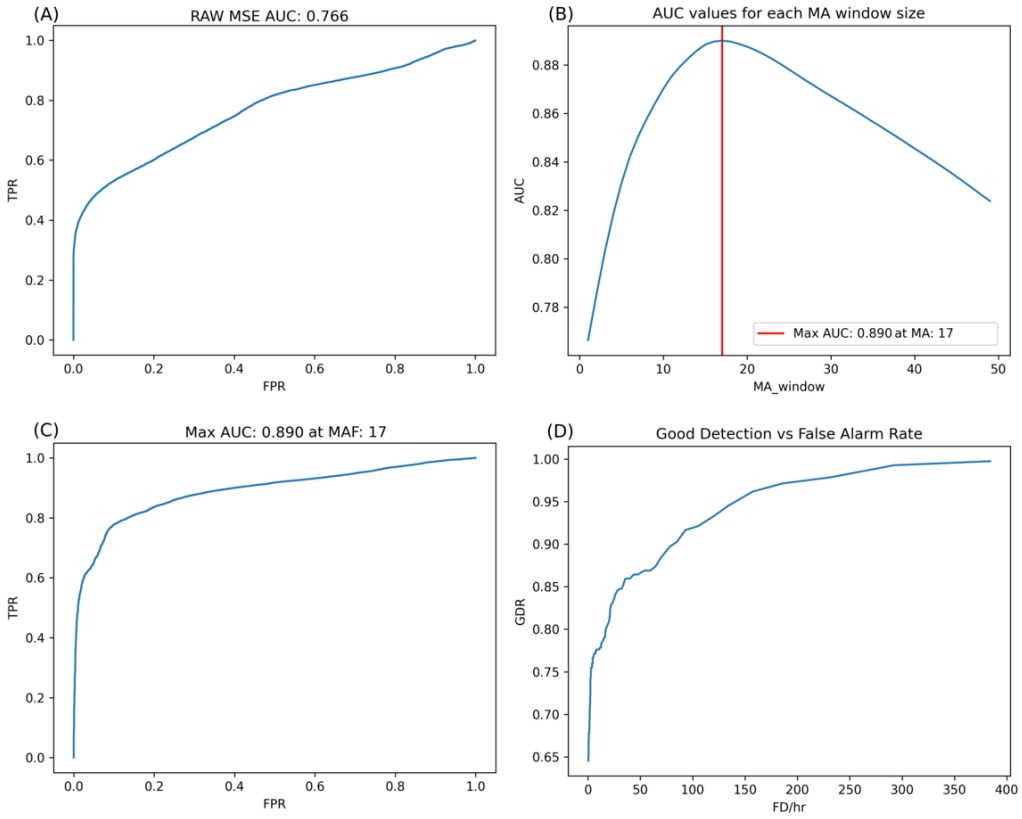


Figure 5.10: Performance of the dropout-based CAE (1024) model. (*A*) Raw MSE AUC; (*B*) AUC versus MAF window length; (*C*) ROC curve at optimal MAF length; and (*D*) GDR performance showing detection versus false-alarm rate.

acceptable: the key indicator of success is not absolute reconstruction accuracy but rather the *relative gap* between errors on normal and anomalous segments. The slight degradation in overall reconstruction quality is a natural consequence of dropout-induced noise and represents a positive trade-off for increased robustness and reduced overfitting.

The inclusion of dropout renders the model less susceptible to memorisation, resulting in reconstructions that are more stable and trustworthy. The model is now more likely to struggle with data that deviate structurally from the learned normal distribution—precisely the behaviour desired for anomaly detection.

Figures 5.11C,D illustrate reconstructions of the `Nmap_T3` anomaly, and this new desired behaviour. In this case, the dropout model performs markedly worse at reproducing the anomaly—an outcome that is both expected and beneficial. In Figure 5.11C, the model fails to follow the abnormal shape of the input, instead reverting to an approximate normal pattern. This inability to reconstruct unseen structures confirms that the model has learned a strong bias toward

normal behaviour. Similarly, in Figure 5.11D, the anomaly exceeds the normal operational amplitude, causing a complete breakdown in reconstruction. The model attempts to impose a familiar shape over the anomalous region, leading to a high MSE—again, an advantageous response.

In Figure 5.11E, the `Hydra_T1` SSH brute-force anomaly on the DragonBoard shows clear evidence of dropout-driven improvement. Although surrounding normal behaviour is reconstructed reasonably well, the model fails to reproduce the anomalous region entirely. Compared with the baseline case, this highlights the increased selectivity introduced by dropout: the model is now unable to replicate the anomaly even when its amplitude lies within the normal range. Figure 5.11F shows the same attack on the Raspberry Pi. Here, the input briefly exceeds the normal operating amplitude, further challenging the model. It defaults to reconstructing a typical normal pattern instead, producing a large reconstruction error—again, a highly desirable outcome.

Figure 5.11G shows the `Hydra_T16` attack on the DragonBoard. The model successfully reconstructs the baseline portion of normal activity but entirely fails to reconstruct the anomaly spike, maintaining a near-constant baseline instead. This behaviour reflects a robust model that resists fitting to unexpected inputs. In Figure 5.11H, the Raspberry Pi's `Hydra_T16` attack further demonstrates this property: the model cannot reconstruct the anomaly at all and instead generates a confused signal pattern somewhere between previously seen normal signatures of both devices. While this confusion may appear suboptimal visually, it is ideal for detection, as it yields a large reconstruction error that strongly differentiates anomalous from normal behaviour.

Figures 5.11I,J depict reconstructions of infiltration-related anomalies, where attackers interacted with the device to execute commands and view files. In both cases, the model shows a complete inability to reconstruct the anomalous regions—another indication that dropout successfully enhances discriminative separation between normal and abnormal data.

Despite the similar metric-driven performance, overall the dropout-regularised model achieves precisely the intended effect:

- Normal segments are reconstructed accurately enough to preserve interpretability.
- Anomalous segments—whether due to shape or amplitude—are no longer faithfully reconstructed.

- The resulting MSE gap between normal and anomalous data is substantially widened, leading to more robust and trustworthy anomaly detection.

A particularly interesting observation is that signal *shape* now plays a much stronger role in driving reconstruction failure. In the baseline model, amplitude dominated the error signal; here, shape and amplitude contribute more equally. Even low-amplitude anomalies that fall within normal operating levels (e.g., Figures 5.11C,E,F) become non-reconstructable due to their structural differences. For anomalies exceeding the normal amplitude, the model's response is also more chaotic than before, often generating distorted attempted shapes rather than simple plateaus, which further increases MSE and detection reliability.

In summary, dropout achieved its intended purpose: it prevented overfitting, forced generalisation across both devices, and significantly sharpened the distinction between normal and anomalous behaviour. The resulting architecture is more resilient, interpretable, and better aligned with the practical demands of anomaly detection in IoT systems.

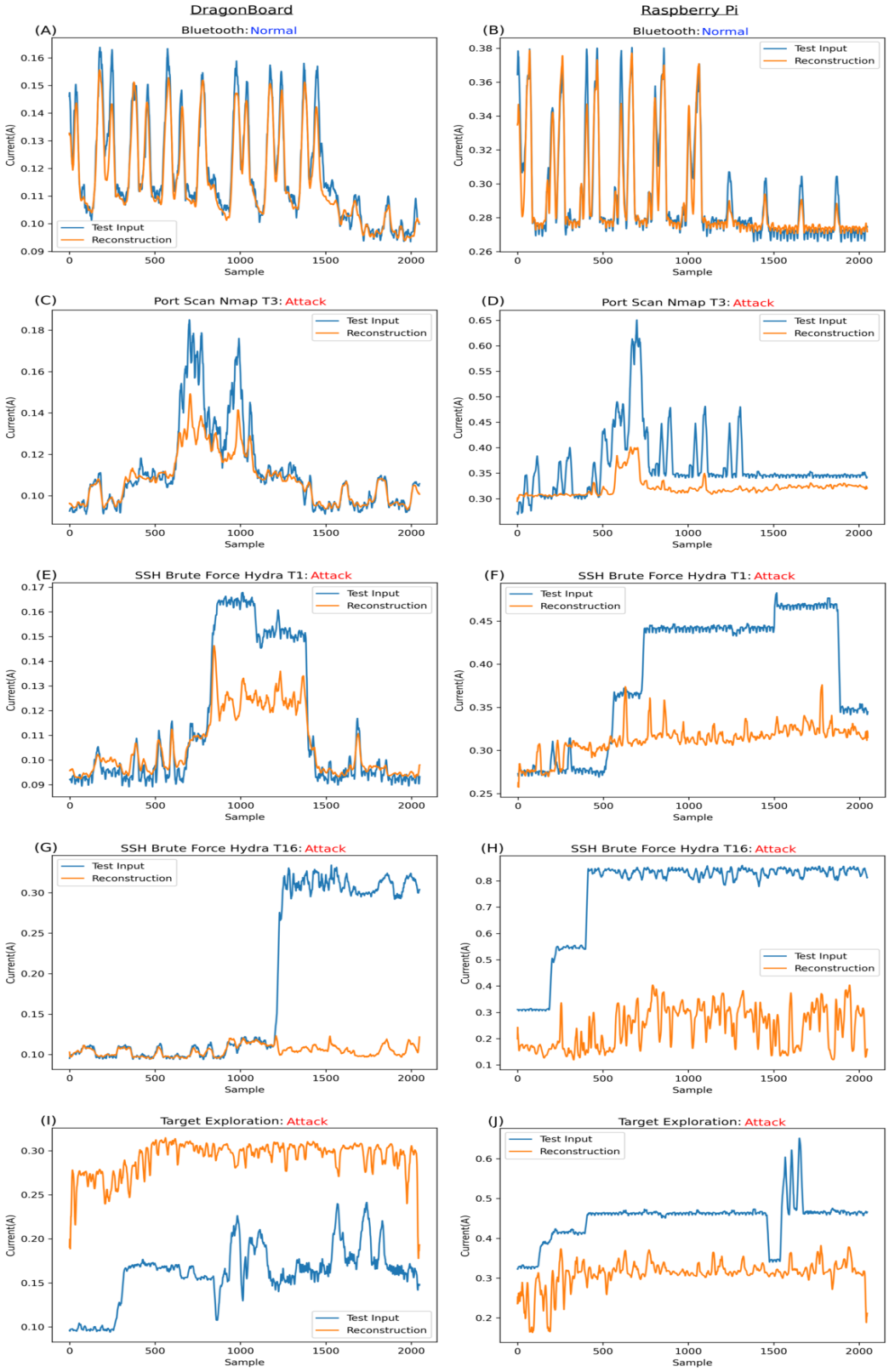


Figure 5.11: Reconstructions using the dropout-regularised CAE (1024) model (DragonBoard (left) vs. Raspberry Pi (right)). (*A*,*B*) normal behaviour; (*C*,*D*) `Nmap_T3` anomalies; (*E*,*F*) `Hydra_T1` anomalies; (*G*,*H*) `Hydra_T16` anomalies; (*I*,*J*) infiltration traces.

5.5.5 MSE Exploration With Dropout

Analysis of the MSE for the dropout-based model further reinforces its effectiveness. When examining the same data segments as in Figure 5.8, a much clearer separation emerges between the reconstruction error of normal Bluetooth behaviour and that of anomalous activity.

As illustrated in Figures 5.12B,C, the normal Bluetooth segments (windows 1400 and 1423) are reconstructed with reasonably low error, albeit slightly higher than in the non-dropout model. In contrast, Figure 5.12D shows the reconstruction error for a `Nmap_T3` port scan anomaly. Here, the error rises sharply, with values nearly six times higher than those of surrounding normal Bluetooth spikes. This clear amplification of anomaly-induced error provides a stronger margin for discrimination. Even when applying a simple threshold—for instance, midway between the average normal and anomalous MSE values—the likelihood of false positives is greatly reduced compared to the non-dropout case.

Overall, the distinct gap between the reconstruction errors of normal and anomalous behaviour demonstrates that dropout significantly improves the reliability and trustworthiness of the model. It compels the CAE to learn robust features that generalise well to unseen data, rather than simply reproducing familiar inputs. A more detailed comparison between the non-dropout and dropout models is presented in Section 5.6, where these improvements are quantitatively and qualitatively assessed.

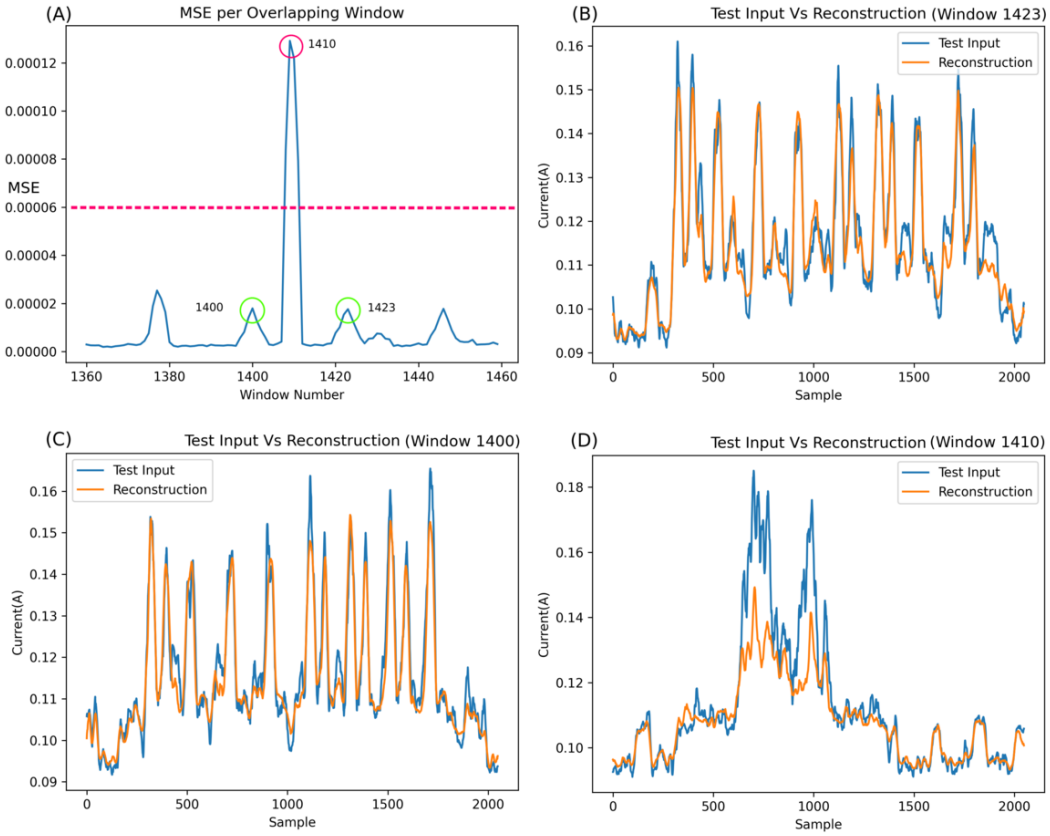


Figure 5.12: MSE analysis of the dropout-based CAE (1024) model. (*A*) Overall MSE sequence; (*B*,*C*) Bluetooth behaviour segments; and (*D*) `Nmap_T3` anomaly. The dropout model produces a distinct separation between normal and anomalous error magnitudes, enhancing robustness.

5.6 Discussions

5.6.1 Self-supervised AI

The defining feature of this Chapter is its adoption of a *self-supervised* approach. As shown in Chapter 4, attack signatures in device power data are highly dynamic, with even minor configuration changes resulting in markedly different patterns. This inherent variability discourages the use of supervised techniques for anomaly detection in IoT power traces.

This caution does not stem from poor test-set performance of supervised models under controlled conditions, but from their lack of *real-world applicability*. A fully supervised anomaly detector might achieve impressive accuracy in a laboratory setting, yet such performance is misleading: anomalies, by definition, are unexpected and unseen. Training on both normal and anomalous data there-

fore risks overfitting to known patterns, yielding models that fail to generalise once deployed.

Some studies attempt to approximate unseen conditions through strategies such as leave-one-out anomaly exclusion during training. However, such approaches still fail to capture the open-ended diversity of real-world anomalies. Even if a model performs well under these contrived conditions, there is no guarantee it will detect entirely novel behaviours outside the training distribution. In contrast, a self-supervised, reconstruction-based strategy that focuses exclusively on normal operation aligns naturally with the IoT context, where normal behaviour is stable, reproducible, and easily collected at scale.

Training on anomaly data would also bias the learning process by over-guiding the model towards specific, pre-defined anomaly forms. The work presented in this Chapter therefore trains solely on held-out normal data collected under controlled conditions, while evaluation is performed using unseen, realistically acquired test data. As a result, model performance directly reflects generalisation to unknown events—an essential property for reliable, real-world anomaly detection.

Requiring only normal data for training also reduced the need for vast amounts of costly anomalous data. Thus, any valuable anomaly data that is obtained can be used solely for validation and testing, and abundant normal data can be used for training.

5.6.2 Human-in-the-Loop

A key strength of the proposed framework lies in its ability to maintain human oversight within the AI loop. Segments (epochs) of concern can be automatically flagged and referred to a human analyst for review. Because the model output is inherently interpretable—comprising the input signal, its reconstruction, and the associated error—security personnel can readily inspect regions of interest and determine whether intervention is warranted. This design ensures that the system augments, rather than replaces, human expertise, enabling faster and more informed decisions during ongoing or suspected attacks.

5.6.3 Explainability of the AI

Explainability is a central advantage of the CAE model presented in this Chapter. Since classification is derived directly from the MSE between the input

and its reconstruction, tracing the reasoning behind each decision is straightforward. Segments with high error can be identified and inspected visually by comparing the input and output waveforms, as shown in earlier figures. The model's difficulty in reconstructing anomalous inputs provides an immediate, tangible explanation for its decision.

This transparency distinguishes the CAE from many other anomaly-detection techniques that output only a class probability or score, offering little insight into *why* a particular decision was made. In contrast, reconstruction-based models—and related architectures such as FCNNs—enable direct attribution: analysts can pinpoint the regions of the signal that contributed most to an anomaly classification.

For less pronounced signatures, such as low-amplitude port scans or brief brute-force attempts, the model reconstructs the surrounding normal behaviour while failing to reconstruct the anomalous portion. This allows analysts to identify precisely where the system diverged from expectation. For high-amplitude attacks, the reconstruction may collapse entirely, reflecting that the input exceeds the normal operating range learned during training. In both cases, the model's output provides clear, human-understandable reasoning for its decision.

5.7 Summary

This Chapter presented a CAE as an initial investigation into a framework for self-supervised intrusion detection in IoT devices using side-channel power data. The work demonstrated that an AE trained solely on normal behaviour can effectively detect anomalies as deviations from learned patterns, even when attack signatures can vary widely across devices and configurations.

Two versions of the CAE were explored: a baseline model and a dropout-regularised variant. The non-dropout model achieved an AUC of 0.778 without post-processing and 0.876 with a mean average filter (MAF) of length 16 applied to its MSE output. Although this version reconstructed normal behaviour nearly perfectly, it also tended to reconstruct anomalies overly well, limiting its discriminative reliability. Introducing dropout—intended to reduce overfitting and counteract identity mapping—substantially improved the model's robustness. The dropout variant achieved an AUC of 0.764 without post-processing and 0.890 with an MAF of length 17. More importantly, it exhibited a significantly larger gap in reconstruction error between normal and anomalous

classes, yielding a more trustworthy and operationally resilient detector.

Reconstructions showed that while both models captured normal behaviour effectively, the dropout model struggled markedly with anomalies, producing higher reconstruction errors that aligned with their abnormal structure. This behaviour is exactly what is desired in a reconstruction-based intrusion detector: accurate reconstruction of normal data and systematic failure on abnormal inputs.

A central advantage of reconstruction-based approaches, such as the CAE used in this Chapter, lies in their explainability. Each detection can be retraced through the input signal, its reconstruction, and the resulting error, providing a clear visual rationale for each detection. This transparency enables the inclusion of human analysts within the loop, supporting both real-time judgement and post-incident forensic analysis.

Despite its strong performance and interpretability, several limitations highlight avenues for future research.

The *evaluation method* could also be refined. Rather than averaging AUC across individual attack files, computing a unified AUC across all anomalies may yield a more holistic view of performance. This could be further enhanced by analysing performance per device and per attack type, computing AUC over these subsets rather than over individual files. This improvement is applied in Chapter 6.

Relying on post-processing, while common, may obscure certain strengths or weaknesses of the model. Chapter 6 therefore prioritises raw model performance without temporal smoothing.

Furthermore, although a validation set was not strictly necessary for this initial investigation, more detailed architectural studies—and any rigorous hyperparameter tuning—should incorporate a dedicated validation split to avoid biasing results toward the test distribution. Hence, in Chapter 6 a carefully chosen validation set is created.

Finally, while this Chapter demonstrated the value of adopting a self-supervised paradigm, the dataset remains limited in scale. Larger and more diverse real-world datasets would allow deeper exploration of model limits and improve statistical confidence in the findings.

In summary, although an initial investigation, this Chapter begins the ground-

work for fulfilling the *third aim of the thesis*: transitioning from the fully supervised approaches of Chapter 3 to self-supervised methods that preserve the anomalous nature of attack data, as motivated by the findings of Chapter 4.

This Chapter forms a stepping stone for Chapter 6, which synthesises the insights gained thus far, and improves on the methodology in this Chapter where possible, to create an advanced, modern Transformer-based architecture trained solely on real normal device data for detecting real attacks.

Chapter 6

Compression Enhanced Masked Autoencoders for Self-Supervised Intrusion Detection in IoT Devices

6.1 Introduction

Building on the findings of Chapter 5—which demonstrated that self-supervised models trained solely on normal data can generalise well and detect unseen anomalies while maintaining interpretability—this Chapter advances both the architecture and the learning strategy of power-based intrusion detection.

In this Chapter, two hybrid architectures, CONFORMER–MAE and COMP-MAE, are developed. These models integrate Conformer-style local–global modelling [24] with Masked Autoencoder (MAE) reconstruction [25]. By combining convolutional inductive bias with Transformer attention, they capture both fine-grained and long-range dependencies in power time-series, addressing representational limitations of the Convolutional Autoencoders (CAEs) introduced in Chapter 5. In doing so, this Chapter delivers the core objective of the third aim of the thesis: the development of an advanced, lightweight, self-supervised intrusion detection model suitable for real-world IoT deployment.

The COMPMAE framework builds directly on the self-supervised reconstruction methods established in Chapter 5, extending them into a Transformer-based paradigm. It integrates three complementary ideas: (i) *Masked Autoencoding* for robust self-supervised representation learning under heavy masking;

(ii) *Conformer*-style blocks that fuse local convolution with global self-attention; and (iii) a lightweight, *AE-inspired novel compressor module* that enforces a latent bottleneck and enhances anomaly sensitivity, while maintaining local and global resolution.

As in Chapter 5, intrusion detection is performed via reconstruction error on time-series power traces. However, the representational capacity and learning dynamics here are substantially more expressive. Following the MAE paradigm, the model is trained exclusively on *normal* power data using a high masking ratio (75%), observing only 25% of the original input during pre-training. Unlike dropout in Chapter 5, masking removes parts of the input itself rather than dropping neurons. This forces the model to learn deeper representations of the training data—far beyond what dropout alone can achieve—while also avoiding the capacity limitations associated with high dropout rates.

Masking is disabled at inference time, enabling the model to reconstruct normal sequences faithfully while producing elevated reconstruction error for anomalous patterns. The novel compressor module reinforces this behaviour by constraining the latent space, encouraging the model to encode only the invariant characteristics of normal operation, and subsequently enhancing its reaction to anomalies.

Two models are developed and evaluated on the DRAGON_PI dataset, as in Chapter 5: (1) CONFORMER–MAE, a baseline demonstrating the benefits of hybrid local–global modelling as well as the innate anomaly detection capabilities of such models; and (2) COMPMAE, a compression-enhanced variant designed to amplify reactions to anomalous data while maintaining stable reconstruction on normal data.

Unlike Chapter 5, no post-processing is used: all results are reported directly from the raw reconstruction output to highlight intrinsic modelling capability and the contribution of the compressor module, rather than tuning reconstruction error to match the annotations of the test set. The evaluation protocol is also more controlled than in Chapter 5: a dedicated validation set is introduced, and test performance is further analysed across device subsets, attack subsets, and device-independent scenarios. Crucially, the Area Under the Curve (AUC) is computed over the entire set or subset at once (rather than averaged per file), providing a fairer global measure of detection performance. The Capture-the-Flag (CTF) data is also treated as a special test set due to its realistic nature, and performance is assessed separately on this subset—giving a glimpse into

how a model would fare during a real attack scenario.

