Détection d'anomalies robuste et non-supervisée : Appliquée à la supervision du trafic réseau

Naji Najari

N°d'ordre NNT : 2022ISAL0124

THESE de DOCTORAT DE L'INSA LYON, membre de l'Université de Lyon

Ecole Doctorale 512
Ecole Doctorale InfoMaths

Spécialité/ discipline de doctorat : Informatique

Soutenue publiquement le 13/12/2022, par :
Naji NAJARI

Robust Unsupervised Anomaly Detection Application to Network Traffic Monitoring

---

Devant le jury composé de :

OWEZARSKI, Philippe	DR - HDR	LAAS-CNRS	Rapporteur
VINCENT, Nicole	Professeur	Université de Paris	Rapporteure
MALLAT, Stéphane	Professeur	Collège de France	Examinateur
CHATEAU, Thierry	Professeur	UCA	Examinateur
DEVIJVER, Emilie	Chercheure	CNRS	Examinatrice
GARCIA, Christophe	Professeur	INSA de Lyon	Directeur de thèse
DUFFNER, Stefan	MdC - HDR	INSA de Lyon	Co-directeur de thèse
LEFEBVRE, Grégoire	Chercheur	Orange	Co-directeur de thèse
BERLEMONT, Samuel	Chercheur	Orange	Co-directeur de thèse

Département FEDORA – INSA Lyon - Ecoles Doctorales

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **https://www.edchimie-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage secretariat@edchimie-lyon.fr	**M. Stéphane DANIELE** C2P2-CPE LYON-UMR 5265 Bâtiment F308, BP 2077 43 Boulevard du 11 novembre 1918 69616 Villeurbanne **directeur@edchimie-lyon.fr**
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** **https://edeea.universite-lyon.fr** Sec. : Stéphanie CAUVIN Bâtiment Direction INSA Lyon Tél : 04.72.43.71.70 secretariat.edeea@insa-lyon.fr	**M. Philippe DELACHARTRE** INSA LYON Laboratoire CREATIS Bâtiment Blaise Pascal, 7 avenue Jean Capelle 69621 Villeurbanne CEDEX Tél : 04.72.43.88.63 **philippe.delachartre@insa-lyon.fr**
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** **http://e2m2.universite-lyon.fr** Sec. : Bénédicte LANZA Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.e2m2@univ-lyon1.fr	**Mme Sandrine CHARLES** Université Claude Bernard Lyon 1 UFR Biosciences Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69622 Villeurbanne CEDEX **sandrine.charles@univ-lyon1.fr**
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** **http://ediss.universite-lyon.fr** Sec. : Bénédicte LANZA Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.ediss@univ-lyon1.fr	**Mme Sylvie RICARD-BLUM** Institut de Chimie et Biochimie Moléculaires et Supramoléculaires (ICBMS) - UMR 5246 CNRS - Université Lyon 1 Bâtiment Raulin - 2ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tél : +33(0)4 72 44 82 32 **sylvie.ricard-blum@univ-lyon1.fr**
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** **http://edinfomaths.universite-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	**M. Hamamache KHEDDOUCI** Université Claude Bernard Lyon 1 Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tél : 04.72.44.83.69 **hamamache.kheddouci@univ-lyon1.fr**
Matériaux	**MATÉRIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec. : Yann DE ORDENANA Tél : 04.72.18.62.44 yann.de-ordenana@ec-lyon.fr	**M. Stéphane BENAYOUN** Ecole Centrale de Lyon Laboratoire LTDS 36 avenue Guy de Collongue 69134 Ecully CEDEX Tél : 04.72.18.64.37 **stephane.benayoun@ec-lyon.fr**
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** **http://edmega.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bâtiment Direction INSA Lyon mega@insa-lyon.fr	**M. Jocelyn BONJOUR** INSA Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69621 Villeurbanne CEDEX **jocelyn.bonjour@insa-lyon.fr**
ScSo	**ScSo*** **https://edsciencessociales.universite-lyon.fr** Sec. : Mélina FAVETON INSA : J.Y. TOUSSAINT Tél : 04.78.69.77.79 melina.faveton@univ-lyon2.fr	**M. Bruno MILLY** Université Lumière Lyon 2 86 Rue Pasteur 69365 Lyon CEDEX 07 **bruno.milly@univ-lyon2.fr**

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

To my parents and my sister.

Remerciements

Il y a des moments dans la vie où les mots peinent à exprimer toute la gratitude que l'on ressent. Ce travail de thèse en est un, et je suis profondément reconnaissant envers toutes les personnes qui ont rendu ce voyage à la fois possible et mémorable.

Je tiens tout d'abord à exprimer ma plus profonde gratitude envers mes directeurs de thèse, Christophe Garcia et Stefan Duffner, pour leur encadrement exceptionnel, leur expertise et leur disponibilité tout au long de ce projet. Leur guidance éclairée et leur soutien indéfectible ont été des piliers essentiels dans l'achèvement de cette thèse.

Mes encadrants, Grégoire Lefebvre et Samuel Berlemont, ont été des piliers de connaissance et de soutien. Votre expertise et votre disponibilité ont grandement contribué à ma croissance personnelle et professionnelle. Votre écoute et votre encouragement ont enrichi mon expérience de manière incommensurable. Je vous suis extrêmement reconnaissant pour votre générosité dans le partage de votre savoir et de votre expérience. J'espère avoir de nouveau l'occasion de collaborer avec vous!

Je souhaite également exprimer ma profonde gratitude envers les membres du jury. Je remercie HDr. Philippe OWEZARSKI et Pr. Nicole VINCENT d'avoir accepté d'être les rapporteurs de ce manuscrit ainsi que Pr. Stéphane MALLAT, Pr. Thierry CHATEAU, et HDr. Emilie DEVIJVER les examinateurs, pour l'intérêt qu'ils ont porté à mes travaux. Merci pour votre implication dans le jury de cette thèse. Votre engagement dans ce processus a été pour moi une source d'inspiration et d'apprentissage inestimable.

Un merci tout particulier à ma famille: Sghaier, Henia, et Nada, qui a toujours été là pour moi. Votre amour inconditionnel, votre soutien moral et votre encouragement constant dans les moments de doute ont été la source de ma force et de ma détermination. Votre sacrifice et votre foi en mes capacités m'ont permis de poursuivre mes rêves et d'atteindre cet objectif.

Je suis également redevable à mes amis, pour leur soutien et les moments de détente partagés. Votre présence, vos encouragements et nos rires ont été un baume dans les périodes difficiles de mon parcours. Un merci particuler pour mes amis: Imed, Bilal, Cylia, Leticia, Ziad, Haythem, Fayrouz, Neil, Amel, Paul, Emna, Wissal, Abdessalam, Mehdi, Khoder, et Wael. Chacun d'entre vous a joué un rôle unique et essentiel dans cette aventure. Votre impact sur ma vie dépasse largement le cadre de ce projet, et j'apprécie profondément le lien que nous avons tissé.

Je tiens également à remercier chaleureusement mes collègues d'Orange pour leur esprit de collaboration, leur soutien et les moments partagés. Travailler à vos côtés a été une expérience enrichissante qui a profondément marqué mon parcours professionnel.

Enfin, je remercie tous ceux qui, de près ou de loin, ont contribué à cette aventure. Chaque rencontre, chaque échange, chaque moment partagé a été un pas de plus vers l'achèvement de ce doctorat.

Ce travail est le résultat de nombreuses mains tendues, de mots d'encouragement et de liens humains profonds. À tous, je vous adresse mon plus sincère merci.

Contents

Remerciements i
Contents iii

Acronyms **ix**

List of Figures **xiv**

List of Tables **xvii**

Abstract **1**

Résumé **3**

Publications Associated With This Thesis **7**

1 Introduction **9**
1.1 Network Monitoring And Anomaly Detection 10
1.1.1 Device Management and Monitoring 11
1.1.2 Internet of Things 12
1.1.3 Limitations of Traditional Monitoring Systems 14
1.1.4 Anomaly Detection for IoT Device Monitoring 15
1.1.5 Privacy And Ethical Principles 15
1.2 Objectives and Contributions 16
1.3 Overview of the thesis 17

2 Anomaly Detection **20**
2.1 Network Traffic Data 22
2.1.1 IP Protocol Overview 23
2.1.1.1 IP Packet Analysis 23
2.1.1.2 IP Flow Analysis 26
2.1.2 Network Traffic Anomalies 27
2.2 Definition And Categorization Of Anomaly Detection Methods 28
2.2.1 Definition Of Anomalies 28
2.2.2 Types Of Anomalies 29
2.2.3 Generic Categorization Of Anomaly Detection Methods 30
2.2.3.1 Categorization Based On Supervision 30
2.2.3.2 Based On The Detection Strategy 32
2.3 Classical Methods for Anomaly Detection 33
2.3.1 Instance-Based Methods 33
2.3.1.1 Distance-Based Methods 33

2.3.1.2 Density-Based Methods . . . 34
2.3.1.3 Strengths And Weaknesses Of Instance-Based Anomaly Detection 36
2.3.2 Statistical Models . . . 36
2.3.2.1 Parametric Statistical Method . . . 37
2.3.2.2 Non-Parametric Statistical Method . . . 38
2.3.3 One-Class Classification-Based Models . . . 40
2.3.3.1 One Class Support Vector Machine . . . 41
2.3.3.2 Support Vector Data Description . . . 42
2.3.3.3 AD With Classification-Based Approaches . . . 43
2.3.3.4 Strengths And Weaknesses Of Classification-Based AD . . . 44
2.3.4 Neural Network-Based One-Class Classification . . . 45
2.3.4.1 Artificial Neurons And Feed-Forward Neural Networks . . . 45
2.3.4.2 One-Class Neural Networks . . . 46
2.3.4.3 Deep Support Vector Data Description . . . 47
2.3.4.4 AD With One-Class Neural Networks And Deep Support Vector Data Description . . . 47
2.3.4.5 Strengthes And Weaknesses Of One-Class Neural Networks And Deep SVDD . . . 47
2.3.5 Reconstruction-Based AD . . . 48
2.3.5.1 Autoencoders . . . 48
2.3.5.2 Variational Autoencoders And Normalizing Flows . . . 49
2.3.5.3 Anomaly Detection With AEs, VAEs, and NFs . . . 51
2.3.5.4 Sequence-to-sequence Models And Transformers . . . 52
2.3.5.5 Anomaly Detection With Sequence-to-sequence Models And Transformers . . . 56
2.3.5.6 Strengths And Weaknesses Of Reconstruction-Based AD . . . 56
2.4 Conclusion . . . 57

3 Robust Autoencoders for Unsupervised Anomaly Detection 59
3.1 Problem Statement . . . 61
3.1.1 Nature of Input Data and Anomalies . . . 61
3.1.2 Unsupervised Anomaly Detection . . . 62
3.1.3 Robustness to Training Contamination . . . 63
3.2 Related Work . . . 64
3.2.1 Principal Component Analysis (PCA) . . . 64
3.2.1.1 PCA Overview . . . 64
3.2.1.2 PCA-based Anomaly Detection . . . 65
3.2.1.3 PCA Sensitivity to Training Outliers . . . 66
3.2.2 Robust Principal Component Analysis (RPCA) . . . 66
3.2.2.1 RPCA Overview . . . 66
3.2.2.2 RPCA-based Anomaly Detection In Network Traffic . . . 67
3.2.2.3 RPCA Limitations . . . 68
3.2.3 Robust Deep Autoencoders (RDAs) . . . 68
3.2.3.1 RDA Overview . . . 68
3.2.3.2 RDA-based Anomaly Detection In Network Traffic . . . 70
3.2.3.3 RDA Limitations . . . 70
3.3 Contribution: RADON, Robust Autoencoder with Dynamic Outlier filteriNg . . . 71
3.3.1 RADON Training Strategy . . . 71
3.3.2 RADON Projection Strategy . . . 72
3.3.3 Unimodal Thresholding of the Reconstruction Histogram . . . 73
3.3.4 Hypotheses . . . 74

3.4 Experiments and Results . . . 74
3.4.1 NSL-KDD dataset . . . 74
3.4.1.1 Dataset Description . . . 74
3.4.1.2 Training Protocols . . . 75
3.4.1.3 Training Parameter Settings and Evaluation Criteria . . . 76
3.4.1.4 Results . . . 77
3.4.2 MedBIoT dataset . . . 78
3.4.2.1 Dataset Description . . . 79
3.4.2.2 Training Protocols . . . 79
3.4.2.3 Training Parameter Settings and Evaluation Criteria . . . 79
3.4.2.4 Results . . . 80
3.5 Conclusion . . . 81

4 Robust Variational Autoencoders for Unsupervised Anomaly Detection 83
4.1 Problem Statement and Background . . . 84
4.1.1 Problem Statement . . . 84
4.1.2 Extreme Value Theory . . . 85
4.2 Related Work: Robust Variational Autoencoder (RVAE) . . . 86
4.2.1 Robust Variational Inference and the β-divergence . . . 86
4.2.2 RVAE Training Strategy . . . 87
4.3 Contribution: GRAnD, Robust VAEs and NFs for Unsupervised Network AD . . . 89
4.3.1 Robust Rejection Strategy . . . 89
4.3.2 Training Loss . . . 91
4.3.3 Hypotheses . . . 92
4.4 Experiments and Results . . . 92
4.4.1 NSL-KDD dataset . . . 92
4.4.1.1 Dataset Description . . . 92
4.4.1.2 Training Protocols . . . 92
4.4.1.3 Training Parameter Settings and Evaluation Criteria . . . 93
4.4.1.4 Results . . . 94
4.4.2 MedBIoT dataset . . . 96
4.4.2.1 Dataset Description and Training Protocols . . . 96
4.4.2.2 Training Parameter Settings and Evaluation Criteria . . . 96
4.4.2.3 Results . . . 96
4.5 Conclusion . . . 98

5 RESIST: Robust Transformer for Unsupervised Time Series Anomaly Detection 101
5.1 Problem Statement . . . 103
5.1.1 Nature of Input Data and Anomalies . . . 103
5.1.2 Unsupervised time Series Anomaly Detection . . . 104
5.1.3 Robustness to Training Contamination . . . 105
5.2 Related Work: AnomalyTransformer, Time Series Anomaly Detection with Association Discrepancy . . . 105
5.2.1 AnomalyAttention . . . 106
5.2.2 Series Association . . . 106
5.2.3 Prior Association . . . 107
5.2.4 Association Discrepancy . . . 107
5.2.5 AnomalyTransformer Global Architecture . . . 107
5.2.6 Minimax Training Strategy . . . 108
5.2.7 Test Point Anomaly Score . . . 109

5.3 Contribution: RESIST, Robust Transformer for Unsupervised Time Series Anomaly Detection . . . 109
5.3.1 Beyond Self-Sequence Dependencies . . . 109
5.3.2 Architecture Overview . . . 110
5.3.3 Siamese Encoder . . . 110
5.3.3.1 Self-attention and Co-attention Module . . . 112
5.3.3.2 FFN Layer . . . 113
5.3.4 Fusion Strategy . . . 113
5.3.5 RESIST Decoder . . . 114
5.3.6 Robust Training Loss . . . 114
5.3.7 Hypotheses . . . 115
5.4 Experiments and Results . . . 116
5.4.1 CICIDS2017 dataset . . . 116
5.4.1.1 Dataset Description . . . 116
5.4.1.2 Data Preprocessing . . . 117
5.4.1.3 Training and Testing Protocols . . . 117
5.4.1.4 Training Parameter Settings and Evaluation Criteria . . . 120
5.4.1.5 Results . . . 120
5.4.1.6 Synthesis and Discussion . . . 126
5.4.2 TON-IoT Dataset . . . 128
5.4.2.1 Dataset Description . . . 128
5.4.2.2 Data Preprocessing . . . 128
5.4.2.3 Training Protocol and Parameter Settings . . . 128
5.4.2.4 Results . . . 129
5.4.2.5 Synthesis and Discussion . . . 130
5.5 Comparison of the Three Contributions: RADON, GRAnD, and RESIST . . . 130
5.6 Conclusion and Perspectives . . . 131

6 Conclusion and Perspectives **133**
6.1 Conclusion . . . 134
6.1.1 RADON . . . 134
6.1.2 GRAnD . . . 135
6.1.3 RESIST . . . 135
6.2 Feasibility Study: End-to-end Monitoring Solution . . . 136
6.2.1 Proposition Of An Architecture For An End-to-end Monitoring Tool . . . 136
6.2.1.1 Data Collection . . . 136
6.2.1.2 Data Processing And Anomaly detection . . . 136
6.2.1.3 Data Visualization . . . 138
6.2.2 Discussion On Ethics . . . 138
6.2.2.1 Data Collection . . . 138
6.2.2.2 Data Processing And Anomaly Detection . . . 138
6.2.2.3 Data Visualization . . . 143
6.3 Research Limitations And Perspectives . . . 143
6.3.1 Data Discussion . . . 143
6.3.1.1 Validation Of Anomaly Detectors . . . 143
6.3.1.2 Privacy Discussion . . . 144
6.3.2 Model Discussion . . . 144
6.3.2.1 Sensitivity of Robust transformEr developed for unSupervised tIme Series anomaly deTection. (RESIST) . . . 144
6.3.2.2 Static Models . . . 144
6.3.2.3 One Model Per Device Family . . . 145

6.3.3 Explainable Anomaly Detection Discussion 145

Bibliography **I**

Acronyms

AD Anomaly Detection. 10, 15, 16, 17, 18, 22, 24, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 40, 43, 44, 45, 46, 47, 48, 49, 51, 52, 53, 56, 57, 61, 62, 63, 64, 65, 66, 68, 71, 74, 76, 79, 81, 84, 87, 92, 98, 103, 104, 110, 112, 124, 134, 135, 136, 138, 143, 144, 145

AE AutoEncoder. 16, 17, 44, 46, 48, 49, 50, 51, 52, 56, 63, 64, 68, 69, 70, 72, 76, 78, 79, 80, 81, 84, 85, 86, 89, 98, 103, 130, 134, 135, 145

ANID Attention for Network Intrusion Detection. 56

AssDis Association Discrepancy. 105, 106, 107, 108, 109

AUROC Area Under the Curve of the Receiver Operating Characteristics. 43, 52, 76, 77, 78, 79, 80, 94, 95, 96, 98, 120, 121, 122, 123, 124, 125, 126, 129, 130

BLSTM-RNNs Bidirectional Long Short Term Memory based Recurrent Neural Networks. 31, 56

CA Co-Attention. 112, 118

CICIDS17 Canadian Institute of Cybersecurity Intrusion Detection System. xv, xvii, 116, 117, 121, 124, 125, 126, 128, 130, 131, 135

CNN Convolutional Neural Network. 51

CPU Central Processing Unit. 13, 14, 138

DAGMM Deep Autoencoding Gaussian Mixture Model. 51

DDoS Distributed Denial of Service. 13, 25, 31, 70

DM Device Management. 10, 11

DoS Denial of Service. 27, 31, 51

DPI Deep Packet Inspection. 24, 25

dSVDD deep Support Vector Data Description. 45, 46, 47, 48

eCDF empirical Cumulative Distribution Function. 91, 92, 135

ELK Elasticsearch, Logstash, and Kibana. 136

EM Expectation–Maximization. 37

EVT Extreme Value Therory. 17, 84, 85, 86, 89, 135

FFN Feed-Forward Network. 45, 46, 47, 49, 51, 53, 55, 56, 76, 80, 93, 107, 110, 112, 113, 114

GAN Generative Adversarial Network. 56

GDPR General Data Protection Regulation. 15, 25

GMM Gaussian Mixture Model. 36, 37, 38

GPD Generalized Pareto Distribution. 85

GPU Graphics Processing Unit. 138

GRAnD Generative Robust autoencoder for unsupervised Anomaly Detection. 17, 84, 88, 89, 92, 93, 94, 95, 98, 130, 131, 135, 144

GRU Recurrent Neural Network. 52, 54

GRU-RNNs Gated Recurrent Unit based Recurrent Neural Networks. 52

HIDS Host Intrusion Detection System. 14

IANA Internet Assigned Numbers Authority. 26

ICMP Internet Control Message Protocol. 43

IDS Intrusion Detection System. 10, 14, 15, 24, 37, 51, 74

IE Information Element. 26, 136

IETF Internet Engineering Task Force. 12, 26

IF Isolation Forest. 76, 80, 92, 94, 95, 129

iid independent and identically distributed. 38, 52, 62, 85, 86, 87, 91, 98, 103, 130, 134

IoT Internet of Things. 10, 11, 12, 13, 14, 15, 22, 23, 25, 31, 78, 79, 81, 134, 136, 138, 143, 144, 145

IP Internet Protocol. xvii, 11, 15, 17, 22, 23, 24, 25, 26, 31, 57

IPFIX IP Flow Information Export. xvii, 26

IQR Inter-Quartile Range. 89, 93, 97, 115

ISO International Organization for Standardization. 23

ISP Internet Service Provider. 11, 13, 24, 27, 28, 81

IT Information Technology. 13, 14

KDE Kernel Density Estimation. 36, 38, 39, 40

KL Kullback-Leiber. xv, 49, 86, 87, 88, 107

KLE Kalman-Loeve Expansion. 67

LAN Local Area Network. 11, 16, 23, 24, 31, 61, 62, 136, 138, 144, 145

LOF Local Outlier Factor. 35, 36

LSTM-RNNs Long Short Term Memory based Recurrent Neural Networks. 51, 52

MIB Management Information Base. 22

MLE Maximum Likelihood Estimation. 37, 85, 86

MLP Multi-Layer Perceptron. 46, 76

MRI Magnetic Resonance Imaging. 10

MSE Mean Squared Error. 114

MTU Maximum Transmission Unit. 25

NF Normalizing Flow. 17, 48, 49, 50, 51, 84, 85, 86, 88, 89, 92, 94, 95, 135

NIDS Network Intrusion Detection System. 14, 51

NLP Natural Language Processing. 52, 54, 56

OCC One-Class Classification. 40, 42, 43

OCNN One-Class Neural Network. 45, 46, 47

OSI Open Systems Interconnect. 23, 25, 26

OSVM One-class Support Vector Machine. 40, 41, 42, 43, 44, 46, 47, 52, 76, 80, 92, 94