A key architectural and behavioural departure from Chapter 5 is that COMPMAE is engineered not only to fail on anomalous inputs, but to amplify their deviation through its compressed latent pathway. This produces more pronounced reconstruction errors for attacks, aiding thresholding, detection, and interpretability—effectively enhancing the model's reaction to anomalies. Despite incorporating modern components such as Multi-head Self-attention (MHSA), the design remains lightweight and edge-feasible. Evolving to modern architectures like Transformers also allows for natural extensibility toward future multimodal fusion (e.g., combining power and network traces) due to their strong representational capabilities.

Finally, an ablation study isolates the contribution of each key component in COMPMAE, clarifying where performance improvements arise and how they trade off against model complexity (parameter count). Together, these analyses aim to establish a principled foundation for deploying self-supervised, reconstruction-based intrusion detection on resource-constrained IoT devices.

Overall, COMPMAE represents a direct evolution of the work in Chapter 5, offering both a more advanced and scalable architecture—via the parallelism and capacity of Transformers, Conformers and MAE style deep representation learning—and a more rigorous evaluation strategy while maintaining a lightweight, edge-deployable nature, therefore fulfilling the third aim of the thesis.

6.1.1 Architectural Assumptions

The self-supervised framework introduced in this Chapter builds upon prior stages and operates under the following architectural and deployment assumptions.

- The model is trained exclusively on **normal power data**, preserving the anomaly-detection paradigm established in earlier chapters.
- Power traces are assumed to contain both local and long-range temporal dependencies, motivating the use of architectures capable of modelling multi-scale structure.
- **Deployment configuration.** Training is performed offline using collected datasets, while inference is designed to operate on resource-constrained

platforms. In this Chapter, the model is also evaluated on-device using CPU-based inference, demonstrating practical edge feasibility. Deployment may also be performed on an adjacent device (e.g., gateway) where required.

- **Data transmission (adjacent inference).** Where inference is performed on an adjacent device, it is assumed that power data are transmitted as time-series segments at a fixed sampling rate, either continuously or in short windows. This transmission is assumed to occur over a lightweight communication interface (e.g., serial, SPI, or network-based transfer), with minimal preprocessing performed on the source device.
- **Computational constraints.** The target deployment platforms are assumed to have limited computational, memory, and power resources. As such, the model is designed to balance representational capacity with efficiency to remain suitable for real-time or near-real-time inference.
- The compression module assumes that normal behaviour can be represented within a reduced latent space, while anomalous behaviour lacks coherent structure and is therefore more difficult to reconstruct.
- **Application workload assumptions.** As in previous chapters, device workloads are assumed to be sufficiently representative of real-world operation, though not exhaustive of all possible behaviours.

These assumptions underpin the design of COMPMAE, enabling it to balance modelling capacity, anomaly sensitivity, and practical deployability within IoT settings.

6.2 Background: Transformers, Conformers, and Masked Autoencoders

6.2.1 Transformers

Since their introduction in [139], Transformers have become a central architecture in modern deep learning, underpinning and accelerating advances across natural language processing, computer vision, and increasingly, time-series modelling. Unlike convolutional or recurrent models, the Transformer relies on self-attention to model global relationships within a sequence, enabling scalable parallel computation and effective long-range context integration.

These capabilities have supported influential models such as BERT [140], GPT [141], Vision Transformers (ViTs) [142], and MAEs [25], ushering in a new era of AI usage and development.

In Figure 6.1, a traditional Transformer architecture, as in [139], is shown, featuring an encoder and decoder. Note the classification head (a linear layer followed by a softmax) which indicates potential usage of this model for sentence classification.

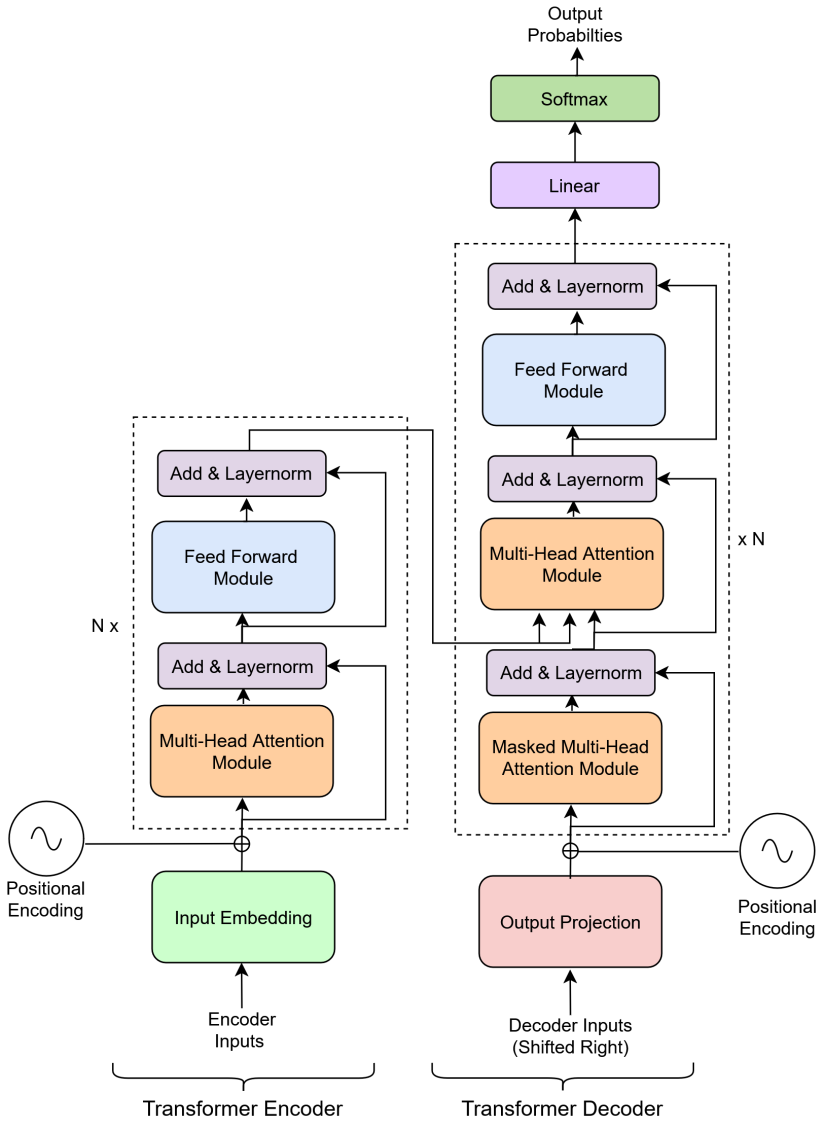


Figure 6.1: Overview of a standard encoder–decoder Transformer architecture as presented in [139]. A classification head attached to the decoder outputs can be used for tasks such as sentence classification.

Importantly, Transformers do not inherently require both an encoder and a decoder. Many well-known architectures, such as BERT [140], are encoder-only,

which is the style adopted in this thesis. Figure 6.2 shows a very simple Transformer encoder. Note there is no classification head; in this case the encoder maps the input sequence to a sequence of contextualised representations, without any task-specific output layer. Masked language modelling could be used in such a setup for pre-training on text, where the model learns to predict missing words in a sequence. Later a classification could be added to the pre-trained encoder for supervised fine-tuning on downstream tasks.

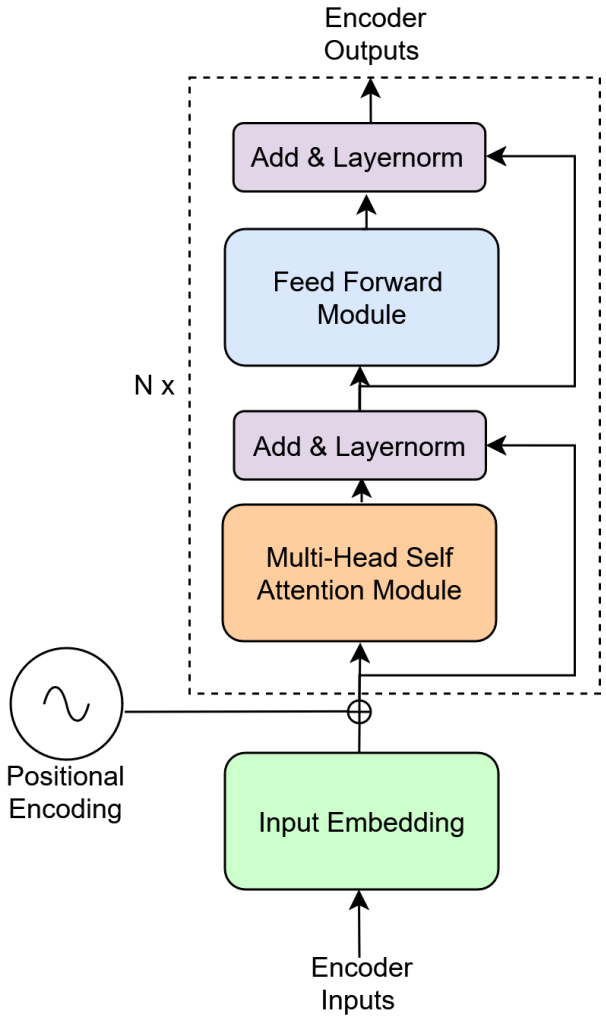


Figure 6.2: Overview of an encoder-only Transformer architecture composed of stacked self-attention and position-wise feed-forward layers.

6.2.2 Self-Attention and Multi-Head Attention

At its core, the fundamental concept introduced with Transformers is the use of self-attention and multi-head attention.

Given an input sequence of embeddings $\mathbf{X} \in \mathbb{R}^{n \times d}$, the model computes learned

projections for queries $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{X}\mathbf{W}_V. \tag{6.1}$$

Self-attention is then computed as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}, \tag{6.2}$$

where d_k is the dimensionality of the key vectors. This allows each token to integrate information from all others, capturing both short- and long-range dependencies.

To learn multiple relational patterns in parallel, Transformers employ Multi-head Attention (MHA):

$$\text{MHA}(\mathbf{X}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\mathbf{W}_O, \tag{6.3}$$

where each head is a separate self-attention operation with its own learned projections. MHSA refers to the common use case where the same sequence provides queries, keys, and values (i.e., $\mathbf{X}$ is used for all three), allowing each token to attend to every other token in the same sequence across multiple heads.

6.2.3 Tokens and Positional Encoding

In Transformer models, inputs are first decomposed into *tokens*—the basic units processed by the architecture (e.g., words or subwords in text, patches in images, or temporal segments in time-series). Each token is mapped to a vector embedding, forming an input sequence $\mathbf{X}$.

Because self-attention is naturally permutation-invariant, positional information must be added explicitly. Transformers therefore incorporate positional encodings—either sinusoidal or learned—which are added to the input embeddings:

$$\mathbf{Z}_0 = \mathbf{X} + \mathbf{P}. \tag{6.4}$$

For modalities beyond text, such as images or time-series, the input can be divided into patches or temporal segments that act as tokens [142], enabling a unified Transformer design across varied data types.

6.2.4 Layer Normalisation and Feed-Forward Networks

Each Transformer block traditionally employs Layer Normalisation [143] for smoother gradients and a position-wise feed-forward network (FFN) to expand feature representation and expressiveness per token:

$$\text{FFN}(\mathbf{x}) = \text{GELU}(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \tag{6.5}$$

where GELU [138] acts as a smooth activation. Residual connections [144] around both the MHA and FFN components improve optimisation stability and convergence. Modern Transformer variants, such as Conformer, increasingly employ alternative activations such as Swish [145].

6.3 Conformer Architectures

The Conformer [24] architecture extends the Transformer by integrating a convolutional module into each layer to improve the modelling of local structure, while retaining the strong global context modelling provided by attention. This hybrid design combines the global receptive field of self-attention with the inductive biases of convolution, resulting in a model with both strong global and local resolution—properties that make it well suited to domains such as speech, biosignal processing, and more broadly, time-series modelling.

A standard Conformer block consists of two macaron-style half FFN layers, MHSA, a *convolutional module*, Layer Normalisation, and residual connections, as shown in Figure 6.3:

$$\mathbf{X}_1 = \mathbf{X} + \tfrac{1}{2}\text{FFN}(\text{LayerNorm}(\mathbf{X})), \tag{6.6}$$

$$\mathbf{X}_2 = \mathbf{X}_1 + \text{MHSA}(\text{LayerNorm}(\mathbf{X}_1)), \tag{6.7}$$

$$\mathbf{X}_3 = \mathbf{X}_2 + \text{ConvModule}(\text{LayerNorm}(\mathbf{X}_2)), \tag{6.8}$$

$$\mathbf{Y} = \text{LayerNorm}(\mathbf{X}_3 + \tfrac{1}{2}\text{FFN}(\text{LayerNorm}(\mathbf{X}_3))). \tag{6.9}$$

Instead of using a single feed-forward layer, the Macaron-style architecture sandwiches the main layer operation between two half feed-forward layers. Each feed-forward output is scaled by a factor of one half, effectively distributing the feed-forward computation across both sides of the layer. By splitting the feed-forward component into two scaled parts, updates are applied symmetrically around the main transformation. This design more closely resembles the

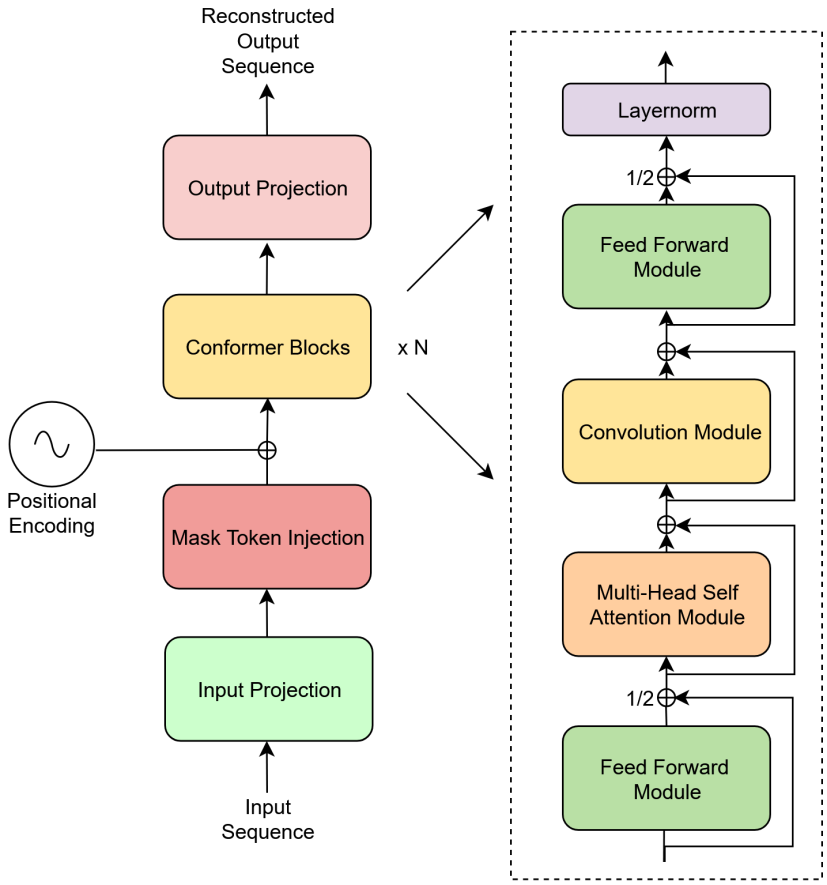


Figure 6.3: Diagram of a standard Conformer block combining macaron-style FFNs, MHSA, and a convolutional module within a residual and LayerNorm framework.

discretisation of an ordinary differential equation (ODE), improving the stability of the representation updates and allowing the model to capture more complex transformations while maintaining stable gradients during training.

6.3.1 Convolutional Module

The convolutional sub-block typically includes a pointwise expansion (using pointwise convolutions for parameter-efficient operation), a Gated Linear Unit (GLU), depthwise convolution, normalisation, activation, and a final projection layer. Specifically, the GLU is defined as:

$$\text{GLU}(\mathbf{X}) = \mathbf{X}_a \odot \sigma(\mathbf{X}_b), \tag{6.10}$$

where $\mathbf{X}_a$ and $\mathbf{X}_b$ are learned affine transforms (splits) of the input. This introduces local feature sensitivity while retaining computational efficiency.

The convolutional module will be covered in further detail in Section 6.5.7, where it is described in the context of the Conformer–MAE and CompMAE models, which follow the Conformer paper closely for this component.

6.4 Masked Autoencoders (MAEs)

MAEs [25] form a self-supervised learning paradigm that reconstructs heavily masked inputs to enable deep representation learning. By randomly removing (masking) a large proportion of tokens and requiring reconstruction from limited context, MAEs compel the encoder to focus on high-level structure rather than low-level detail. In contrast to traditional AEs, which often see the full input and risk learning trivial identity mappings, MAEs are explicitly trained to infer missing content from sparse observations.

Traditionally, a defining characteristic of MAEs is their asymmetrical design, as visible in Figure 6.4:

- the *encoder* processes only the unmasked tokens (e.g., 25% of the original sequence), learning a compact representation of the visible subset;
- the *decoder* is lightweight and receives both the encoded tokens and mask tokens to reconstruct the full sequence.

This separation places most of the modelling capacity in the encoder, encouraging it to learn robust, generalisable features that can be reused for downstream tasks, while keeping the reconstruction-specific decoder relatively shallow and inexpensive.

This architecture greatly reduces encoder computation while still enabling reconstruction over the complete input, making MAEs particularly effective for large-scale self-supervised learning across multiple domains. High mask ratios (e.g., 75% masking) further strengthen the pressure to model global relationships rather than simply copying local details. In Figure 6.4, note how the encoder only views the unmasked portions of the input; these are then reinserted into the full token sequence (with mask tokens) before being passed to the lightweight decoder for reconstruction. This design naturally aligns with sequence and patch-based modalities, including images, time-series, and sensor data, where reconstructing missing regions from limited context is a powerful proxy task for learning deep representations.

While MAEs are traditionally formulated with an encoder–decoder architecture,

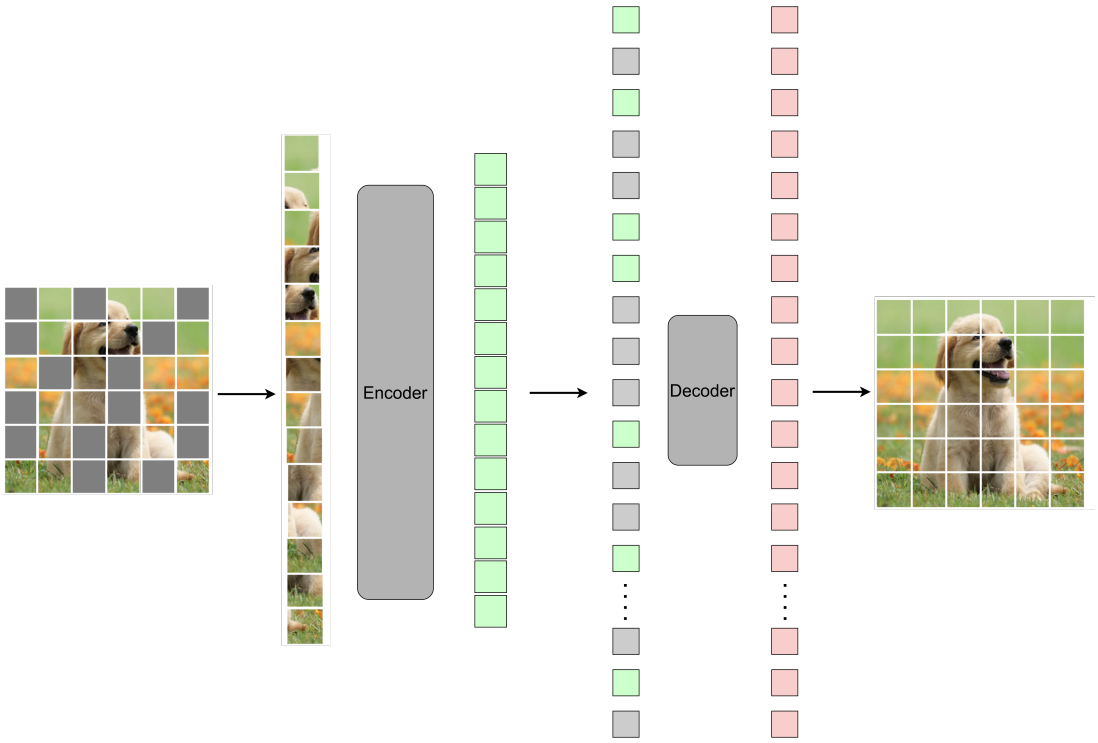


Figure 6.4: Illustration of an MAE with an asymmetric encoder–decoder design.

it is still feasible to operate in an encoder-only setting. In this case, the computational savings are reduced, as the encoder must process both mask tokens and visible tokens, but it remains a viable option and a natural extension of encoder-only models.

6.4.1 Architecture and Objective

An MAE consists of a Transformer encoder applied only to visible tokens and a lightweight decoder that reconstructs the full input:

$$\mathbf{Z}_v = E_\theta(\mathbf{X}_v), \qquad \hat{\mathbf{X}} = D_\phi(\mathbf{Z}_v, \mathcal{M}, \mathbf{P}). \tag{6.11}$$

The loss, in the original MAE paper, is computed only over the masked positions, similar to BERT:

$$\mathcal{L}_{\text{MAE}} = \frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \|\mathbf{X}_i - \hat{\mathbf{X}}_i\|_2^2. \tag{6.12}$$

High masking ratios promote robust, generalisable representations for images [25] as well as time-series [146, 147].

6.5 CompMAE Architecture Design

6.5.1 CompMAE Encoder

COMPMAE adopts an encoder-only architecture, similar in spirit to the Conformer [24] and MAEEG [147] frameworks. Each COMPMAE block closely follows the Conformer structure, consisting of MHSA, convolutional, and feed-forward modules, with two key additions: (i) self-supervised learning via masked reconstruction, and (ii) a dedicated *compressor* module. COMPMAE also uses traditional rather than macaron-style feed-forwards. These elements are detailed in the sections that follow.

This encoder-only formulation differs from that of MAE [25], which employs an asymmetric encoder–decoder structure with a lightweight decoder. In preliminary experiments, adopting a similar decoder for COMPMAE led to the encoder learning weak representations while the decoder dominated the reconstruction process. This imbalance is a known dynamic in MAE-style training: in the original MAE, the decoder is intentionally small to ensure that the encoder captures the semantic content. In COMPMAE, the encoder is comparatively lightweight, making it difficult to design a decoder sufficiently smaller to maintain this balance. As a result, a decoder-based configuration caused the model to over-rely on the decoder, undermining the self-supervised objective.

A consequence of the encoder-only design is that it does not expose a single fixed-size latent vector, which would be useful for downstream fine-tuning if needed, in contrast to traditional AEs. While this has no impact on the reconstruction-based anomaly detection task considered in this thesis, it may limit flexibility for certain downstream applications. If such use cases arise, a projection head or small auxiliary decoder could be attached to produce a task-specific latent representation.

The combination of multi-head self-attention, convolution, and the compression module in a sequential pipeline forms each COMPMAE block, as illustrated in Figure 6.5. Analogous to the Conformer, these components operate at every layer, enabling both global context modelling and local feature extraction, while the compressor progressively refines and contracts the latent representation, enhancing anomalous reaction.