PCA Principal Component Analysis. 17, 44, 61, 63, 64, 65, 66, 67, 68, 81

POT Peaks-Over-Threshold. 85, 89

QKV Query-Key-Value. 54

QoE Quality of Experience. 22

QoS Quality of Service. 11, 22

RADON Robust Autoencoder with Dynamic Outlier filteriNg. 16, 17, 61, 70, 71, 72, 73, 74, 75, 77, 78, 79, 80, 81, 84, 98, 99, 130, 131, 135, 144

RBF Radial Basic Function. 70

RBRP Recursive Binning and Re-Projection. 34

RDA Robust Deep Autoencoder. xiv, xvii, 17, 61, 64, 68, 69, 70, 71, 72, 75, 76, 77, 78, 80, 81, 84, 88

ReLU Rectified Linear Unit. 46, 52, 55, 93

RESIST Robust transformEr developed for unSupervised tIme Series anomaly deTection.. vi, xv, 17, 18, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 133, 135, 144

RPCA Robust Principal Component Analysis. 17, 61, 64, 66, 67, 68, 69, 81, 84

RVAE Robust Variational Autoencoder. 17, 76, 78, 84, 86, 87, 88, 92, 94, 98

SA Self-Attention. 112, 118

SNMP Simple Network Management Protocol. 22, 23

SoC System On Chip. 138

SPI Shallow Packet Inspection. 24, 25, 143, 144

SVDD Support Vector Data Description. 40, 42, 43, 44, 47

SVM Support Vector Machine. 31, 38, 41, 70

SVT Singular Value Thresholding. 67

TCP Transmission Control Protocol. 11, 25, 31, 43, 75

TDR True Detection Rate. 40

TON-IoT Telemetry, Operating systems logs and Network traffic of IoT device. xv, 116, 127, 128, 129, 130, 131, 135

TOS Type Of Service. 26

UDP User Datagram Protocol. 11, 22, 23, 43, 75

VAE Variational Autoencoder. 17, 48, 49, 50, 51, 52, 56, 84, 85, 86, 87, 88, 89, 92, 94, 95, 96, 135, 145

VQA Visual Question Answering. 112, 121

WAN Wide Area Network. 11, 23

XAD eXplainable Anomaly Detection. 145

XAI eXplainable Artificial Intelligence. 145

List of Figures

1.1 Illustration of a smart home network. . . . 12

2.1 Illustration of the SNMP architecture, extracted from [188] . . . 23
2.2 The OSI seven-layer model. . . . 24
2.3 The typical structure of a network packet, extracted from [274]. . . . 24
2.4 Illustration of punctual (left), contextual (middle), and collective (left) anomalies, extracted from [144]. . . . 30
2.5 Supervised (left), semi-supervised (middle), and unsupervised (right) anomaly detection. . . . 31
2.6 An example of a 2-dimensional outlier. . . . 34
2.7 Illustration of Kernel Density Estimation sensitivity with respect to the bandwidth hyperparameter h, extracted from [37]. We note that h controls the smoothness, where a very small h (top) may result in a noisy density model and a very large h (bottom) results in a less informative density. . . . 39
2.8 Illustration of the one-class classification paradigm, extracted from [252]. . . . 41
2.9 Overview of the OSVM approach, extracted from [252]. . . . 41
2.10 Overview of the SVDD approach, extracted from [129]. . . . 43
2.11 Hybrid model workflow. . . . 44
2.12 Single- (left) and multi-layer (right) neural networks, extracted from [6]. . . . 45
2.13 The workflow of one-class neural networks, extracted from [58] . . . 46
2.14 The architecture of AEs, adapted from [80]. . . . 48
2.15 The architecture of variational autoencoders, extracted from [103]. . . . 50
2.16 An example of a normalizing flow model adapted from [273] . . . 50
2.17 An example of a contextual anomaly in a temperature time series data, extracted from [63]. . . . 52
2.18 Sequence-to-sequence model, extracted from [73]. . . . 53
2.19 The transformer architecture, extracted from [260]. . . . 54

3.1 Illustration of punctual (left), contextual (middle), and collective (left) anomalies, extracted from [144] . . . 62
3.2 The main difference between noise and outlier according to Robust Deep Autoencoder (RDA) reference [292]. On the left, the image of the digit 7 is an outlier among 2 digit images. This outlier corresponds to a structured row that is corrupted in the data matrix. On the right, noisy data are visualized, where some pixels of the images are corrupted. . . . 69
3.3 RADON training strategy, which alternates between updating the AE parameters (top) and updating the subsets $\mathbb{X} = \mathbb{L} \cup \mathbb{S} \cup \mathbb{U}$ (bottom). . . . 71
3.4 The thresholding strategy at the end of the first iteration (left), and at the end of the following ones (right). . . . 73
3.5 Variation of $\mathbb{L}$ and $\mathbb{S}$ thresholds during the training stage. . . . 78

4.1 The POT method of EVT, adapted from [90]. . . . 85

4.2 Comparison of the robustness of Kullback-Leiber (KL) and β-divergences, extracted from [11]. . . . 88
4.3 The workflow of the proposed approach, GRAnD. . . . 90
4.4 Illustration of the rejection strategy using the POT approach. . . . 91
4.5 Empirical cumulative distribution function of $\mathbb{U}$ samples 91
4.6 NSL-KDD experimental results: comparison of AD methods based on average AUROCs and deviations over five runs for multiple contamination ratios. . . . 94
4.7 Sensitivity analysis of GRAnD-PF on the NSL-KDD dataset. . . . 95
4.8 The MedBIoT experimental results, for the the four device categories: fan, light, lock, and switch. We report the average AUROC with the standard variation over five runs. . . . 97
4.9 Sensitivity analysis of RVAE and GRAnD-PF on MedBIoT dataset. We report the average AUROC with the standard variation over five runs. . . . 98

5.1 Univariate (left) and multivariate (right) time series, extracted from [40]. O1, O2, O3 are three anomalies. . . . 104
5.2 AnomalyTransformer architecture, adapted from [279]. . . . 106
5.3 Non-overlapping sliding windows of a toy sequence. . . . 109
5.4 RESIST architecture. . . . 111
5.5 Self-attention unit 112
5.6 Co-attention unit 112
5.7 The general robust loss function proposed in [30]. . . . 115
5.8 Three configurations of RESIST based on the modular composition of co-attention and self-attention modules. . . . 118
5.9 Comparison between RESIST three variants: RESIST-SS, RESIST-SC, and RESIST-CC, on the Canadian Institute of Cybersecurity Intrusion Detection System (CICIDS17) dataset. . . . 121
5.10 The experimental results for RESIST trained with different loss functions: L2 loss for RESIST-MSE, Charbonnier loss for RESIST-Ch, Cauchy loss for RESIST-Cauchy, and Geman-McClure loss for RESIST-GM. . . . 122
5.11 RESIST performances for different history length K and with different fusion strategies: RESIST-Concat denotes RESIST with the concatenation fusion, while RESIST-Mixup denotes RESIST with the *mixup* fusion. . . . 123
5.12 Evolution of RESIST-Concat and Resist-Mixup training times with the history length K. . . . 124
5.13 Comparison between RESIST and the competing methods on CICIDS17 dataset. 125
5.14 RESIST sensitivity analysis according to the history length hyperparameter K. . 126
5.15 RESIST sensitivity analysis according to the scale hyperparameter c of training robust loss. . . . 127
5.16 RESIST sensitivity analysis according to the number of heads h of multi-head attention units. . . . 127
5.17 Telemetry, Operating systems logs and Network traffic of IoT device (TON-IoT) experimental results. . . . 129

6.1 Architecture of the end-to-end monitoring tool. . . . 137
6.2 The *overview* interface of our monitoring solution. . . . 139
6.3 The *flow* interface of our monitoring solution. . . . 140
6.4 The *service* interface of our monitoring solution. . . . 141
6.5 The *anomaly* interface of our monitoring solution. . . . 142

List of Tables

2.1 A list of some Internet Protocol (IP) header-based metadata extracted by Wireshark. 25
2.2 A list of the most popular IP Flow Information Export (IPFIX) Information Elements. 26

3.1 Statistics about NSL-KDD dataset 75
3.2 The hyperparamters used in the NSL-KDD experiments. 76
3.3 NSL-KDD experimental results, with different percentages γ of training outliers. * indicates that the result is significant compared to RDA, according to Welch's test with a p-value=0.05. 77
3.4 The hyperparameters used in the MedBIoT experiments. 80
3.5 MedBIoT experimental results, with $\gamma = 0.1$. * indicates that the result is significant, according to Welch's test with a p-value=0.05. 80

4.1 The hyperparameters used in the NSL-KDD experiments. 93
4.2 The hyperparameters used in the MedBIoT experiments. 97

5.1 The label classes present in the CICIDS17 dataset 117
5.2 The hyperparameters used in the experiments of this chapter. 120
5.3 The experimental results on the CICIDS17 dataset. 131

Abstract

With the advent of the Internet of Things (IoT), connected devices are becoming ubiquitous and progressively introduced into domestic networks. These heterogeneous constrained objects, are potential targets for a wide spectrum of anomalies, e.g., software and hardware dysfunctions and malicious attacks. Orchestrating such fleet of devices, ensuring their nominal operation, and diagnosing anomalous activities are becoming a primordial asset for Telecommunication operators and service providers.

The present thesis explores the problem of unsupervised and robust anomaly detection based on network traffic analysis, and applied to Internet of Things (IoT) device monitoring. The principle is to firstly model the nominal behavior of a connected device using a set of metadata extracted from its network traffic. Then, any atypical behavior, i.e., any significant deviation from the nominal behavior expected by the model, is considered as an anomaly.

In particular, we explore the topic of representation learning with artificial neural networks, and specifically the autoencoder architecture. To model the norm, classical anomaly detection approaches train an autoencoder to reconstruct the nominal data, minimizing an error criterion between the initial data and the output of the network. As the new anomalous observations are structurally different, their reconstruction is accompanied by a significant loss of information, with a large reconstruction error. However, the constitution of a training dataset devoid of non-nominal data remains costly, time-consuming, and most of the time infeasible, as experts cannot update domain knowledge with new anomalies exactly as they arise. Thus, we strove to develop robust autoencoders, which are able to model the norm, even if the training dataset is contaminated by an unknown amount of anomalies.

In particular, we propose three contributions. The first two contributions focus on the robust detection of punctual anomalies and the third one extends the scope of our study to contextual and collective anomaly detection in time series.

First, we propose to analyze the distribution of reconstruction scores in order to perform a self-supervision. We dynamically estimate thresholds from the reconstruction histogram to identify potential anomalies in the training set. Then, we leverage them to enhance the discriminative potential of the model. This contribution has been realized with RADON (Robust Autoencoder with Dynamic Outlier filteriNg).

In a second step, we propose to harness the power of variational autoencoders and normalizing flows to improve the anomaly estimation process. The thresholding criterion on the reconstruction error histogram is replaced by a statistical modeling thanks to the Extreme Value Theory (EVT). The combination of these contributions leads to our model GRAnD (Generative Robust autoencoder for unsupervised Anomaly Detection).

Finally, we propose a third contribution, RESIST (Robust transformEr developed for unSupervised tIme Series anomaly deTection), which relies on sequence-to-sequence models, and in particular Transformers, to model the temporal dependencies between tokens network flow sequence and detect any contextual and collective deviation. The impact of contaminants during learning is significantly mitigated thanks to a Siamese architecture and a robust objective function. Indeed, infrequent and atypical observations have less weight than common data and have less influence on the optimization of our model parameters.

We demonstrate the advantage of our methods over classical approaches of the literature, and

complete the performance characterization with ablation and sensitivity studies. These experimental analyses are based on 4 public datasets, NSL-KDD, MedBIoT, CICIDS17, and TON-IoT, including network traffic metadata from real IoT devices.

Keywords*: Anomaly Detection, IoT Devices Monitoring, Network Traffic Analysis, Machine Learning, Robust and Unsupervised Learning, Self-supervised Learning, Artificial Neural Networks, Autoencoders, Variational Autoencoders, Normalizing Flows, Transformers, Unimodal Thresholding, Extreme Value Theory.*

Résumé

Le contexte de cette thèse est la gestion des équipements connectés et installés chez les clients, alias Device Management (DM), assurée par Orange, un opérateur de télécommunications multi-services. Particulièrement, l'objectif du DM est de s'assurer que les équipements connectés des clients fonctionnent de manière nominale. Le DM couvre, entre autres, les opérations de diagnostic, d'assistance aux clients, de configuration de service, et de maintenance des équipements connectés. Avec le développement de l'Internet des Objets (Internet of Things, IoT), la gestion des réseaux locaux des clients, alias Local Area Networks (LANs), devient plus complexe [18]. En effet, des nouveaux objets connectés hétérogènes, et qui fournissent divers services, sont introduits dans les réseaux domestiques. Ces équipements peuvent dépendre de solutions de DM différentes : il n'est plus possible pour l'opérateur de diagnostiquer l'ensemble des équipements du LAN. Orchestrer ces objets hétérogènes, assurer leur fonctionnement nominal, et diagnostiquer les équipements dysfonctionnels deviennent un enjeu important pour les opérateurs de télécommunications et pour les fournisseurs de services.

Cette thèse étudie la détection non-supervisée et robuste des anomalies à partir du trafic réseau des équipements connectés, appliquée à la supervision des objets IoT. Nous proposons d'analyser le trafic produit et ingéré par chaque objet IoT d'un réseau local, accessible au niveau de la passerelle Internet de l'opérateur, afin de détecter tout équipement défaillant et dysfonctionnel. Le principe est de modéliser, en un premier lieu, le comportement nominal d'un équipement connecté à partir des métadonnées extraites de son trafic réseau. En effet, les objets IoT participent à un nombre de services limité, avec une activité réseau prévisible [182], à l'inverse de smartphones et autres PCs. Ensuite, tout comportement atypique, i.e., toute déviation significative par rapport au comportement attendu par le modèle, est considérée comme une anomalie. Vu l'hétérogénéité des équipements à diagnostiquer, et la diversité des anomalies à détecter, nous optons pour des approches d'apprentissage automatique semi-supervisé et/ou non-supervisé, notamment des Autoencodeurs et des Transformeurs, pour modéliser le comportement nominal d'un équipement IoT et pour détecter les événements anormaux et irréguliers dans son trafic. On peut alors envisager de déployer ce service de détection d'anomalies dans le LAN sans aucun besoin d'intervention humaine lors du traitement des données, seulement lors du traitement des alertes.

Pour modéliser la norme, les approches classiques de détection d'anomalies entraînent un Autoencodeur à reconstruire les données nominales, en minimisant un critère d'erreur entre les données initiales et celles produites en sortie du réseau. Comme les nouvelles observations anormales sont structurellement différentes, leur reconstruction est accompagnée par une significative perte d'information, avec une large erreur de reconstruction. Toutefois, la constitution d'une base d'apprentissage qui ne contient que des données nominales reste coûteuse, chronophage, et parfois infaisable lorsque l'anomalie n'a pas encore été identifiée par les experts. Ainsi, nous avons cherché à développer des Autoencodeurs robustes, c'est-à-dire capables de modéliser la norme, même si la base d'apprentissage est contaminée par des anomalies.

Dans cette thèse, nous proposons trois contributions. Les deux premières contributions se focalisent sur la détection robuste d'anomalies ponctuelles et la troisième étend le champ de notre

étude aux anomalies contextuelles et collectives dans les séries temporelles.

Dans un premier chapitre, nous nous intéressons à la détection non-supervisée des anomalies avec des Autoencodeurs robustes. Notre étude de la littérature montre que même si les *Robust Deep Autoencoders (RDAs)* [292] classiques sont robustes aux anomalies contaminantes, ils sont significativement sensibles à des hyperparamètres prédéfinis. La sélection de ces hyperparamètres nécessite l'accès à une base de données de validation qui contient des données étiquetées. Pour pallier à ce problème, nous proposons une nouvelle stratégie d'apprentissage qui analyse de façon incrémentale la distribution des scores de reconstruction afin d'opérer une auto-supervision. Nous estimons dynamiquement des seuils à partir de l'histogramme de reconstruction des données d'apprentissage, afin d'identifier les potentielles anomalies contaminantes. Enfin, nous les exploitons pour renforcer le potentiel de discrimination du modèle. Cette contribution a été concrétisée avec *RADON (Robust Autoencoder with Dynamic Outlier filteriNg)*. Nous démontrons l'avantage de cette contribution par rapport aux approches de la littérature, avec une étude expérimentale s'appuient sur deux jeux de données publics, NSL-KDD [196] et MedBIoT [109], comprenant des métadonnées du trafic réseau des équipements IoT. Contrairement aux RDAs, notre contribution est plus robuste au choix de ses hyperparamètres vu que les performances restent relativement stable suite à une variation de son hyperparamètre λ qui contrôle la sparsité de la projection.

Dans un deuxième chapitre, nous proposons d'exploiter la puissance des autoencodeurs variationnels et des *normalizing flows* pour améliorer le processus d'estimation des anomalies. Notre étude de l'état de l'art des approches robustes révèle leur sensibilité à la sélection des hyperparamètres. Nous proposons une deuxième contribution qui combine autoencodeurs variationnels et des *normalizing flows* d'un part et la théorie des valeurs extrêmes d'autre part. Nous entraînons ces modèles génératifs à extraire une représentation du processus de génération de données de manière probabiliste, c'est-à-dire inférer les distributions de probabilité des observations d'apprentissage dans les espaces latente et de sortie. Pour adapter le critère de rejet des contaminants à cette modélisation probabiliste, nous proposons d'utiliser la théorie des valeurs extrêmes. La combinaison de ces contributions aboutit à notre modèle GRAnD (Generative Robust autoencoder for unsupervised Anomaly Detection). Nous évaluons le potentiel de cette contribution sur les deux jeux de données NSL-KDD et MedBIoT, et notre étude expérimentale souligne confirme l'avantage de GRAnD par rapport aux méthodes concurrentes de la littérature [10].

Enfin, comme les deux premières contributions ne détectent que des anomalies ponctuelles, nous cherchons à explorer de nouvelles architectures neuronales pour la détection d'anomalies contextuelles et collectives. Nous optons pour des approches basées sur les Transformeurs, du fait de leur capacité à modéliser efficacement de longues séries temporelles. En effet, les Transformeurs sont des réseaux de neurones artificiels dits de *séquence-à-séquence* (*sequence-to-sequence*), qui peuvent modéliser les interdépendances contextuelles d'une séquence grâce à un module d'attention. Ce module apprend à se concentrer spécifiquement sur les données contextuelles pertinentes afin de modéliser les relations qui lient chaque observation avec son contexte. Contrairement aux réseaux de neurones récurrents (Reccurent Neural Networks RNNs), les Transformeurs ne traitent pas les données de façon séquentielle. Ainsi, l'apprentissage de ces modèles peut être parallélisé et devient plus rapide, grâce à l'utilisation des cartes graphiques. Plusieurs publications récentes ont proposé des détecteurs des anomalies en se basant sur les Transformeurs [260]. La détection d'anomalie avec un Transformeur implique également une architecture de type autoencodeur, à la distinction que les données traitées sont des séries temporelles. L'encodeur encode les echantillons d'une séquence des données avec une représentation vectorielle qui considère les dépendances temporelles par paires, et le decodeur utilise la sortie de l'encodeur pour reconstruire la séquence initiale de façon autoregressive. Cependant, ces méthodes basées sur les Transformeurs ne sont pas robustes. Elles présupposent que seules des données nominales sont utilisées lors de l'apprentissage. Pour

lever ce verrou, nous proposons une troisième contribution, RESIST (Robust transformEr developed for unSupervised tIme Series anomaly deTection), qui adapte l'apprentissage d'un Transformeur autoencodeur pour modéliser les dépendances temporelles entres les tokens d'une séquence de flux réseaux et détecter toute déviation contextuelle et collective. L'impact des contaminants lors de l'apprentissage est significativement atténué grâce à une architecture Siamoise et une fonction objective robuste. En effet, les observations peu fréquentes et atypiques ont un poids moins important que les données communes et ont moins d'influence sur l'optimisation des paramètres de notre modèle. Nous démontrons l'avantage de notre méthode par rapport aux approches de la littérature [279], et complétons la caractérisation des performances par des études d'ablation et de sensibilité. Cette analyse expérimentale s'appuie sur deux jeux de données publics, CICIDS17 [237], et TON-IoT [179], comprenant des métadonnées du trafic réseau des équipements IoT contenant des anomalies contextuelles et collectives.

En outre, nous présentons une comparaison expérimentale entre nos trois contributions sur le jeu de données CICIDS17. RESIST atteint les meilleures performances, avec une meilleure précision de détection, mesurée à l'aide de la métrique AUROC, qui signifie l'aire sous la courbe ROC (Receiver Operating Characteristic), et une durée d'apprentissage significativement plus courte. Cette meilleur performance est cependant obtenue au prix d'une sensibilité à un hyperparamètre : *le paramètre de l'échelle* de la fonction objective robuste, qui contrôle l'amplitude du gradient [30].

Dans le dernier chapitre, nous proposons une synthèse des trois contributions de la thèse. Une étude de faisabilité démontre la validité de notre approche dans le contexte industriel de l'étude. Nous illustrons l'arcitecture d'une chaine de traitement de bout en bout qui collecte les flux réseau, extrait les métadonnées et les enregistre dans une base de données. Un modèle robuste est entraîné pour chaque équipement, et utilisé pour détecter des anomalies en temps réel. Ces anomalies sont enfin visualisées dynamiquement sur un tableau de bord permettant de superviser l'ensemble du réseau local. Finalement, nous discutons les limitations de notre recherche et nous presentons des potentielles perspectives, utiles pour des activités de recherche futures sur la détection d'anomalies et l'analyse du trafic réseau.