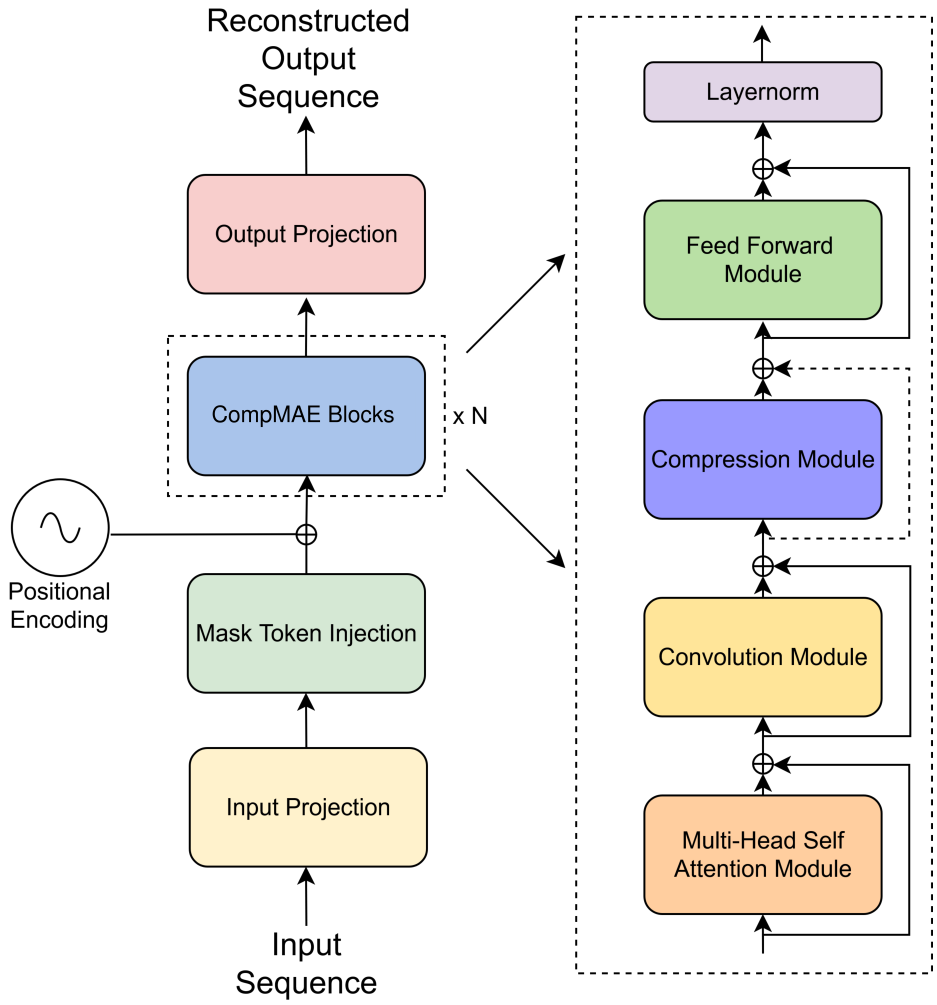


Figure 6.5: Overview of the COMPMAE encoder architecture.

6.5.2 Masking

To enable self-supervised learning using only normal power time-series data (as in Chapter 5) and to encourage the model to learn deep representations of this behaviour, a high masking ratio is employed in both COMPMAE and its CONFORMER–MAE baseline. Training with substantial masking creates a challenging reconstruction task, requiring the model to infer missing information from limited context. Consistent with findings from MAE [25] and the EEG-oriented MAEEG framework [147], a masking ratio of 75% was found to provide the best balance between stability and representational richness (see Section 6.8 for further confirmation).

With 75% masking, the model observes only a small subset of input patches, forcing reliance on global correlations and contextual dependencies across the unmasked regions. This encourages the formation of semantically meaningful

representations of normal operation, aligning with the self-supervised learning principle of reconstruction-based pre-training [140]. Functionally, high masking plays a role similar to compression in classical AEs: the model must reconstruct the signal from an incomplete or compact representation.

If the masking ratio is too low, the model can trivially interpolate missing values from adjacent patches rather than learning deeper structure. Conversely, if the ratio is too high, the available context becomes insufficient and training becomes unstable. Empirically, the 75% regime offered a strong compromise, producing informative latent representations while maintaining reliable optimisation. Although a full MAE-style configuration—where the encoder sees only unmasked tokens and a lightweight decoder reconstructs the full sequence—could further emphasise this behaviour, such a design was not adopted here for the reasons outlined in Section 6.5.1.

Several effective masking strategies have been proposed in prior work. MAE [25] found that masking patches *uniformly at random without replacement* performs best for visual data, while MAEEG [147] reported that masking a *single contiguous segment* is more effective for 1-D EEG signals, also at around 75%. For COMPMAE and CONFORMER–MAE, random patch sampling was found to be more effective than contiguous masking. This difference likely reflects differences in data type between EEG and power time-series data (with less redundancy per window in the latter), as well as the objective of this Chapter: the aim is not to maximise reconstruction fidelity on normal signals, but to heighten anomaly sensitivity. A slight decrease in normal-case reconstruction resolution is acceptable if it increases the error gap between normal and anomalous behaviour.

Finally, unlike MAE, which computes loss only over masked patches, COMPMAE (similarly to MAEEG) compares the reconstructed full sequence to the input. This is essential because evaluation during inference is derived from the full reconstructed sequence, not from masked regions alone.

6.5.3 Input and Reconstruction Targets: Training and Inference

COMPMAE processes one-dimensional power time-series segments, each representing a one-second window sampled at 2048 Hz (as in Chapter 5). Each segment is *patchified* into 64 patches of 32 samples, and each patch is linearly

projected into a 128-dimensional embedding before being passed to the encoder. This configuration offers a practical balance between (i) the number of patches available for global contextual modelling and (ii) the amount of information contained within each patch.

Only normal data are used for training, again following Chapter 5, but here stochastic heavy masking replaces the dropout-based regularisation previously employed. During inference, no masking is applied (zero-mask): the model receives either normal or anomalous sequences and reconstructs them directly. The reconstruction error—computed as MSE between the full input and output—is compared against a threshold to detect anomalies. Thus, the core detection mechanism remains reconstruction-based.

Masking and positional encodings are applied prior to the encoder input, as illustrated in Figure 6.5. Although relative positional encoding (RPE) [148] is commonly effective for series data and is used in Conformer [24] and technically in MAEEG [147], it performed poorly under the zero-masking inference regime considered here. The discrepancy between heavily masked training and unmasked inference led to unstable reconstructions. To mitigate this train–test mismatch, absolute positional encodings (as in MAE [25]) were adopted, yielding stable reconstruction. While absolute encodings can potentially interfere with waveform-like data due to sinusoidal components being added to the signal, this did not pose issues for the DRAGON_PI data due to its inherent frequency characteristics.

Both COMPMAE and CONFORMER–MAE reconstruct the entire input sequence by predicting all patches. In the encoder-only configuration, the final Transformer layer outputs per-token embeddings of dimension 128, which are linearly projected ("patch head") back to 32-sample patches and then *unpatchified* into the full one-second segment.

As mentioned, unlike MAE—which computes reconstruction MSE loss only over masked regions—COMPMAE follows MAEEG in computing MSE loss over the entire reconstructed signal. Restricting the loss to masked regions was found to hinder convergence at high masking ratios, particularly in the absence of a decoder. Computing loss over the full sequence improved training stability and aligns with the intrusion detection objective, which depends on global reconstruction error rather than local differences.

This global formulation ensures consistent supervision across the input space

and promotes robust reconstruction quality under zero-masking inference.

6.5.4 Self-Supervised Learning

In this problem setting, the model is trained exclusively on normal power time-series data, forming a single-class learning scenario as in Chapter 5. A self-supervised reconstruction objective enables the model to learn normal behaviour without requiring labelled anomalies, analogous to the approach used in conventional AEs. This paradigm is particularly relevant in cybersecurity contexts, where anomalous or attack behaviours are inherently diverse, sparse, and difficult to label reliably (see Chapter 4). By learning to reconstruct only normal patterns, the model naturally produces larger reconstruction errors when encountering anomalous inputs, forming the basis for unsupervised anomaly detection.

A further advantage of the self-supervised framework in this Chapter is its effective data efficiency under high masking ratios. While Transformer architectures are often data-hungry, the DRAGON_PI dataset is relatively small. Heavy masking mitigates this by exposing the model to many stochastic partial views of the same underlying signals across training epochs: at each iteration, a different subset of patches is visible, forcing the encoder to infer missing content from limited context. This process acts as a form of implicit data augmentation and regularisation, improving generalisation despite limited dataset size. Although larger-scale data would undoubtedly enhance performance, as with most deep learning approaches, this property demonstrates that the approach remains practical and scalable even under modest data availability.

6.5.5 Zero-Masking Inference

Although training employs a high masking ratio (75%), inference with COMPMAE and CONFORMER–MAE is conducted under a *zero-masking* condition. This introduces a mild but permissible train–test mismatch: the model learns to reconstruct signals containing masked tokens but is evaluated on fully visible inputs. On normal sequences, this can slightly increase reconstruction error relative to the masked training regime, yet in practice it helped improve anomaly sensitivity. Because the model is optimised to infer missing information, the absence of masking at inference makes deviations in anomalous signals more salient—yielding larger reconstruction errors and improved detectability. The

application of masking at inference, even at small amounts (10%), reduced the reactivity of COMPMAE considerably.

This design choice is further justified because the reconstruction loss is computed over the entire input sequence rather than only the masked regions. In contrast, objectives that restrict loss to masked positions—as in BERT [140] or MAE [25]—provide no direct supervision for unmasked tokens, which can hinder encoder-only (decoder-free) configurations that rely on a consistent full input–output mapping. By optimising the complete reconstruction, COMPMAE achieves stable learning and robust zero-masking inference without additional fine-tuning.

Finally, due to the nature of the data, the entire input must be available for the model at inference. Any masking could hide potential anomalies and cause false negatives in attack detection.

Overall, zero-masking inference accentuates the model's reactive behaviour: it reconstructs normal signals faithfully while amplifying deviations induced by anomalous activity. This enhances both interpretability and the diagnostic value of reconstruction error as an indicator of intrusion.

6.5.6 Multi-Headed Self-Attention Module

In COMPMAE, MHSA is applied at each layer to learn the global context of the unmasked patches. As in the Conformer architecture, pre-norm residual units are used [24]. Each MHSA module consists of a LayerNorm operation (to normalise over the feature dimension and stabilise training in deep models), MHSA (to capture the global context of the sequence), and dropout for regularisation. The MHSA strategy applied can be seen in Figure 6.6.

In a pre-norm residual unit, LayerNorm is applied to the input before the MHSA block, while the residual connection bypasses both operations. This design preserves an identity path through the residual connection, which results in much more stable gradient flow compared to post-norm architectures. As a consequence, pre-norm Transformers typically train faster, are easier to optimise at greater depth, and exhibit improved stability in self-supervised learning settings.

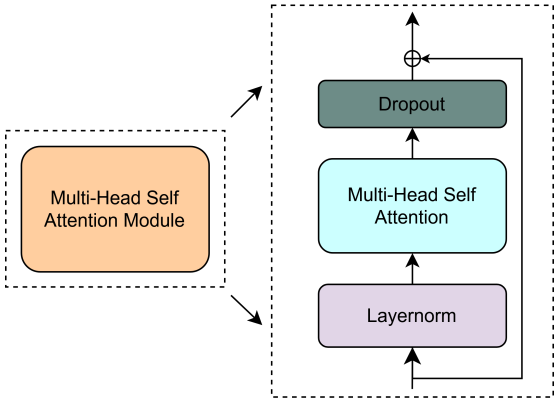


Figure 6.6: Schematic of MHSA within a COMPMAE block. Each head models contextual dependencies across masked and unmasked tokens before convolutional refinement.

6.5.7 Convolution Module

As in the Conformer architecture [24], a convolutional module follows the attention module to capture local contextual dependencies within the sequence, implemented within a pre-normalised residual structure, as in Figure 6.7. The module begins with a *pointwise* (1×1) convolution that expands the feature dimension by a factor of two, applied independently to each token. A *Gated Linear Unit* (GLU) [149] then provides a gating mechanism that selectively filters these expanded features. After gating, the channel dimension is projected back to the original encoder size.

The gated features are subsequently passed through a single one-dimensional *depthwise convolution* [150]. Depthwise convolutions operate separately on each channel while sliding across the temporal dimension, allowing the model to capture short-range correlations in local power fluctuations. This complements the global attention of the MHSA block by providing fine-grained temporal sensitivity, which is particularly relevant to IoT power traces and other time-series. Immediately after the depthwise convolution, batch normalisation (BatchNorm) [100] is applied to stabilise feature distributions and support training stability.

Following normalisation, a *Swish* (SiLU) activation [145] introduces nonlinearity while maintaining smooth gradient flow. Swish reduces the likelihood of dead activations (in comparison to ReLU) and improves optimisation stability, benefits that are especially useful in lightweight convolutional architectures. Consistent with observations in Conformer [24], its use here enhances convergence and overall representational flexibility.

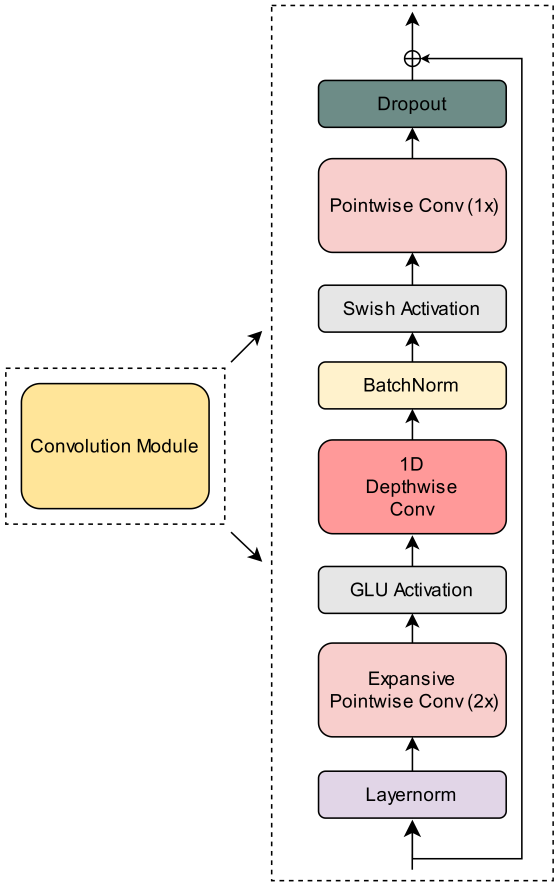


Figure 6.7: Conformer-style convolution module used in both CONFORMER–MAE and COMPMAE.

Finally, a second pointwise (1×1) convolution mixes the channel-wise information produced by the depthwise convolution. If needed, it can also restore the representation to the original encoder dimensionality. The overall sequence of operations—pointwise expansion, GLU gating, depthwise convolution, BatchNorm, Swish activation, and a final pointwise projection—refines local temporal structure while maintaining computational efficiency within a residual block.

6.5.8 Compression Module

Building on principles from the convolutional module and traditional AEs, and bearing similarity to some concepts from Squeezeformer [151], this Chapter introduces a novel compression module, illustrated in Figure 6.8. Whereas the standard Conformer-style convolutional module temporarily expands the token representation for feature enrichment before gating and projecting back, the compression module intentionally reduces the encoder dimensionality, en-

forcing an information bottleneck that promotes compact and behaviourally meaningful representations (similar to traditional AEs). The model is therefore encouraged to reconstruct normal signals accurately while naturally producing higher reconstruction error for anomalous inputs, consistent with classical AE-based anomaly detection.

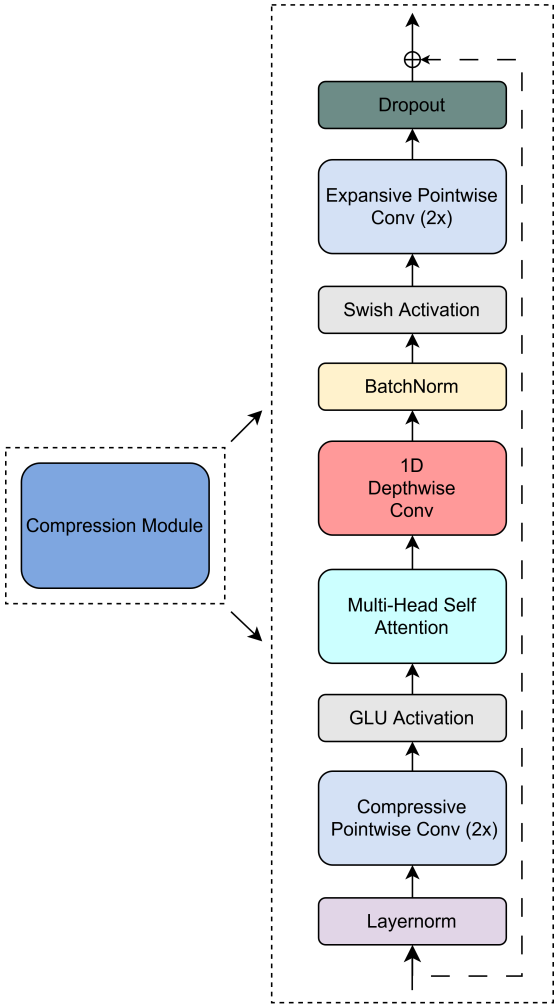


Figure 6.8: Compression module used in CompMAE. Note the similarities to the convolutional module, albeit inverted in key areas.

As mentioned, the compression module inverts part of the Conformer block: a pointwise (1×1) convolution first reduces the feature dimension from $128 \rightarrow 64$—an initial $2\times$ compression. Because pointwise convolutions act independently per token while mixing information across channels, they provide a parameter-efficient and locality-preserving alternative to dense bottleneck layers.

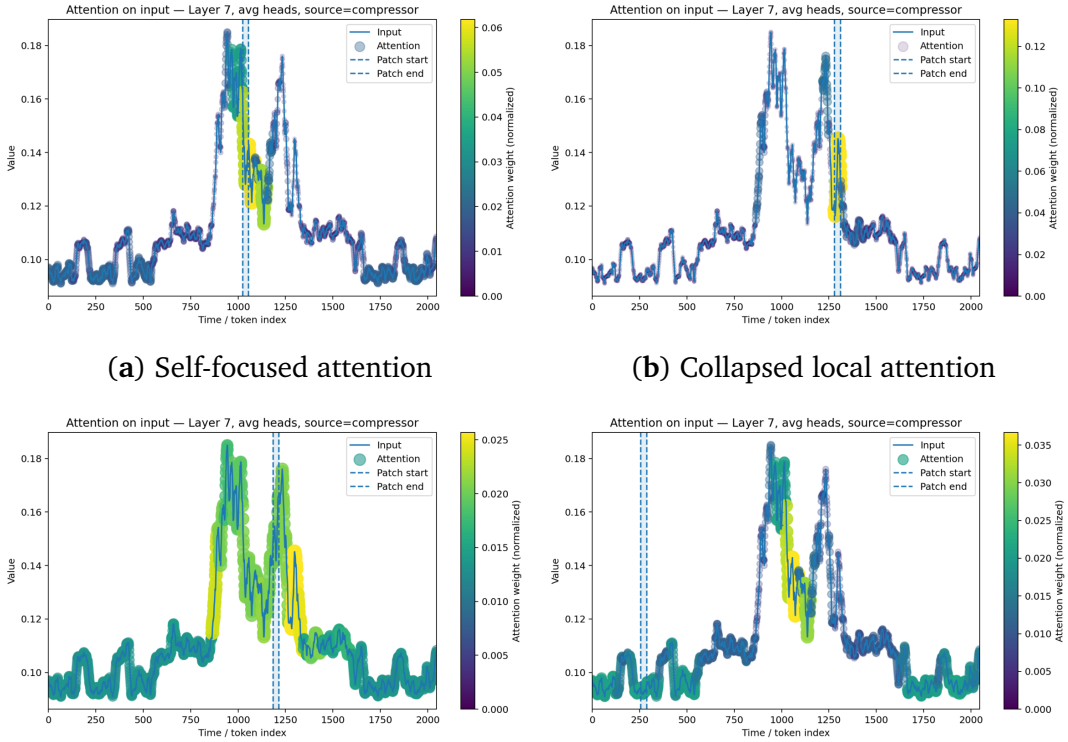
A Gated Linear Unit (GLU) [149] is then applied to the compressed representation. In the original Conformer block, the GLU operates in an expanded feature

space and primarily controls feature flow. Here, the GLU further compresses the representation by splitting the 64-dimensional vector into two 32-dimensional partitions and retaining only the gated output. The GLU therefore acts as a compression-aware selector, passing only features most predictive of normal operation—further reducing the feature space beyond the initial $2\times$ for a total of $4\times$ compression. This compression-aware gating provided by the combination of the pointwise convolution and GLU helps increase sensitivity to deviations: anomalous inputs disrupt the compressed representation more strongly than they would in a higher-dimensional latent space.

To restore expressive capacity within this reduced latent space, and to increase the model's reaction to anomalies by attending over the compressed token representations, a secondary MHSA layer is applied after the GLU. Because this attention is trained solely on normal data, the resulting compressed embeddings encode consistent behavioural structure. Under anomalous conditions, the attention patterns become atypical (e.g., overly concentrated self-attention, collapse to strictly local attention, diffuse attention across all tokens, or disproportionate influence from anomalous tokens), with some examples illustrated in Figure 6.9. These disruptions can propagate through the network and amplify reconstruction error, enhancing anomaly detectability and reactivity. Importantly, this secondary attention layer remains naturally computationally efficient due to residing within the reduced feature dimension.

Following the MHSA, a one-dimensional depthwise convolution layer [150] models short-range temporal dependencies within the compressed representation and restores some local resolution that may have been lost during compression. As in the Conformer block [24], this is followed by batch normalisation [100] and a Swish activation [145]. A final pointwise convolution then projects the representation back to the original encoder dimensionality ($32 \rightarrow 128$), completing the module. The entire compression pathway operates within a pre-norm residual structure to aid convergence and stability.

A further key design element is the application of dropout along the residual branch (the dashed skip connection around the compressor in Figure 6.8) of the compressor. Without regularisation, the residual pathway in a reconstruction-style objective can dominate the block's behaviour. This is especially problematic in the case of the compressor module, which by its very nature makes convergence and reconstruction harder; the model may instead choose the path of least resistance around it.



(**a**) Self-focused attention

(**b**) Collapsed local attention

(**c**) Diffuse/meaningless attention

(**d**) Disproportionate influence of anomalous tokens

Figure 6.9: Examples of disrupted attention patterns under anomalous inputs within the compressed latent space (attention for a specific token).

In a residual formulation of the form

$$y = x + f(x),$$

the model can minimise reconstruction loss by letting $f(x) \approx 0$, allowing the uncompressed shortcut x to reproduce the target directly. This causes the compressor to be effectively ignored during training, preventing the network from learning a meaningful low-dimensional representation of normal behaviour. To mitigate this failure mode, a small amount of dropout is applied on the residual branch, so that during training the block effectively behaves as

$$y = \text{Dropout}(x) + f(x),$$

occasionally weakening the shortcut and forcing the model to rely on the compressed transformation $f(x)$. As a result, the compressor becomes behaviourally relevant, and anomalies that disrupt $f(x)$ propagate more strongly to the final reconstruction, improving downstream detectability.

6.5.9 Feed-Forward Module

Rather than adopting the Macaron-style dual feed-forward structure used in Conformer [24], COMPMAE and CONFORMER–MAE employ a more conventional Transformer feed-forward design, similar to that used in MAE [25] and [139]. As shown in Figure 6.10, the feed-forward module consists of a linear expansion of the token embedding dimension ($4\times$), followed by a Swish (SiLU) activation [145], dropout, and a final linear projection that compresses the representation back to the original encoder size. Empirically, substituting the Macaron configuration resulted in noticeably diminished anomaly reactivity, suggesting that the additional smoothing introduced by the dual feed-forward structure reduced the contrast between normal and anomalous reconstructions.

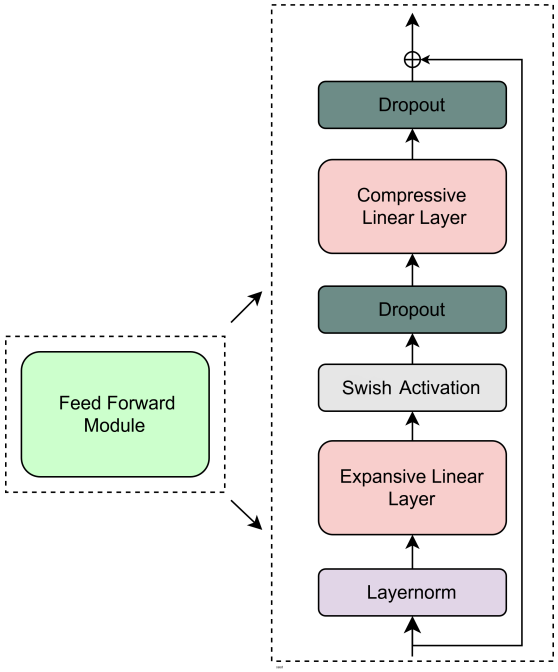


Figure 6.10: Feed-forward module used in COMPMAE, comprising a linear expansion, Swish (SiLU) activation, dropout, and linear compression.

This formulation preserves simplicity and computational efficiency while providing sufficient expressive capacity for the self-supervised reconstruction task. As with the other modules in the architecture, the feed-forward layer operates within a pre-normalised residual block. Dropout is applied both internally and externally after the residual connection as in Figure 6.10. This combination supports stable optimisation, prevents overfitting, and maintains smooth gradient flow throughout the encoder stack.