***Keywords**: Détection d'Anomalies, Supervision des Equipements Connectés, Analyse du Trafic Réseau, Apprentissage Automatique, Apprentissage Robuste et Non-supervisé, Apprentissage Auto-supervisé, Réseaux de Neurones Artficiels, AutoEncodeurs, Autoencodeurs Variationnels, Normalizing Flows, Transformeurs, Seuillage Unimodal, Théorie des Valeurs Extrêmes.*

Publications Associated With This Thesis

Publication In Peer-Reviewed Journals (Under Submission)

1. Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "Robust Unsupervised Time Series Anomaly Detection with Transformers". In: *Neurocomputing (under submission)*. 2022

Publication in Peer-Reviewed Conferences And Workshops

1. Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "RESIST: Robust Transformer for Unsupervised Time Series Anomaly Detection". In: *7th Workshop on Advanced Analytics and Learning on Temporal Data (AALTD) of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*. 2022, p. 16

2. Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "Robust Variational Autoencoders and Normalizing Flows for Unsupervised Network Anomaly Detection". In: *International Conference on Advanced Information Networking and Applications (AINA)*. Springer. 2022, pp. 281–292

3. Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "RADON: Robust Autoencoder for Unsupervised Anomaly Detection". In: *2021 14th International Conference on Security of Information and Networks (SIN)*. 2021, pp. 1–8

4. Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "Network Traffic Modeling For IoT-device Re-identification". In: *2020 International Conference on Omni-layer Intelligent Systems (COINS)*. 2020, pp. 1–6

Patent Applications at the National Institute for Intellectual Property

1. Naji Najari, Samuel Berlemont, and Grégoire Lefebvre. *Anomaly Detection for the Applicative Maintenance of IoT Devices.* FR2205936, 2022

2. Samuel Berlemont, Grégoire Lefebvre, and Naji Najari. *Remote support solution to aid in the diagnosis of identification of dysfunctional equipment not administered by the service provider.* FR2012433, 2021

1 Introduction

"To study the abnormal is the best way of understanding the normal."
William James

Contents

1.1 Network Monitoring And Anomaly Detection 10
1.1.1 Device Management and Monitoring 11
1.1.2 Internet of Things 12
1.1.3 Limitations of Traditional Monitoring Systems 14
1.1.4 Anomaly Detection for IoT Device Monitoring 15
1.1.5 Privacy And Ethical Principles 15
1.2 Objectives and Contributions 16
1.3 Overview of the thesis 17

Anomaly Detection (AD) is a fundamental research problem that has a long history and is routinely applied in almost all scientific fields and applications. This cross-domain interest is justified by the fact that discovering anomalies helps to understand innovative insights about novel phenomena that are not well explained with existing knowledge and beliefs. According to Kunh [141],

> "Discovery commences with the awareness of anomaly, i.e., with the recognition that nature has somehow violated the paradigm-induced expectations that govern normal science. It then continues with a more or less extended exploration of the area of anomaly. And it closes only when the paradigm theory has been adjusted so that the anomalous has become the expected".

Besides the scientific discovery of unknown phenomena, anomalies may reveal important assets about the data. For instance, network anomalies indicate an atypical behavior that may threaten the network integrity, e.g., network attacks or failures. Credit card fraud detection allows bankers to block fraudulent transactions. An anomalous Magnetic Resonance Imaging (MRI) image indicates a potential disease that should be cured. In some cases, anomalies reveal positive phenomena, e.g., a sudden business gain resulting from a big sale event.

The large landscape of AD literature has evolved over time, with many advances in the last few decades thanks to the advent of machine learning and computer hardware [6]. Tuning these approaches for optimal performance requires a comprehensive understanding of their fundamental concepts and their applicability scope.

The purpose of this thesis is to explore the detection of anomalous behaviors of Internet of Things (IoT) devices from their network traffic. A wealth of books, surveys, and studies have been proposed to address this problem [6, 67, 43, 264, 265]. However, most of these approaches present some shortcomings that limit their applicability in real-world environments.

This first chapter aims to present the context of our study and provide an overall overview of AD applied to IoT device monitoring. Section 1.1 presents the thesis context and defines the used terminology of Device Management (DM), monitoring, and IoT. We conclude this first part with the main limitations of traditional Intrusion Detection Systems (IDSs). Section 1.2 precises the objective of the present thesis as well as our contributions to the problem of robust unsupervised AD. Finally, Section 1.3 depicts the global structure of the thesis, with a brief description of each chapter.

1.1 Network Monitoring And Anomaly Detection

This Ph.D. thesis is part of Orange research activities on the management of IoT devices installed at customer smart homes. In particular, Orange aims to ensure that customers' connected devices are operating nominally and detect any malicious and non-malicious anomalous network activity as soon as possible.

This section presents the thesis context for network traffic AD. First, we introduce DM objectives and we define the classical Orange DM monitoring process. Second, we present the new challenges introduced with the IoT and discuss the limitations of classical monitoring systems, mainly traditional IDSs. Third, we shed the light on the need for more flexible AD tools based on unsupervised machine learning models. Finally, we present the ethical principles that we consider in our research, and which are aligned with Orange's strategy in terms of personal data processing, privacy, and transparency.

A computer network can be defined as "a collection of interconnected hosts, via some shared media which can be wired or wireless." [227]. In the early 1960s, computers were used as

independent individual machines that cannot communicate or exchange data with each other. In the late-1960s, ARPANET [2] was introduced as a small network of interconnected computers. The main idea was to enable these computers to communicate by connecting to a special computer, called an Interface Message Processor. This idea was extended to connect computers of two or multiple different networks in 1972 [271]. A set of protocols were introduced later to organize the communication between networks, e.g., IP, User Datagram Protocol (UDP), and Transmission Control Protocol (TCP) protocols. They define fundamental concepts such as the structure of the unit of the data to be transmitted and the mechanism for routing network packets.

Networks can be either Local Area Networks (LANs), which are defined over a small geographic area, e.g., an office or a user home, or Wide Area Networks (WANs), comprising distant interconnected devices across cities or towns. Nowadays, the Internet consists of a network of networks that interconnects a large number of computers and connected devices from the entire world. New networks and computers are continuously integrated into this large web.

Regarding this complex structure, most end users rely on Internet Service Providers (ISPs) to define and manage their network of devices. This functionality is mainly ensured by DM, which will be developed in the next section.

1.1.1 Device Management and Monitoring

DM allows ISPs and telecommunication operators to ensure that the devices of end users are functional and to prevent potential anomalies that disrupt the provided services. To this end, a set of operations are remotely executed to continuously monitor the customer devices. DM monitor customer LANs using the following four main operations:

- *Provisioning* installs and manages the configuration files that determine the device functionalities and enable services that the end-user is subscribed to;
- *Maintenance* updates the outdated software of connected devices;
- *Assistance* aims to correct service malfunctions and failures reported by customers, using diagnostic tasks to detect potential root causes and applying corrective measures in a timely manner;
- *Tracking* logs device state information and alarms declared by the managed devices.

DM is a critical asset for both end users and Orange, ISPs and Telecommunication operators in general. From a customer point of view, maintaining their own LAN infrastructure and troubleshooting emerging issues is not intuitive. A study by Grinter et al. [107] investigated the complexities of managing a LAN in the United Kingdom and the United States. It deduced that "the work required to implement home networking presents significant challenges" for householders. The difficulty of the own configuration of the household network was reported in numerous studies [281, 282].

From an ISP perspective, DM is a main asset that controls Quality of Service (QoS) and improves user experience. Indeed, when customers encounter an issue in their LAN, they seek assistance from Orange. The diagnosis and the troubleshooting of the anomaly require expert analysis of the network and can be costly and time-consuming.

In the last decade, the user LANs have evolved from a few interconnected computers to a large collection of connected devices such as smart door locks, connected light bulbs, smart printers, and connected coffee machines. A study [226] estimates the number of connected devices worldwide will exceed 75 billion at the end of 2025, which approximately corresponds to more than nine connected devices per person. It is also estimated that every second, 127 new devices are connected to the Internet [275]. Managing such complex networks that integrate multiple heterogeneous devices is significantly more challenging for ISPs and Telecom operators. The following section introduces the challenges introduced by the IoT.

1.1.2 Internet of Things

The term “Internet of Things” was introduced by Ashton in 1999 [4] and refers to Internet-connected objects that, unlike traditional computers, can sense the world with less dependency on human interventions. Since then, more definitions have been proposed for this novel concept. Recently, the Internet Engineering Task Force (IETF) has defined IoT as follows: “The Internet of Things (IoT) is the network of physical objects or "things" embedded with electronics, software, sensors, actuators, and connectivity to enable objects to exchange data with the manufacturer, operator, and/or other connected devices” [254].

The advent of the IoT in recent years has had a profound impact on almost every industrial sector, e.g., Smart Home, Healthcare, Agriculture, and Industry 4.0. The key asset of this emerging paradigm is that it allows the development and the monitoring of an environment not completely dependent on human command. These smart environments allow the expansion of new services that improve human daily life and create large business values. From the end-user perspective, IoT-based services will interfere on several sides, e.g., health monitoring, assisted living, and work facilities. From the business actor and enterprise point of view, IoT plays an increasingly important role in smart manufacturing, intelligent transportation, and even smart city development [20].

There is a growing concern about the reliability of IoT-based networks in the network community, as such ecosystems are susceptible to diverse anomalies and threats. Indeed, the IoT introduces new challenges compared to traditional computer networks.

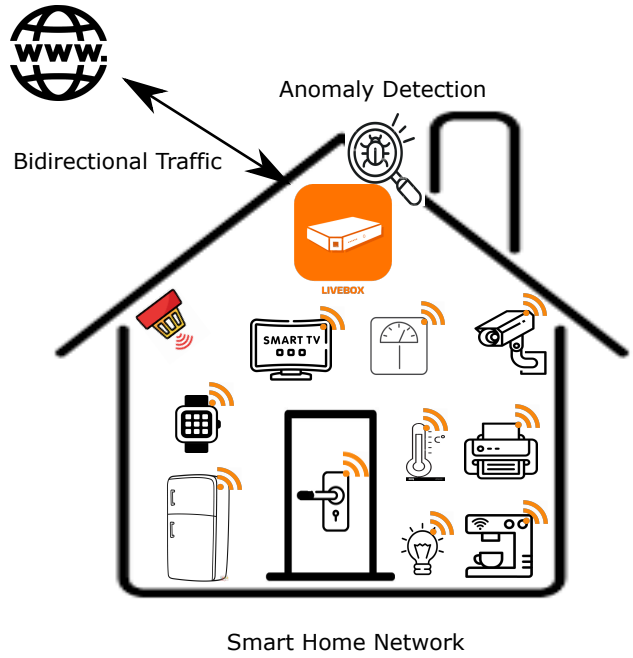


Figure 1.1: Illustration of a smart home network.

First, the IoT introduces a high diversity of heterogeneous connected devices, as illustrated in Figure 1.1. This large fleet of devices operates over a plethora of network protocols, is developed by different manufacturers, and has diverse life-cycle profiles. While some devices, have short life-cycles, evolve quickly, and should be replaced frequently, e.g., vocal assistants, network gateways, and smart watches, other ones are designed for long-term usage, e.g., healthcare sensors. The scale and heterogeneity of IoT devices challenges traditional monitoring tools and require more flexible solutions. Besides the heterogeneity and the scalability, a more challenging issue is related to the service and the connectivity interdependency of IoT devices. For example, a connected bulb activity may depend on a Wi-Fi extender, a gateway, and a vocal assistant. Currently, these devices

are managed by multiple operators and manufacturers with siloed management tools. However, a global and unified vision of these interdependent device behaviors is of paramount importance to provide useful insights into the root causes of the underlying dysfunctions.

Second, unlike traditional computers, most IoT devices, e.g., sensors, are developed to provide specific functionalities that do not require large resource consumption. These devices are generally constrained, with limited Central Processing Unit (CPU), memory, and power resources [205].

Third, the IoT is continuously growing, where the main manufacturers' objective is to quickly release low-cost devices [194]. This "rush-to-market strategy" leads sometimes to the release of vulnerable devices with inherent security issues. Such devices become the weakest point of the network and the target of malicious outsider attacks [194]. A recent study conducted on 10 common studies has discovered around 250 threats, such as running outdated obsolete firmware, having open telnet ports, and using clear-text protocols for sensitive data exchange [85]. A Zscaler report [189] has analyzed around 56 million IoT device transactions of over 1000 enterprise networks and found that 91.5% of these data transactions were unencrypted.

Finally, communication is key in the IoT. Despite the heterogeneity and the profile diversity, the network devices are constantly communicating to exchange information. For example, smoke sensors can exchange the recorded data with the locks to unlock the doors in case of fire [286]. This cross-device communication can be exploited by malicious users to spread indirect and reflective attacks more rapidly [241].

All the aforementioned challenges make IoT devices more prone to anomalies and their monitoring significantly more complex. There is a huge concern reported by different network actors about the security and reliability of IoT networks, because failures in such networks are more dangerous and disastrous than traditional Information Technology (IT) systems. To begin with, IoT devices are integrated into everyday life to closely sense and control end user daily activities. Wearable devices monitor human activities throughout the entire day, ranging from burned calories to steps walked, real-time heart rate, and even the quality of sleep [166]. Some health monitoring devices, e.g., blood pressure and sugar monitors, exchange user personal data using clear-text protocols [277]. All the data collected by such devices can be collected, analyzed, and exploited, potentially without clear and transparent user consent. This introduces many privacy and ethical concerns that are still debated among the network community. Besides, IoT devices are conceptually designed to act on the local environment without explicit user intervention. A compromised device may result in severe detrimental consequences. One of the worst real-life examples of such consequences was reported in 2015, when hackers exploited a firmware update vulnerability and totally controlled a smart Jeep car [155]. They were able to remotely control the in-car temperature, the speed, and the braking system, and they finally veered it off from the road. Another example was the zero-day vulnerability of St. Jude Medical's implantable cardiac devices that allowed hackers to access these devices, control the heart pacing, and even execute some electrical shocks [278]. Finally, a peculiarity of IoT devices is their large scale, with innumerable exposed interconnected nodes. This worsens the situation with such a massive volume of precarious targets. Malicious activities can spread faster on a large scale, compared to traditional computer networks that comprise a few reliable and homogeneous machines. One of the most popular large-scale IoT-based attacks was discovered in early 2014 [208], where over 100000 IoT devices e.g., TVs and fridges, was weaponized to implement Distributed Denial of Service (DDoS) attacks. These attacks targeted customers and enterprises worldwide and are estimated to cost 30,000 $ per hour for compromised victims [158]. In 2019, Kaspersky detects over 100 million attacks that aim to compromise IoT devices and to use them in sophisticated botnets [123]. Besides malicious anomalies, numerous network downtimes have been reported due to hardware and software failures, with outdated firmware and battery issues [253].

In summary, the IoT paradigm introduces new challenges in the network landscape. Unless meticulously investigated, these challenges result in detrimental effects for both end users as well organization and ISPs. One intuitive solution consists in applying classical monitoring tools used

in traditional IT networks to this new ecosystem. However, this strategy is inherently flawed and sub-optimal.

1.1.3 Limitations of Traditional Monitoring Systems

This section objective is to introduce traditional monitoring systems, mainly IDSs, and to discuss the limitations of such tools in IoT networks.

IDSs are among the most conventional monitoring tools used to protect computer networks from intrusions and threats that endanger their nominal functioning. Ghorbani et al. [100] define intrusion detection as "the process of identifying and (possibly) responding to malicious activities targeted at computing and network resources". IDSs are used along with firewalls as a two-line defense strategy to defend the network from malicious attacks. Besides detecting hostile network activities, IDSs can provide monitoring reports to ease the problem diagnosis by network administrators [92].

IDSs can be categorized into two main classes:

- Host Intrusion Detection System (HIDS): a HIDS monitors the behavior of a single host, i.e., one computer, and detects any local anomalous activity relative to this host. These systems are sometimes called *agent-based* IDS [92] because they must be installed on the monitored device. Once installed, they analyze the host log files, e.g., operating system and application log files, and assess the host security.

  Although the popularity of such systems in the network community, this agent-based system applicability in the IoT paradigm is limited. Indeed, we mentioned previously that IoT devices are generally constrained with limited CPU and memory resources. Integrating such resource-consuming tools in these frugal environments is not an option. This observation was highlighted by Yu et al. [286], with the following statement: "even antivirus systems for embedded systems, such as Commtouch Antivirus [72], require 128 MB RAM, while most IoT devices use single-thread microcontroller (8051, MSP430, ATMEL series) with 2 MB RAM";

- Network Intrusion Detection System (NIDS): an NIDS analyzes the whole network to detect any atypical traffic activity and protect all network endpoints. Unlike HIDSs, these systems provide a global view of the network as well as the interactions between nodes. They can be easily added to existing networks and only a few NIDSs are required to monitor large networks [92]. The key idea of NIDSs is to passively inspect the network traffic exchanged between the devices to identify any unusual communication pattern. A typical configuration consists in mirroring, i.e., streaming a copy, all the network traffic that go through the gateway to one specific NIDS, e.g., *Snort* [243] or *Zeek* [255].

Owing to their flexibility and ease of use, NIDS are more suitable for application in IoT networks. To detect anomalous traffic, most NIDSs rely on signature-based methods, a.k.a., knowledge-based and misuse detection [92]. They identify anomalies based on a predefined set of rules that models known attack signatures. When a network traffic activity matches with one signature, an alarm is generated. These systems have shown promising results in computer networks, which run a limited range of operating systems and undergo frequent security inspections by experts. However, the situation is completely different in the IoT, with a significant increase in the occurrence of new unknown anomalies, e.g., zero-day attacks. According to [289], around 80% of network vulnerabilities reported in early 2020 belong to zero-day attacks. These novel anomalies generally go under the radar of these traditional network security solutions. An even more critical issue is due to the diversity of IoT device profiles and their dynamic behaviors. Since every IoT system is different, with different specifications depending on the manufacturer and the application context, the defense strategy must be adapted according to device specificities. It requires an extensive untenable effort to define a set of signatures per device type. Furthermore,

IoT device behaviors are dynamic as they change with the operating context [286]. The continuous update of the signature-based IDS database can be costly and does not enable a real-time action.

Alternatively, anomaly detection-based methods have been proposed to mitigate the aforementioned limitations.

1.1.4 Anomaly Detection for IoT Device Monitoring

Many emergent studies proposed to leverage IoT systems singularities and proposed anomaly-detection approaches [74, 87]. In fact, unlike general-purpose computers, IoT devices are designed to perform one or some specific functionalities. Hence, it is more straightforward and easier to model their nominal behaviors than encoding all the signatures of the countless anomalies.

The crux of AD, which will be developed further in the next chapter, is to develop a baseline profile of the expected typical behavior of the network. Then, any traffic activity that deviates from this nominal profile is deemed anomalous. A wealth of methods can be used to create the baseline nominal profile (cf. Chapter 2 Section 2.3), ranging from manually developed specifications to statistical and machine learning-based approaches.

The main strength of AD-based IDSs is their ability to detect known as well as unknown malicious and non-malicious anomalies. This strategy is more suitable for the IoT, with predictable device behaviors. It allows the discovery of novel anomalies and can be used to bring to light zero-day attacks [92]. However, the main shortcoming is the potential increase of false positive rates. Dynamic to dynamic nominal behaviors that evolve over time, some novelties may be reported as anomalous unless the expected profile is properly updated. In addition, anomalies may occur during the training phase, i.e., the creation of the normal profile, and result in a skewed model that does not accurately reflect the expected activity. This problem is known as the *sensitivity to training contamination* and will be developed in detail in the remainder of this thesis. Robust unsupervised anomaly detection has been a point of interest to mitigate these limitations and develop reliable and secure networks.

1.1.5 Privacy And Ethical Principles

AD-based methods are data-driven approaches that require collecting a large volume of data, i.e., network traffic data in our specific application. Nonetheless, the use of customer data may introduce some privacy and ethical issues, unless framed through a clear and well-defined data strategy.

First, Orange data processing policy is compliant with European obligations, e.g., General Data Protection Regulation (GDPR). These directives aim to regulate the use of sensitive data and its storage and are applied to all companies and organizations that collect data from European users. In fact, in order to ensure legal compliance, companies undertake a comprehensive set of measures that filter out user personal data, e.g., the content of network traffic and IP addresses (cf. Section 2.1). The comprehensive list of rules defined by GDPR can be found here[1].

Second, besides these obligations, Orange is committed to a set of ethical principles in order to protect the personal data and privacy of their customers. Indeed, Orange maintains trusted relationships with its customers and partners. As such, users must be assured that their data are secure and used responsibly. To this end, it is required to maintain a balance between collecting data to provide novel services, and keeping user trust. Among the privacy measures ensured by Orange, the data should be collected after user consent and with a clear and transparent processing process. The data are only kept for a predefined length of time needed to fulfill the provided service and can be subject to user subscription, which immediately stops any further data analysis. In

[1] https://gdpr-info.eu/

addition, Orange ensures that the user collected data remains secure and confidential[2].

Building on this vision, we discuss in this section the ethical aspects involved in our research problem: network traffic-based AD. In particular, we articulate our data analysis on the following principles, related to two aspects:

- Data privacy: the data should be processed in the customer premises, i.e., LAN, to minimize personal data transfer to outsiders, i.e., company servers. Indeed, user data, which may include personal and sensitive information, should be kept inside the user's private sphere. In contrast, the computational analysis should be moved from Enterprise servers to user LAN. Besides, the local processing aspects, the careful selection of the metadata that should be analyzed is critical. the content of network traffic packets may include user personal information, e.g., healthcare data or confidential email content. In our study, we exclude certain data inspection strategies, i.e., deep packet inspection (cf. Section 2.1.1.1) that involve the analysis of the *content* of the network traffic, i.e., the payload, as this violates the privacy of users and uncovers their personal data [121];
- Model privacy: the training of the AD models are performed in LANs, as the data should be kept inside the LAN. In this case, it is not conceivable to annotate the learning data under a supervised training scenario. Instead, we opt for unsupervised machine learning approaches. As such, Orange can flexibly deploy the AD service in the customer LAN without any need for human intervention to annotate the collected data.