6.6 Training

6.6.1 Dragon_Pi Dataset

All experiments are conducted on the DRAGON_PI dataset, consistent with Chapter 5. As this Chapter extends the work introduced previously, a dedicated validation set is introduced here to support more comprehensive hyperparameter exploration and performance analysis. Unlike Chapter 5, where evaluation was primarily focused on aggregate anomaly detection, the present study requires finer control over model tuning and generalisation assessment.

The validation set is carefully curated from segments of individual attack files to ensure a balanced composition of normal and abnormal data, while excluding CTF recordings. The CTF sessions are reserved exclusively for a special test set due to their higher realism and operational relevance. Approximately the first 20% of each attack file is allocated to validation, with care taken to avoid splitting anomalies across training and evaluation subsets.

Dataset pre-processing follows the same protocol as in Chapter 5. Power signals are downsampled to 2 kHz, and a moving-average filter (MAF) is applied to suppress high-frequency measurement noise from the power analyser. Each sequence is then segmented into overlapping windows of 2048 samples (1 s) with an 87.5% overlap, consistent with prior work in Chapter 5. Although this high overlap could introduce redundancy between windows, the heavy stochastic masking used during self-supervised training mitigates its effect by ensuring that each batch presents a distinct, randomly masked view of the data.

6.6.2 Automatic Mixed Precision

In another improvement on the work in Chapter 5, training of the models makes extensive use of *Automatic Mixed Precision (AMP)* [152]. AMP accelerates training and reduces GPU memory usage by combining 16-bit (FP16) and 32-bit (FP32) floating-point arithmetic. Computationally stable operations such as matrix multiplications, convolutions, and attention are executed in FP16, while numerically sensitive components—such as loss computation, normalisation, and gradient updates—are retained in FP32 to preserve accuracy.

To prevent gradient underflow in FP16, AMP employs dynamic loss scaling, automatically adjusting scaling factors during training. This allows stable gradient propagation across mixed-precision operations without manual tuning. In

practice, AMP yielded close to a $2\times$ reduction in training time and significantly improved GPU utilisation, with negligible loss in final reconstruction fidelity. It also enabled the use of larger batch sizes within the same memory footprint, further stabilising convergence and reducing training variance across epochs.

While mixed precision was utilised in this work, more aggressive configurations, such as full FP16 training, could offer additional speed-ups at the potential cost of numerical stability and representational precision.

6.6.3 CONFORMER–MAE vs. COMPMAE Training

To establish a baseline for evaluating the effect of the compression module, a self-supervised Conformer variant was first implemented following the MAE-style reconstruction paradigm (CONFORMER–MAE). This provided a benchmark for assessing a masked Conformer's inherent anomaly detection capability and for quantifying the improvement achieved by introducing the compression module in COMPMAE.

Wherever applicable (see Table 6.1), hyperparameters were kept consistent between the CONFORMER–MAE baseline and COMPMAE to ensure a fair comparison and simplify experimental design. Both models were trained with a fixed learning rate of 5×10^{-5}, zero weight decay, and a batch size of 128.

The architectural hyperparameters in Table 6.1 were chosen to preserve lightweight operation suitable for edge deployment on IoT hardware, guided in part by the Conformer configuration [24]. Despite incorporating an additional compression module in every layer, COMPMAE adds only ~ 148k parameters compared with CONFORMER–MAE. Both models were trained for 1600 epochs, corresponding to roughly 22 hours of wall-clock time with AMP acceleration. After training, the optimal checkpoint for each model was selected post hoc based on validation AUC.

Conceptually, COMPMAE can be regarded as a masked Conformer augmented with a compression module in series. Consequently, the two architectures share core operational principles, including identical depth and attention configurations. Both employ 8 encoder layers and 8 attention heads, providing sufficient representational depth to model local and global dependencies within power traces. To constrain complexity, the encoder dimension is fixed at 128—substantially smaller than in standard Conformer or Transformer configurations—reducing parameter count and overfitting risk while retaining

Table 6.1: Conformer vs. CompMAE hyperparameters.

Model	**Conformer**	**CompMAE**
Number of Parameters (M)	2.02	2.15
Encoder Layers	8	8
Encoder Dimension	128	128
Attention Heads	8	8
Conformer Kernel Size	17	17
Compression Ratio	–	4
Compressor Heads	–	8
Compressor Kernel Size	–	5

meaningful expressivity.

The convolutional module's kernel size was set to 17. Although the original Conformer paper [24] reported optimal values between 32 and 64 for large-scale speech tasks, a proportionally smaller kernel was selected here to match the reduced encoder width and the shorter contextual dependencies characteristic of power-domain signals.

Importantly, although each compression block introduces additional components—most notably an extra attention mechanism—the parameter increase remains modest at approximately ~ 148k. This efficiency arises because all operations within the compressor—including attention and depthwise convolution—are performed at the reduced feature width ($D = 32$), corresponding to a $4\times$ compression ratio, rather than the baseline $D = 128$. As the computational cost of depthwise convolution scales approximately with $D \times k$, this reduction yields significant savings in both parameter count and inference latency. The 1×1 pointwise projections used for compression are parameter-equivalent to per-token linear layers; the principal efficiency gains come from performing subsequent computation in the narrower latent space rather than from the projection itself.

6.7 Results on Dragon_Pi

Like Chapter 5, while MSE is used during training, AUC is used as the overall performance metric for the models on the Dragon_Pi dataset. The MSE nevertheless underpins the anomaly detection process by providing the individ-

Table 6.2: Performance comparison between the Conformer (CONFORMER–MAE) and CompMAE (COMPMAE) models across various subsets of the DRAGON_PI dataset. Metrics are AUC and detection delay in seconds.

	Conformer		CompMAE	
Data	AUC	Delay(s)	AUC	Delay(s)
Full Dragon_Pi	0.784	0.472	**0.821**	**0.421**
Dragon Only	0.763	0.462	**0.786**	**0.302**
Pi Only	0.811	0.507	**0.865**	**0.407**
Portscan Only	0.727	0.102	**0.820**	**0.100**
DoS Only	0.762	**0.767**	**0.936**	0.933
Bruteforce Only	0.773	0.548	**0.796**	**0.337**
CTF Only	0.819	**0.498**	**0.897**	0.585

ual anomaly scores from which the AUC is computed across different decision thresholds.

Across the DRAGON_PI dataset, COMPMAE consistently outperforms its CONFORMER–MAE baseline, as summarised in Table 6.2. Initially, models are compared using the AUC over the entire test set (including CTF data) and the average detection delay (in seconds), providing complementary views of classification accuracy and temporal responsiveness.

Building upon the evaluation methodology of Chapter 5, performance is reported not only across the complete DRAGON_PI test set, but also by testbed, by attack type, and separately for the CTF scenarios. This multi-granular evaluation offers a clearer perspective on model generalisation and robustness under varied operational conditions. Treating the CTF sequences as a held-out test set—never used during validation—provides insight into real-world deployment behaviour, as these traces combine multiple concurrent attacks in realistic execution contexts.

In all detection scenarios (AUC) in Table 6.2, COMPMAE outperforms its CONFORMER–MAE baseline. Across the full DRAGON_PI dataset, the improvement in AUC appears modest (0.821 vs. 0.784). This trend is echoed in several subsets, including the Dragon-only portion (0.786 vs. 0.763) and the Bruteforce-only subset (0.796 vs. 0.773). However, the performance gap widens substantially in others: for Pi-only data, COMPMAE achieves 0.865 vs. 0.811; for Portscan-only, 0.820 vs. 0.727; for DoS-only, 0.936 vs. 0.762; and notably for the CTF subset, 0.897 vs. 0.819.

The limited separation in some subsets can be attributed to label noise, a recognised limitation of the Dragon_Pi dataset. This is particularly evident in the Bruteforce scenarios, where normal windows interleaved between login attempts were labelled as anomalous because the device was still technically under attack (even if the power signature looks completely normal). Since the majority of Bruteforce examples were obtained from the DragonBoard, this likely contributes to the reduced gap observed in that subset and in the overall Dragon-only results.

While seeming numerically small, qualitative inspection of reconstruction traces (next section) shows that CompMAE exhibits clearer anomaly localisation and far greater reaction to anomalies. The CTF subset—being cleaner and more consistently annotated—strongly accentuates this effect: CompMAE reaches an AUC of 0.897 compared to 0.819 for Conformer–MAE, a significant improvement of approximately 7.8%—especially as the CTF data constitutes a more realistic test set.

Temporal detection delay results further highlight these differences. Over the full dataset, CompMAE detects attacks on average 51 ms faster than Conformer–MAE (0.421 s vs. 0.472 s), corresponding to a reduction of approximately 10.8%. Both models respond rapidly overall—typically within 500 ms of an anomalous event—although part of the residual delay arises from annotation imprecision (e.g., windows labelled anomalous despite containing only a small anomalous segment) and from the inherent temporal smoothing introduced by overlapping windows.

Overall, despite only moderate gains in some noisy subsets, the more controlled and realistic CTF evaluations reveal the true strength of the compressor: it increases anomaly sensitivity, improves reconstruction contrast, and accelerates detection in edge-deployable conditions.

Another methodological distinction from Chapter 5 is the absence of post-processing. Here, reconstruction errors are evaluated in their raw form to isolate the contribution of architectural improvements. In Chapter 5, a moving-average smoothing filter was introduced; this processing step was omitted in the current analysis to reflect model performance directly and to preserve real-time interpretability.

The AUC computation methodology has also been refined. In Chapter 5, AUC was computed per test file and then averaged, which treats all files equally but

can underrepresent longer sequences. In this Chapter, reconstructions from all test files are pooled and a single global AUC is computed over the aggregated distribution. This provides a stricter, distribution-level estimate of discrimination capability. Accordingly, the evaluation procedure adopted in this Chapter constitutes an improvement over that used previously, ensuring that reported gains more closely reflect underlying performance.

6.7.1 Masked Reconstruction Performance

To assess the quality of the deep representations learned for normal data by both the Conformer–MAE baseline and CompMAE, inference on unseen normal data was conducted under varying masking ratios, following the MAE protocol. Although this does not directly measure anomaly-detection performance, it provides valuable insight into the strength and robustness of the learned representations of normal behaviour. Figure 6.12 illustrates reconstruction quality on the same unseen test segment of normal Bluetooth activity (the original input is shown in Figure 6.11) across a range of masking ratios applied only at inference, with the training masking ratio fixed at 75%.

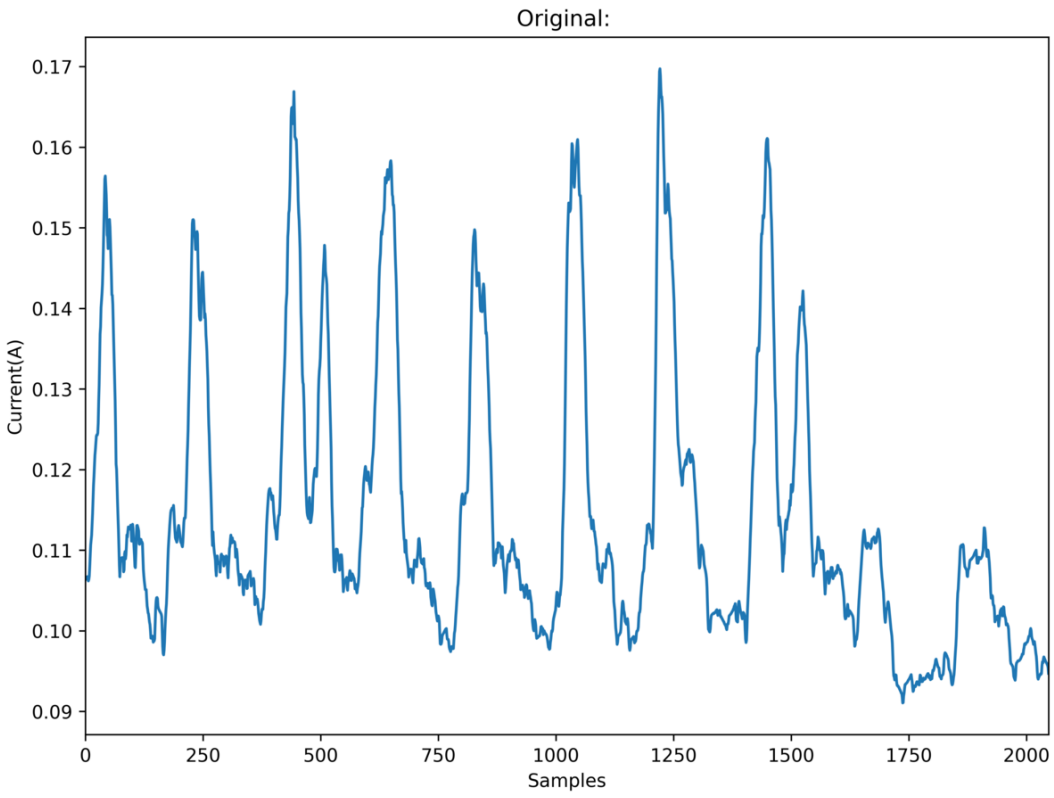


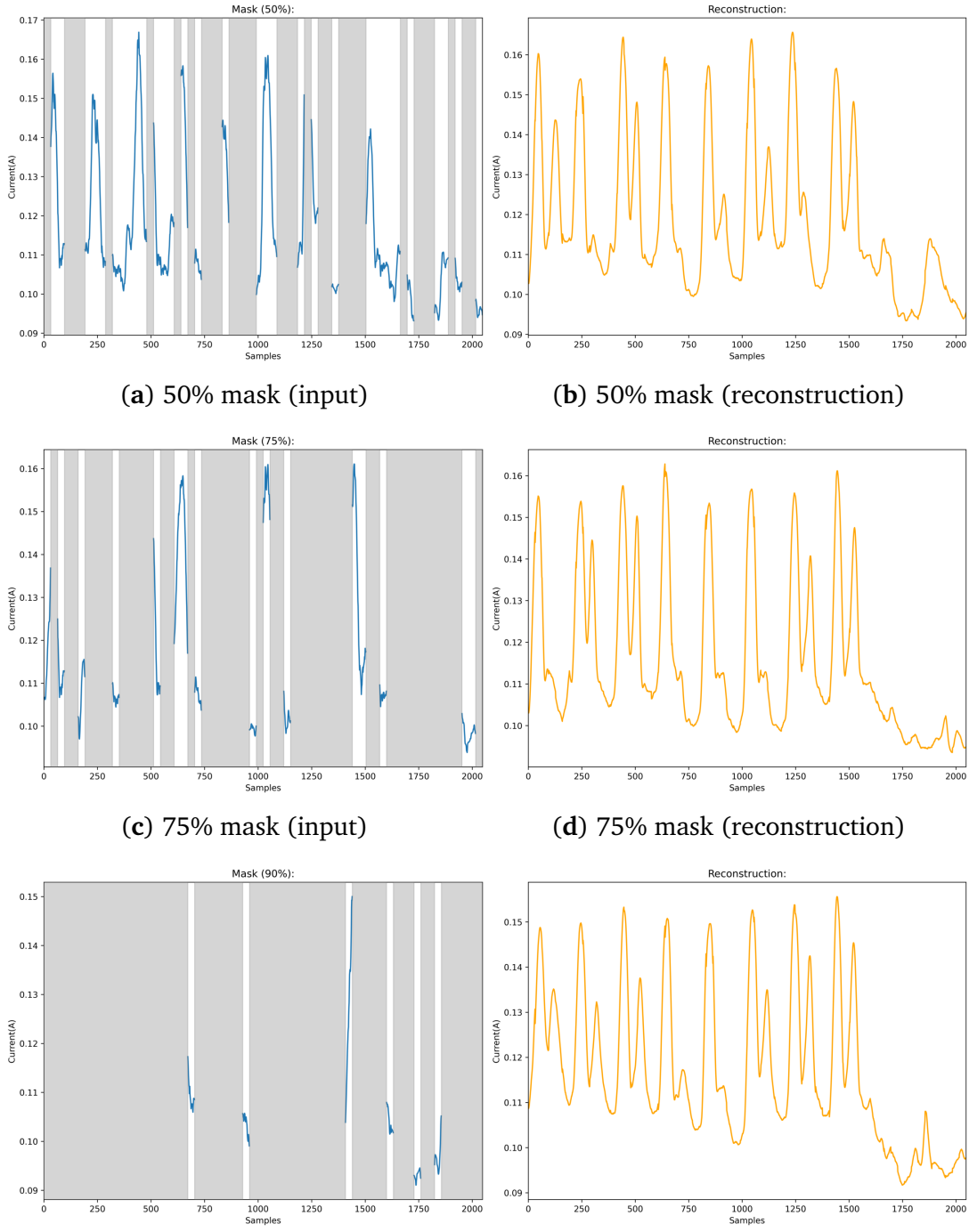
Figure 6.11: Original normal Bluetooth segment used for masked-reconstruction evaluation.

For models trained under the MAE paradigm [25], the objective is not to reproduce the input exactly but to generate a *semantically plausible* reconstruction that preserves salient structure and temporal dynamics. As shown in Figure 6.12, COMPMAE maintains stable, coherent reconstructions even under extreme masking (up to 90%), with reconstruction fidelity improving progressively as the inference masking ratio decreases toward the training value (75%) and below—as expected. This behaviour indicates that the model captures sufficiently rich contextual representations to plausibly infer missing information from sparse input—one of the core signatures of effective self-supervised learning [25, 140].

For example, in Figure 6.12d, COMPMAE omits a small spike near the end of the signal that was fully masked but fills the interval with a plausible continuation consistent with neighbouring visible context. Likewise, a modest temporal shift or amplitude variation—provided the overall periodic structure of the Bluetooth pattern is preserved—remains acceptable, as the objective is to reconstruct behaviourally valid sequences rather than achieve pixel-level fidelity. Even with minimal visible context, as in Figure 6.12e, the model continues to produce structurally coherent reconstructions that capture the rhythm and morphology of normal operation.

This robustness arises partly from the regular, periodic nature of normal DRAGON_PI power traces, but it also reflects the representational benefits of high-ratio masking. By forcing the model to infer large missing portions from sparse context, heavy masking encourages abstraction over local detail and promotes learning of high-level behavioural structure [25, 147]. The resulting representations generalise more effectively to unseen or incomplete normal inputs—an essential property for real-world IoT anomaly detection, where noise, interference, and partial observability are common.

Under very high masking levels such as 90%, the model will not necessarily produce identical or strictly semantically aligned reconstructions on every run. However, the fact that it can generate plausible and structurally consistent outputs at all despite minimal visible context demonstrates the strength and robustness of the learned representations.



(**a**) 50% mask (input)

(**b**) 50% mask (reconstruction)

(**c**) 75% mask (input)

(**d**) 75% mask (reconstruction)

(**e**) 90% mask (input)

(**f**) 90% mask (reconstruction)

Figure 6.12: Reconstruction of the same normal Bluetooth segment under varying inference masking ratios (rows: 50%, 75%, 90%). Left: masked inputs. Right: corresponding reconstructions.

6.7.2 Reconstruction Performance

An important aspect of this investigation is not merely the aggregate performance metrics, but also how the models *behave* when exposed to normal and anomalous data, as reflected through their reconstructions (as in Chapter 5). Examining these reconstructions provides a more interpretable view of model performance, especially as a key goal is an enhanced reaction to anomalies. Importantly, no masking is applied in these experiments (consistent with the main performance evaluation), as the entire sequence may contain informative content. This section therefore illustrates each model's response to the full input.

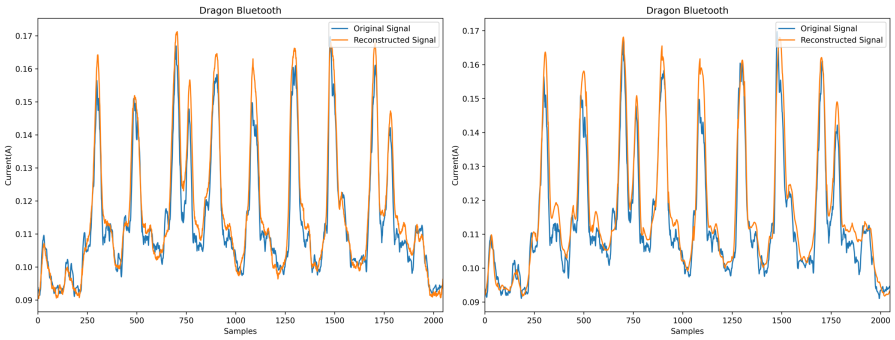
Before considering anomalous behaviour, reconstruction performance on normal data is evaluated. Accurate reconstruction of normal patterns is essential for minimising false positives and ensuring a clear separation between normal and anomalous reconstruction errors. A well-trained self-supervised model should capture the underlying distribution of normal behaviour robustly across diverse operational conditions, maintaining low reconstruction error while generalising across varying device states.

Figure 6.13 compares reconstructions from the CONFORMER–MAE baseline and COMPMAE on normal data only. Both models exhibit strong performance in reproducing normal power behaviour, with CONFORMER–MAE showing a slight advantage in raw reconstruction fidelity—an expected outcome, given that COMPMAE must reconstruct through an additional compressive bottleneck and therefore incurs a small degradation. This effect is more pronounced in Figures 6.13d and 6.13h, where a modest increase in reconstruction error is observed for Raspberry Pi Bluetooth and background behaviour, respectively. The increased variance of normal patterns within the Pi data likely contributes to this difference.

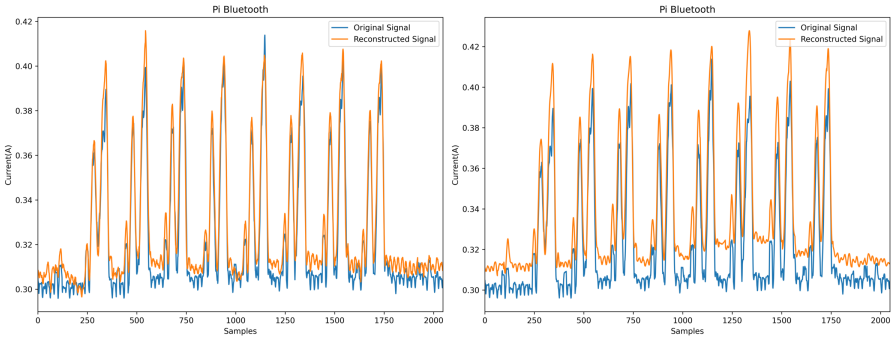
Despite this minor reduction in fidelity, COMPMAE achieves consistently low reconstruction error across all normal scenarios, demonstrating its capacity to model normal device behaviour effectively even under aggressive compression. This supports the architectural premise that compression does not impair normal reconstruction quality in practice, but instead enhances the model's ability to distinguish anomalous deviations—preserving essential normal structure while amplifying reconstruction contrast for irregular inputs.

Consulting Figure 6.14, the reconstruction behaviour of the CONFORMER–MAE baseline and COMPMAE is compared across several anomalous events from the

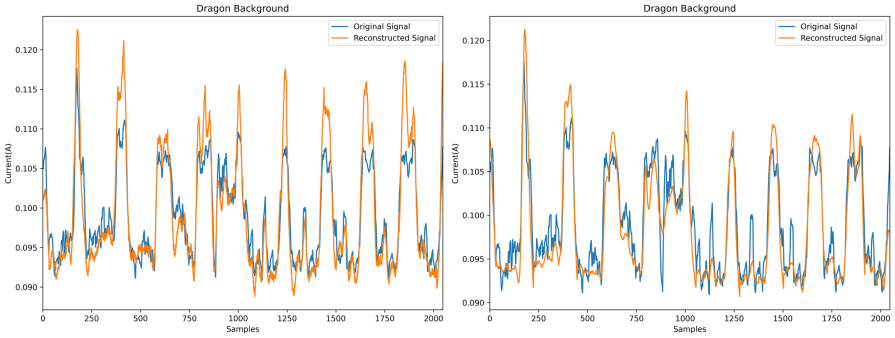
CONFORMER–MAE **COMPMAE**



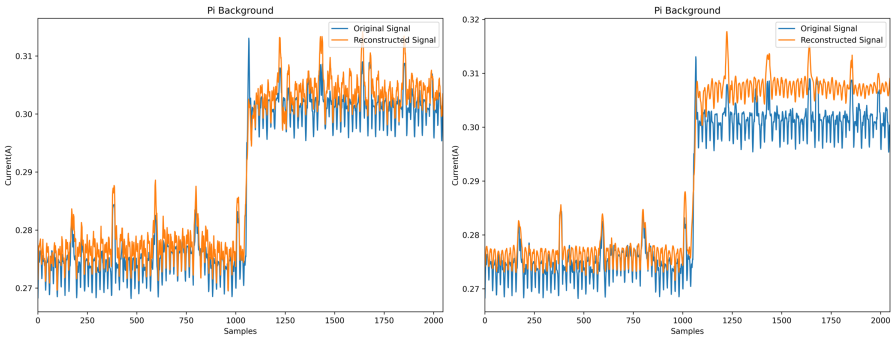
(**a**) Bluetooth (DragonBoard) (**b**) Bluetooth (DragonBoard)



(**c**) Bluetooth (Raspberry Pi) (**d**) Bluetooth (Raspberry Pi)



(**e**) Background (DragonBoard) (**f**) Background (DragonBoard)



(**g**) Background (Raspberry Pi) (**h**) Background (Raspberry Pi)

Figure 6.13: Reconstruction of *normal* power behaviour across devices and activities. Left column: CONFORMER–MAE baseline. Right column: COMPMAE.