These principles will guide our AD strategy throughout the present study. In the following, we present this thesis objective as well as our three main contributions.

1.2 Objectives and Contributions

This thesis focuses on robust unsupervised anomaly detection, i.e., robust representation learning under contaminated data, using artificial neural networks applied to network traffic monitoring. Indeed, these models, and particularly AutoEncoders (AEs), have shown great potential in modeling complex high-dimensional data and have become a staple baseline in AD. One main limitation of such classical approaches is that their training requires access to anomaly-free training data, with completely nominal instances. This thesis aims to relax this assumption by investigating robust AEs, which can learn a robust data projection transformation that is not biased by training outliers.

The present problem requires developing two crucial aspects: (1) a rejection strategy that filters training anomalies and (2) an optimization process that finds the optimal parameters of the model, which accurately reflects the data properties.

We propose 3 main contributions in this thesis:

- Robust Autoencoder with Dynamic Outlier filteriNg (RADON) We first propose a robust autoencoder that can learn a robust representation of the nominal class. Our unsupervised method automatically filters training anomalies thanks to the iterative unimodal thresholding of the reconstruction error histogram. To further improve the AD performance, its training was enhanced to simultaneously minimize the reconstruction scores of nominal instances and maximize that of the filtered anomalies. In summary, we propose a training strategy that allows RADON to learn a robust projection in a latent space, where nominal data are similar to their reconstruction, but also where anomalies are explicitly badly reconstructed;

[2]https://www.orange.com/en/privacy-notice-protecting-your-personal-data

- Generative Robust autoencoder for unsupervised Anomaly Detection (GRAnD) We propose to leverage the potential of Variational Autoencoder (VAEs) and Normalizing Flows (NFs) to further improve the AD performance. Indeed, these models are used to provide a probabilistic reformulation of the data reconstruction errors. In particular, we propose to model the distribution of the training reconstruction scores using the Extreme Value Therory (EVT) to define an automatic rejection criterion of extreme values. All in all, we introduce a training strategy that alternates between filtering outliers contaminating the training dataset and learning a robust representation of the norm. Unlike recent robust generative methods, our approach makes no assumption about the anomaly distribution, or about the fraction of training outliers;
- RESIST We propose a third contribution tailored to contextual and collective AD in time series. We introduce a robust learning strategy that trains a Transformer to model the nominal behavior of the network activity. Relying on a contrastive learning-based robust loss function and a Siamese encoder-decoder architecture, RESIST automatically down-weights atypical corrupted training data, to reduce their impact on the training optimization.

1.3 Overview of the thesis

This section depicts an overview of each chapter of this thesis.

- Chapter 2 presents a global overview of the AD problem. In particular, this problem is reviewed under three dimensions: (1) network traffic data, (2) anomaly properties and categories, and (3) detection approaches. First, we present the IP network traffic data characteristics and the relevant metadata that can be used in our study. Then, we propose to define the connotation of an *anomaly* with a common that will be retained in the remainder of the thesis. Next, we discuss the different categories of anomalies that are defined in the literature. We particularly distinguish three groups: punctual, contextual, and collective anomalies. Finally, we review the different approaches to AD. We start by surveying punctual AD and we then extend our survey to contextual and collective AD.
- Chapter 3 presents our first contribution to the problem of robust unsupervised AD: RADON. After formalizing the problem that we address, robust unsupervised AD in non-sequential data, we introduce the most relevant methods of the literature, designed to solve this problem. We mainly review three dimensionality-reduction-based anomaly detectors: Principal Component Analysis (PCA), Robust Principal Component Analysis (RPCA) [54], and RDA [10]. Then, we present our first contribution of the thesis: RADON, which seeks to improve the performance of the classical RDAs. We finally depict the experimental protocols and results. We highlight that the contributions of this chapter have been published in [183].
- Chapter 4 presents our second contribution GRAnD, which extends RADON learning capacity using generative AEs, and particularly VAEs and NFs, and EVT. First, we start by introducing the theoretical background of the variational-based unsupervised AD, by developing the the variation inference setting, with VAEs and NFs, and the uncertainty estimation with EVT. Second, we outline the state-of-the-art robust generative AE: the Robust Variational Autoencoder (RVAE) [10]. We discuss the strengths and weaknesses of this approach and we present our second contribution: GRAnD. This contribution is compared against state-of-the-art methods with a set of experimental protocols. We discuss the experimental results and we conclude this chapter with the main conclusions and further research perspectives. The contributions of this chapter have been published in [186].

- Chapter 5 presents our third contribution RESIST. Unlike the first two contributions, this approach is particularly designed to detect contextual and collective anomalies in sequential data, i.e., time series. First of all, we introduce the peculiarities of AD in time series. following up on this problem formalization, we survey the state-of-the-art method tailored to solve the present problem. We identify the main limitations of this method and we then propose our third contribution RESIST. We experimentally validate this contribution and we conclude this chapter with an empirical comparison between the three contributions of this thesis.
- Chapter 6 concludes this thesis with an overview of our contributions, the limitations of our approaches, and the potential perspectives of future works. In addition, we discuss the ethical aspects involved in this domain, and how they impact the eventual operationalization of our propositions.

2
Anomaly Detection

"Research is seeing what everybody else has seen and thinking what nobody else has thought." Albert Szent-Györgyi

Contents

2.1 Network Traffic Data 22
2.1.1 IP Protocol Overview 23
2.1.1.1 IP Packet Analysis 23
2.1.1.2 IP Flow Analysis 26
2.1.2 Network Traffic Anomalies 27
2.2 Definition And Categorization Of Anomaly Detection Methods 28
2.2.1 Definition Of Anomalies 28
2.2.2 Types Of Anomalies 29
2.2.3 Generic Categorization Of Anomaly Detection Methods 30
2.2.3.1 Categorization Based On Supervision 30
2.2.3.2 Based On The Detection Strategy 32
2.3 Classical Methods for Anomaly Detection 33
2.3.1 Instance-Based Methods 33
2.3.1.1 Distance-Based Methods 33
2.3.1.2 Density-Based Methods 34
2.3.1.3 Strengths And Weaknesses Of Instance-Based Anomaly Detection 36
2.3.2 Statistical Models 36
2.3.2.1 Parametric Statistical Method 37
2.3.2.2 Non-Parametric Statistical Method 38
2.3.3 One-Class Classification-Based Models 40
2.3.3.1 One Class Support Vector Machine 41
2.3.3.2 Support Vector Data Description 42
2.3.3.3 AD With Classification-Based Approaches 43
2.3.3.4 Strengths And Weaknesses Of Classification-Based AD . . . 44

2.3.4 Neural Network-Based One-Class Classification 45
2.3.4.1 Artificial Neurons And Feed-Forward Neural Networks . . . 45
2.3.4.2 One-Class Neural Networks 46
2.3.4.3 Deep Support Vector Data Description 47
2.3.4.4 AD With One-Class Neural Networks And Deep Support Vector Data Description 47
2.3.4.5 Strengthes And Weaknesses Of One-Class Neural Networks And Deep SVDD 47
2.3.5 Reconstruction-Based AD 48
2.3.5.1 Autoencoders 48
2.3.5.2 Variational Autoencoders And Normalizing Flows 49
2.3.5.3 Anomaly Detection With AEs, VAEs, and NFs 51
2.3.5.4 Sequence-to-sequence Models And Transformers 52
2.3.5.5 Anomaly Detection With Sequence-to-sequence Models And Transformers 56
2.3.5.6 Strengths And Weaknesses Of Reconstruction-Based AD . . 56
2.4 Conclusion 57

In today's world of digitization, technological advances have favored the collection of large and continuously growing volumes of data in diverse application fields. Service providers aim to leverage available sources of data to offer new services and improve provided features. In particular, the advent of the IoT and Industry 4.0 allowed manufacturers to rely on connected objects to continuously monitor provided services and improve QoS, Quality of Experience (QoE), and security [9]. Nevertheless, experts can no longer manually process this ever-growing volume of the generated data, resulting in a need for automating data processing and analysis. Detecting anomalous behaviors is among the most important steps in data analysis. This task allows the prevention of system failures, security breaches, data leakage, and losses of money.

AD has received an increasing attention and has been explored in diverse research domains, including network intrusion detection and cybersecurity, fraud detection, healthcare and medical image processing [57, 93, 201]. For example, detecting anomalous network traffic allows administrators to detect, as soon as possible, any malicious activity that threatens their infrastructures. Detecting these attacks prevents the leakage of sensitive data and protects the network endpoints. In healthcare image processing, detecting outliers allow the discovery of malign diseases. Credit card fraud detection helps bankers identify card theft and malicious transactions.

For these reasons, AD dates back to at least the 19^{th} century [159], serving as the stage for the development of a plethora of methods [63]. The increasing ease-of-access to ever-growing computational methods motivated an interest in machine learning-based methods, whether general [228, 201, 40] or application-specific [93, 74]. The present chapter aims to survey the generic anomaly detectors, highlighting their strengths, weaknesses, and applicability to our problem. As the literature is substantial, and to ease the understanding of this chapter, this problem is reviewed under three dimensions: (1) network traffic data, (2) anomaly properties and categories, and (3) detection approaches.

After a first overview of the present problem, we discuss in Section 2.1 the network traffic data peculiarities. We start with outlining the characteristics of the IP traffic and the relevant metadata that can be used in our study. Section 2.2 defines the meaning of an *anomaly* and categorizes AD methods. Finally, we develop in detail the classical approaches of AD and we discuss the strengths and weaknesses of each approach.

2.1 Network Traffic Data

One of the most critical steps investigated by researchers before the development of an anomaly detection model is to understand and characterize the data. Particularly, for network data analysis, different data sources with different levels of granularity can be used, which control the type of anomalies that could be detected and impact the performance of the underlying model [92]. This section discusses the most common data types used in network monitoring.

One of the earliest protocols used for network monitoring is the Simple Network Management Protocol (SNMP). SNMP was introduced in 1988 to help network administrators and engineers to collect numerous metrics from their fleet of connected devices and gain more visibility of their networks [172]. This UDP-based protocol can passively collect statistical information from different connected devices, such as routers and switches. It extracts some metrics from the monitored devices, such as the CPU and the memory usage, the temperature of the device, and the throughput. SNMP relies on a server-client paradigm, with SNMP *agents* and *managers*. An agent is installed on each connected device to collect and store the metrics. A manager, which is installed on a server machine, queries the agents to collect and visualize the overall data. In addition, this information is locally stored in a database, a.k.a., Management Information Base (MIB), for further processing. The global architecture of SNMP is illustrated in Figure 2.1.

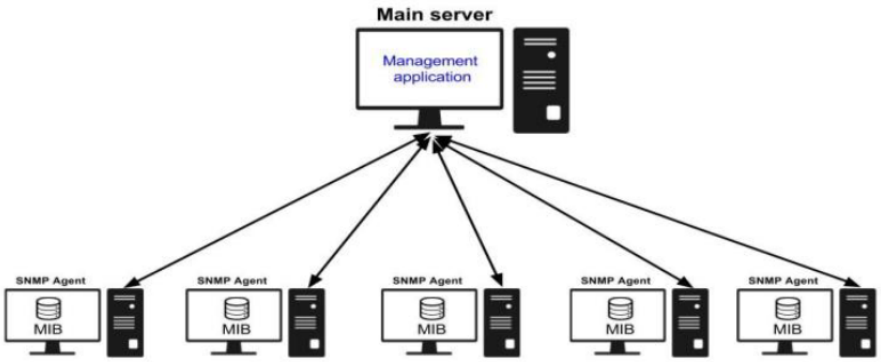


Figure 2.1: Illustration of the SNMP architecture, extracted from [188]

Diverse studies have investigated SNMP metrics to detect any anomalous network activity [188, 117, 122]. Nonetheless, SNMP-based network analysis presents multiple technical limitations. Firstly, this protocol is built on the well-known UDP protocol. While UDP was used to prevent overhead and congestion, loss of important data may occur in a real-world environment [15]. This loss may result in a limited and biased vision of the network activity. Secondly, SNMP can be applied only on SNMP-enabled devices. In the IoT, diverse constrained devices do not integrate this protocol, which results in excluding them from the monitoring process. Lastly, some studies report the scalability issues of this protocol, as it fails to collect and process a large volume of data [219].

Analyzing the IP traffic has been proposed as an alternative to SNMP-based monitoring. With the increasing complexity of LANs and the evolution of the proposed services, researchers looked for other sources of data that provide more detailed information, with more fine-grained attributes. A promising source of data that shows the further potential is IP *flows*. Before presenting the IP flow structure and the useful metadata, let us first provide an overall description of the IP protocol and packets.

2.1.1 IP Protocol Overview

A Network is a set of interconnected devices that communicate using a wired or wireless medium. In this ecosystem of devices, communication is key to continuously sharing and exchanging data and information. The IP protocol is the glue that sticks the network components, as it was designed to structure the transport of network packets from a source to a destination. It defines an abstract way to communicate between devices, regardless of their types and whether they belong to the same network, i.e., LANs, or distinct networks, i.e., WANs.

For this purpose, the International Organization for Standardization (ISO) defined in 1978 a reference model, referred to as Open Systems Interconnect (OSI), which defines the overall structure of an internet packet that can be exchanged between network devices. This reference model organizes the communication into a stack of seven layers, in the following order (from bottom to top): physical, data link, network, transport, session, presentation, and application. Each layer has a specific function, which are described in Figure 2.2, extracted from [274].

2.1.1.1 IP Packet Analysis

Following this layered structure model, a network packet is formed through the encapsulation of a series of fields, corresponding to these layers(cf. Figure 2.3). The overall structure of a network packet contains two main fields: a header, which is an envelope that contains the metadata required

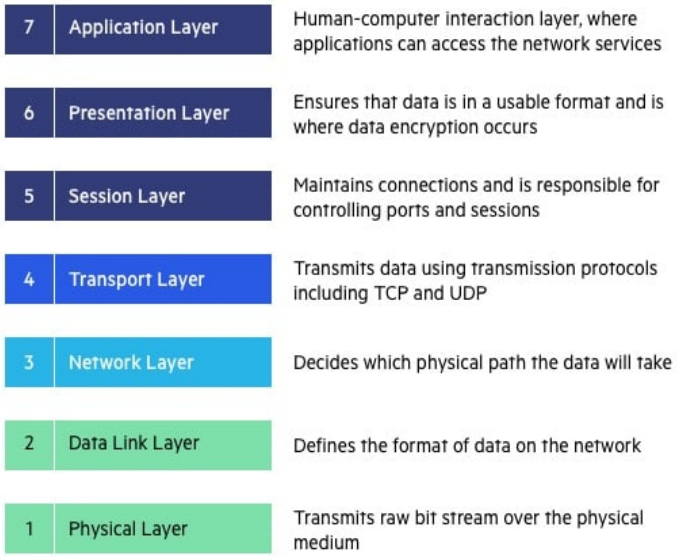


Figure 2.2: The OSI seven-layer model.

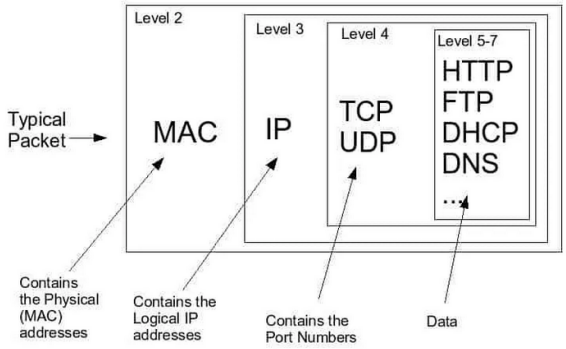


Figure 2.3: The typical structure of a network packet, extracted from [274].

to route the packet in the network, e.g., the source and destination IP addresses, and a payload, which contains the content or the data to be transmitted.

The question now is, which is the useful information that can be extracted and used in our AD task? There exist a dichotomy between two popular strategies used in network packet analysis: Deep Packet Inspection (DPI) and Shallow Packet Inspection (SPI).

Deep Packet Inspection

DPI approaches analyze both the payload and the header of network packets to decide whether the communication is anomalous or not. In particular, it enables ISPs and specific network actors to inspect the content of the data transmitted in the networks, to identify any suspicious and potentially malicious behavior. DPI was initially developed to secure LANs of customers to guarantee that no harmful data have been seeped into the network. It typically relies on comparing the payload patterns with signatures of known attacks and alerts any matched pattern. This classical approach is used by numerous existing IDSs, which are thoroughly reviewed in this study [161].

However, despite their popularity in the network traffic analysis community, this strategy presents various technical and ethical challenges. First, DPI is an intrusive approach that introduces major privacy concerns. In this case, the customer data, which may contain sensitive data, e.g.,

health-related data collected by healthcare monitoring devices, can be inspected by external entities [77]. Many legal provisions protect user privacy, and DPI must conform to these requirements. For example, in the European Union, the GDPR[1] regulates an organization's use of personal data with multiple specifications. Second, there exists a technical limitation that hinders the applicability of DPI in modern networks. Indeed, DPI assumes that the content of the network packet is clearly accessible for inspection. However, encryption has become pervasive in nowadays networks, with over 87% of the whole Internet traffic in 2019 [79]. This means that the data are no longer exchanged with clear-text protocols and it is no longer possible to access the deciphered content of payloads.

Shallow Packet Inspection

Unlike DPI, SPI examines only packet headers, related to the first four layers of the OSI model, to detect anomalous activities. It is less intrusive as the contents of the packets are never inspected.

Header metadata are essential to route the packets between different devices of the network and may convey insightful information about the overall communication behavior. This is especially noticeable in IoT networks, where connected devices are constantly communicating with other ones, e.g., local sensors or remote servers. The communication pattern is critical in determining whether a device is behaving as usual [32]. For example, a door sensor that starts sending numerous voluminous packets, each of which has a large size close to the Maximum Transmission Unit (MTU), can result from a compromised device under a DDoS attack.

Numerous metadata can be extracted from a single network packet. The comprehensive list of the header metadata that can be extracted by Wireshark[2], one of the most popular open-source full-packet capture tools, with over 271000 fields is available online[3]. Some of the most common features are depicted in Table 2.1.

Table 2.1: A list of some IP header-based metadata extracted by Wireshark.

Attribute Name	Attribute Description
len	entire length of the IP packet, i.e., header and payload, in bytes.
ip.src	IP source address of the packet.
ip.dst	IP destination address of the packet.
tcp.srcport	TCP source port of the packet.
tcp.dstport	TCP destination port of the packet.
tcp.windowsize	TCP window size of the packet.

Even though network packets provide a fine-grained vision of the network, analyzing such very comprehensive data is generally costly and time-consuming. Full packet capture and analysis is computationally expensive and can lead to data overload [12]. The real-time analysis of every packet in high-speed networks that contains multiple interconnected devices is impracticable, with high resource consumption and latency. Furthermore, full packet capture provides too comprehensive information, and in some cases, the relevant attributes may be lost among irrelevant details [12].

An alternative strategy with a higher-level-data analysis has been proposed in the literature, which consists in analyzing network flows.

[1] https://gdpr-info.eu/
[2] https://www.wireshark.org/
[3] https://www.wireshark.org/docs/dfref/

2.1.1.2 IP Flow Analysis

A network flow is defined as "a set of IP packets passing through an observation point over a predefined time interval" [71]. The packets that belong to the same flow have a set of common attributes: the source and destination IP addresses, the ports, the protocol, and the Type Of Service (TOS). A flow provides various metadata about the aggregated packets, e.g., byte and packet counts, flow duration, and packet inter-arrival time, which will be detailed later in this section.

In 1996, Cisco pioneered this field by introducing the NetFlow protocol to structure the flow production and processing [116]. NetFlow proposes an exporter-collector architecture: a probe, i.e., exporter, is embedded into the monitored network to collect the packets and create the corresponding flows. These flows are then sent to a NetFlow collector to analyze and visualize the received flows.

Over time, other protocols that propose additional features have emerged such as sFlow and IPFIX [92]. The same principles remain valid, and only slight differences are reported, mainly regarding the flow extraction process. For example, some protocols use a random packet sampling strategy during the flow export phase to reduce the overall delays. In 2004, the IETF decided to standardize the protocol IPFIX and provide formal specifications for a uniform flow structure. The attributes that can be exported in a flow are formalized as Information Elements (IEs) and the exhaustive list of the standard IEs is maintained by the Internet Assigned Numbers Authority (IANA). Overall, an IE is defined by an ID, i.e., a unique number, a name, a length, i.e., the number of bytes required to store the attribute, a type, and a status [18]. It can be extracted from multiple layers of the OSI model: from layer 2 to layer 7 (cf. Section 2.1.1). The comprehensive list of standard IEs can be found here[4] and the most well-known IEs are reported in Table 2.2. This short list is supported by most flow collection tools. As shown in Table 2.2, extracted features can be either numerical or categorical. In addition to these standard IEs, a set of enterprise-specific IEs can be added to the list [116], which allows the definition of use-case-specific attributes.

Table 2.2: A list of the most popular IPFIX Information Elements.

Information Element Name	Information Element Description
flowStartMilliseconds	Timestamp of the first packet of the flow
flowEndMilliseconds	Timestamp of the last packet of the flow
sourceIPv4Address	IP source address of the flow packets
destinationIPv4Address	IP source address of the flow packets
sourceTransportPort	Source port of the transport layer
destinationTransportPort	Source port of the transport layer
protocolIdentifier	IP protocol number
packetDeltaCount	Number of packets aggregated in the flow
octetDeltaCount	Number of octets used in the flow

Network flow-based traffic analysis provides numerous advantages compared to complete packet analysis:

- Lower costs and delays: flow records represent bidirectional communication between two devices through a set of statistical metrics about aggregated packets. Rather than analyzing each packet individually, flows provide a higher-level view of the communication, which

[4] https://www.iana.org/assignments/ipfix/ipfix.xhtml

reduces the volume of the data to process, resource consumption, and processing latency [92]. This strength point makes these methods well-suited for high-speed networks and high-bandwidth links [192];

- Less intrusive: since flow-based approaches only process header metadata and do not analyze the user data, i.e., the payload, this results in fewer privacy issues. Furthermore, these approaches are more flexible and can be used even if the traffic is encrypted [116];

- Most network routing devices support nowadays NetFlow export and are equipped with built-in options that can be activated for flow collection and export. As such, no additional hardware is required [116].

For these reasons, we will focus in the remainder of the thesis on flow-based analysis. It is nevertheless important to note that the discussed methods as well as our contributions are generic and are not data-specific.

Following up on our description of network traffic data, we turn now to network traffic anomalies. We particularly discuss in the next section the most common anomalies encountered by network administrators and ISPs.

2.1.2 Network Traffic Anomalies

Network anomalies can be differentiated according to their causes. Anomalies are not always caused by malicious attacks that are run by an outsider to threaten the network integrity. Some outliers can result from non-malicious events such as hardware and software failure, network congestion, and corrupted sensors. Numerous studies [169, 29, 92] have distinguished four main causes of network anomalies:

- **Operational events**, a.k.a., misconfiguration failures, are non-malicious anomalies. The cause of these anomalies can be either hardware and software failures or configuration issues. Numerous examples of network anomalies fall in this category, including server crashes, traffic congestion, inappropriate resource allocation, and power outage [92];

- **Network attacks**, a.k.a., network abuse anomalies, are malicious events that aim to disrupt or degrade the provided network services and ecosystem [92]. These attacks are destined to compromise network availability and access to personal data. A plethora of attack types have been identified in the literature, ranging from *phishing* to *spoofing*, *intrusion* attacks (e.g. Denial of Service (DoS) attacks), *worms*, *viruses*, *botnets*. We refer the reader to this publication [49] for a detailed review of network attacks;

- **Flash crowds** are non-malicious anomalies that are defined as "a large surge in traffic to a particular website, causing a dramatic increase in server load" [126]. Unlike DoS, flash crowds are legitimate events that occur due to a rapid increase of users that simultaneously request access to a resource. A typical example of a flash crowd anomaly is the surge of access to a website when an online market proposes big sales offers. Unless the appropriate resources are allocated to cope with this burden, such anomalies may result in a service failure and system unavailability [92];

- **Measurement anomalies** are non-malicious issues related to the data collection process. For example, an overloaded router prioritizes packet routing over traffic collection and mirroring, which can result in a loss of some network packets [92]. Another common example of measurement anomalies is the loss of traffic data due to the use of unreliable UDP-based protocols.

Numerous *adhoc* approaches that target the detection of a specific category of network anomalies have been developed in the literature [49, 162, 99]. Nonetheless, we aim in this thesis at developing a *generic* anomaly detector that is flexible enough to detect both malicious and non-malicious anomalies. Indeed, as an ISP that aims to ensure that customer networks are operational, we aim to prevent any anomalous event that degrades the provided services. We have no prior knowledge about the anomaly types that may occur in the future. Consequently, we advocate for the use of flexible methods that does not only focus on specific anomaly types.

To this end, we now consider a broader and application-agnostic exploration of the AD. The following section aims to provide definition and taxonomy for anomalies, based on an in-depth analysis of the broad literature.

2.2 Definition And Categorization Of Anomaly Detection Methods

As mentioned previously, AD is a fundamental research problem that is studied in diverse scientific fields and various definitions of an "anomaly" have been proposed in the literature. In this part, we provide the most popular definitions that are commonly used in the community. Afterwards, we present an overall categorization of the literature AD methods.

2.2.1 Definition Of Anomalies

In 1969, Grubbs [108] proposed to define an anomaly as follows: "an outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs". The most common definition was articulated by Hawkins in 1980 [110], which states that an anomaly is "an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism". Fourteen years later, Barnett and Lewis proposed a similar definition in their seminal book [95]: an anomaly is "an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data". More recently, Ramaswamy et al. [211] have defined anomalous data points as follows: "an outlier in a set of data is an observation or a point that is considerably dissimilar or inconsistent with the remainder of the data".

These four definitions provide a relatively similar meaning: anomalies are non-conforming observations that violate an expected common nominal behavior. These definitions are generic and not limited to a specific application area: they remain valid regardless of the types of data processed. It is however important to mention that there are many domain-specific definitions of anomalies that copy with particular data types. For example, Huang et al. [118] define a network traffic anomaly as follows: "Network traffic anomalies are unusual and significant changes in the traffic of a network."

All the above definitions converge on two fundamental assumptions about anomalies:

Assumption 2.1 *Anomalies present remarkably dissimilar characteristics compared to the nominal data.*

Assumption 2.2 *The majority of the data is nominal. Anomalies are rare and constitute a small subset of the data.*

These two critical observations have a direct impact on the selection of the AD method. For example, classical supervised classifiers that require a balanced training dataset are not optimal for such tasks [46].

Until now, a mix-up between names has been reported in the literature as anomalies has been interchangeably referred to as *outliers*, *rare events*, *novelties*, *aberrations*, *surprises*, *noise*,

discordants, and *contaminants* [56]. The most common terms are *outliers*, *anomalies*, *noise* and *novelties*. There is no consensus about the differences between these terms. However, some recent studies have proposed to define the boundaries, by focusing on the peculiarities of each concept [6].

- **Noise**: According to Braei and Wagner [46], "noise can be a mislabeled example (class noise) or errors in the attributes of the data (attribute noise) which are not of interest to data analysts". Even though the noise is not interesting to analyze, its detection and removal are critical to clean the data and prevent any modeling bias;
- **Outlier**: Unlike noise, outliers are generally considered as extreme data points that "contain interesting and useful information about the underlying system" [230];
- **Anomaly** is considered as a broader term that includes both outliers and noise [292];
- **Novelty**: Masana et al. [171] define novelty as new "samples that share some common space with the trained distribution." For example, if one trains a neural network to identify cat breeds, a novelty here can be a dog image that belongs to a new breed, not considered during the training. This is different from an anomaly or an outlier, e.g., a cat or a horse image.

Although the fine differences between these terms, especially anomaly, outlier, and noise, a tremendous number of publication interchangeably use these worlds [6, 230, 229]. Similarly, we will consider in the remainder of the thesis that anomaly, outlier, and noise refer to the same connotation, according to the common Hawkins's definition:

Definition 2.1 *An anomaly is "an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism."*

Accordingly, the AD problem is related to various similar topics, including novelty detection, one-class classification, out-of-distribution detection, and open-set recognition [229]. These tasks share considerably similar goals, despite the slight differences and specificities. Recently, Salehi et al [229] provided a comprehensive survey where they proposed to unify these differences based on the main cross-task commonalities. The majority of the methods can be cross-domain interconnected and easily adapted to another sub-field.

The objective of the present part is to clearly define the anomaly taxonomy to avoid any potential confusion in the remainder of the thesis. We focus in the following section on depicting a generic categorization of anomalies that impact the design of the detection method.

2.2.2 Types Of Anomalies

Various studies have proposed different categorizations of anomalies [40, 118, 74]. Following this recent survey [40], we provide one of the most common categorizations that are not limited to a specific application. This categorization groups anomalies into three classes, according to their inherent properties and characteristics.

- **Punctual anomalies**: a punctual anomaly, a.k.a., point anomaly, is an individual data point that is non-conform with the remaining data (cf. Figure 2.4 (left)). To illustrate, a credit card punctual anomaly example could be a single extreme transaction that is very high compared to the average transaction range. This is the most straightforward type of anomaly and most outlier detection studies focus on the detection of this anomaly type [144];
- **Contextual anomalies**: a contextual anomaly, a.k.a., conditional anomaly, is an observation that is anomalous *only in a specific context* (cf. Figure 2.4 (middle)). For example, a temperature of 30° C in France during summer is nominal. However, this temperature becomes anomalous if it is reported during the winter. The *context* is determined based

on the structure of the data and their relative inter-dependencies. For example, in time-series data, the sequential order, i.e., temporal order, defines the context. In spatial data, the context is determined by the position of the data, i.e., the longitude and latitude of the location. Detecting contextual anomalies requires developing specific methods that are able to model the underlying data context, which makes the task more challenging. Therefore, these anomalies are less explored in the literature, and particularly investigated in time-series analysis [40];

- **Collective anomalies**: a collective anomaly, a.k.a., a group or a sub-sequence anomaly, is a set, i.e., a group, of consecutive observations that are anomalous regarding the remaining data set. In this case, the individual samples can be nominal if considered separately, while the entire pattern is anomalous. Figure 2.4 shows an example of a collective anomaly, highlighted in red. Collective AD also requires the modeling of the context. This task has been explored in time series, spatial, and graph data [63].

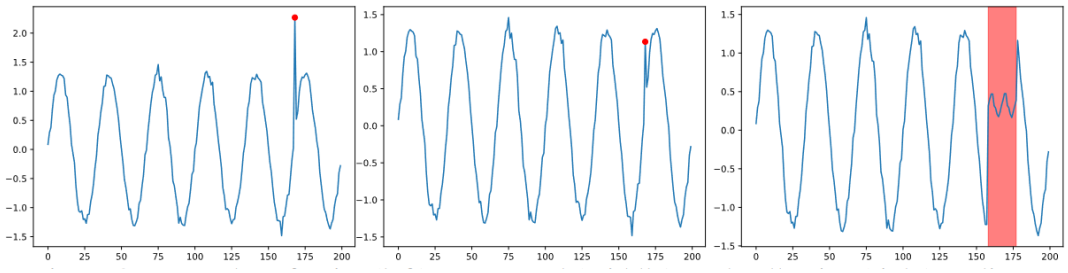


Figure 2.4: Illustration of punctual (left), contextual (middle), and collective (left) anomalies, extracted from [144].

The type of anomalies that should be detected has a direct impact on the design of the optimal detector. Detecting contextual anomalies with a method that ignores the sequential properties of the data may result in suboptimal performance, with a significant false negative rate [6].

We turn now to the literature detection methods.

2.2.3 Generic Categorization Of Anomaly Detection Methods

A plethora of methods tailored to detect anomalous data points have been developed in different communities [229, 222, 63, 201]. We can categorize AD methods of the literature according to two criteria: (1) the use of training labels and (2) the detection strategy.

2.2.3.1 Categorization Based On Supervision

The first categorization is based on the level of supervision with data labels. According to the availability of the labels, anomaly detectors can be grouped into three categories: (1) supervised, (2) semi-supervised, and (3) unsupervised approaches. The difference between these three categories is visualized in Figure 2.5.

Supervised Methods

Supervised methods, a.k.a., classification methods, (cf. Figure 2.5) assume the availability of a training dataset with labeled nominal and anomalous instances. A binary or multi-class classifier is trained on these labeled data to distinguish the different classes. Then, this classifier is used to assign labels to novel unseen data.

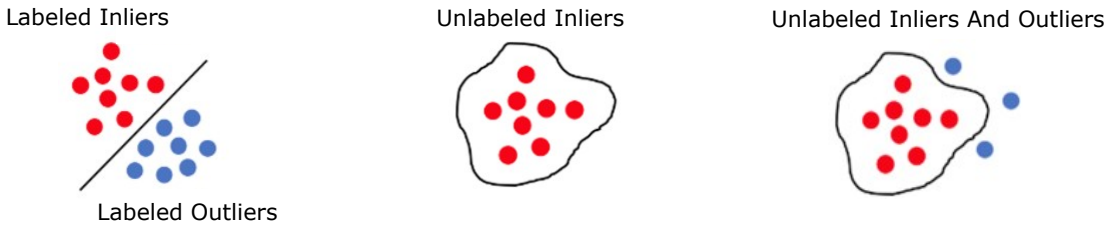


Figure 2.5: Supervised (left), semi-supervised (middle), and unsupervised (right) anomaly detection.

Among the most popular supervised anomaly detectors, we find Logistic Regression, Decision Trees, Naive Bayes, and Support Vector Machines [19]. Some studies have proposed to particularly leverage these classical methods to network traffic AD. For example, Anthi et al. [16] applied the J48 decision tree approach [210] to detect network attacks, including DoS, Man-in-the-Middle, and spoofing attacks. To this end, they collected a network traffic dataset from a testbed that comprises eight IoT devices: Belkin NetCam camera, TP-Link NC200 Camera, TP-Link Smart Plug, Samsung Smart Things hub, Amazon Echo Dot, British Gas Hive connected to a motion sensor and a window/door sensor, and a Lifx Lamp [16]. To collect anomalous data, the authors ran some network attacks during data collection, and labeled the corresponding packets as abnormal. Finally, they extracted header-based metadata from the raw packets, e.g., packet length, IP protocol, and TCP flags, and they train the J48 decision tree to separate the classes. The testing protocols have shown that this method has a high detection accuracy and can accurately detect seen attacks. Doshi et al. [86] focus on the detection of DDoS in LAN. The authors compare the performance of five supervised machine learning models: K-nearest neighbors, Support Vector Machine (SVM) with linear kernel, Decision Trees, Random Forest, and Artificial Neural Networks. These five models present promising results, with an accuracy score higher than 99%. Similarly, McDermott et al. [175] use Bidirectional Long Short Term Memory based Recurrent Neural Networks (BLSTM-RNNs) to detect botnet activities within IoT networks. Brun et al. [50] train random neural networks to detect network attacks based on traffic metadata. Finally, Restuccia et al. [214] discuss the importance of using machine-learning-based approaches to mitigate network anomalies. They provide a taxonomy of existing classical IoT anomaly detectors, introduce the emerging challenges in IoT Security, and discuss the relevance of machine-learning-based approaches to address these challenges.

Despite the high detection accuracy and ease of implementation, supervised anomaly detectors have two main limitations. First, collecting a representative labeled dataset with diverse anomalous instances is challenging. Indeed, anomalies are rare and usually unknown in advance. The lack of availability of labeled up-to-date network traffic anomaly datasets is a major hurdle to the development of fully supervised anomaly detectors. Novel anomalies frequently occur in real-world environments, e.g., zero-day attacks, which makes existing labeled training datasets outdated and the collected anomalous data do not well represent the true anomalies that can be found in reality. This requires the continuous update of trained classifiers to integrate the novel anomaly properties. Second, the ratio of labeled anomalies is generally significantly lower than inliers. This class imbalance impacts the performance of classical supervised classifiers and leads to sub-optimal results [59].

Owing to these main technical issues, supervised AD is less explored in the literature and we focus in the remainder of the thesis on semi-supervised and unsupervised approaches.

Semi-Supervised Methods

Generally, the labels of nominal instances are more affordable and easier to collect compared to outliers, all the more relevant in the IoT application area. Indeed, IoT devices are generally developed to perform a specific well-defined task. It is thus easier to model the nominal behavior

than to model all anomalous classes. Following this reasoning, semi-supervised anomaly detectors, a.k.a, one-class approaches, (cf. Figure 2.5) learn a representation of the *normality*. Then, any deviation from the norm is considered an anomaly. In this setting, the training data must only contain instances generated from the nominal class.

Since labeled anomalies are not required during the training, these approaches are prevalent in the AD community. A plethora of semi-supervised anomaly detectors have been developed in the literature, including statistics-based, classification-based, distance-based, and reconstruction-based methods [57]. These categories will be detailed later in Section 2.3.

Semi-supervised AD efficacy depends on the availability of anomaly-free training data, and performance may degrade significantly when this assumption is violated. Unfortunately, this violation is likely to occur in real-world applications. For example, in network traffic monitoring, collected network packets may comprise defective data sent by faulty sensors, damaged fiber connectors, or caused by network congestion [139]. Finally, due to data volumes and potentially unknown anomalies, manual labeling of training samples is not guaranteed to be correct..

Unsupervised Methods

Unsupervised methods relax the assumption that the training data are completely anomaly-free (cf. Figure 2.5). They assume that the training data may comprise an unknown ratio of outliers, a.k.a., contaminated data. They implicitly assume that nominal instances are much more frequent than outliers, since the latter are by definition scarce. The aim is to develop a *robust* model that accurately represents the *normality* without being impacted by the training contaminants. This task is arduous since no labels are provided and the ratio of contamination is generally unknown in advance. Numerous studies adapt semi-supervised approaches to this scenario. They propose two steps-based approaches where they first filter potential training contaminants with a dedicated rejection strategy. Then, they use the recovered anomaly-free subset to model the norm [202]. We will consider a detailed review of these approaches later in the thesis.

2.2.3.2 Based On The Detection Strategy

The previous categorization focused on label supervision and grouped anomaly detectors into three classes: supervised, semi-supervised, and unsupervised approaches. Concurrently, it is possible to categorize AD methods based on the detection strategy. Here, there is a clear dichotomy between *instance-based* and *model-based* methods.

Instance-Based Methods

Instance-based learning, a.k.a., memory-based learning and lazy learning, does not explicitly create a model of normality. It compares each observation with the most similar instances of the dataset. The main assumption is that anomalies are significantly dissimilar to the majority of the data points, while the nominal data share common properties. That is, the criterion to distinguish both classes is according to the relative comparison between an observation and the most relevant instances, i.e., the closest or the most similar data points.

Instance-based models are widely used in AD thanks to their effectiveness and simplicity [6]. Two of the most common instance-based anomaly detectors are distance-based and density-based approaches, which will be discussed in Section 2.3.1.

Model-Based Methods

Unlike instance-based learning, model-based approaches, a.k.a., explicit generalization methods, operate in two steps.

- Training phase: this first step trains an algorithm that accurately represents the normality. This representation must model all the properties of the nominal behavior, and therefore represents an *explicit generalization* of the norm;
- Inference phase: once the nominal data are well-modeled, the trained model is used to compute novel instance anomalousness scores, according to their deviation from the norm. A large deviation indicates that the data point presents uncommon atypical behavior, and the point is deemed anomalous.

The main singularity of this paradigm over instance-based learning is that the data must not be memorized and stored for inference. Instead, the model itself is used as a reference for normality. This results in a faster and more flexible inference. Furthermore, by virtue of the prosperous development of machine learning-based methods, namely artificial neural networks, model-based learning has become a staple of AD. Model-based AD mainly regroups, inter alia, statistics-based (cf. Section 2.3.2, one-class classification-based (cf. Section 2.3.3), neural network-based one-class classification (cf. Section 2.3.4), and reconstruction-based approaches (cf. Section 2.3.5). All these paradigms will be described in the following sections.

2.3 Classical Methods for Anomaly Detection

This section reviews the most common AD approaches. We start by developing two popular instance-based methods: distance-based and density-based methods. We outline the main limitations of this paradigm. Then, we develop the more prevalent model-based learning strategies: statistics-based, one-class classification-based, neural network-based one-class classification, and reconstruction-based approaches. We highlight that most of these models are tailored to semi-supervised AD (cf. Section 2.2.3.1). Unless explicitly mentioned, the approaches described in this section fall in the paradigm of semi-supervised learning.

2.3.1 Instance-Based Methods

2.3.1.1 Distance-Based Methods

Distance-based methods define the anomaly score of an observation according to the distance to its k-nearest neighbors. They assume that outliers are far from their local neighbors, while inliers are located close to each other. For example, the point A of Figure 2.6 is considered anomalous because it is isolated from almost all the remaining data points.

Distance-based methods quantify the proximity of an instance from its neighbors using a distance function. The most widely-used distance function is the Euclidean function [6]. The key idea is to compute the pairwise distance between the input data instances. The distance of an outlier to its closest neighbors is much larger than that of inliers. We note however that this is the most basic distance-based overview and different variants have been proposed in the broad literature [136, 211].

Distance-based methods are among the most popular methods that are widely used to detect anomalies in diverse application fields [264]. Knorr and Ng [136] were among the first researchers that apply distance-based approaches to AD. They particularly proposed a non-parametric approach leveraging an indexed-based search algorithm that identifies the closest data points to each test instance and assesses its anomalousness accordingly. This algorithm however presents numerous limitations, namely its computational complexity of $O(d \times N^2)$, where d is the dimension of the data and N is the overall number of instances included in the dataset. The quadratic complexity

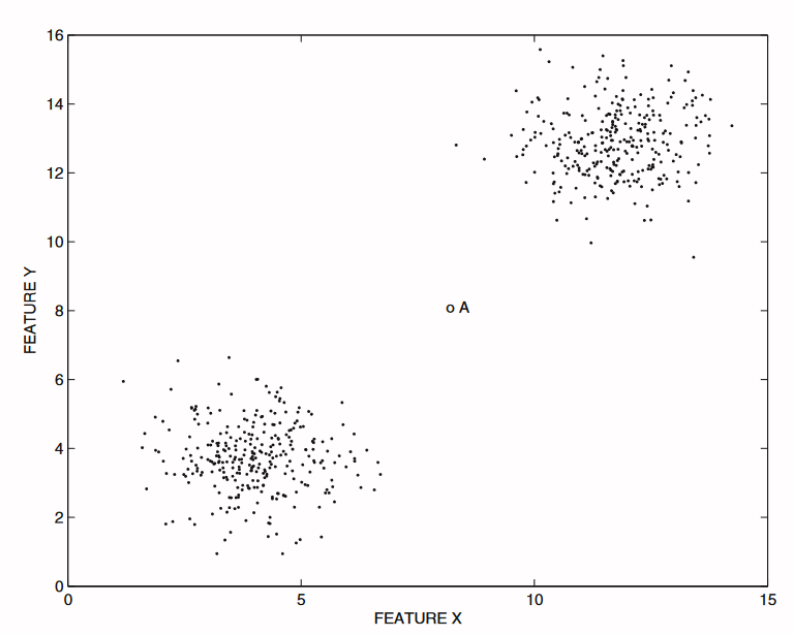


Figure 2.6: An example of a 2-dimensional outlier.

with respect to N makes this approach intractable for large data. Ramaswamy et al. [211] studied the limitations of the previous proposal and focused on reducing the computational cost. They proposed a partition-based method that divides the data into disjoint subsets and iteratively filters out points that have the lowest distance to their k-nearest neighbors. They found that pruning out these points in small partitions rather than the entire dataset can filter out numerous instances. Then, only the k-nearest neighbors of the few remaining outlier candidates need to be computed, which significantly reduces the computational cost and preprocessing overhead. One main concern of this method is its sensitivity to the partition size. It is difficult to find k close neighbors in very small reduced-size partitions, and in this case, very few instances can be pruned out during the first stage. In contrast, performing the pruning in multiple large-size partitions is computationally expensive. Similarly, multiple other studies have explored this technical issue in the literature [101, 36, 267]. A common way to improve the computational complexity is through approximate nearest neighbor estimation. This strategy approximates the traditional exhaustive search of neighbors with a more efficient search strategy. For example, Ghoting et al. [101] proposed a fast method, referred to as Recursive Binning and Re-Projection (RBRP). This method splits the data into partitions, finds neighbors in partitions, and proposes a fast merging algorithm to establish global k-nearest neighbors. The authors proved that this approach "scales log-linearly as a function of the number of data points and linearly as a function of the number of dimensions", i.e., with a computational complexity of $O(NlogN \times d)$, where d is the dimensionality and N is the data volume.