Dragon subset of the Dragon_Pi dataset. The Dragon signals generally exhibit lower overall amplitude than those of the Pi subset, producing anomalies whose magnitudes often fall below or approximate the normal Bluetooth activity levels observed elsewhere.

A clear pattern emerges in Figure 6.14. For the Nmap T3 anomaly in Figure 6.14a, the Conformer–MAE baseline reacts only minimally, whereas CompMAE displays a pronounced, amplified response to the same event (Figure 6.14b). This demonstrates the additional reactivity introduced by the compressor in CompMAE. Notably, the anomaly here is neither high in amplitude nor long in duration—its magnitude is comparable to the normal Bluetooth oscillations—yet CompMAE still reacts to it distinctly. This underscores the model's heightened sensitivity to deviations that superficially resemble normal patterns but diverge subtly in temporal or structural context.

A similar trend is evident for the Hydra T1 SSH brute-force anomaly (Figures 6.14c and 6.14d). Again, the anomaly's amplitude is modest, close to that of normal Bluetooth activity. In this case, the Conformer–MAE baseline exhibits a slight but inconsistent reaction, indicating a degree of inherent anomaly reaction. The compressor module in CompMAE substantially amplifies this response, producing a strong reconstruction divergence at the point of attack. While the baseline occasionally reacts in a similar fashion, Figure 6.14 shows that such behaviour is unreliable compared with the consistent reactivity achieved by CompMAE. In this sense, the compressor can be viewed as enhancing reactivity that is already latent within the underlying Conformer–MAE architecture.

With the Hydra T16 SSH brute-force attack (Figures 6.14e–6.14f), the anomalous signal reaches approximately 0.38 A—roughly twice the DragonBoard's typical Bluetooth current (cf. Figure 6.13a). Despite this, the Conformer–MAE baseline reconstructs this high-amplitude sequence with surprising fidelity and minimal reaction. Such accurate reconstruction is desirable for normal inputs but completely undesirable for anomalies, as it yields a low reconstruction error that can obscure detection. This illustrates a fundamental limitation of unconstrained reconstruction models: if the anomaly remains within the amplitude and temporal statistics of the training distribution, the model may not perceive it as abnormal. By contrast, CompMAE makes little attempt to reproduce the anomalous window faithfully, producing a completely exaggerated reconstruction and correspondingly large error—precisely the desired outcome

for reliable anomaly detection.

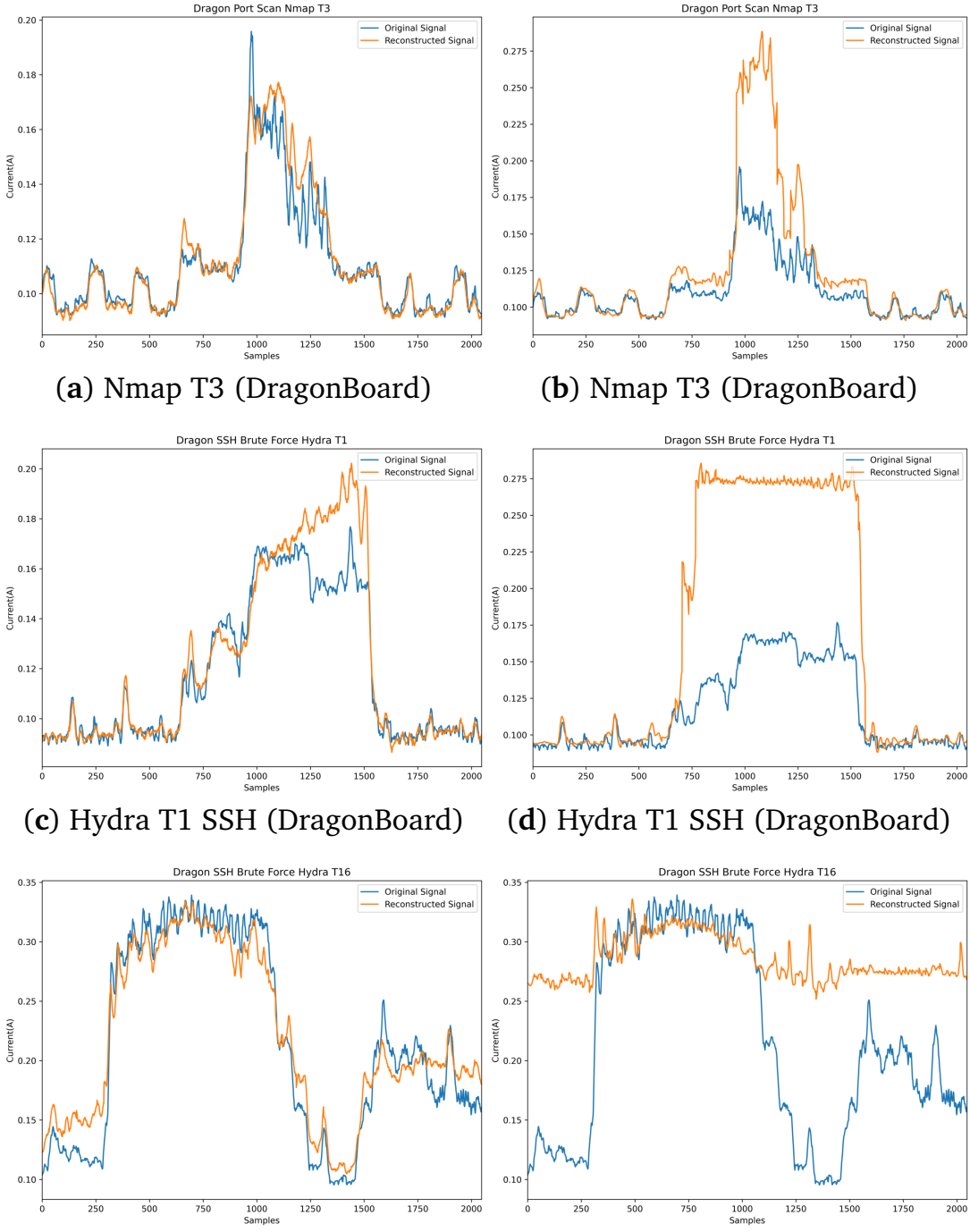
Focusing on the Dragon subset initially as a whole, the benefit of the compressor in COMPMAE is evident: it produces a consistently stronger reaction to anomalies across a spectrum of scales, from subtle, short-duration deviations (Figure 6.14b), through low-amplitude extended events (Figure 6.14d), to high-amplitude, longer-duration anomalies (Figure 6.14f). It is important to reiterate that all models in this Chapter are trained jointly on *both* Dragon and Pi normal data, as in Chapter 5—demonstrating that a single unified model can generalise effectively across devices while maintaining distinct anomaly sensitivity for each.

Focusing on the Pi subset in Figure 6.15, a similar pattern emerges, albeit with a more reactive CONFORMER–MAE baseline than in the Dragon case. The Pi signals generally exhibit higher amplitudes than those of the DragonBoard (cf. the inputs in Figures 6.13a and 6.13c). This is also true for many Pi-side anomalies, whose amplitudes and temporal signatures frequently extend well beyond the learned normal operating range, prompting both CONFORMER–MAE and COMPMAE to react strongly.

In Figures 6.15a and 6.15b, a relatively low-amplitude Pi anomaly is shown: an Nmap T3 port-scan segment whose full signature typically exhibits much greater magnitude. Focusing on this short-duration, low-amplitude case enables a clear comparison of cross-device reactivity. Its amplitude is only slightly above the Dragon example in Figure 6.14e and comparable to normal Bluetooth data 6.13c. The CONFORMER–MAE baseline reconstructs the first half of the window poorly but gradually recovers, achieving a reasonable fit towards the latter half of both the anomaly and the surrounding window. This would likely still trigger detection via the elevated early reconstruction error. By contrast, COMPMAE exhibits a stronger and more persistent reaction across the entire segment, producing a distorted reconstruction that diverges even in non-anomalous portions of the window. The anomaly itself elicits a direct, focused response in COMPMAE that is largely absent in CONFORMER–MAE, improving anomaly visibility. Overall, the broader and more consistent reactivity seen in Figure 6.15b suggests a higher detection likelihood than in Figure 6.15a, although both would likely be detected.

In Figure 6.15c, a larger Hydra T1 SSH brute-force signature appears on the Pi, extending slightly beyond the normal operating range. Due to its elevated amplitude, neither model reconstructs the input faithfully (Figures 6.15c

Conformer–MAE **CompMAE**



(**a**) Nmap T3 (DragonBoard) (**b**) Nmap T3 (DragonBoard)

(**c**) Hydra T1 SSH (DragonBoard) (**d**) Hydra T1 SSH (DragonBoard)

(**e**) Hydra T16 SSH (DragonBoard) (**f**) Hydra T16 SSH (DragonBoard)

Figure 6.14: Reconstruction behaviour of Conformer–MAE (left column) and CompMAE (right column) on selected anomalous DragonBoard events from the Dragon_Pi dataset.

and 6.15d). The Conformer–MAE baseline still attempts to trace the overall waveform somewhat, whereas CompMAE instead *repels* the anomalous

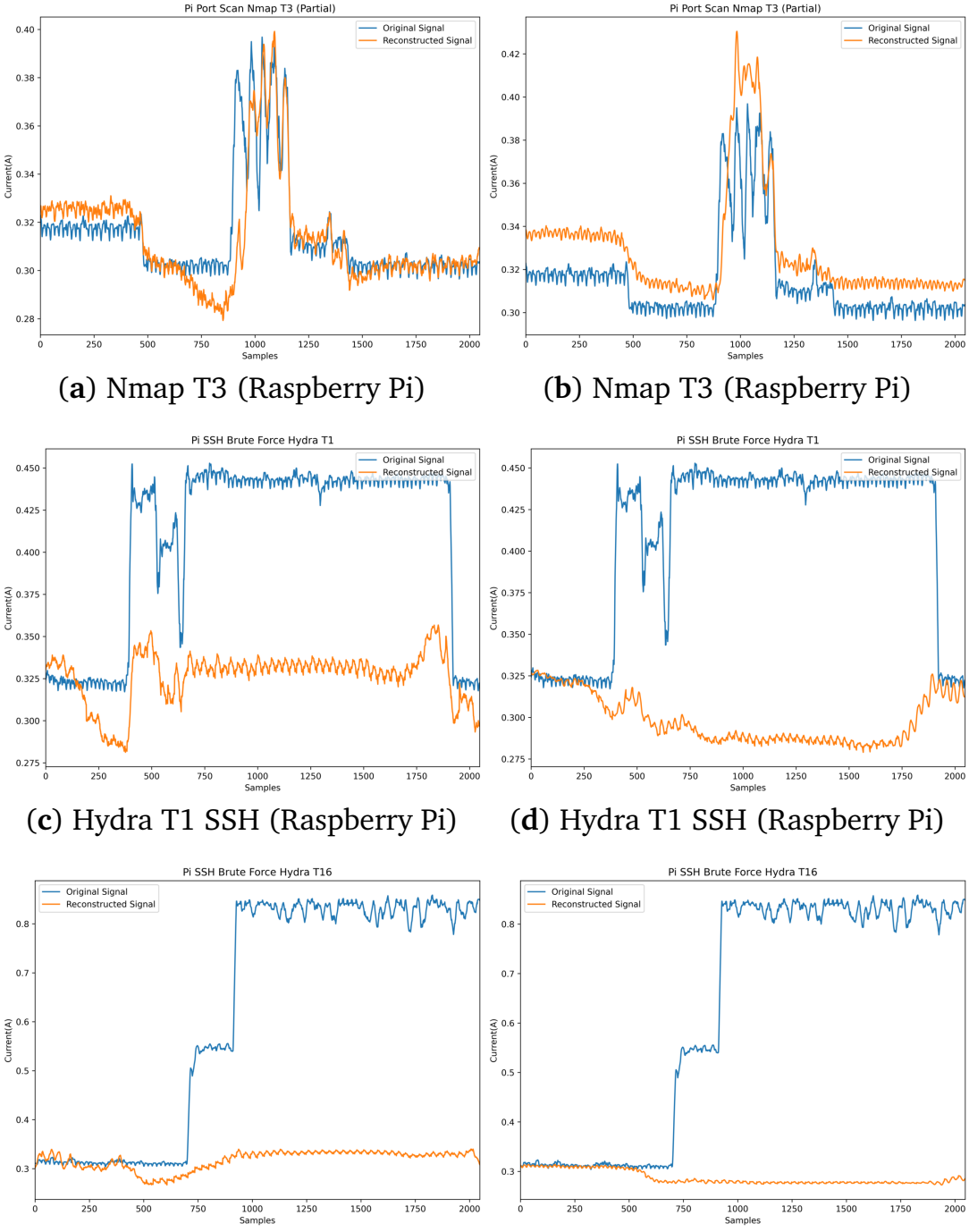
input, producing a reconstruction that deviates sharply away from the input. Importantly, while the anomalous data exceed the normal operating range, the reconstructions from both models remain below the maximum amplitudes observed during training, confirming that this behaviour is not simply a "reconstruct up to a cap" artefact as in Chapter 5. Even when both models fail to reconstruct due to amplitude saturation, COMPMAE still yields a higher reconstruction error, providing stronger anomaly separability by making little attempt to follow the anomalous waveform. This mirrors behaviour observed in Chapter 5, where anomalies far outside the normal amplitude range were unreconstructable by either model.

A consistent pattern appears in Figures 6.15e and 6.15f, which depict the Hydra T16 SSH brute-force anomaly. This attack exceeds twice the maximum normal amplitude seen during training, and thus neither model attempts a faithful reconstruction. Again, both produce outputs below the expected dataset amplitude ceiling. COMPMAE demonstrates a pronounced negative reconstruction response—actively pulling the reconstruction away from the anomaly and producing a large reconstruction error. In these high-amplitude scenarios, the anomaly would be successfully detected using the error of either model, but COMPMAE provides a wider error margin and therefore greater robustness to threshold variation.

Comparing the reconstruction performance of the CONFORMER–MAE baseline and COMPMAE across these cases highlights the additional benefit—and notably stronger anomaly reactivity—introduced by the compression module. Rather than the compressor's added components merely marginally increasing parameter count or improving the fit to normal data through greater representational capacity, the opposite effect is observed. The compression pathway enhances the model's latent sensitivity to deviations, making its reactions to anomalies both more pronounced and more consistent.

In COMPMAE, anomalous inputs do not simply become "hard to reconstruct"; the model exhibits a distinct and visibly reactive response to them. This behaviour contrasts with the outcome observed in Chapter 5, where the CAE tended to reconstruct anomalies only partially—reproducing them "up to a point." By comparison, COMPMAE actively distorts or rejects anomalous patterns, producing a strong reconstruction divergence that amplifies the anomaly signal in the error domain. This reactive reconstruction mechanism yields higher reconstruction error for anomalous inputs and, consequently,

CONFORMER–MAE **COMPMAE**



(**a**) Nmap T3 (Raspberry Pi) (**b**) Nmap T3 (Raspberry Pi)

(**c**) Hydra T1 SSH (Raspberry Pi) (**d**) Hydra T1 SSH (Raspberry Pi)

(**e**) Hydra T16 SSH (Raspberry Pi) (**f**) Hydra T16 SSH (Raspberry Pi)

Figure 6.15: Reconstruction behaviour of CONFORMER–MAE (left column) and COMPMAE (right column) on selected anomalous Raspberry Pi events from the DRAGON_PI dataset.

more clearly separable and detectable anomalies.

The robust and reactive reconstructions of COMPMAE in comparison to the

CONFORMER–MAE baseline further highlight the performance noise caused by annotation inaccuracy in the DRAGON_PI dataset. A similar trend was observed in Chapter 5, where a more robust model with higher reconstruction error on anomalies still exhibited only a small gap in metric performance in some cases. Fortunately, the CTF data help illustrate a more representative picture of model performance, due to clearer labels and greater attack diversity.

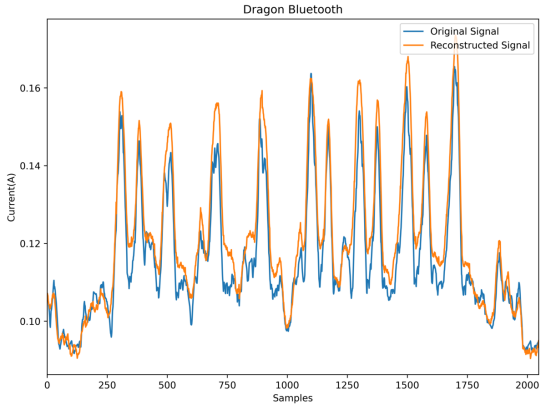
6.7.2.1 Over-Time Performance

As in Chapter 5, the effect of introducing the compressor block was further examined by analysing reconstruction performance over a sequence of contiguous windows rather than isolated segments. This evaluation provides a more realistic view of detection dynamics as they would manifest during deployment over time, where the model operates continuously on streaming data. Both the CONFORMER–MAE baseline and COMPMAE were evaluated across 200 successive windows, each one second in duration. The sequence is predominantly normal, with an Nmap port-scan centred near the midpoint, allowing direct observation of how each model behaves as a sliding detector traverses an anomaly in real time.

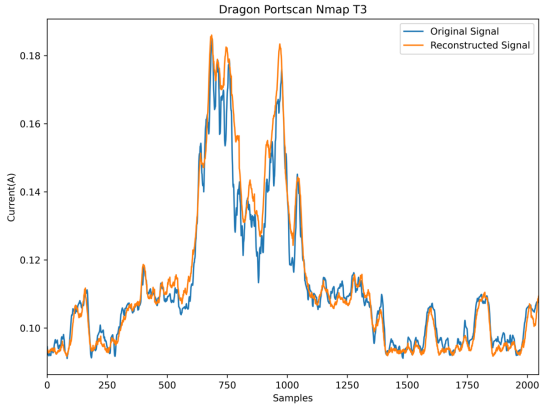
Considering first the CONFORMER–MAE baseline in Figure 6.16, proficient reconstruction of normal Bluetooth behaviour is evident in Figures 6.16a and 6.16d. This outcome is desirable: low reconstruction error on normal data reduces false positives and improves separation between the normal and anomalous error distributions. However, as shown in Figure 6.16b, the CONFORMER–MAE baseline also reconstructs the Nmap port-scan segment with unexpectedly high fidelity. This behaviour is contrary to the desired outcome—when anomalous windows are reconstructed too accurately, their reconstruction errors become nearly indistinguishable from those of normal data.

The impact of the lack of compressive elements, which effectively creates a trivial reconstruction task for CONFORMER–MAE and encourages overfitting on this anomaly, is evident in Figure 6.16c, where the MSE trace of the anomalous windows overlaps almost entirely with that of normal operation. As a result, the port-scan anomaly becomes effectively undetectable when viewed through reconstruction error alone. In fact, all behaviour is indistinguishable anomalous or not. This illustrates a key limitation of the uncompressed baseline: despite capturing general temporal regularities, it lacks the selective reactivity required

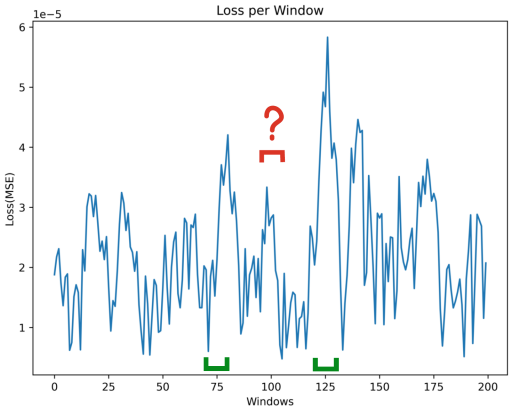
to amplify deviations during continuous, over-time inference.



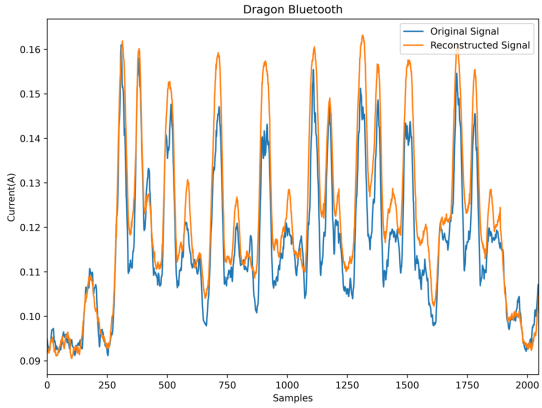
(**a**) Normal window (Bluetooth)



(**b**) Anomalous window (Nmap port-scan)



(**c**) Over-time MSE trace



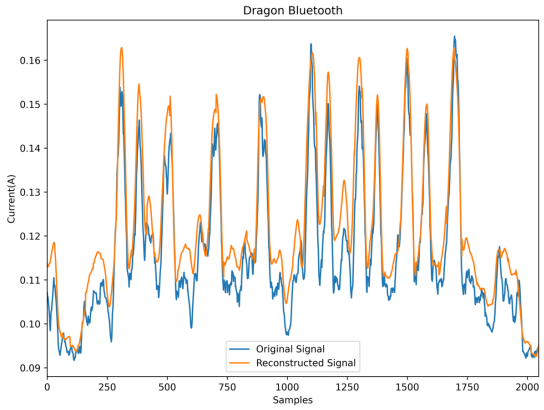
(**d**) Normal window (Bluetooth)

Figure 6.16: Reconstruction behaviour of the Conformer–MAE baseline over 200 windows. Normal windows are reconstructed accurately, but the Nmap port-scan anomaly is also reproduced with high fidelity, yielding an MSE trace in which the anomalous response is almost indistinguishable from normal operation (the anomaly lies within the region marked in red, while normal Bluetooth activity appears in the regions highlighted in green).

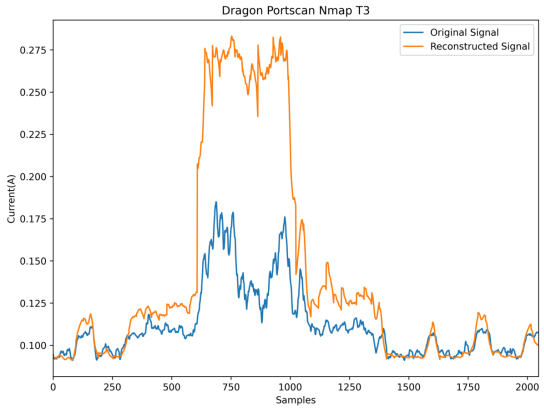
Turning to the over-time performance of CompMAE in Figure 6.17, strong reconstruction of normal data remains evident (Figures 6.17a and 6.17d). Although its fidelity does not quite match that of the Conformer–MAE baseline, CompMAE nonetheless reconstructs normal inputs with high consistency, yielding low reconstruction error across normal windows. This confirms that the introduction of the compression pathway does not impair the model's capacity to represent regular operational behaviour.

In stark contrast to both the normal case and the baseline behaviour, CompMAE displays a pronounced and desirable reaction to the same Nmap port-

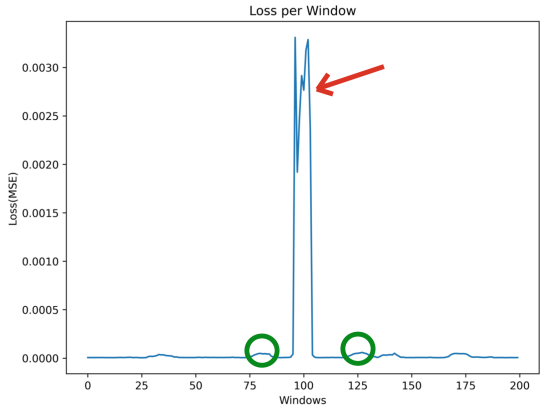
scan anomaly (Figure 6.17b). The anomaly elicits a strong reconstruction divergence, producing a marked increase in reconstruction error relative to surrounding windows—tens of times larger. As shown in Figure 6.17c, the anomalous overlapping windows for the portscan attack are clearly distinguishable, standing several orders of magnitude above the normal error levels. This distinct separation highlights the effect of the compression enhancement, which amplifies the model's sensitivity to deviations while maintaining stability on normal data.



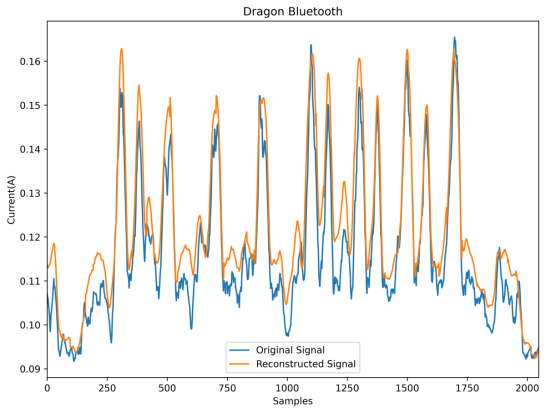
(**a**) Normal window (Bluetooth)



(**b**) Anomalous window (Nmap portscan)



(**c**) Over-time MSE trace



(**d**) Normal window (Bluetooth)

Figure 6.17: Reconstruction behaviour of COMPMAE across 200 windows. Normal windows are reconstructed with high consistency and consequently low error, as seen in adjacent normal windows A and D (highlighted in green), whereas the Nmap anomaly produces a strong divergence and a large, easily identifiable peak in the MSE trace (annotated by the red arrow).

Overall, these results confirm that the compressor in COMPMAE promotes a more *reactive reconstruction regime*, wherein normal behaviour is reconstructed accurately while anomalies are deliberately distorted, yielding highly separa-

ble error distributions. This enhancement is evident when compared to both the CONFORMER–MAE baseline and the approach in Chapter 5, where models reacted far less strongly to anomalies over the same segment of data (cf. Figures 6.17c and 5.12). Such reactive behaviour to anomalous input directly supports reliable anomaly detection in continuous, over-time operation.

6.8 Ablation

6.8.1 CONFORMER–MAE versus COMPMAE

To assess the specific contributions of the individual architectural components introduced in COMPMAE, an ablation study was conducted. The most significant ablation involves the complete removal of the compressor block, leaving only the masked CONFORMER–MAE baseline. This configuration isolates the impact of the compression mechanism while preserving all other architectural and training parameters. The results of this comparison have already been presented throughout this Chapter comparing CONFORMER–MAE to COMPMAE (see Table 6.2).

6.8.2 Conformer Module Depthwise Kernel

A central hyperparameter in the Conformer architecture is the kernel size of the depthwise convolution, which determines the receptive field for local feature extraction [24]. In this section, the influence of the convolutional module's kernel size—here referred to as the *external kernel*, as it resides in the Conformer-style module outside the compressor—is systematically evaluated for both masked CONFORMER–MAE and masked COMPMAE variants.

Kernel sizes of $\{3, 5, 7, 17\}$ were tested. In all cases, the same external kernel size was applied across all layers, while the *internal* kernel size within the compressor remained fixed at 5. The results are summarised in Table 6.3.

For the CONFORMER–MAE baseline, performance improves as the kernel increases from 3 to around 5–7, but declines again at the largest size ($k = 17$). This trend mirrors that reported in the original Conformer study, where moderate kernel sizes yield the best trade-off between local context modelling and parameter efficiency [24]. In this work, $k = 7$ provides the strongest overall performance, whereas $k = 5$ performs marginally better on the CTF subset. The difference between these two settings is small, and either can be considered

reasonable. As expected, increasing the kernel size slightly raises the number of parameters, with a maximum increase of roughly 20k between $k = 3$ and $k = 17$.

For COMPMAE, the relationship between kernel size and performance is less predictable. Performance initially improves from $k = 3$ to $k = 5$, decreases at $k = 7$, and then rises again at $k = 17$. This non-linear behaviour likely reflects the dual objective of the model: to reconstruct normal data accurately while maintaining strong reactivity to anomalies. Larger external kernels expand the temporal receptive field, enabling the model to capture longer-range dependencies. While this can enhance anomaly sensitivity, it may occasionally come at the cost of reduced normal reconstruction fidelity. The fluctuation across kernel sizes therefore represents a shifting balance between accurate reconstruction and reactive anomaly response. Another plausible explanation lies in kernel synergy, which will be covered in the next ablation.

Table 6.3: Comparison of Conformer and CompMAE with varying depthwise convolution kernel sizes.

	Conformer			CompMAE		
Kernel Size	Overall AUC	CTF Only AUC	Params (M)	Overall AUC	CTF Only AUC	Params (M)
3	0.775	0.763	2.00	0.767	0.762	2.139
5	0.793	0.839	2.00	0.803	0.832	2.141
7	0.797	0.833	2.01	0.762	0.759	2.143
17	0.784	0.819	2.02	**0.821**	**0.897**	2.153

The near-identical performance observed at $k = 5$ for both CONFORMER–MAE and COMPMAE suggests that the baseline Conformer's intrinsic representational quality and mild anomaly sensitivity form a foundation that the compressor amplifies. The compressor thus strengthens this inherent reactivity without degrading normal reconstruction performance. As with the baseline, parameter growth from $k = 3$ to $k = 17$ remains modest (approximately 20k parameters), confirming that kernel variation primarily influences the effective receptive field rather than computational scale.

6.8.3 Compressor Module Depthwise Kernel

In addition to the external Conformer kernel, the *internal kernel*—the depthwise convolution within the compressor block—was also investigated in Table 6.4. Because larger external kernels increase the temporal receptive field and appeared to enhance anomaly reactivity, it was important to determine whether a similar relationship holds for the compressor's internal convolution. In this experiment, the external kernel was fixed at $k_{\text{ext}} = 17$ for all configurations.

As with the external kernel, the internal-kernel results exhibit a non-monotonic trend. Rather than showing consistent improvement with increasing kernel size—or the clear rise–fall pattern observed for the CONFORMER–MAE baseline—performance fluctuates in a wavelike manner across k_{int}. This suggests that kernel size alone does not linearly determine reconstruction or detection performance within the compressor.

A plausible explanation lies in kernel synergy: the internal (k_{int}) and external (k_{ext}) depthwise convolutions act in series and therefore interact, rather than contributing independently. For example, an internal kernel of $k_{\text{int}} = 5$ may be particularly effective when operating on representations already structured by a larger external kernel $k_{\text{ext}} = 17$. In such cases, the external convolution captures broader temporal context, while the internal compressor kernel refines it into compact, anomaly-sensitive latent features.

Table 6.4: Effect of compressor *internal* depthwise kernel size (k_{int}) with fixed external kernel $k_{\text{ext}} = 17$.

Internal Depthwise k_{int}	Overall AUC	CTF Only AUC	Params (M)
3	0.788	0.814	2.153
5	**0.821**	**0.897**	2.153
7	0.801	0.819	2.154
17	0.807	0.857	2.156

A further observation is that larger internal kernels do not necessarily improve quality. This supports the view that COMPMAE's gains arise not from added parameters or generic capacity increases, but from the deliberate interplay of components within a tightly coupled architecture. In other words, effective anomaly reactivity depends on balanced design choices—particularly the syn-

ergy between external context capture and internal latent refinement—rather than raw model capacity.

6.8.4 Compressor Module Attention Heads

A key factor in Transformer performance is the number of attention heads (H) used within MHSA—in this case specifically for the secondary MHSA layer within the compressor module. Each head can specialise to different relational patterns, enabling a richer set of dependencies to be modelled. Increasing H therefore expands the number of attention "perspectives" but reduces the dimensionality per head.

Equally important is the dimension per head, $d_{\text{head}} = D/H$, where D denotes the feature dimension presented to the MHSA. With D fixed, increasing H proportionally decreases d_{head}, so each head operates on a smaller subspace and captures a more limited portion of the total representational capacity. Balancing the number of heads and per-head width is therefore a non-trivial design decision [139].

It was hypothesised that a moderately small d_{head} could act as an implicit compressive constraint, reinforcing the AE-like character of the compressor by limiting the information available to each head. As shown in Table 6.5, a medium configuration of $H{=}8$ (yielding $d_{\text{head}}{=}4$ under the compressed $D{=}32$) achieved the best overall performance.

Table 6.5: Effect of compressor attention heads H (and resulting per-head dimension $d_{\text{head}} = D/H$, with $D{=}32$) on performance.

Attention Heads H	Per-head Dimension d_{head}	Overall AUC	CTF Only AUC	Params (M)
2	16	0.732	0.686	2.153
4	8	0.742	0.788	2.153
8	4	**0.821**	**0.897**	2.153
16	2	0.732	0.720	2.153

Taken together, these results suggest that consistency between the compressor's internal MHSA ($H{=}8$) and the external MHSA in the main encoder ($H{=}8$) is more beneficial, rather than relying on extreme compression via very small d_{head} alone, thus disproving the hypothesis.

Maintaining the same number of heads across modules likely aids representational alignment and reduces inter-module mismatch, stabilising training. Effective performance therefore arises from a balanced head configuration—trading off parallel attention views against sufficient per-head capacity—rather than from maximal parallelisation or minimal per-head width.

6.8.5 Compression Strategy Ablation

Compression within the latent (hidden) space is a critical design factor for AE-style models. Conventional AEs typically perform compression by applying a fully connected bottleneck after flattening the entire feature space. In contrast, COMPMAE avoids flattening the sequence and instead compresses per token: each sequence element is reduced along its channel (encoder) dimension while preserving temporal length. This design maintains temporal resolution and avoids coupling unrelated time steps. A per-token, channel-mixing pointwise (1×1) convolution performs the initial reduction, and a gated linear unit (GLU) can optionally apply an additional learned halving of the feature dimension.

Three methodologies for feature-space reduction were evaluated:

- **Pointwise-only**: a 1×1 projection that compresses $D \to D/2$ or $D \to D/4$ within each token (no GLU).
- **GLU-only compression**: after a pointwise convolution which has no compressive factor, a GLU is applied per token, halving its input dimension (e.g., $128 \to 64$).
- **Pointwise + GLU (4×)**: a sequential pointwise reduction $D \to D/2$, followed by a GLU $D/2 \to D/4$, yielding a total 4× compression.

Given that the encoder dimension entering the compressor is fixed at 128, a 2× compressor yields an internal dimension of 64, while a 4× compressor reduces this to 32.

As shown in Table 6.6, the combined pointwise (2×) + GLU (2×) configuration achieves the strongest overall performance. This combination outperforms the 4× pointwise-only variant, indicating that a purely linear 1×1 projection at that strength is overly coarse and discards fine-grained local structure. Incorporating the GLU introduces a data-dependent gating mechanism that selectively filters features, preserving salient content while enforcing a deeper, more expressive bottleneck.

Table 6.6: Comparison of compression strategies within the compressor block. Compression ratio refers to the total reduction applied to the encoder dimension ($D = 128$).

Compression Ratio	Compression Components	Other Components	Overall AUC	CTF Only AUC	Params (M)
2	Pointwise Conv. (1×) + GLU (2×)	–	0.782	0.792	2.354
2	Pointwise Conv. (2×)	–GLU, +Swish	0.809	0.822	2.288
2	Pointwise Conv. (1×) + GLU (2×)	16 Heads (4 dims/head)	0.775	0.747	2.354
4	Pointwise Conv. (2×) + GLU (2×)	–	**0.821**	**0.897**	2.153
4	Pointwise Conv. (4×)	–GLU, +Swish	0.779	0.776	2.120

Without the GLU, 2× pointwise performs better than 4× pointwise, again consistent with the observation that excessive linear compression weakens reconstruction fidelity and anomaly reactivity. Conversely, the GLU-only (2×) variant slightly outperforms the 4× pointwise-only model, but the gain is modest: a single halving of the 128-dimensional space does not create as selective or expressive a bottleneck as the combined pathway.

An extra test examined the interaction between compression and per-head dimensionality. All other experiments used eight attention heads, corresponding to 8 dims/head under 2× compression (64/8) and 4 dims/head under 4× compression (32/8). To test whether maintaining a smaller d_{head} could serve as an implicit constraint, a 16-head variant was trained at the 2× setting ($64/16 = 4$ dims/head). This model performed marginally worse overall and substantially worse on the CTF subset, confirming that excessively low per-head dimensionality degrades attention expressivity rather than providing a useful compressive effect.

6.8.6 Compressor Component Ablation

One of the most critical additions to the compressor was hypothesised to be the MHSA layer. It was expected that MHSA, operating within the compressed feature space produced by the pointwise projection and GLU, would bias the model towards normal behaviour—since it operates on a subspace of salient normal features—and thereby heighten reactivity when anomalies are encountered. As shown in Table 6.7, removing MHSA from the compressor produces the largest drop in performance (CTF AUC $= 0.750$, Overall AUC $= 0.767$). This confirms the importance of attention within the compressor and suggests that other components rely on the contextual dependencies it establishes. Notably, the dropout-only variant (containing only a single $4\times$ pointwise projection) performs better than the "no-MHSA" configuration, implying that both the GLU and

depthwise convolution derive much of their effectiveness from the representational structure created by MHSA within the compressed space.

Within the compressor, the principal components under investigation are the GLU, MHSA, and depthwise convolution, each influencing anomaly sensitivity in distinct ways. When a component is removed, supporting elements (e.g., BatchNorm, Swish) are also removed or adjusted as necessary to maintain stability. The full compressor serves as the baseline for comparison.

Removing the depthwise convolution after MHSA results in a moderate performance decline (Overall AUC $0.821 \rightarrow 0.808$, CTF AUC $0.897 \rightarrow 0.867$). The depthwise convolution was designed to restore local resolution potentially lost during compression. Its removal indicates that while it contributes useful local context, it is not the principal driver of anomaly sensitivity. As expected, associated BatchNorm and Swish layers—whose roles are to stabilise and regularise the convolutional path—are omitted in this variant, as they exist primarily to support the depthwise operation (Nonlinearity within the compressor module remains due to the GLU and the softmax operation in MHSA, preventing the block from collapsing into a purely linear mapping).

Next, the compressor is reduced to a single pointwise projection with dropout: the GLU, MHSA, and depthwise convolution are all removed, and the pointwise compression is increased to $4\times$ to compensate for the loss of the GLU. A modest dropout ($p = 0.2$) is applied. This lean variant performs surprisingly well (Overall AUC $= 0.800$, CTF AUC $= 0.857$), achieving respectable results, especially on the CTF data, despite using approximately 69k fewer parameters and omitting key components. However, it still underperforms relative to the full compressor, underscoring the benefits of attention and gating mechanisms within the bottleneck. This variant also highlights the innate reactivity of the baseline CONFORMER–MAE model and suggests that there may be multiple viable pathways to enhance it other than that taken in COMPMAE.

The GLU plays a key role in front-end gating and compression. The initial pointwise projection halves the feature dimension, after which the GLU—operating within the reduced space—applies a selective, data-dependent halving that introduces nonlinearity and smoother feature suppression compared with an ungated linear 1×1 projection [149]. Removing the GLU leads to performance drops: for $2\times$ pointwise compression, CTF AUC falls to 0.822 (Overall 0.809); for $4\times$ pointwise compression, CTF AUC declines further to 0.776 (Overall 0.779). These results highlight the GLU's importance in compression-aware gating—

Table 6.7: Ablation of compressor components, ordered by CTF-only AUC.

Pointwise Conv.	GLU	MHSA	Depthwise Conv.	Other	Overall AUC	CTF Only AUC	Parameters (M)
✓	✓	✓	✓	n/a	0.821	**0.897**	2.153
✓	✓	✓	✗	– BatchNorm – Swish	0.808	0.867	2.151
$4\times$	✗	✗	✗	+ Dropout	0.800	0.857	2.084
$2\times$	✗	✓	✓	n/a	0.809	0.822	2.288
$4\times$	✗	✓	✓	n/a	0.779	0.776	2.120
✓	✓	✗	✓	n/a	0.767	0.750	2.120

without it, features are reduced too abruptly for MHSA to meaningfully exploit.

Finally, removing the MHSA component causes the most severe degradation (CTF AUC $= 0.750$, Overall AUC $= 0.767$), confirming that attention within the compressed space is essential. The dropout-only variant outperforming the "no-MHSA" model reinforces this conclusion: in the absence of MHSA, the remaining convolutional and gating components cannot capture or preserve long-range dependencies within the latent representation at all, performing worse than the baseline CONFORMER–MAE model.

Overall, these ablations demonstrate that MHSA provides the contextual structure upon which the GLU and depthwise convolution operate effectively, forming a hierarchical synergy that underpins COMPMAE's anomaly sensitivity. The results also highlight the delicate interdependence of these components: when one is removed, the remaining elements can lose much of their effectiveness and balance.

6.8.7 Residual Dropout

During early development of COMPMAE, a key concern was that the compressor module—by design—increases training difficulty, as the network must reconstruct signals from a heavily reduced feature space. This raised the possibility that the model might learn to bypass the compressor entirely via its residual skip connection, effectively nullifying the intended compression.

To mitigate this, a small amount of dropout ($p = 0.2$) was applied to the residual branch before recombining it with the compressor output. This *residual dropout* acts as a stochastic gating mechanism, preventing the model from relying exclusively on the shortcut path and thereby encouraging active utilisation of the compressor during training. At inference, the dropout is disabled, so this

Table 6.8: Effect of residual dropout (external kernel size fixed at 17).

Residual	Overall AUC	CTF Only AUC	Params (M)
Full	0.804	0.815	2.153
Dropout	**0.821**	**0.897**	2.153
None	0.773	0.822	2.153

intervention affects only the learning dynamics. A small dropout rate was chosen deliberately to minimise the discrepancy between training and inference conditions while still discouraging residual dominance.

Table 6.8 evaluates three configurations applied to the compressor block only: (i) a full residual (no dropout), (ii) a residual with dropout (dropout applied to the residual path prior to addition), and (iii) no residual connection.

Residual dropout yielded the best overall performance. The dropout-attenuated residual outperformed both the full-residual and no-residual variants, supporting the hypothesis that mild stochastic disruption of the skip path encourages the network to rely on the compressed representation rather than circumvent it during training. The full-residual results closely matched those of the CONFORMER–MAE baseline, suggesting that the compressor was largely bypassed and the model functioned similarly to an uncompressed CONFORMER–MAE variant.

Conversely, removing the residual connection entirely slightly improved the CTF-only metric but degraded overall performance. This indicates that, while eliminating the shortcut can force stronger use of the compressor, it also hinders gradient flow and stability, reducing convergence quality and generalisation, thus impairing the reconstruction of all data, normal or anomalous. Retaining the residual with light dropout therefore provides the optimal balance between compression enforcement and trainability.

This approach is conceptually similar to techniques such as stochastic depth [153] and Drop-Path [154], which randomly disable residual connections to improve robustness and encourage effective use of intermediate layers. In COMPMAE, the same principle is applied selectively to prevent residual dominance and ensure that the compressor contributes meaningfully to representation learning during training.

6.8.8 Masking Strategy and Ratio

Among all design factors in masked reconstruction, the *masking ratio* and *masking strategy* are arguably the most influential. Prior work provides informative anchors across different modalities. In masked language modelling, as exemplified by BERT [140], a masking ratio of around 15% is typically optimal: linguistic tokens carry high semantic density, and excessive masking renders the prediction task impossible. In contrast, for images, substantially higher ratios are feasible, as there exists a much higher ratio of redundant data—MAE [25] reports strong results around 75%—since spatial redundancy allows reconstruction from sparse observations. For waveform and linear time-series domains, similarly high ratios have proven effective; for example, MAEEG [147] demonstrated that a 75% masking ratio performs well for electrophysiological signals.

Table 6.9: Effect of masking strategy and ratio on reconstruction-based anomaly detection performance.

Masking Ratio	Masking Strategy	Overall AUC	CTF Only AUC	Params (M)
45%	Single Chunk	0.783	0.844	2.153
45%	Individual Patches	0.786	0.825	2.153
60%	Single Chunk	0.766	0.874	2.153
60%	Individual Patches	0.812	0.860	2.153
75%	Single Chunk	0.765	0.814	2.153
75%	Individual Patches	**0.821**	**0.897**	2.153

While these domains support large masking ratios, they differ markedly in masking strategy. MAE found that masking patches uniformly at random yields the best performance, whereas MAEEG reported that masking a single contiguous chunk improved temporal coherence for time-series data. Given these contrasting findings, both the masking strategy (random vs. contiguous) and the ratio were systematically ablated in this study.

As shown in Table 6.9, higher masking ratios combined with random sampling proved most effective in COMPMAE. Performance increased steadily as the random-mask ratio rose, consistent with the objective of encouraging deep, context-aware representation learning. Interestingly, at lower masking ratios, the single-chunk strategy—although weaker overall—outperformed random masking on the CTF-only subset.

This suggests that when masking is light, contiguous gaps may promote more

localised temporal reasoning, aligning with the sequential, multi-attack structure of the CTF recordings. However, at the higher 75% ratio, random masking decisively outperformed all alternatives, supporting the intuition that a heavily masked, randomly sampled objective provides the strongest self-supervised signal for learning robust normal sequence structure.

6.8.9 Model Depth and Capacity Considerations

Finally, while a number of architectural and hyperparameter factors were explored in this ablation study, varying the number of encoder layers or the overall model capacity (encoder dimension) was deemed unnecessary. Both CONFORMER–MAE and COMPMAE maintain identical architectural depth and comparable parameter counts, ensuring that observed differences arise from the compressor's internal design rather than from changes in representational capacity. Altering layer count or encoder dimension would primarily assess scaling behaviour—useful for efficiency benchmarking but orthogonal to the present objective of isolating the compressor's functional contribution to reconstruction quality and anomaly reactivity.

6.9 Edge Implementation Results: Preliminary Investigation

A central motivation of this thesis is the development of real-time, on-device intrusion detection for IoT systems. Although model training and validation were conducted on high-performance compute infrastructure, it is essential to verify that the resulting architectures remain viable for use under realistic edge constraints. The training workstation used in this study—equipped with 112 CPU cores, 500 GB RAM, and dual NVIDIA V100 GPUs (32 GB VRAM each)—far exceeds the computational capabilities of typical IoT or edge hardware. To demonstrate practical feasibility, COMPMAE was therefore deployed on representative low-power hardware and evaluated under real-time operational conditions.

For this preliminary investigation, the Qualcomm RB3 Gen 2 development kit was selected as the edge platform. RB3 Gen 2 is built around the Qualcomm QCS6490 SoC and targets edge-AI and robotics applications. The system integrates an octa-core Kryo 670 CPU complex, an Adreno 643-class GPU/visual

Table 6.10: Inference performance and power across server and edge devices. For server-class hardware, reported values reflect maximum power spec.

Device	Avg. inference time	Avg. power draw (W)	Avg. current (A)
NVIDIA® Quadro® GV100	9.520 ms	n/a—max power 250 W	n/a
Intel® Xeon® Platinum 8276	26.441 ms	n/a—max power 165 W	n/a
Qualcomm® RB3 Gen 2 (Kryo™ 670)	163.450 ms	6.954	0.5798

subsystem, and a 6th-generation Qualcomm AI Engine (Hexagon) for on-device acceleration. It also provides high-speed I/O and full Linux/Android support, making it a suitable environment for testing lightweight AI models under realistic power and memory constraints.

As an initial deployment exploration, the full COMPMAE model was executed on the RB3's Kryo CPU (without using the dedicated Neural Processing Unit (NPU) or GPU offload). During training, an overlap of 87.5% was used to maximise the learning signal (a 1 s window with a 125 ms hop). At inference, this overlap can be reduced to lower computational demand and increase time between inferences. For this investigation, a moderate update interval of 333 ms per window was adopted—equivalent to a 66.7% overlap for 1 s segments—allowing approximately 333 ms per cycle for both inference and data handling.

As shown in Table 6.10, COMPMAE remains highly efficient despite its hybrid Transformer architecture, with only 2.15 M parameters. On the NVIDIA V100 GPU, inference completes in under 10 ms, leaving substantial headroom for pre- and post-processing. On the server CPU (Intel® Xeon® Platinum 8276), inference completes in under 27 ms, demonstrating that even general-purpose CPUs can sustain real-time throughput. Most importantly, on the RB3's Kryo CPU, inference completes in under 164 ms. Under the 333 ms update cadence set out for this test, this leaves more than 160 ms per cycle for auxiliary tasks such as buffering and feature preparation—confirming that real-time operation is achievable even without hardware acceleration.