Finally, numerous researchers have applied distance-based methods to network traffic AD [45, 266, 44]. We refer the reader to this survey [130] for a detailed review of these research publications.

All the above methods rely on the distance to the neighbors as a criterion to distinguish inliers and outliers. An alternative popular criterion is relative to the local density of a data point.

2.3.1.2 Density-Based Methods

Density-based AD assumes that nominal data are located in dense regions, while outliers appear in low-density, a.k.a., sparse, areas. Thus, these methods compare the density of each instance against its local neighbor densities. A significant difference is generally considered an indication of

irregularity and the corresponding point is deemed anomalous.

It is however worth mentioning that density and distance are closely related to each other in such a way that some distance-based methods are sometimes presented as density-based approaches, and conversely [6]. Indeed, the distance to the k-nearest neighbor of an instance can be seen as the radius of a sphere that is centered in the given instance and that contains the relative closest k neighbors. A larger k-nearest neighbor distance means that the data point is located in lower-density regions. As such, this distance can be considered as an approximation of the inverse of the density [63].

One of the most popular density-based anomaly detectors was introduced by Breunig et al. [48] and denoted as Local Outlier Factor (LOF).

Local Outlier Factor

LOF quantifies the comparison between the densities of a data point and its local neighbors using the *outlier factor*. To begin with, the authors formalize the local density of a point using the *reachability distance*.

Let $\mathbf{x} \in \mathbb{R}^d$ be a data point of dimension d, $D^k(\mathbf{x})$ the distance to its k-nearest neighbor, and $N^k(\mathbf{x})$ a set that contains all the points within the hypersphere centered at $\mathbf{x}$ and of radius $D^k(\mathbf{x})$, i.e., the points that are less far than the k-nearest neighbor. The authors define the reachability distance between two points $\mathbf{x}$ and $\mathbf{y}$ as follows:

$$R^k(\mathbf{x},\mathbf{y}) = \max(dist(\mathbf{x},\mathbf{y}), D^k(\mathbf{y})). \tag{2.1}$$

In other words, the reachability distance is the distance between two points, smoothed by the k-nearest neighbor distance of the second point when this distance is low. We highlight however that the reachability distance is not conforming with the mathematical definition of a distance, since it is not symmetric. The *local reachability density* (lrd) is defined as:

$$lrd(\mathbf{x}) = \frac{1}{MEAN_{\mathbf{y}\in N^k(\mathbf{x})} R^k(\mathbf{x},\mathbf{y})}, \tag{2.2}$$

where *MEAN* refers to the mathematical mean of a set of values. That is, the local reachability density of a point $\mathbf{x}$ is the inverse of the average reachability distance with respect to its local k-nearest neighbors. Finally, the local outlier factor score is computed as follows :

$$LOF^k(\mathbf{x}) = MEAN_{\mathbf{y}\in N^k(\mathbf{x})} \frac{lrd(\mathbf{y})}{lrd(\mathbf{x})}. \tag{2.3}$$

A score that is close to 1 indicates that the point local density is similar to its local neighbors. A score larger than 1 means that the instance is located in a lower-density region compared to its neighbors. This point is therefore considered anomalous.

LOF has shown huge success in the AD community and multiple variations have been proposed, such as Connectivity-based Outlier Factor, Local Correlation Integral, Cluster-Based Local Outlier Factor, and Dynamic-Window Outlier Factor. Reviewing these variants is out of the scope of this thesis. We however refer the reader to this recent survey [13], which develops all these variants in detail.

In the following, we will explore the most relevant publications that apply LOF and more generally density-based methods to unsupervised AD.

Anomaly Detection With LOF And Density-Based Approaches

Density-based approaches are widely used in the AD in diverse application domains such as fraud detection, network intrusion detection, and medical image analysis [13]. We limit the scope of this part to network traffic analysis. Paulauskas and Bagdonas [204] apply the classical

LOF method to analyze the network flow data collected from a large network that comprises 350 connected computers and to detect any suspicious and malicious behavior. Brute force attacks were executed to collect and label anomalous flow data. Overall, the dataset contains 479 labeled flows, where 54 are labeled anomalous. In this study, LOF shows promising results, particularly when the hyperparameter k was set as equal to 10% of all the data flows. More recently, Auskalnis et al. [22] investigate the efficacy of LOF-based AD in network intrusion detection. They particularly demonstrated its sensitivity to the hyperparameters, and mainly the number of neighbors k, on the NSL-KDD public network traffic dataset [196]. This sensitivity was also criticized in [87]. Indeed, the abnormality of a data point is controlled by this hyperparameter k, which must be carefully selected by the analyst. When k is very low, only a few neighbors are considered in the comparison. In this case, and even for an outlier, its few neighbors may also be anomalous, which results in comparable low densities, i.e., a local outlier score close to 1. Conversely, the smoothing strategy of the reachability distance can be dominant when k is very large.

We will conclude this part by developing the strengths and weaknesses of instance-based AD.

2.3.1.3 Strengths And Weaknesses Of Instance-Based Anomaly Detection

This section aims at examining the advantages and the challenges of instance-based AD.

On the one hand, these methods are extremely popular thanks to their accuracy and ease of implementation [6]. Indeed, instance-based approaches are non-parametric and do not fit a model to represent the norm. As such, no prior knowledge or predefined assumptions about the nominal distribution are required in this case.

On the other hand, these classical methods have a number of technical challenges [6]. First, the main concern of instance-based AD is their computational complexity. Indeed, the computation of pairwise distances has a large computational cost of $O(N^2)$ in the worst case, which limits the applicability of these methods on very large datasets. Second, even though some approximate solutions have been proposed to reduce this quadratic complexity, an even more critical issue is the linear dependency with respect to data dimensionality. Most existing instance-based methods involve computing distances between data points. However, in high-dimensional spaces, distances become less informative as all high-dimensional vectors become almost equidistant [6]. This issue is known as the curse of dimensionality, which is mainly due to the sparsity of high-dimensional data. Consequently, instance-based AD performance may significantly degrade when the input data has many features. Finally, numerous studies have criticized the sensitivity of these methods with respect to their hyperparameters, namely the number of the local neighbors [87, 22]. As mentioned above, this hyperparameter must be carefully selected to adapt the methods to the given data and grant their optimal performance. Model-based anomaly detectors were developed in the literature as an alternative paradigm that addresses some of the aforementioned limitations

2.3.2 Statistical Models

The earliest works on model-based AD dates back to the 19^{th} century [159] and were concentrated on statistical and probabilistic models [6]. The key idea is to fit a specific model to the nominal data. Then, the likelihood fit of a novel data point to the underlying model is used as an anomaly score. Extreme values indicate that the given point is poorly fitted and does not follow the common properties of the majority of the data, i.e., an outlier.

There are two main families of statistical methods: parametric and non-parametric approaches. While parametric methods require knowledge or conjectures about the distribution of the normality, before fitting the model, non-parametric models do not require any prior assumption about the nominal behavior distribution. We will now develop the theoretical background of two extremely popular statistical anomaly detectors: the Gaussian Mixture Model (GMM) and the Kernel Density Estimation (KDE).

2.3.2.1 Parametric Statistical Method

Gaussian Mixture Model

The Gaussian distribution is one of the most prevalent distributions in probabilistic modeling. The assumption that the nominal data follow a Gaussian distribution or, more generally, a GMM, is ubiquitous in multiple fields thanks to the simplicity of the resulting methods. For univariate data, the density function of the Gaussian distribution is defined as follows:

$$f(x) = \mathcal{N}(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{(2\pi)}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \tag{2.4}$$

where μ is the mean and σ is the standard deviation of the data. The optimal two parameters that accurately model the nominal data can be estimated using the Maximum Likelihood Estimation (MLE) method, i.e., the Expectation–Maximization (EM) algorithm.

This model assumes that the nominal data are normally distributed according to μ and σ. Outliers are extreme values that are poorly fitted under this model, and consequently are located in the tail of the distribution. These extreme can be identified using the *z-value* test. The z-value score of an observation $x \in \mathbb{R}$ is defined as follows:

$$z = \frac{|x-\mu|}{\sigma}, \tag{2.5}$$

where |.| is the absolute value. This score quantifies the distance of a point from the mean, normalized by the standard variation to account for the data variation. Points that are very far from the mean are deemed anomalous. A rule of thumb consists in rejecting all data points with $z > 3$ [6].

In some cases, the nominal behavior may present multiple patterns having different characteristics, e.g., a connected printer that can print documents, scan manuscripts, and even send e-mails. Modeling these distinct behaviors with a single Gaussian may be sub-optimal. An alternative approach consists in fitting a mixture of Gaussian distributions, a.k.a., the GMM.

GMM assumes that the data are generated from k Gaussian distributions, where each distribution has its own specific parameters: mean and standard deviation. The corresponding density function is defined as:

$$p(x) = \sum_{i=1}^{k} \pi_i \mathcal{N}(x|\mu_i,\sigma_i), \tag{2.6}$$

where $\mathcal{N}(x|\mu_i,\sigma_i)$ is a Gaussian distribution of parameters μ_i and σ_i, a.k.a., component of the mixture, and π_i is the mixing component, which determines the weight of the component i in the overall model. There are two constraints regarding these mixing weights. First, their sum must be equal to 1 to ensure that integrating the density function is normalized, i.e.,

$$\sum_{i=1}^{k} \pi_i = 1. \tag{2.7}$$

Second, since $\mathcal{N}(x|\mu_i,\sigma_i)$ are by definition positive, each mixing coefficient must be positive, i.e.,

$$0 \leq \pi_i \leq 1, \forall i \in \{1,...,k\}. \tag{2.8}$$

AD With Parametric Statistical Methods

The GMM, and less specifically parametric-based methods, are among the most prevalent statistical methods that are applied to AD. Bahrololum et al. [26] proposed a network IDS based on the GMM approach. Their approach learns the optimal parameters of the GMM that accurately model the network traffic activity. Then, extreme values with a low likelihood fit, i.e., a likelihood

that is below a predefined threshold, are reported as anomalous. Bitaab and Hashemi [39] proposed a hybrid network anomaly detector, called DT-GMM, that combines decision trees and GMM. The decision trees are a supervised algorithm that is trained on labeled nominal and anomalous network flows. They are used as a first step to detect and filter out known malicious attacks. The GMM is trained on an anomaly-free subset and applied to report any unseen attack that may go under the radar of the first method. This hybrid system was experimentally evaluated on the NSL-KDD benchmark data [196] and compared against a hybrid system of decision trees and the classical SVM, called DT-SVM. DT-GMM outperforms DT-SVM on this dataset, with higher detection accuracy and a lower false positive rate. However, the main limitation of this approach is that it requires labeled training data for the supervised learning of the decision trees. Liu et al. [154] performed a sensitivity analysis of the GMM and demonstrated that the number of components k is of paramount importance in the modeling, which must be fine-tuned and carefully selected with a dedicated validation strategy.

Besides GMM-based AD, other parametric statistical and probabilistic models have been proposed in the literature, ranging from a mixture of Poisson distributions [168], to Chi-squared distribution [283], and the generalized Pareto distribution [157].

Strengths And Weaknesses Of Parametric Statistical AD

In summary, the main advantage of parametric models is their efficiency, with their low computational cost compared to instance-based learning. Instead of memorizing all the data for the inference, these approaches build a model which stands as a reference for normality. Then, only this model is leveraged and no pairwise distance computations are involved during the detection. Nonetheless, the choice of the parametric model is critical in this paradigm, which is generally data-specific and requires prior knowledge about the nominal data distribution. With an unsuitable selected model, even the nominal data may be poorly modeled with extremely low likelihoods of fit and can be reported as false positives. Furthermore, learning the nominal data distribution consists in empirically optimizing the model parameters, which requires the availability of completely clean anomaly-free training data. The presence of noisy data and outliers that contaminate the training data may result in a biased model of the norm that does not reliably represent the underlying data. This biased model can incorrectly report nominal data points as outliers and dismiss the true anomalies, with high confidence [6].

2.3.2.2 Non-Parametric Statistical Method

Kernel Density Estimation

In the previous section, we focused on parametric AD with predefined statistical and probabilistic models, having known mathematical forms and defined with a finite number of parameters that can be estimated from the data. Here, we investigate non-parametric methods that presuppose fewer and weaker assumptions about the norm. One of the most well-known non-parametric methods in the AD community is KDE.

KDE, a.k.a., the Parzen's window [203], assumes that the data are generated from an unknown distribution function $p(x)$ that does not belong to a familiar parametric family. KDE empirically estimates this density from the data.

Let $X = x_1, x_2, .., x_N$ be N independent and identically distributed (iid) variables, such that $x_i \in \mathbb{R}\ \forall i \in \{1, .., N\}$. These samples are supposed to be generated from a continuous distribution $F(x)$ with a density function $f(x) = \frac{d}{dx}F(x)$. The distribution $F(x)$ can be estimated by the empirical distribution function $\hat{F}(x) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(x_i \leq x)$. By definition, and regardless of the family distribution of $F(x)$, the density can be written as:

$$f(x) = \lim_{h \to 0} \frac{\hat{F}(x+h) - \hat{F}(x-h)}{2h} = \lim_{h \to 0} \frac{P(x-h < X < x+h)}{2h}, \tag{2.9}$$

where $h > 0$ is a very small variable. One can estimate the probability $P(x-h < X < x+h)$ by counting the number of points that are located in this interval:

$$P(x-h<X<x+h) = \frac{1}{N}\sum_i \mathbb{1}(\frac{|x_i - x|}{h} < 1). \tag{2.10}$$

Let us define the following uniform function K:

$$K(x) = \begin{cases} \frac{1}{2} & \text{if } |x| < 1 \\ 0 & \text{Otherwise.} \end{cases} \tag{2.11}$$

Then, the empirical density function can be written as:

$$f(x) = \frac{1}{N}\sum_i K(\frac{x_i - x}{h}). \tag{2.12}$$

K is called a *kernel function* and h is the bandwidth of the kernel. Here, we provided a simple example of a kernel function that can be used in KDE. Other smooth kernel functions can be used to obtain a smoother density function. A common choice is the Gaussian Kernel, which defines the following density model:

$$f(x) = \frac{1}{N}\sum_i \frac{1}{h\sqrt{2\pi}} \exp\left(-\frac{||x-x_i||_2^2}{2h^2}\right). \tag{2.13}$$

The kernel bandwidth is the Gaussian standard deviation. That is, Gaussian KDE is achieved by representing each data point with a Gaussian component and averaging all the Gaussian to obtain the overall model. The hyperparameter h controls the smoothness of the model, since a very small h may result in a noisy density model, and a very large h results in a less informative density. This is illustrated in Figure 2.7, extracted from [37].

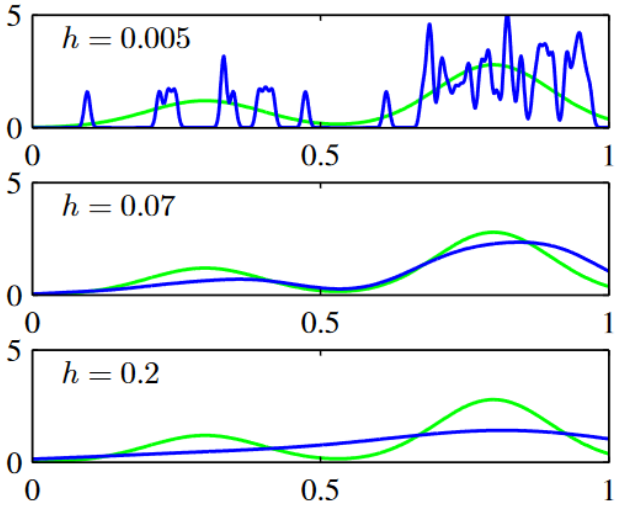


Figure 2.7: Illustration of Kernel Density Estimation sensitivity with respect to the bandwidth hyperparameter h, extracted from [37]. We note that h controls the smoothness, where a very small h (top) may result in a noisy density model and a very large h (bottom) results in a less informative density.

AD With Non-parametric Statistical Methods

A large body of research publications has proposed to apply KDE and other non-parametric methods, e.g., histogram-based modeling, to AD, and particularly to detect network traffic anomalies [285, 284, 239]. Yeung and Chow [284] applied the KDE strategy to network intrusion detection. They proposed to model the network nominal data using a KDE with Gaussian Kernels. Then a threshold is set to isolate novel instances with small log-likelihoods under this model. The KDD Cup 1999 [76] benchmark dataset was used to explore the efficacy of this non-parametric method, which reported a high True Detection Rate (TDR). This method showed competitive AD results compared to supervised algorithms trained with labeled data. In addition, they ran additional experiments to assess the KDE sensitivity to the bandwidth h. They found that the results remain relatively stable over some values of h. Shen and Agrawal [239] investigated KDE-based network intrusion detection by comparing four different kernel functions: the Epanechnikov, the Triangular, the Biweight, and the Gaussian kernels, which report relatively similar results. In this study, the authors used a heuristic rule to set the bandwidth hyperparameter equal to $h = 1.06\sigma N^{-\frac{1}{5}}$, where σ is the data standard deviation and N is the sample size.

Strengths And Weaknesses Of Non-parametric Statistical AD

Non-parametric statistical approach present various advantages and disadvantage. The main strong point of these methods is their flexibility and adaptability to data change, as no prior knowledge about the norm is required in advance. Nonetheless, this approach performance generally depends on user-defined hyperparameters, e.g. the kernel bandwidth of the KDE. Furthermore, a more fundamental limitation is observed with the increasing dimensionality of the data. Indeed, this problem arises due to the sparsity of the data in high dimensionality spaces, and computing the kernel matrix may be suboptimal [6].

2.3.3 One-Class Classification-Based Models

AD can be seen as a classification problem where one class, i.e., the anomalous class, is partially or completely unobserved. That is, only the labeled nominal instance are provided and anomalous data points are missing. This setting is frequent in network AD where some labeled attacks may exist, but numerous malicious and non-malicious anomalies are still unknown and constantly discovered over time [6]. In the case where the anomaly labels are partially available, the problem can be formulated as an imbalanced supervised classification problem. The more extreme situation, and the more common setting, where only labeled nominal data are observed refers to the problem of One-Class Classification (OCC).

OCC [180], a.k.a., unary classification and semi-supervised classification, aims to develop a classifier that distinguishes inliers and outliers, when the anomalous class data is poorly represented or completely absent. The objective is therefore to estimate the boundary of the target class, i.e., inliers, without any prior knowledge about the negative class, i.e., outliers. Then, novel instances are assessed based on whether they belong to the target class region or not. Figure 2.8, which is extracted from [252], illustrates the key principle of OCC. This task is significantly harder than the conventional two-class classification problem, since only one side of the boundary can be estimated [129].

Multiple surveys have been proposed to review the most recent OCC-based approaches in different application fields [129, 206, 236]. Two of the most common OCC-based approach that are extremely popular in AD are One-class Support Vector Machine (OSVM) and Support Vector Data Description (SVDD). These methods will be reviewed in the next section.

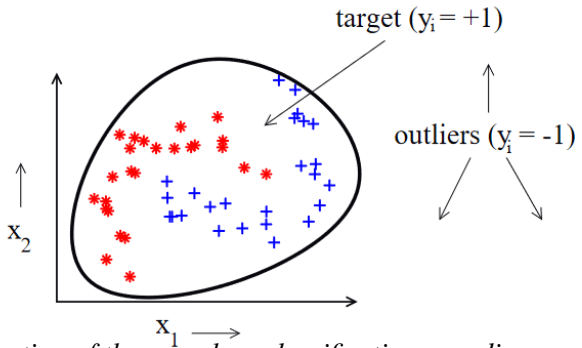


Figure 2.8: Illustration of the one-class classification paradigm, extracted from [252].

2.3.3.1 One Class Support Vector Machine

The classical SVM is designed to separate two classes using a decision boundary that maximizes the corresponding margins [42]. OSVM [233] extends the traditional SVM to the situation where only one class data are observed, usually inliers. This approach learns a kernel-based transformation that maps the data to another space. The *origin* of this space is supposed to belong to the anomalous class, while the training data are considered nominal. OSVM estimates the optimal hyperplane that separates the data points from the origin, such that this hyperplane is very close to the data and as far as possible from the origin, i.e., the proxy of the anomalous class. The overview of OSVM is visualized in Figure 2.9, extracted from [6].

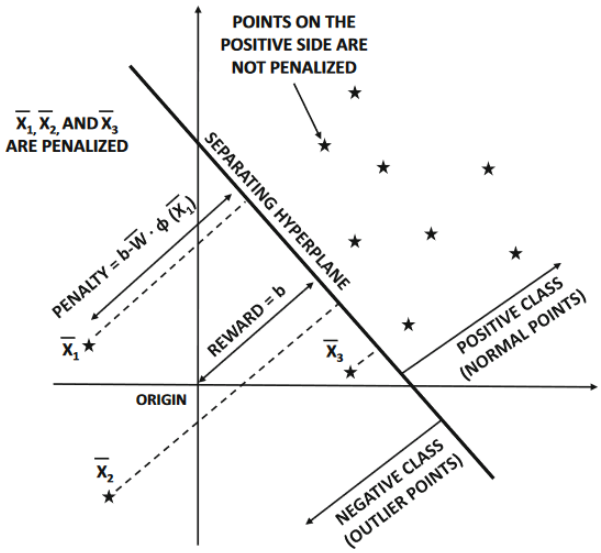


Figure 2.9: Overview of the OSVM approach, extracted from [252].