In terms of energy efficiency, direct power telemetry was unavailable for the server-class systems, whose thermal design powers (TDPs) far exceed those of embedded hardware (GPUs $\sim$250 W, CPUs $\sim$165 W). By contrast, the RB3 Gen 2 operates at a board-level max draw of $\leq$10 W. Using an external power analyser, an average consumption of 6.95 W was measured during repeated COMPMAE inference runs on the RB3, illustrating the substantial power advantage of edge execution for this workload.

Although the Kryo CPU is the slowest of the three compute platforms tested,

inference was still possible within a reasonable window. Importantly, however, the RB3 Gen 2 also incorporates a dedicated Qualcomm AI Engine (Hexagon NPU) specifically designed for neural inference. While this preliminary evaluation focused on CPU-only execution, future work can exploit the NPU and associated toolchains to further reduce inference latency and energy consumption—providing additional headroom for faster update rates or larger models.

Overall, these results confirm that COMPMAE, despite its hybrid Transformer architecture, remains well within the computational and power budgets of modern edge-AI platforms. This reinforces the practical viability of self-supervised, real-time intrusion detection at the network edge.

6.9.1 Summary

This Chapter presented the culmination of the methodological AI development in this thesis: the design and implementation of the *Compression-Enhanced Masked Autoencoder (*COMPMAE*)*, a novel architecture that unifies Conformer-style local–global modelling with a bespoke compression mechanism tailored for IoT power-based intrusion detection. Building on the trajectory established in Chapter 5—where CAEs and their dropout-enhanced variants demonstrated the promise of self-supervised reconstruction—this Chapter extended those foundations into the Transformer domain, refining both the architecture and the evaluation protocol. The resulting framework is lightweight, explainable, and fully compatible with edge deployment, while capable of modelling complex temporal dependencies in power traces.

At the core of the contribution lies the proposed *compression module*, which introduces several architectural innovations not present in existing designs. First, by inverting the standard Conformer convolutional block, the encoder reduces feature dimensionality early, forcing the model to learn under a deliberately restrictive bottleneck, prioritising salient structure and suppressing redundant activations. This enforced constraint improves representational efficiency and enhances the contrast between normal and anomalous behaviour, therefore fulfilling the third aim of the thesis.

Secondary multi-head self-attention (MHSA) within the compressed latent space enables global reasoning at low computational cost. Because the model reasons over reduced tokens, disruptions to normal dependency structure become more pronounced, naturally amplifying anomaly sensitivity. Com-

plementing this, residual dropout prevents information from bypassing the compressor, ensuring that each stage meaningfully contributes to representation learning.

COMPMAE makes use of AE-style channel compression without temporal flattening. By compressing channels (tokens) with pointwise convolutions while preserving sequence length, COMPMAE retains temporal coherence yet still enforces a strict bottleneck. This balance allows the model to reconstruct normal data faithfully while responding sharply to out-of-distribution inputs.

Together, these design choices yield a model that not only reconstructs normal behaviour accurately but also reacts strongly to anomalies. Channel reduction (e.g., $128 \rightarrow 64 \rightarrow 32$) transforms subtle deviations into pronounced reconstruction failures—turning anomaly detection into a clear structural signal rather than a small statistical fluctuation. This provides a diagnostic advantage: anomalous regions become easily identifiable both numerically and visually through the reconstruction itself.

In the broader context of the thesis, COMPMAE represents the synthesis of all preceding chapters:

1. the proof of concept established in Chapter 3;
2. the dataset development and real-world attack analysis of Chapter 4;
3. the initial self-supervised reconstruction work of Chapter 5.

It consolidates robustness, interpretability, and efficiency into a single model aligned with the thesis's overarching aim: to realise effective, explainable, and edge-feasible intrusion detection via side-channel power monitoring with AI.

The successful demonstration of COMPMAE on embedded hardware further confirms that modern Transformer-based architectures can operate within realistic IoT power and memory constraints, bridging the gap between state-of-the-art AI methods and practical, deployable cybersecurity.

Chapter 7

Conclusions and Future Work

This Chapter concludes this thesis by revisiting its primary research aims laid out in Chapter 1 and summarising how each has been addressed through the methodologies, experiments, and analyses presented in Chapters 3–6. Each section corresponds to one of the major aims defined in Chapter 1, reflecting the progressive evolution from synthetic to real data and from supervised to self-supervised learning approaches, finally culminating in a novel state-of-the-art self-supervised model for IoT intrusion detection using side-channel power data.

7.1 Aim 1: Power-Based Intrusion Detection Proof of Concept

The first aim examined whether useful behavioural features are present within the side-channel power consumption of IoT devices, and whether lightweight, edge-deployable AI models can feasibly detect anomalies in this modality or whether such patterns would simply be lost within the broader signal. Due to the scarcity of real anomalous data at the outset, synthetic anomalies were injected into otherwise normal power traces. In essence, this aim asked whether power data alone contains sufficiently repetitive and structured information for anomaly detection, and whether this structure can be learned by models small enough to run on constrained IoT hardware.

Chapter 3 addressed this aim through a host-based Intrusion Detection System (IDS) implemented on a low-power MAX78000 microcontroller equipped with a Neural Processing Unit (NPU). Using a supervised learning paradigm, a

lightweight Convolutional Neural Network (CNN) was trained on a combination of normal power traces and synthetically injected anomalies. Despite the tight resource constraints of the hardware platform, the system achieved strong performance: the *Comp3* and *AIO* variants obtained F1-scores of 0.902 and 0.903, respectively. These results demonstrated that device power behaviour contains consistent, discriminative patterns that can be effectively learned even by small, embedded models.

Crucially, the MAX78000 deployment confirmed that such an IDS is not only conceptually viable but also practically feasible. The CNN required only *55 KB* of model storage and drew approximately *15 mW* during inference while achieving faster-than-real-time performance. This provided the first concrete evidence in this thesis that power-based anomaly detection can operate entirely at the edge without relying on cloud resources or specialised hardware beyond what is already available in low-power IoT systems.

However, several limitations emerged from this early exploration. The most significant was the dependence on synthetic anomalies, which—although useful for controlled experimentation—cannot capture the diversity and unpredictability of real-world adversarial behaviour. Additionally, the anomaly-injection process introduced a subtle dataset bias, since the model encountered both normal and perturbed versions of identical sequences. Finally, relying on supervised learning raised concerns about the long-term generalisability of such systems: real IoT environments are characterised by evolving behaviour and previously unseen attack types, making it impractical to assume that labelled anomalies will always be available for training.

These limitations motivated the transition to Aim 2 (Chapter 4), where the investigation shifted to real attack data captured directly from IoT devices. Nonetheless, Aim 1 successfully established the foundational premise of the thesis: that side-channel power data is a viable and informative signal for intrusion detection, providing a strong platform on which the subsequent aims build.

7.2 Aim 2: Synthetic vs. Real Anomalies

The second aim examined the validity and limitations of synthetic anomaly generation for power-based intrusion detection. In Chapter 3, synthetic anomalies were used as a practical workaround for the scarcity of labelled attack data,

enabling controlled experiments on a constrained embedded platform (particularly one lacking network connectivity). While this approach successfully established the feasibility of power-based intrusion detection, it raised an important question: do real cyberattacks exhibit consistent, learnable signatures, or do they vary to such an extent that synthetic anomalies will struggle to approximate them?

Chapter 4 addressed this question by presenting a systematic investigation of real attack signatures as reflected in IoT device power traces. Two realistic testbeds were constructed—based on the Raspberry Pi and DragonBoard 410c platforms—to capture both normal and abnormal power behaviour under a variety of cyberattacks. These included reconnaissance, denial-of-service, and brute-force attacks, as well as a compound "Capture-the-Flag" (CTF) scenario in which multiple attack types were chained together. This experimental setup provided a detailed view of how actual intrusions manifest in device power consumption over time, independent of any particular detection model.

The analysis showed that, while in some cases real attacks can share similar signatures between devices within the same attack tool setting, they generally exhibit complex, multi-phase power signatures that can vary not only between attack types but also within a single type, depending on the attacker's tools, parameters, and network conditions. Even minor configuration changes produced noticeably different power profiles. By contrast, the synthetic anomalies used in Chapter 3 represent comparatively simple, stationary perturbations designed for controlled experimentation. While such anomalies are valuable for proof-of-concept validation and stress-testing models, they inevitably fall short of capturing the full variability and temporal structure observed in genuine cyberattacks. Although this thesis does not perform a direct, quantitative comparison between models trained on synthetic versus real anomalies, the qualitative differences observed in Chapter 4 highlight the risks of relying solely on synthetic perturbations when targeting real-world deployment.

Despite these challenges, valuable insights were gained from the attack data. The captured power traces provided a unique perspective on how conventional network-layer attacks translate into the physical side-channel domain, revealing previously under-documented relationships between computational activity and attack stages. This view can be particularly valuable for forensic analysis, enabling security practitioners to reconstruct the timeline of an intrusion using only a device's power behaviour. Importantly, Chapter 4 also demonstrates

the concept of the *Correct Key Event* (CKE)—a distinctive, repeatable power signature indicating successful password usage after a brute-force attack—demonstrating that meaningful operational milestones can be inferred directly from side-channel data.

From an AI-based intrusion detection standpoint, these findings reinforce that signature-focused, supervised approaches are difficult to sustain in the power domain. While such methods might achieve excellent laboratory performance on known attacks, the scarcity of labelled attack data, combined with the inherent variability of real-world intrusions, makes it impractical to construct comprehensive training sets of anomalous behaviour. Moreover, there is a risk of biasing models toward known attack patterns, potentially degrading performance on unseen threats. Instead, the results support adopting anomaly-based or self-supervised methodologies that rely solely on abundant normal behaviour to define expected operation. This shift improves model generalisability, eliminates the need for large amounts of attack data for model training and better reflects the practical realities of IoT systems, where exhaustive attack coverage is infeasible.

In summary, Aim 2 clarified the role of synthetic anomalies: they are valuable for initial feasibility studies and controlled experimentation, but insufficient on their own for robust, real-world intrusion detection. The real power traces captured in Chapter 4 revealed the true variability and complexity of attack behaviour, motivating a transition toward learning paradigms that do not depend on predefined attack labels. These insights directly informed Aim 3, developed in Chapters 5 and 6, which explores self-supervised learning strategies capable of detecting unseen anomalies using only normal data during training.

7.3 Aim 3: Model Evolution and Innovation

The final aim of this thesis focused on the progressive refinement and innovation of AI architectures for intrusion detection using IoT power data. Having established the feasibility of power-based anomaly detection (Aim 1) and demonstrated the distinctive nature of real attack signatures (Aim 2), this aim explored learning paradigms beyond supervised classification and pursued iterative architectural modernisation. The overarching goal was to develop a lightweight, explainable, and edge-feasible model capable of capturing both local and global temporal dependencies in power time-series.

This aim unfolded through the evolutionary path from the Convolutional Autoencoder (CAE) in Chapter 5 to the Compression-Enhanced Masked Autoencoder (COMPMAE) in Chapter 6. The CAE established a robust self-supervised baseline in which the model learned normal device behaviour exclusively from normal data, detecting anomalies as deviations in reconstruction quality. Introducing dropout regularisation in the CAE–DROP variant improved robustness by mitigating overfitting to minor fluctuations, ensuring that the model reconstructed normal traces faithfully while failing more reliably on anomalous inputs. Crucially, the reconstruction-based approach offers intrinsic interpretability: analysts can visually inspect inputs, reconstructions, and error patterns to understand and validate detections.

Building upon this foundation of self-supervised, reconstruction-based intrusion detection, the COMPMAE represented a substantial architectural and conceptual advancement for this thesis. Integrating the strengths of Masked Autoencoders (MAEs) and Conformer blocks, COMPMAE combined global self-attention with local convolutional context modelling, enabling it to capture both fine-grained and long-range temporal dependencies in power traces. Unlike the earlier CAE–DROP model which used dropout, COMPMAE employed heavy random masking (75%) during training, forcing the encoder to infer missing information from context and thereby learn rich representations of normal behaviour. During testing, this allowed the model to reconstruct normal sequences with high fidelity while amplifying reconstruction errors for anomalous inputs. A key evolution observed in Chapter 6 was that COMPMAE did not merely fail to reconstruct anomalies but often reacted to them strongly—producing exaggerated, structurally distorted outputs that made abnormal behaviour visually and numerically pronounced.

A central innovation introduced in Chapter 6 was the novel *compression module*, which reinterpreted both gating and attention mechanisms to operate within a reduced latent space. By constraining representational capacity, the module forced the network to prioritise salient, information-dense features of normal operation (similar to how a normal AE operates). Anomalies, lacking coherent latent structure, produced markedly larger reconstruction distortions that improved detectability. This architecture also minimised the computational cost of additional components: attention was performed within the compressed space on compressed tokens, and temporal coherence was preserved through pointwise convolutions rather than the sequence-flattening operations used in Chapter 5.

Together, these design elements advanced the state of power-based anomaly detection, demonstrating that modern Transformer-inspired architectures can remain interpretable, reactive, and practical for embedded deployment. Despite their conceptual sophistication, the resulting models retained a lightweight profile suitable for real-time edge operation. While only implemented on the CPU of an embedded device for an initial implementation experiment, the COMPMAE model was able to achieve a reasonable inference time of only 163.45 ms, while drawing 6.95 W on average.

Analytically, this stage deepened understanding of how reconstruction-based models perceive anomalies. The observation that COMPMAE not only fails to reconstruct anomalous inputs but actively *responds* to them—producing distinctive structural deviations—marks an important step forward in both explainability and sensitivity. This "reactive reconstruction" behaviour transforms detection from a passive assessment of reconstruction error into an active, interpretable signal that directly exposes the model's reasoning process.

In summary, this final aim completed this thesis's developmental trajectory: from lightweight supervised convolutional detectors trained on synthetic data to advanced hybrid Transformer–Conformer architectures trained in a self-supervised manner solely on normal data and evaluated on real attacks. The resulting frameworks combine real-time edge feasibility with transparent, explainable operation, aligning fully with the overarching goals of this thesis.

7.4 Conclusions and Future Directions

The objectives set forth at the beginning of this thesis have been progressively realised, with each Chapter building upon the insights and limitations of the previous to fulfil these aims. From the initial exploration of the feasibility of power-based intrusion detection to the development of a modern, self-supervised hybrid architecture, this research has demonstrated both the practicality and untapped potential of using side-channel power data as a defensive signal for IoT cybersecurity. Each Chapter contributed to a cumulative deepening of understanding—methodologically, technically, and conceptually—culminating in a comprehensive, edge-feasible intrusion detection framework.

Collectively, the results presented in Chapters 3–6 demonstrate that side-channel power data constitute a rich yet underutilised behavioural modality for

identifying abnormal activity. Chapter 3 established that even lightweight supervised CNNs can detect synthetic anomalies under strict resource constraints, providing a practical proof of concept. Chapter 4 extended this foundation by capturing and analysing real cyberattacks, revealing the nuanced temporal structure of genuine intrusion behaviour and underscoring the limitations of synthetic perturbations and reliance on fully supervised learning paradigms. Chapters 5 and 6 advanced the modelling paradigm further, transitioning from fully supervised to self-supervised reconstruction-based approaches that learn exclusively from normal behaviour. This progression began with the CAE and culminated in COMPMAE, a compression-enhanced Transformer–Conformer hybrid that combines global self-attention, local convolutional structure, and a novel compression module to create a highly sensitive, interpretable, and edge-compatible anomaly detector.

This developmental trajectory mirrors a broader evolution in AI—from reliance on handcrafted supervision and predefined anomaly classes toward autonomous, representation-driven learning. Crucially, the models developed here demonstrate that such modern learning paradigms can be adapted to the strict computational budgets of IoT devices. Although evaluated only on CPU, the demonstration of COMPMAE on embedded hardware confirms that sophisticated, explainable AI models can operate within realistic power and memory limits, bridging the gap between state-of-the-art research and deployable IoT security solutions.

Beyond the specific architectures proposed, this thesis contributes a methodological framework for IoT security research that embraces:

- the use of side-channel data as a primary behavioural signal,
- self-supervised learning as a means of avoiding dependence on scarce attack labels, and
- edge-implemented intelligence as a pathway to scalable, privacy-preserving intrusion detection.

This framework establishes a foundation upon which more comprehensive and autonomous IoT defence systems may be built.

7.5 Limitations and Scope of the Study

While this thesis demonstrates the feasibility and effectiveness of power-based intrusion detection for IoT systems, it is important to clearly delineate what has been established through experimentation and what remains uncertain.

7.5.1 What Has Been Demonstrated

This work has shown that side-channel power data contain sufficiently rich and structured information to enable anomaly detection using both lightweight and advanced AI models. Across multiple experimental stages, it has been demonstrated that:

- Lightweight CNNs can detect anomalous behaviour under constrained embedded conditions.
- Real cyberattacks produce observable and often distinctive signatures in device power consumption.
- Self-supervised, reconstruction-based models can learn normal behaviour and detect previously unseen anomalies without reliance on labelled attack data.
- Modern architectures, such as CompMAE, can improve anomaly sensitivity while remaining compatible with edge deployment.

These findings collectively establish power-based monitoring as a viable and promising modality for IoT intrusion detection.

7.5.2 What Has Not Been Fully Ascertained

Despite these results, several aspects remain untested or only partially explored. In particular, the generalisability of the proposed methods to highly diverse real-world environments has not been exhaustively validated.

The datasets used, while realistic and carefully constructed, represent a subset of possible device behaviours, workloads, and attack scenarios. As such, it cannot be assumed that all forms of normal or anomalous behaviour are fully captured within the scope of this work.

7.5.3 Sensitivity to Real-World Variability

The performance of the proposed approaches may be influenced by factors not fully explored in this thesis.

Application workloads and operational context. The models developed in this work assume relatively stable and repeatable baseline behaviour. In real-world deployments, IoT devices may operate under a wide range of application loads, including dynamic, multi-process, or user-driven activity. Such variability may alter power consumption patterns in ways that resemble anomalous behaviour, potentially increasing false positives or reducing detection sensitivity. Further investigation is required to assess robustness under highly heterogeneous and evolving workloads.

Range of attacker behaviours. Although this thesis evaluates multiple representative attack types, including reconnaissance, brute-force, and denial-of-service scenarios, it does not exhaustively cover all possible adversarial strategies. Certain classes of attacks—particularly low-intensity, stealthy, or highly distributed attacks—may produce only subtle deviations in power consumption and could therefore be more difficult to detect using the proposed techniques. Additionally, adversaries may adapt their behaviour to minimise observable side-channel impact.

In summary, while the results demonstrate strong potential for power-based intrusion detection, they should be interpreted within the context of the experimental conditions considered. Addressing these limitations represents an important direction for future work and is essential for achieving robust real-world deployment.

Future work should extend this foundation by:

- creating more IoT-based security datasets, especially those that explore under-represented channels such as power;
- exploring multimodal IDS frameworks that integrate side-channel power data with network telemetry to enhance detection context and reduce false positives;
- further optimising compression-aware Transformer architectures for ultra-low-power inference, ensuring scalability and long-term energy efficiency;

In summary, this thesis establishes both the theoretical and practical feasibility

of using side-channel power data as a primary defensive signal in IoT deployments protected by AI intrusion detection mechanisms. Through a systematic progression—from feasibility demonstration to real-world attack characterisation and advanced self-supervised modelling—it shows that effective, explainable, and deployable intrusion detection is achievable directly on the edge. The groundwork laid here provides not only a functional prototype but also a roadmap for the next generation of intelligent, autonomous, and adaptive IoT security systems.

References

[1] D. Lightbody, D.-M. Ngo, A. Temko, C. Murphy, and E. Popovici, "Host-based intrusion detection system for iot using convolutional neural networks," in *2022 33rd Irish Signals and Systems Conference (ISSC)*, 2022, pp. 1–7. DOI: 10.1109/ISSC55427.2022.9826188.

[2] D. Lightbody, D.-M. Ngo, A. Temko, C. Murphy, and E. Popovici, "Dragon_pi: Iot side-channel power data intrusion detection dataset and unsupervised convolutional autoencoder for intrusion detection," *Future Internet*, vol. 16, no. 3, 2024, ISSN: 1999-5903. DOI: 10.3390/fi16030088. [Online]. Available: https://www.mdpi.com/1999-5903/16/3/88.

[3] C. Griffiths, *The latest 2025 cyber crime statistics (updated october 2025)*, https://aag-it.com/the-latest-cyber-crime-statistics/ visited on November. 21, 2025, 2023.

[4] World Economic Forum, *Partnership against cybercrime, insight report 2020*, https://www.weforum.org/reports/partnership-against-cybercrime/ visited on April. 06, 2023, 2020.

[5] CISA, *Stop ransomware | cisa*, https://www.cisa.gov/stopransomware visited on April. 06, 2023.

[6] CISA, *What is wannacry/wanacrypt0r?* https://www.cisa.gov/sites/default/files/FactSheets/NCCIC ICS_FactSheet_WannaCry_Ransomware_S508C.pdf visited on April. 06, 2023.

[7] B. Chappell and S. Neuman, *U.s. says north korea 'directly responsible' for wannacry ransomware attack*, https://www.npr.org/sections/thetwo-way/2017/12/19/571854614/u-s-says-north-korea-directly-responsible-for-wannacry-ransomware-attack visited on April. 06, 2023.

[8] M. Kapko, *Cisa's priority sectors for 2023: Water, hospitals, k-12*, https://www.cybersecuritydive.com/news/CISA-water-schools-healthcare/634657/ visited on April. 06, 2023.

[9] A. Zacharakos, *No relief in sight for ransomware attacks on hospitals*, https://www.techtarget.com/searchsecurity/feature/No-relief-in-sight-for-ransomware-attacks-on-hospitals visited on April. 06, 2023.

[10] B. Fowler, *Ransomware cost us schools 3.56 billion in 2021, study says*, https://www.cnet.com/tech/services-and-software/ransomware-cost-us-schools-3-56-billion-in-2021-study-says/ visited on April. 06, 2023.

[11] N. C. S. C. NCSC, *Ransomware attack on health sector - update 2021-05-16*, https://www.ncsc.gov.ie/pdfs/HSE_Conti_140521_UPDATE.pdf visited on April. 06, 2023.

[12] P. McGrath, *Nuig it systems remain offline after attempted cyber attack*, https://www.rte.ie/news/2021/0930/1249912-nuig-cyber-attack/ visited on April. 06, 2023.

[13] O. Dwyer, *It services remain disrupted at two colleges after ransomware attacks*, https://www.thejournal.ie/tu-dublin-ransomware-attack-ongoing-5403034-Apr2021/ visited on April. 06, 2023.

[14] A. Daly, *Tu dublin's tallaght campus investigating 'significant' ransomware attack*, https://www.thejournal.ie/tu-dublin-ransomware-attack-5401763-Apr2021/ visited on April. 06, 2023.

[15] I. D. C. IDC, *Future of industry ecosystems: Shared data and insights*, https://blogs.idc.com/2021/01/06/future-of-industry-ecosystems-shared-data-and-insights/ visited on April. 06, 2023.

[16] T. Monte, *Why cybercrime is an increasing threat to the iot*, https://blog.nordicsemi.com/getconnected/why-cybercrime-is-an-increasing-threat-to-the-iot visited on April. 12, 2023.

[17] CISA, *Heightened ddos threat posed by mirai and other botnets*, https://www.cisa.gov/news-events/alerts/2016/10/14/heightened-ddos-threat-posed-mirai-and-other-botnets visited on April. 06, 2023.

[18] CISA, *Cyber-attack against ukrainian critical infrastructure*, https://www.cisa.gov/news-events/ics-alerts/ir-alert-h-16-056-01 visited on April. 06, 2023.

[19] H. Kilpatrick, *5 infamous iot hacks and vulnerabilities*, https://www.iotsworldcongress.com/5-infamous-iot-hacks-and-vulnerabilities/ visited on April. 06, 2023.

[20] Verkada, *Summary: March 9, 2021 security incident report*, https://www.verkada.com/security-update/report/ visited on January 12, 2024.

[21] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016. DOI: 10.1109/COMST.2015.2494502.

[22] L. Ruff et al., "A unifying review of deep and shallow anomaly detection," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 756–795, 2021. DOI: 10.1109/JPROC.2021.3052449.

[23] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology — CRYPTO' 99*, M. Wiener, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 388–397, ISBN: 978-3-540-48405-9.

[24] A. Gulati et al., *Conformer: Convolution-augmented transformer for speech recognition*, 2020. arXiv: 2005.08100 [eess.AS]. [Online]. Available: https://arxiv.org/abs/2005.08100.

[25] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, *Masked autoencoders are scalable vision learners*, 2021. arXiv: 2111.06377 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2111.06377.

[26] D. Lightbody, D.-M. Ngo, A. Temko, C. Murphy, and E. Popovici, "Attacks on iot: Side-channel power acquisition framework for intrusion detection," *Future Internet*, vol. 15, no. 5, 2023, ISSN: 1999-5903. DOI: 10.3390/fi15050187. [Online]. Available: https://www.mdpi.com/1999-5903/15/5/187.

[27] D. Lightbody, D.-M. Ngo, A. Temko, C. Murphy, and E. Popovici, "Compression enhanced masked autoencoders for iot intrusion detection," in *2025 10th International Workshop on Advances in Sensors and Interfaces (IWASI)*, 2025, pp. 1–6. DOI: 10.1109/IWASI66786.2025.11122006.

[28] K. Ashton et al., "That 'internet of things' thing," *RFID journal*, vol. 22, no. 7, pp. 97–114, 2009.

[29] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010, ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2010.05.010. [On-

line]. Available: https://www.sciencedirect.com/science/article/pii/S1389128610001568.

[30] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013, Including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services & Cloud Computing and Scientific Applications — Big Data, Scalable Analytics, and Beyond, ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2013.01.010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X13000241.

[31] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015. DOI: 10.1109/COMST.2015.2444095.

[32] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016. DOI: 10.1109/JIOT.2016.2579198.

[33] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, ser. MCC '12, Helsinki, Finland: Association for Computing Machinery, 2012, pp. 13–16, ISBN: 9781450315197. DOI: 10.1145/2342509.2342513. [Online]. Available: https://doi.org/10.1145/2342509.2342513.

[34] A. Khraisat and A. Alazab, "A critical review of intrusion detection systems in the internet of things: Techniques, deployment strategy, validation strategy, attacks, public datasets and challenges," *Cybersecurity*, vol. 4, no. 1, p. 18, 2021, ISSN: 2523-3246. DOI: 10.1186/s42400-021-00077-7. [Online]. Available: https://doi.org/10.1186/s42400-021-00077-7.

[35] D. Evans, "The internet of everything: How more relevant and valuable connections will change the world," Cisco IBSG, Tech. Rep., 2012. [Online]. Available: https://www.cisco.com/c/dam/global/en_my/assets/ciscoinnovate/pdfs/IoE.pdf.

[36] S. Sicari, A. Rizzardi, L. Grieco, and A. Coen-Porisini, "Security, privacy and trust in internet of things: The road ahead," *Computer Networks*, vol. 76, pp. 146–164, 2015, ISSN: 1389-1286. DOI: https://doi.org/

10.1016/j.comnet.2014.11.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128614003971.

[37] Center for Internet Security (CIS), *Election security spotlight – cia triad*, https://www.cisecurity.org/insights/spotlight/ei-isac-cybersecurity-spotlight-cia-triad visited on April. 12, 2023.

[38] M. Antonakakis et al., "Understanding the mirai botnet," in *26th USENIX Security Symposium (USENIX Security 17)*, Vancouver, BC: USENIX Association, Aug. 2017, pp. 1093–1110, ISBN: 978-1-931971-40-9. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis.

[39] R. M. Lee, M. J. Assante, and T. Conway, "Analysis of the cyber attack on the ukrainian power grid," *E-ISAC SANS Report*, 2016. [Online]. Available: https://nsarchive.gwu.edu/sites/default/files/documents/3891751/SANS-and-Electricity-Information-Sharing-and.pdf.

[40] CISA, *St. jude merlin@home transmitter vulnerability (update a)*, https://www.cisa.gov/news-events/ics-medical-advisories/icsma-17-009-01a visited on Nov. 22, 2025.

[41] S. Curry, *Hacking kia: Remotely controlling cars with just a license plate*, *Sam Curry's Blog*, Accessed: 2025-11-18, 2024. [Online]. Available: https://samcurry.net/hacking-kia.

[42] CISA, *Aa21-042a: Compromise of u.s. water treatment facility*, https://www.cisa.gov/news-events/cybersecurity-advisories/aa21-042a visited on Nov. 22, 2025.

[43] C. Cimpanu, *15 percent of all iot device owners don't change default passwords*, https://www.bleepingcomputer.com/news/security/15-percent-of-all-iot-device-owners-dont-change-default-passwords/ visited on April. 12, 2023.

[44] T. Zillner and S. Strobl, "Zigbee exploited: The good, the bad and the ugly," in *Black Hat USA Conference Proceedings*, 2015. [Online]. Available: https://blackhat.com/docs/us-15/materials/us-15-Zillner-ZigBee-Exploited-The-Good-The-Bad-And-The-Ugly-wp.pdf.

[45] V.-L. Nguyen, P.-C. Lin, and R.-H. Hwang, "Energy depletion attacks in low power wireless networks," *IEEE Access*, vol. 7, pp. 51 915–51 932, 2019. DOI: 10.1109/ACCESS.2019.2911424.

[46] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. New York: Springer, 2007, ISBN: 978-0-387-30857-9. DOI: 10.1007/978-0-387-38162-6.

[47] F.-X. Standaert, "Introduction to side-channel attacks," in *Secure Integrated Circuits and Systems*, I. M. Verbauwhede, Ed. Boston, MA: Springer US, 2010, pp. 27–42, ISBN: 978-0-387-71829-3. DOI: 10.1007/978-0-387-71829-3_2. [Online]. Available: https://doi.org/10.1007/978-0-387-71829-3_2.

[48] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013, Towards a Science of Cyber Security Security and Identity Architecture for the Future Internet, ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2012.12.018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128613000054.

[49] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems (idps)," National Institute of Standards and Technology, Tech. Rep. Special Publication 800-94, 2007. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-94.pdf.

[50] Z. Azam, M. M. Islam, and M. N. Huda, "Comparative analysis of intrusion detection systems and machine learning-based model analysis through decision tree," *IEEE Access*, vol. 11, pp. 80 348–80 391, 2023. DOI: 10.1109/ACCESS.2023.3296444.

[51] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, p. 20, 2019, ISSN: 2523-3246. DOI: 10.1186/s42400-019-0038-7. [Online]. Available: https://doi.org/10.1186/s42400-019-0038-7.

[52] IBM, *What is supervised learning?* https://www.ibm.com/topics/supervised-learning visited on Dec 16, 2024, 2024.

[53] A. Guezzaz, S. Benkirane, and M. Azrour, "A novel anomaly network intrusion detection system for internet of things security," in *IoT and Smart Devices for Sustainable Environment*, M. Azrour, A. Irshad, and R. Chaganti, Eds. Cham: Springer International Publishing, 2022, pp. 129–138, ISBN: 978-3-030-90083-0. DOI: 10.1007/978-3-030-90083-0_10.

[Online]. Available: https://doi.org/10.1007/978-3-030-90083-0_10.

[54] M. Roesch, "Snort - lightweight intrusion detection for networks," in *Proceedings of the 13th USENIX Conference on System Administration*, ser. LISA '99, Seattle, Washington: USENIX Association, 1999, pp. 229–238.

[55] KDD Cup. "Kdd cup 1999 data set," Accessed: Jan. 12, 2024. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[56] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. Anwar, "Ton_iot telemetry dataset: A new generation dataset of iot and iiot for data-driven intrusion detection systems," *IEEE Access*, vol. 8, pp. 165 130–165 150, 2020. DOI: 10.1109/ACCESS.2020.3022862.

[57] M. Karthikeyan, D. Manimegalai, and K. RajaGopal, "Firefly algorithm based wsn-iot security enhancement with machine learning for intrusion detection," *Scientific Reports*, vol. 14, no. 1, p. 231, 2024, ISSN: 2045-2322. DOI: 10.1038/s41598-023-50554-x. [Online]. Available: https://doi.org/10.1038/s41598-023-50554-x.

[58] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, O. Watanabe and T. Zeugmann, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 169–178, ISBN: 978-3-642-04944-6.

[59] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6. DOI: 10.1109/CISDA.2009.5356528.

[60] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102 419, 2020, ISSN: 2214-2126. DOI: https://doi.org/10.1016/j.jisa.2019.102419. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214212619305046.

[61] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, ser. MLSDA'14, Gold Coast, Australia QLD, Australia: Association for Computing Machinery, 2014, pp. 4–11, ISBN: 9781450331593.

DOI: 10.1145/2689746.2689747. [Online]. Available: https://doi.org/10.1145/2689746.2689747.

[62] M. Almiani, A. AbuGhazleh, A. Al-Rahayfeh, S. Atiewi, and A. Razaque, "Deep recurrent neural network for iot intrusion detection system," *Simulation Modelling Practice and Theory*, vol. 101, p. 102 031, 2020, Modeling and Simulation of Fog Computing, ISSN: 1569-190X. DOI: https://doi.org/10.1016/j.simpat.2019.102031. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1569190X19301625.

[63] T. Saba, A. Rehman, T. Sadad, H. Kolivand, and S. A. Bahaj, "Anomaly-based intrusion detection system for iot networks through deep learning model," *Computers and Electrical Engineering*, vol. 99, p. 107 810, 2022, ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2022.107810. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790622001100.

[64] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019, ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2019.05.041. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X18327687.

[65] S. Bhosale, *Network intrusion detection*, https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection visited on May. 9th, 2023.

[66] P. Singh, J. J. P, A. Pankaj, and R. Mitra, "Edge-detect: Edge-centric network intrusion detection using deep neural network," in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, 2021, pp. 1–6. DOI: 10.1109/CCNC49032.2021.9369469.

[67] I. Ullah and Q. H. Mahmoud, "Design and development of a deep learning-based model for anomaly detection in iot networks," *IEEE Access*, vol. 9, pp. 103 906–103 926, 2021. DOI: 10.1109/ACCESS.2021.3094024.

[68] S. Garcia, A. Parmisano, and M. J. Erquiaga, *Iot-23: A labeled dataset with malicious and benign iot network traffic*, version 1.0.0, Zenodo, 2020. DOI: 10.5281/zenodo.4743746. [Online]. Available: https://doi.org/10.5281/zenodo.4743746.

[69] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, Jul. 2009, ISSN: 0360-0300. DOI: 10.1145/1541880.1541882. [Online]. Available: https://doi.org/10.1145/1541880.1541882.

[70] M. Ahmed, A. Naser Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016, ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2015.11.016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804515002891.

[71] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1, pp. 37–52, 1987, Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists, ISSN: 0169-7439. DOI: https://doi.org/10.1016/0169-7439(87)80084-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0169743987800849.

[72] IBM, *What is unsupervised learning?* https://www.ibm.com/topics/unsupervised-learning visited on Dec 16, 2024, 2024.

[73] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001, ISSN: 0899-7667. DOI: 10.1162/089976601750264965. eprint: https://direct.mit.edu/neco/article-pdf/13/7/1443/814849/089976601750264965.pdf. [Online]. Available: https://doi.org/10.1162/089976601750264965.

[74] J. Liu, D. Yang, M. Lian, and M. Li, "Research on intrusion detection based on particle swarm optimization in iot," *IEEE Access*, vol. 9, pp. 38 254–38 268, 2021. DOI: 10.1109/ACCESS.2021.3063671.

[75] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, 1995, 1942–1948 vol.4. DOI: 10.1109/ICNN.1995.488968.

[76] M. Eskandari, Z. H. Janjua, M. Vecchio, and F. Antonelli, "Passban ids: An intelligent anomaly-based intrusion detection system for iot edge devices," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 6882–6897, 2020. DOI: 10.1109/JIOT.2020.2970501.

[77] M. Sheikhan and H. Bostani, "A hybrid intrusion detection architecture for internet of things," in *2016 8th International Symposium on*

Telecommunications (IST), 2016, pp. 601–606. DOI: 10.1109/ISTEL.2016.7881893.

[78] Y. Meidan et al., "N-baiot—network-based detection of iot botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, 2018. DOI: 10.1109/MPRV.2018.03367731.

[79] M. Catillo, A. Pecchia, and U. Villano, "A deep learning method for lightweight and cross-device iot botnet detection," *Applied Sciences*, vol. 13, no. 2, 2023, ISSN: 2076-3417. DOI: 10.3390/app13020837. [Online]. Available: https://www.mdpi.com/2076-3417/13/2/837.

[80] M. Dissem, M. Amayri, and N. Bouguila, "Neural architecture search for anomaly detection in time-series data of smart buildings: A reinforcement learning approach for optimal autoencoder design," *IEEE Internet of Things Journal*, vol. 11, no. 10, pp. 18 059–18 073, 2024. DOI: 10.1109/JIOT.2024.3360882.

[81] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in iot," *Sensors*, vol. 17, no. 9, 2017, ISSN: 1424-8220. DOI: 10.3390/s17091967. [Online]. Available: https://www.mdpi.com/1424-8220/17/9/1967.

[82] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *2010 IEEE Symposium on Security and Privacy*, 2010, pp. 305–316. DOI: 10.1109/SP.2010.25.

[83] M. Saied, S. Guirguis, and M. Madbouly, "Review of artificial intelligence for enhancing intrusion detection in the internet of things," *Engineering Applications of Artificial Intelligence*, vol. 127, p. 107 231, 2024, ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2023.107231. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095219762301415X.

[84] R. Lippmann et al., "Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation," in *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2, 2000, 12–26 vol.2. DOI: 10.1109/DISCEX.2000.821506.

[85] S. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. Chan, "Cost-based modeling for fraud and intrusion detection: Results from the jam project," in *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2, 2000, 130–144 vol.2. DOI: 10.1109/DISCEX.2000.821515.

[86] N. Moustafa and J. Slay, "Unsw-nb15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6. DOI: 10.1109/MilCIS.2015.7348942.

[87] S. S. Clark et al., "WattsUpDoc: Power side channels to nonintrusively discover untargeted malware on embedded medical devices," in *2013 USENIX Workshop on Health Information Technologies (HealthTech 13)*, Washington, D.C.: USENIX Association, Aug. 2013. [Online]. Available: https://www.usenix.org/conference/healthtech13/workshop-program/presentation/clark.

[88] F. Ding et al., "Deeppower: Non-intrusive and deep learning-based detection of iot malware using power side channels," in *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '20, Taipei, Taiwan: Association for Computing Machinery, 2020, pp. 33–46, ISBN: 9781450367509. DOI: 10.1145/3320269.3384727. [Online]. Available: https://doi.org/10.1145/3320269.3384727.

[89] B. Copos and S. Peisert, *Catch me if you can: Using power analysis to identify hpc activity*, 2020. arXiv: 2005.03135 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2005.03135.

[90] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943, ISSN: 1522-9602. DOI: 10.1007/BF02478259. [Online]. Available: https://doi.org/10.1007/BF02478259.

[91] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958. DOI: 10.1037/h0042519.

[92] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Available: https://www.deeplearningbook.org/.

[93] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986, ISSN: 1476-4687. DOI: 10.1038/323533a0. [Online]. Available: https://doi.org/10.1038/323533a0.

[94] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10, Haifa, Israel: Omnipress, 2010, pp. 807–814, ISBN: 9781605589077.

[95] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html.

[96] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. DOI: 10.1109/5.726791.

[97] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, ISSN: 0001-0782. DOI: 10.1145/3065386. [Online]. Available: https://doi.org/10.1145/3065386.

[98] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Cham: Springer International Publishing, 2014, pp. 818–833, ISBN: 978-3-319-10590-1.

[99] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, *Striving for simplicity: The all convolutional net*, 2015. arXiv: 1412.6806 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1412.6806.

[100] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, 2015. arXiv: 1502.03167 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1502.03167.

[101] Analog Devices, *Max78000: Ultra-low-power neural network microcontroller*, https://www.analog.com/en/products/max78000.html, Accessed: 2025-10-23, 2021.

[102] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, vol. 32, pp. 323–332, 2012, Selected Papers from IJCNN 2011, ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2012.02.016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608012000457.

[103] H. Bosman, A. Liotta, G. Iacca, and H. Wörtche, "Anomaly detection in sensor systems using lightweight machine learning," in *2013 IEEE International Conference on Systems, Man, and Cybernetics*, 2013, pp. 7–13. DOI: 10.1109/SMC.2013.9.

[104] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, ISSN: 1476-4687. DOI:

10.1038/nature14539. [Online]. Available: https://doi.org/10.1038/nature14539.

[105] B. Jacob et al., *Quantization and training of neural networks for efficient integer-arithmetic-only inference*, 2017. arXiv: 1712.05877 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1712.05877.

[106] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412.6980 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1412.6980.

[107] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Computers & Security*, vol. 86, pp. 147–167, 2019, ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2019.06.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016740481930118X.

[108] R. Pi, *Raspberry pi 3 model b*, https://www.raspberrypi.com/products/raspberry-pi-3-model-b/ visited on April. 12, 2023.

[109] Qualcomm, *Dragonboard 410c development board*, https://www.qualcomm.com/processors/application-processors/products/dragonboard-410c visited on April. 12, 2023.

[110] Qualcomm, *Snapdragon 410 processor*, https://www.qualcomm.com/smartphones/products/4-series/snapdragon-processors-410 visited on April. 12, 2023.

[111] J. Hildenbrand, *Qualcomm's snapdragon 410 used in over 550 designs and they've shipped more than 200 million units*, https://www.androidcentral.com/qualcomms-snapdragon-410-used-over-550-designs-and-theyve-shipped-more-200-million-units visited on April. 12, 2023.

[112] S. Collins, *The life of pi: Ten years of raspberry pi*, https://www.cam.ac.uk/stories/raspberrypi visited on April. 12, 2023.

[113] Raspberry Pi Foundation, *Raspberry pi os*, https://www.raspberrypi.com/software/, Accessed: 2025-10-23, 2023.

[114] Linaro, *Linaro debian for dragonboard 410c*, https://releases.linaro.org/96boards/dragonboard410c/, Accessed: 2025-10-23, 2018.

[115] T. Ylonen and C. Lonvick, *The secure shell (ssh) protocol*, RFC 4251, IETF, https://www.rfc-editor.org/rfc/rfc4251, 2006.

[116] Texas Instruments, *Ti cc2650stk sensortag development kit*, https://www.ti.com/tool/CC2650STK, Accessed: 2025-10-23, 2016.

[117] I. Harvey, *Sensortag.py-repository*, https://github.com/IanHarvey/bluepy/blob/master/bluepy/sensortag.py visited on April. 12, 2023.

[118] Agilent Technologies, *Agilent n6705a dc power analyzer*, https://www.keysight.com/us/en/product/N6705A/dc-power-analyzer-mainframe.html, Accessed: 2025-10-23, 2013.

[119] Matplotlib, *Choosing colormaps in matplotlib*, https://matplotlib.org/stable/tutorials/colors/colormaps.html visited on May. 10th, 2023.

[120] CISA, *National cyber incident scoring system*, https://www.cisa.gov/sites/default/files/2023-01/cisa_national_cyber_incident_scoring_system_s508c.pdf visited on May. 10th, 2023.

[121] Nmap.org, *Nmap: The network mapper*, https://nmap.org/ visited on April. 12, 2023.

[122] OffSec, *Kali linux*, https://www.kali.org/ visited on April. 12, 2023.

[123] T. S. Messerges, "Using second-order power analysis to attack dpa resistant software," in *Cryptographic Hardware and Embedded Systems — CHES 2000*, Ç. K. Koç and C. Paar, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 238–251, ISBN: 978-3-540-44499-2.

[124] W. J. Burns, *Common password list (rockyou.txt)*, https://www.kaggle.com/datasets/wjburns/common-password-list-rockyoutxt visited on April. 12, 2023.

[125] A. Ferrante and R. Buzzi, *Gtfobins: Curated list of unix binaries for privilege escalation*, https://gtfobins.github.io/, Accessed: 2025-10-23, 2018.

[126] M. van Hauser Heuse, *hydra*, version 9.2, Mar. 2021. [Online]. Available: https://github.com/vanhauser-thc/thc-hydra.

[127] die.net, *Hping3(8) - linux man page*, https://linux.die.net/man/8/hping3 visited on April. 12, 2023.

[128] V. Švábenský, P. Čeleda, J. Vykopal, and S. Brišáková, "Cybersecurity knowledge and skills taught in capture the flag challenges," *Computers & Security*, vol. 102, p. 102 154, 2021, ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2020.102154. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404820304272.

[129] A. Sayakkara, N.-A. Le-Khac, and M. Scanlon, "Leveraging electromagnetic side-channel analysis for the investigation of iot devices," *Digital Investigation*, vol. 29, S94–S103, 2019, ISSN: 1742-2876. DOI: https://

doi.org/10.1016/j.diin.2019.04.012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1742287619301616.

[130] die.net, *Ping(8) - linux man page*, https://linux.die.net/man/8/ping visited on April. 12, 2023.

[131] B. Strom, A. Applebaum, D. Miller, K. Nickels, A. Pennington, and C. Thomas, "Mitre att & ck: Design and philosophy," in *MITRE Technical Report*, 2018.

[132] die.net, *Scp(1) - linux man page*, https://linux.die.net/man/1/scp visited on April. 12, 2023.

[133] W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, Eds., *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer, 2019. DOI: 10.1007/978-3-030-28954-6.

[134] N. Moustafa, N. Koroniotis, M. Keshk, A. Y. Zomaya, and Z. Tari, "Explainable intrusion detection for cyber defences in the internet of things: Opportunities and solutions," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 3, pp. 1775–1807, 2023. DOI: 10.1109/COMST.2023.3280465.

[135] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006. DOI: 10.1126/science.1127647. eprint: https://www.science.org/doi/pdf/10.1126/science.1127647. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.1127647.

[136] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Artificial Neural Networks and Machine Learning – ICANN 2011*, T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 52–59, ISBN: 978-3-642-21735-7.

[137] H. Wu and X. Gu, "Towards dropout training for convolutional neural networks," *Neural Networks*, vol. 71, pp. 1–10, Nov. 2015, ISSN: 0893-6080. DOI: 10.1016/j.neunet.2015.07.007. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2015.07.007.

[138] D. Hendrycks and K. Gimpel, *Gaussian error linear units (gelus)*, 2023. arXiv: 1606.08415 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1606.08415.

[139] A. Vaswani et al., *Attention is all you need*, 2023. arXiv: 1706.03762 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1706.03762.

[140] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, 2019. arXiv: 1810.04805 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1810.04805.

[141] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," in *OpenAI Blog*, vol. 1, 2018, p. 8.

[142] A. Dosovitskiy et al., *An image is worth 16x16 words: Transformers for image recognition at scale*, 2021. arXiv: 2010.11929 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2010.11929.

[143] J. L. Ba, J. R. Kiros, and G. E. Hinton, *Layer normalization*, 2016. arXiv: 1607.06450 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1607.06450.

[144] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1512.03385.

[145] P. Ramachandran, B. Zoph, and Q. V. Le, *Searching for activation functions*, 2017. arXiv: 1710.05941 [cs.NE]. [Online]. Available: https://arxiv.org/abs/1710.05941.

[146] Z. Li, Z. Rao, L. Pan, P. Wang, and Z. Xu, *Ti-mae: Self-supervised masked time series autoencoders*, 2023. arXiv: 2301.08871 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2301.08871.

[147] H.-Y. S. Chien, H. Goh, C. M. Sandino, and J. Y. Cheng, *Maeeg: Masked auto-encoder for eeg representation learning*, 2022. arXiv: 2211.02625 [eess.SP]. [Online]. Available: https://arxiv.org/abs/2211.02625.

[148] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, *Transformer-xl: Attentive language models beyond a fixed-length context*, 2019. arXiv: 1901.02860 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1901.02860.

[149] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, *Language modeling with gated convolutional networks*, 2017. arXiv: 1612.08083 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1612.08083.

[150] F. Chollet, *Xception: Deep learning with depthwise separable convolutions*, 2017. arXiv: 1610.02357 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1610.02357.

[151] S. Kim et al., *Squeezeformer: An efficient transformer for automatic speech recognition*, 2022. arXiv: 2206.00888 [eess.AS]. [Online]. Available: https://arxiv.org/abs/2206.00888.

[152] P. Micikevicius et al., *Mixed precision training*, 2018. arXiv: 1710.03740 [cs.AI]. [Online]. Available: https://arxiv.org/abs/1710.03740.

[153] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Weinberger, *Deep networks with stochastic depth*, 2016. arXiv: 1603.09382 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1603.09382.

[154] G. Larsson, M. Maire, and G. Shakhnarovich, *Fractalnet: Ultra-deep neural networks without residuals*, 2017. arXiv: 1605.07648 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1605.07648.