Let us consider N training observations $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_N)$, where $\mathbf{x}_i \in \mathbb{R}^d$, $\phi(.)$ be a feature map that project the original data point into the novel space, and K be the corresponding kernel function, defined as follows:

$$K(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle , \tag{2.14}$$

where $\langle . \rangle$ denotes the dot product between two vectors. The objective of OSVM is to learn the parameters of the hyperplane that separates the target data and the origin. The decision boundary equation can be formulated as follows:

$$\mathbf{w}\phi(\mathbf{x}) - b = 0, \tag{2.15}$$

where $\mathbf{w}$ and b are the parameters of the hyperplane. All the nominal training data should have $\mathbf{w}\phi(\mathbf{x}) - b \geq 0$, while the origin is negative.

To this end, this approach penalizes the nominal data points that are on the wrong side of the boundary, which is translated with a regularization penalty of $\max\{0, b - \mathbf{w}\phi(\mathbf{x})\}$. In addition, the boundary is encouraged to be as far as possible from the origin by maximizing the value of b.

In summary, OSVM optimizes the following objective function:

$$\frac{1}{2}||\mathbf{w}||^2 + \frac{C}{N}\sum_i \max\{0, b - \mathbf{w}\phi(\mathbf{x}_i)\} - b. \tag{2.16}$$

$C > 1$ is a hyperparameter that balances the regularization and the nominal data point penalty. Increasing the value of C over-penalizes misclassified nominal data points and enlarges the region that encloses inliers.

We note that the optimization problem is defined using the unknown function $\phi(.)$. One can reformulate the optimization task using the kernel trick, a.k.a., a dual formulation of the problem. Indeed, the optimization task can be reformulated as follows:

$$\begin{gathered} \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{N}\sum_i \xi_i - b \\ \text{subject to } \langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle \geq b - \xi_i, \text{and } \xi_i \geq 0. \end{gathered} \tag{2.17}$$

Here, ξ_i are slack variables that are introduced to relax the hard boundary assumption. Indeed, they are destined to balance the compromise of maximizing the overall margin and minimizing the number of data points that lie outside the target region. It can be shown that this problem may be solved with the Lagrange dual formulation:

$$\begin{gathered} \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j \left\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \right\rangle \\ \text{subject to } 0 \leq \alpha_i \leq \frac{C}{N}, \text{ and } \sum_i \alpha_i = 1, \end{gathered} \tag{2.18}$$

where α_i are the Lagrange multipliers. We highlight that, in this dual formulation, all mappings are indirectly computed with the kernel trick, where the inner products $\left\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \right\rangle = K(\mathbf{x}_i, \mathbf{x}_j)$. As such, the non-linear mapping function $\phi(.)$ may not be explicitly defined. The most common kernel functions used in the literature are the linear and the Gaussian kernels.

Finally, with this dual formulation, the binary label associated with a novel data point $\mathbf{x}$ is defined by the following function:

$$f(\mathbf{x}) = \text{sign}\left(\sum_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) - b\right). \tag{2.19}$$

"sign" is the sign function, which outputs -1 if the corresponding function is negative and $+1$ otherwise.

2.3.3.2 Support Vector Data Description

Another similar OCC-based approach, inspired from OSVM, was developed by Tax et al. [251] and called SVDD. Rather than learning a linear hyperplane that separates the data from the origin in the transformed space, SVDD surrounds the nominal data with a hypersphere of radius $R \in \mathbb{R}^+$ and center $\mathbf{a} \in \mathbb{R}^d$, as illustrated in Figure 2.10. The aim is to find the optimal hypersphere with the minimum radius that encloses all the nominal data points.

Concretely, given N training data points of dimension $d \in \mathbb{N}^*$, $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_N)$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $\phi(.)$ a function that maps the original data point into a novel space, SVDD minimizes the

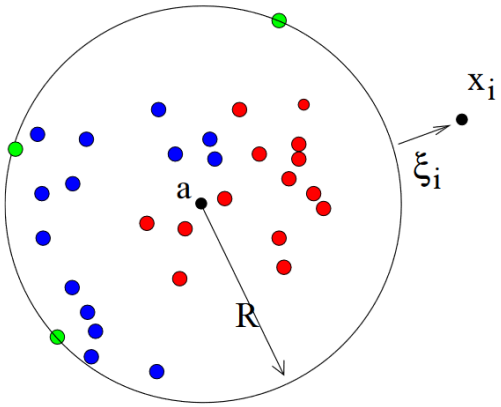


Figure 2.10: Overview of the SVDD approach, extracted from [129].

following objective function:

$$R^2 + C\sum_i \xi_i \tag{2.20}$$
$$\text{subject to } ||\phi(\mathbf{x}_i) - \mathbf{a}||^2 \leq R^2 + \xi_i, \text{ and } \xi_i \geq 0,$$

where ξ_i are slack variables and C is a trade-off hyperparameter that balances the hypersphere radius minimization and the violation penalty of the boundaries with the slack variables. The resulting model segregates test inliers and outliers based on the distance from the hypersphere center, where a data point is deemed anomalous if:

$$||\phi(\mathbf{x}_i) - \mathbf{a}||^2 \geq R^2. \tag{2.21}$$

Similar to OSVM, a dual formulation of this constrained optimization task can be defined using the Lagrange multipliers:

$$\sum_i \alpha_i(\mathbf{x}_i, \mathbf{x}_i) + \sum_{i,j} \alpha_i \alpha_j K(\mathbf{x}_i), \mathbf{x}_j) \tag{2.22}$$
$$\text{subject to } 0 \leq \alpha_i \leq C, \text{ and } \sum_i \alpha_i = 1,$$

where $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ is the kernel matrix. Finally, we note that OSVM is equivalent to SVDD, when trained with the Gaussian kernel.

2.3.3.3 AD With Classification-Based Approaches

Classification-based approaches, and particularly OSVM and SVDD, are widely studied and applied in network AD. An interesting discussion about OCC can be found in [5]. Tran et al. [256] proposed a OSVM-based method, tailored to network traffic AD. The authors first collect the nominal network traffic and extract some flow statistics with the open-source network monitor tool: *Tcpstat* [114]. Tcpstat computes 20 metadata features, but only 5 features were involved in this study: the number of Internet Control Message Protocol (ICMP) packets, the number of TCP packets, the number of UDP packets, the average packet size, and the standard deviation of packet size. Then, the classical OSVM is used to learn a classifier that recognizes the normal behavior of the network based on these metadata. A Gaussian kernel was selected and the evolving training model method [257] was used to find the optimal hyperparameters: C and the standard deviation σ of the Gaussian. Evaluated on the DARPA intrusion detection dataset, OSVM reports promising results, with high Area Under the Curve of the Receiver Operating Characteristics (AUROC) and a

low false alarm rate. More recently, Chen and Li [65] proposed a lightweight SVDD-based anomaly detector, designed to recognize malicious attacks in wireless sensor networks. The main distinction of this approach is its low power consumption, which is adapted to resource-constrained wireless sensors. In the same line, the study of Jha and Ragha [124] investigates the efficacy of OSVM and SVDD when applied to network intrusion detection. They mainly discuss the strengths and limitations of these classical methods, which will be presented in the next section.

2.3.3.4 Strengths And Weaknesses Of Classification-Based AD

Among all AD methods, classification-based approaches have the advantage of relying on the kernel trick and optimizing a few free parameters, the parameters of the decision boundary [6]. Nonetheless, they have numerous technical limitations that should be addressed. To begin with, the kernel matrix computation with respect to all the training data is computationally expensive, with a quadratic complexity cost. A second shortcoming is related to its significant sensitivity to user-defined hyperparameters, i.e., the balancing parameter C and the choice of the kernel family [6]. This issue was criticized by Manevitz and Yousef [165] in their study that investigates OSVM performance for document classification. They stated that this approach "turns out to be surprisingly sensitive to specific choices of representation and kernel in ways which are not very transparent. [...] Since the difference in performance is very dramatic based on these choices, this means that the method is not robust without a deeper understanding of these representation issues". Additionally, OSVM assumes that the origin of the kernel-based transformed space represents the anomalous class, and the data can be linearly separated from the origin in this space. This assumption may be sub-optimal and can result in a degenerate solution, i.e., where $\mathbf{w} = 0$ and $b = 0$, if the used kernel is inappropriate. For example, mean-centering the kernel matrix may cause a degenerate solution [6]. An even more critical limitation is encountered with high-dimensional data points. We note that the dual formulation of the optimization process requires the computations of the data kernel matrix. Besides the quadratic cost of the kernel computation, in high-dimensional space, the data become sparse, and computing the relative distance may be less informative [6, 7]. The presence of irrelevant features may impact the performance of these classical methods and feature engineering and selection are critical in this setting [223]. Unfortunately, the feature selection requires prior knowledge about inliers and outliers, which is lacking in semi- and unsupervised AD (cf. Section 2.2.3.1).

Hybrid models have been proposed to improve the sub-optimal performance of classical methods for high-dimensional data. Two-step approaches are introduced in the literature. Firstly, a dimensionality-reduction-based method, e.g. PCA or AE, is used to retain the most important features and discard irrelevant ones. Then, a classical classification approach, e.g., OSVM or SVDD, is applied to this compressed representation to detect anomalous data. The main intuition here is to project the input data into a lower-dimensional feature space, where pairwise distance computation is easier and more constructive. The overall workflow of this paradigm is illustrated in Figure 2.11.

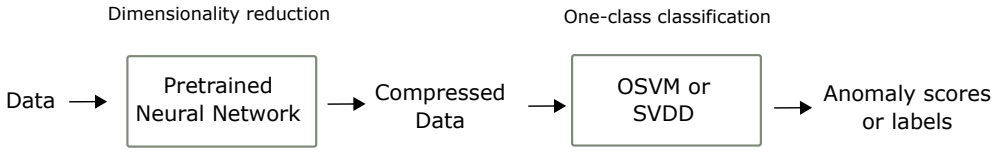


Figure 2.11: Hybrid model workflow.

The literature review shows that hybrid methods are more convenient than classical one-class classifiers, particularly with high-dimensional data [177, 96]. In this case, hybrid models improve

detection accuracy and reduce the overall training time. Nonetheless, a closer look reveals multiple shortcomings that need to be addressed. The first limitation relates to the first step of dimensionality reduction, which is not optimized for the task AD. Indeed, the compressed data representation is learned separately before optimizing the one-class classifier. As such, the first step focuses on reducing the data dimensionality and is unaware of the subsequent AD task. Important features that are irrelevant to the dimensionality reduction task but represent key information regarding anomalies may be disregarded [293, 58, 198]. The second shortcoming of these hybrid models consists in their sensitivity to training contaminants. The first step of dimensionality reduction assumes the availability of anomaly-free training data to train the underlying model. In the case when the data are contaminated with unknown outliers, which is common in real-world applications, the model may learn a biased representation of the norm, with a skewed lower-dimensional space where inliers and outliers are well-represented. That is, the extracted features become less distinguishable and it becomes difficult to segregate them with the subsequent one-class classifier AD [160].

Another promising line of research solely relies on artificial neural networks, where the representation learning objective is directly tailored to AD. The subsequent section presents a review of recent literature on neural network-based one-class AD.

2.3.4 Neural Network-Based One-Class Classification

The aim of this part is to review two of the most prevalent neural network-based one-class classifiers: One-Class Neural Network (OCNN) and deep Support Vector Data Description (dSVDD). In the following, we will introduce the concept and the functional form of artificial neurons and Feed-Forward Network (FFNs). Then, we will develop OCNN and dSVDD.

2.3.4.1 Artificial Neurons And Feed-Forward Neural Networks

Artificial neural networks are mathematical computational learning models that were initially developed to simulate information processing in biological systems [174]. Indeed, the biological nervous system is constituted of a large network of computational cells, denoted as *neurons*. Neurons receive stimuli from other neurons and learning new knowledge consists in adjusting the synaptic connection between these cells. Artificial neural networks build on an analogy with this biological system and propose a mathematical model, a.k.a., Perceptron model [174]. This most basic model is illustrated in Figure 2.12, extracted from [6].

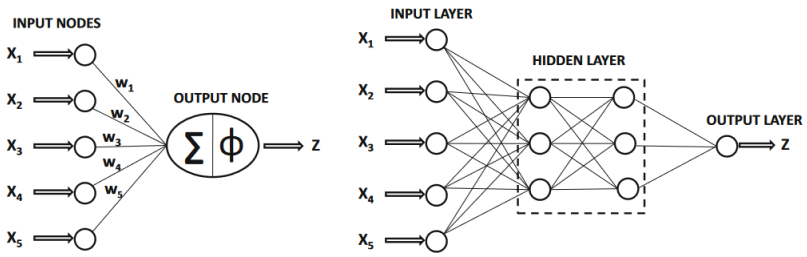


Figure 2.12: Single- (left) and multi-layer (right) neural networks, extracted from [6].

This model contains two layers of neurons: a first layer referred to as the input layer and a second layer denoted as the output layer, with a single artificial neural. The number of neurons in the input layer is equal to the dimension of the processed data. Formally, let $\mathbf{x} = (x_1, ..., x_d)$ a d-dimensional input data point, $\mathbf{x} \in \mathbb{R}^d$. The output neuron receives stimuli from the input neurons, with weighted synaptic connections $\mathbf{w} \in \mathbb{R}^d$. This cell performs a computations function and accordingly, outputs the result $o \in \mathbb{R}$ as follows:

$$o = \Phi(\mathbf{w}\mathbf{x} + b) = \Phi(\sum_i w_i x_i + b). \tag{2.23}$$

$b \in \mathbb{R}$ is the bias term and Φ is the *activation function*. This activation function is often a nonlinear function that is used to introduce nonlinearity in the learning process. The most common activation functions are the sigmoid, the hyperbolic tangent, and the Rectified Linear Unit (ReLU) functions.

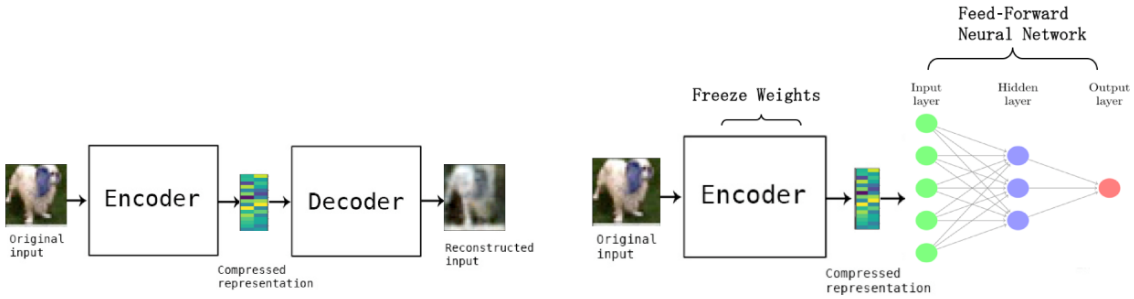
The Perceptron is the most basic architecture of neural networks. A more advanced architecture is the FFN, a.k.a., the Multi-Layer Perceptron (MLP). This model is constituted of multiple layers, with weighted synaptic connections between all the neurons of two adjacent layers (cf. Figure 2.12). The information can propagate only from the input to the output, i.e., from left to right, which justifies the name *feed-forward neural networks*. This model comprises at least three layers: an input layer, one or multiple hidden layers, and an output layer.

can be used to model any nonlinear representation, hence the name of "universal function approximators". These models aim to infer a complex nonlinear function from a finite set of data points. In general, the supervised learning paradigm is used for their training. The underlying data are labeled and the parameters of the model are iteratively adjusted to minimize the training error, i.e., the difference between the actual output and the expected one. For this reason, the gradients of the classification errors are computed and "back-propagated" to adjust the weights. The objective is to determine the global minimum of the training error that allows the model to generalize the inferred function to novel unseen data.

However, the classical supervised training of FFNs requires a labeled training subset, which is lacking in AD (cf. Section 2.2.3.1). OCNN and dSVDD are specific FFNs that was semi-supervised AD and their training does not require data labels. The following section presents an overview of these two models.

2.3.4.2 One-Class Neural Networks

A first attempt to define a one-class training objective of an end-to-end FFN was introduced by Chalapathy et al. [58]. The key insight consists in replacing the classical one-class classifier of the hybrid models with a single-layer FFN. Indeed, this approach is inspired from *deep learning-based transfer learning*, which stacks two neural networks: the first model is a deep neural network that is pre-trained on a large dataset and that serves as a generic feature extractor, and the second model is a classifier that is fine-tuned on the input data to perform a specific task[104].



(a) Autoencoder . (b) one-class neural networks.

Figure 2.13: The workflow of one-class neural networks, extracted from [58]

Building on this insight, the authors propose a neural network architecture comprised of a pre-trained encoder that compresses the high dimensional data, followed by a one-hidden-layer FFN (cf. Figure 2.13). They particularly propose a two-step training strategy that firstly trains a vanilla AE, which will developed later in Section 2.3.5.1, to reconstruct the nominal data and then uses the frozen parameters of the encoder as a feature extractor. These encoder outputs are fed to the second neural network classifier to segregate inliers and outliers. The proposed objective function of the second step is very similar to that of OSVM and is defined as follows:

$$\min_{\mathbf{w},\mathbf{v},b} \frac{1}{2}||\mathbf{w}||^2 + \frac{1}{2}||\mathbf{v}||^2 + \frac{C}{N}\sum_i \max\{0, b - \langle \mathbf{w}, s(\mathbf{v}\mathbf{x}_i)\rangle\} - b, \tag{2.24}$$

where $\mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ are the parameters of the hyperplane that separates inliers and outliers, $\mathbf{v}$ is the weight matrix of the classifier network, and s is the Sigmoid activation function. In fact, this objective function simply replaces the kernel mapping function of OSVM with the output of the new FFN.

However, this objective function is not convex and the optimization may get stuck in local optima. To address this issue, the authors propose an alternate minimization approach, which alternates between minimizing the parameters $\{\mathbf{w}, \mathbf{v}, b\}$. It first fixes the parameter b, i.e., the distance of the hyperplane to the origin, and optimizes the weights $\mathbf{w}$ and $\mathbf{v}$. Then, these optimal parameters are fixed and it minimizes the objective with respect to b. We refer the reader to [58] for a detailed description of the optimization process and the hyperparameter selection.

OCNN inherits most limitations of hybrid classifiers, principally the disjoint training of the encoder and the classifier (cf. Section 2.3.3). Besides, the non-convex formulation of the objective function may result in non-global optima and requires a more complex training strategy. An alternative approach, referred to as deep SVDD, which was inspired by the classical SVDD, was proposed in the literature.

2.3.4.3 Deep Support Vector Data Description

Ruff et al. [225] introduced dSVDD, that builds on the kernel-based SVDD and trains a FFN to find the optimal hypersphere that encloses the nominal data. The main distinction between SVDD and dSVDD is that the latter jointly learn a data representation that extracts the most useful features along with the one-class classification objective.

Let $\mathscr{X} \in \mathbb{R}^d$ be a d-dimensional input space, $\mathscr{F} \in \mathbb{R}^p$ be a p-dimensional output space, and $f(.,\mathbf{W}) : \mathscr{X} \to \mathscr{F}$ a neural network comprised of $L \in \mathbb{N}$ layers and defined with a matrix weights $\mathbf{W} = \{\mathbf{w}^1, ..., \mathbf{w}^L\}$, where $\mathbf{w}^l$ are the layer weights. The goal of dSVDD is to jointly learn the parameters $\mathbf{W}$ that project the input data into a lower-dimensional space $\mathscr{F}$ and minimize the volume of the hypersphere that encloses the nominal data. This sphere is defined with a radius $R \in \mathbb{R}^+$ and a center $\mathbf{c} \in \mathbb{R}^p$. All in all, dSVDD optimizes the following objective function:

$$\min_{R,\mathbf{W}} R^2 + \frac{C}{N}\sum_{i=1}^{N} \max\{0, ||f(\mathbf{x}_i,\mathbf{W}) - c||^2 - R^2\} + \frac{\lambda}{2}\sum_{l=1}^{L} ||\mathbf{w}^l||_F^2. \tag{2.25}$$

Similar to SVDD, this objective includes three terms: the first part minimizes the hypersphere volume R^2, the second part penalizes the points that fall outside the boundaries, and the last term is a regularization over the weight parameters $\mathbf{W}$.

Once the training is complete, the model can be applied to novel unseen samples to assess their anomalousness. The data is projected into the output space $\mathscr{F}$ and the distance of the representation from the center is used as the anomaly score, i.e., $||f(\mathbf{x}_i,\mathbf{W}) - c||^2$.

2.3.4.4 AD With One-Class Neural Networks And Deep Support Vector Data Description

Despite the outstanding popularity of OCNN and dSVDD-based AD in some fields, e.g., computer vision and video analysis [236], their application in network traffic analysis remains limited. Additional studies are required to understand more completely the potential of these approaches in this application domain.

2.3.4.5 Strengthes And Weaknesses Of One-Class Neural Networks And Deep SVDD

The advantages of these approaches are twofold. First, one-class methods and their deep variants are well explored in the literature, with a strong theoretical background and foundations.

Second, it is no longer required to select a suitable kernel function that is adapted to the underlying data. Deep models jointly learn the transformation from the data, which alleviates the limitation of the sensitivity to the kernel choice [201]. Besides, this reduces the quadratic complexity of the kernel matrix computation, which is more efficient with large training datasets. However, the main limitation of dSVDD is related to the assumption that the data can be enclosed in a hypersphere. Indeed, when the nominal data present complex distribution with a mixture of multiple behaviors, projecting these data into a single compact hypersphere may be sub-optimal and can lead to poor results [201]. Furthermore, in some situations, particularly when including the hypersphere center in the optimization (cf. Equation 2.25), the training of dSVDD may result in a trivial solution, a.k.a., hypersphere or mode collapse, where the network projects all the data points in the center of the hypersphere. As such, specific requirements are defined to prevent such collapses, e.g., only unbounded activation functions must be used and bias terms should be omitted [70].

More recent attention in the literature has focused on reconstruction-based AD. These methods will be developed in the following section.

2.3.5 Reconstruction-Based AD

Reconstruction-based AD, learns to compress the nominal data points into a low-dimensional representation and reconstruct the original data based on these compressed encodings. In other words, these methods learn to extract the most important information of the norm by mapping the data into a subspace of lower dimensionality, with the least reconstruction error. Since anomalies generally comprise non-representative features, it is harder to project them in this subspace without significant loss of information, which results in a larger reconstruction error.

Reconstruction-based anomaly detectors are extremely prevalent in the literature and various architectures have been proposed. This section presents the most common reconstruction-based models, namely AEs, VAEs, NFs, and sequence-to-sequence models.

2.3.5.1 Autoencoders

AEs, a.k.a., replicator and auto-associative neural networks, [33] are a specific type of artificial network networks that are developed for unsupervised dimensionality reduction. These neural networks have been introduced to learn an encoding function that projects the data into a lower-dimensional space, with a compressed representation, a.k.a., encoding. Then, this compressed representation is decoded back into the original space in order to *reconstruct* the original data. The aim is to find the optimal compression/decompression function with the minimal information loss, such that the reconstructed data and the input ones are similar.

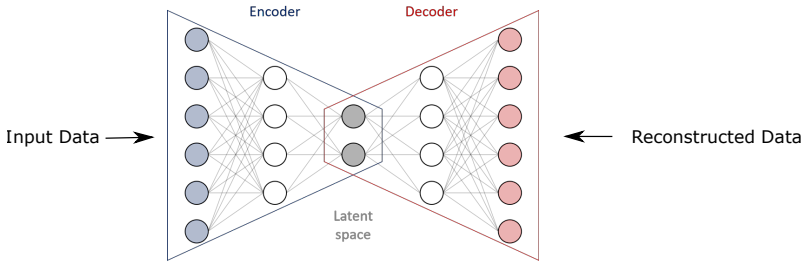


Figure 2.14: The architecture of AEs, adapted from [80].

Following this first description, AEs present an encoder-decoder architecture, constituted of two parts: a first "encoder" function $f(.)$ that maps the data into the latent space, and the second "decoder" function $g(.)$ that reconstructs the input. The overall architecture is displayed in Figure 2.14. The objective is to minimize the *reconstruction error*, i.e., the difference between the initial

data $\mathbf{x} \in \mathbb{R}^d$ and the reconstructed data $g(f(\mathbf{x}))$:

$$\min_{f,g} ||\mathbf{x} - g(f(\mathbf{x}))||_2^2. \tag{2.26}$$

One trivial solution to this optimization problem is the the identity functions, i.e., $f(\mathbf{x}) = x$ and $g(\mathbf{x}) = x$. However, this solution is not useful for the underlying representation learning task. To prevent this trivial solution, a form of regularization is required. The most common regularization introduced in AEs is to constrain the dimension of the latent space to be less than the input dimension, a.k.a., a *bottleneck*. As such, the model is forced to keep only the most relevant information required to reconstruct the data, and consequently learns the data informative properties. It is noteworthy that there exist numerous other regularization techniques, such as sparse encoding [191] and denoising AEs [263].

Generally, the encoder and the decoder of AEs are two FFN comprising multiple layers, each of which is followed with a non-linear activation function. This non-linearity allows the AE to learn a non-linear manifold and extract the non-linear data properties. Multiple possible configurations have been proposed in the literature. We refer the reader to this survey [28], which reviews the most common types of AEs. AEs can be trained in an end-to-end manner or trained layer-by-layer separately, a.k.a., stacked AEs.

Recently, generative AEs, and particularly VAEs and NFs, have been extensively used in AD [207]. They allow to model the underlining generative process of the nominal data, and outliers are extreme observations that violate this generative model. Even though their global design is similar to the classical discriminative AEs, an additional generative assumption about the data is required. The following section introduces the variational inference setting and focus on VAEs and NFs.

2.3.5.2 Variational Autoencoders And Normalizing Flows

In general, generative models aim to find the optimal parameters θ that maximize the likelihood $p_\theta(\mathbf{x}) = \mathbb{E}_{p(\mathbf{z})}[p_\theta(\mathbf{x}|\mathbf{z})]$, where $\mathbf{z}$ is the model latent variable and $p(\mathbf{z})$ is a predefined prior. They empirically select the optimal parameters that maximize the log-likelihood

$$\hat{\theta} = \text{argmax}_\theta \log p_\theta(\mathbf{x}). \tag{2.27}$$

However, this likelihood is intractable because of the marginalization over the latent variable $\mathbf{z}$:

$$p_\theta(\mathbf{x}) = \mathbb{E}_{p(\mathbf{z})}[p_\theta(\mathbf{x}|\mathbf{z})] = \int p_\theta(\mathbf{x}|\mathbf{z}) p(\mathbf{z}) d\mathbf{z} \tag{2.28}$$

Variational inference aims to approximate the posterior probability $p(\mathbf{z}|\mathbf{x})$ with a parametric distribution $q_\phi(\mathbf{z}|\mathbf{x})$, parameterized by ϕ. Regardless of the choice of the latent distribution, we can reformulate the log-likelihood as follows:

$$\log p_\theta(x) \geq \mathbb{E}_q[\log p_\theta(x|z)] - \mathbb{D}_{KL}[q_\phi(z|x)||p(z)] = -\mathscr{F}(x), \tag{2.29}$$

where $q_\phi(z|x)$ is the approximate posterior distribution for the latent variables, and $\mathscr{F}$ is the negative free energy, a.k.a., the evidence lower bound (ELBO). This energy comprises two terms. The first term is the reconstruction error, and the second one represents the KL divergence between the approximate distribution and the prior distribution.

A common choice of the approximate posterior distribution $q_\phi(\mathbf{z}|\mathbf{x})$ is the multivariate Gaussian with the diagonal covariance structure:

$$\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x}) = \mathscr{N}(\mathbf{z}; \mu, \sigma^2 \mathbf{I}), \tag{2.30}$$

μ is the mean of the approximate posterior, and $\sigma^2\mathbf{I}$ is the diagonal covariance matrix.

VAE, introduced in [133], can be seen as a probabilistic version of vanilla AEs, where the encoder (cf. $e_\mathbf{z}$ in Figure 2.15) models the data in the latent space with a parametric probability

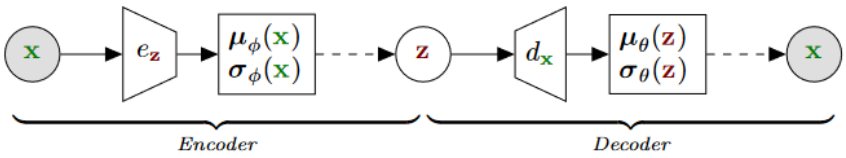


Figure 2.15: The architecture of variational autoencoders, extracted from [103].

distribution, i.e., the multivariate Gaussian with the diagonal covariance. Similarly, the decoder (cf. $d_{\mathbf{z}}$ in Figure 2.15)) learns to reconstruct the input data using latent representations sampled from the latent distribution. To this end, the decoder estimates the parameters of the output distribution, represented by a second multivariate Gaussian. This probabilistic formulation allow the model not only to reconstruct the initial data but also to generate novel data points, hence the name "generative AE". VAE training consists in minimizing the training data free energy defined in Equation 2.29.

VAEs assume that the data can be well-modeled in the latent space with a Gaussian distribution, In some cases, the input data present multiple complex patterns, which can not be accurately represented with such parametric distribution (cf. Section 2.3.2.1).

To model more complex nominal patterns, NFs propose a richer parametric family of approximate posterior distributions. NFs transform an initial density function, i.e., the classical Gaussian distribution, to a more sophisticated one, by applying a sequence of invertible transformations (cf. Figure 2.16). Formally, let $\mathbf{z}_0 \in \mathbb{R}^d$ be a random variable that follows a probability distribution

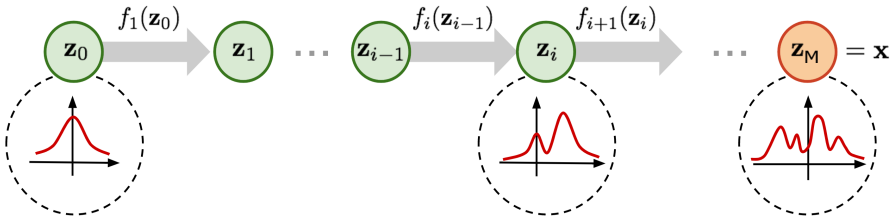


Figure 2.16: An example of a normalizing flow model adapted from [273]

$q_0(\mathbf{z}_0)$ and $f : \mathbb{R}^d \mapsto \mathbb{R}^d$ an invertible mapping. We can transform $\mathbf{z}_0$ to $\mathbf{z}_k$ by applying a series of $M \in \mathbb{N}^*$ mappings:

$$\mathbf{z}_k = f_M \circ \ldots \circ f_1(\mathbf{z}_0). \tag{2.31}$$

The probability distribution of the novel vector $\mathbf{z}_k$ is defined as follows:

$$q_k(\mathbf{z}_k) = q_0(\mathbf{z}_0) \prod_{k=1}^{M} \left| \det \frac{\partial f_k}{\partial \mathbf{z}_{k-1}} \right|^{-1}, \tag{2.32}$$

where $q_0(\mathbf{z}_0)$ is the initial distribution before applying the M mapping. Generally, $q_0(\mathbf{z}_0) = \mathcal{N}(z; \mu, \sigma^2 \mathbf{I})$. Thus, the log likelihood of the transformed density $q_M(\mathbf{z}_M)$ can be written as :

$$\log q_M(\mathbf{z}_M) = \log q_0(\mathbf{z}_0) - \sum_{k=1}^{M} \left| \det \frac{\partial f_k}{\partial \mathbf{z}_{k-1}} \right|. \tag{2.33}$$

Finally, the free-energy function can be rewritten as :

$$\mathscr{F}(x) = \mathbb{E}_{q_0(\mathbf{z}_0)}[\log q_0(\mathbf{z}_0) - \sum_{k=1}^{M} \log \left| \det \frac{\partial f_k}{\partial \mathbf{z}_{k-1}} \right| - \log p(\mathbf{z}_M)] - \mathbb{E}_{q_0(\mathbf{z}_0)}[\log p(\mathbf{x}|\mathbf{z}_M)]. \tag{2.34}$$

In practice, and for a scalable inference, NFs must satisfy several requirements. Particularly, the transformation needs to be invertible and with easy computation of the Jacobian determinant $\det \frac{\partial f_k}{\partial \mathbf{z_{k-1}}}$. Several architectures have been proposed in the literature, including planar and radial flows [216], coupling flows [83], autoregressive flows [135]. For example, planar flows [215] transform the latent variable $\mathbf{z}$ as follows:

$$f(\mathbf{z}) = \mathbf{z} + \mathbf{u}h(\mathbf{w}^T\mathbf{z} + b), \tag{2.35}$$

where $\mathbf{w} \in \mathbb{R}^d$, $\mathbf{u} \in \mathbb{R}^d$, and $b \in \mathbb{R}$ are trainable parameters, and $h(.)$ is the non-linear *tanh* function. The Jacobian determinant of this transformation is:

$$\left|\det \frac{\partial f}{\partial \mathbf{z}}\right| = |1 + \mathbf{u}^T \psi(\mathbf{z})|, \tag{2.36}$$

where

$$\psi(\mathbf{z}) = h'(\mathbf{w}^T\mathbf{z} + b)\mathbf{w} \tag{2.37}$$

Using equation 2.33, we deduce the novel density:

$$\log q_M(\mathbf{z_M}) = \log q_0(\mathbf{z_0}) - \sum_{k=1}^{M} \left|1 + \mathbf{u}^T \psi(\mathbf{z}_{M-1})\right|. \tag{2.38}$$

In the case of NFs, the posterior distribution $q_\phi(\mathbf{z}|\mathbf{x})$ is approximated with the flow density $q_M(\mathbf{z_M})$, i.e., $q_\phi(\mathbf{z}|\mathbf{x}) \cong q_M(\mathbf{z}_M)$. Consequently, the free energy is given by [216]:

$$\mathscr{F}(\mathbf{x}) = \mathbb{E}_{q_0(\mathbf{z}_0)}[\log q_0(\mathbf{z}_0)] - \mathbb{E}_{q_0(\mathbf{z}_0)}[\sum_{k=1}^{M} \log \left|\det \frac{\partial f_k}{\partial z_{k-1}}\right|] - \mathbb{E}_{q_0(\mathbf{z}_0)}[\log p(\mathbf{x}, \mathbf{z_M})]. \tag{2.39}$$

2.3.5.3 Anomaly Detection With AEs, VAEs, and NFs

Reconstruction-based AE involves a two-step approach. Firstly, an AE, VAE, or NF, is trained to reconstruct the nominal network traffic data. Secondly, the reconstruction scores of this model are used as anomaly scores: the higher the reconstruction score, the higher the probability that the observation is anomalous. Multiple popular AE-based architectures have been proposed in the AD literature, e.g., Deep Autoencoding Gaussian Mixture Model (DAGMM) [293], Adversarial AEs [163], USAD [21], MSCRED [291].

A series of recent studies has explored the efficacy of these models in network traffic AD [244, 23, 113, 66]. Song et al. [244] have discussed the design of an AE-based NIDS. The performance of this approach was empirically assessed on three intrusion detection benchmark datasets: the NSL-KDD [196], the IoTID20 [259], and the NB-IoT [176] datasets. In this study, AEs have shown very competitive results on the three datasets. Additionally, the authors demonstrated that the results are significantly sensitive to the model capacity and the number of neurons in the latent layer. They particularly observed that performance tends to improve on these datasets with higher model sizes, i.e., with more trainable parameters. As such, the model design and the overall architecture must be carefully selected to grant optimal detection performance. A recent study by Altaha et al. [14] has compared the accuracy of AEs in detection network attacks, against different supervised deep learning algorithms, including FFNs, Convolutional Neural Networks (CNNs), and Long Short Term Memory based Recurrent Neural Networks (LSTM-RNNs). For this purpose, they installed a network testbed comprising three computers and they used a Kali Linux machine to execute network attacks: DoS and malicious packet injection. AEs show better results than the other neural networks. Another AE-based IDS was proposed by Choi et al. [68]. Four well-known architectures were explored: a basic AE, denoising AE, stacked AE, and variational AE. The four models share the same architecture, with three hidden layers comprised of 32, 16, and 32 units, followed by the

ReLU activation function. The experiments were conducted on the NSL-KDD dataset [196]. The four models show relatively similar performances and the best results are reported with the stacked AEs.

Zavrak et al. [288] proposed a VAE-based anomaly detector designed to detect unknown network attacks from flow metadata. The authors conducted an empirical experiment on the CICIDS17 benchmark dataset, where they compared VAE, vanilla AE, and OSVM performances. Overall, VAE show the best performance is reported, with a higher AUROC.

2.3.5.4 Sequence-to-sequence Models And Transformers

All the aforementioned methods focus on the problem of AD in iid data, where it is supposed that there is no chronological order between the samples, and each data point can be analyzed independently from the other points. This assumption can be violated in practice, where there may exist data temporal dependencies, and modeling these relationships is crucial to detect contextual and collective outliers (cf. Section 2.2.2). In this section, we present the most common reconstruction-based approaches that address this problem: sequence-to-sequence models and Transformers. More comprehensive reviews can be found in [41, 156, 144, 69, 46, 74]. This topic will be revisited in Chapter 5 of the thesis.

Despite a similar outline, anomaly detection in temporal-data presents key characteristics that set it aside from that in non-temporal data. Indeed, the criterion used to detect anomalies in non-temporal data is according to their deviations from the rest of the data. The task is slightly different in time series data. Here, the context is key, where a sample observed at a timestamp t can influence the immediate following samples occurring in the following timestamps. A data point or a set of data points that are considered anomalous in a specific context can be nominal in another one. Let us consider an example of a contextual anomaly in the monthly values of temperature, illustrated in Figure 2.17. A low value of temperature recorded in December, cf. t_1 in Figure 2.17, is normal. However, this same value reported in June, i.e., t_2 in Figure 2.17, is considered anomalous.

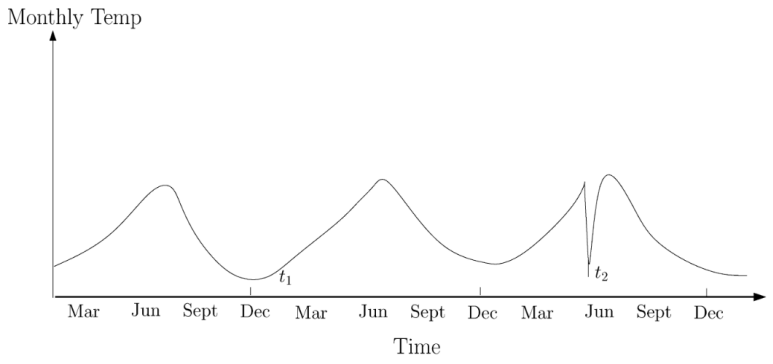


Figure 2.17: An example of a contextual anomaly in a temperature time series data, extracted from [63].

Sequence-to-sequence models [247] are among the most common approaches used to model sequential data. These models have an architecture that combines AEs and Recurrent Neural Network (RNNs), as illustrated in Figure 2.18. The encoder and the decoder of this model are two RNNs, e.g., LSTM-RNNs or Gated Recurrent Unit based Recurrent Neural Networks (GRU-RNNs). The encoder learns a latent representation of the input sequence by updating its hidden state and the decoder learns to regenerate the original data using the last hidden state of the encoder. Similar to vanilla AEs, the training of these sequence-to-sequence models consists in minimizing the reconstruction error (cf. Equation 2.26).

Within the reconstruction-based category, Transformers [260] have shown exemplary performance on diverse tasks and various domains such as Natural Language Processing (NLP), computer

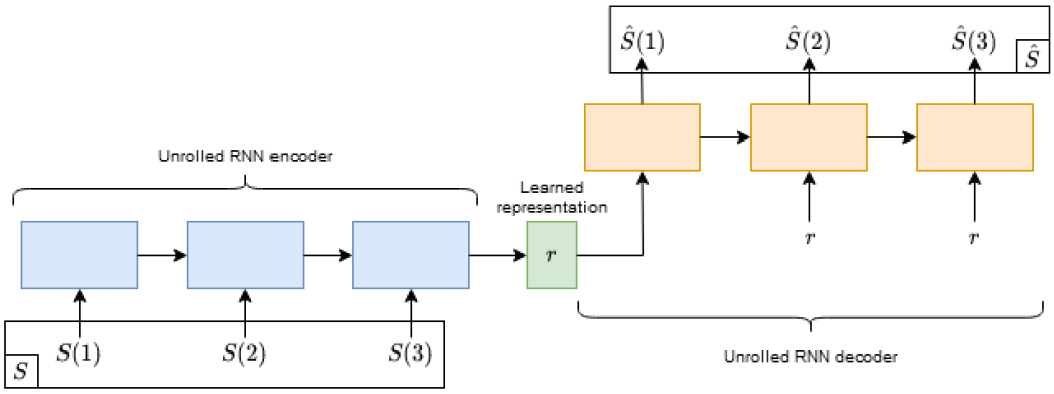


Figure 2.18: Sequence-to-sequence model, extracted from [73].

vision, audio processing, and AD [152]. Recently, these models have attracted a growing attention in the literature and a wealth of recent surveys have been proposed to recover Transformer-based approaches and architectures [152]. Indeed, this increasing interest resulted in various derivative models tailored to AD, e.g., TranAD [258] and AnomalyTransformer [279]. In the following section, we aim to introduce the vanilla architecture of Transformers and its components. Then, we review the latest studies that apply Transformers to network traffic AD. Finally, we conclude this part with the strengths and weaknesses of these models.

The classical Transformers [260] are sequence-to-sequence models with a multi-layered encoder-decoder architecture. The overall architecture of Transformers is depicted in Figure 2.19. Both encoder and decoder are constituted by stacking a set of identical blocks. Each bloc comprises two sub-layers: a multi-head self-attention layer and a position-wise FFN. In the following subsections, we will present the main building components of the vanilla Transformer.

Input Embedding And Positional Encoding The input sequence comprises an ordered set of tokens, e.g., words in text processing or flows in network traffic analysis. The first step is to convert these tokens into a unified format, defined with a numerical vector representation. The objective of the embedding layer is to map the input data into a multi-dimensional space of dimension $d_{model} \in \mathbb{N}$. In this space, semantically similar tokens should have *close* representations. This transformation is modeled in vanilla Transformers with a linear layer parameterized with a set of learnable parameters.

Then, the positional encoding module is used to model the token order in the sequence. Since no recurrence is involved, this operation can be seen as an additional information injection to explicitly provide the model with the temporal order of the sequence data points. Vanilla Transformers use cosine and sine functions to model the location of a token in a sequence. Indeed, the authors propose encoding each dimension of the encoding vector with a specific sinusoid. Let $\mathbf{x}_{pos} \in \mathbb{R}^{d_{model}}$ be a d-dimensional input vector, where $pos \in \mathbb{N}$ is the position of the token in the sequence Even dimensions $1 \leq 2i \leq d_{model}$ of $\mathbf{x}_{pos}$ are represented with the following vector:

$$\mathbf{PE}(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right). \tag{2.40}$$

Odd dimensions $1 \leq 2i+1 \leq d_{model}$ are modeled as follows:

$$\mathbf{PE}(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right). \tag{2.41}$$

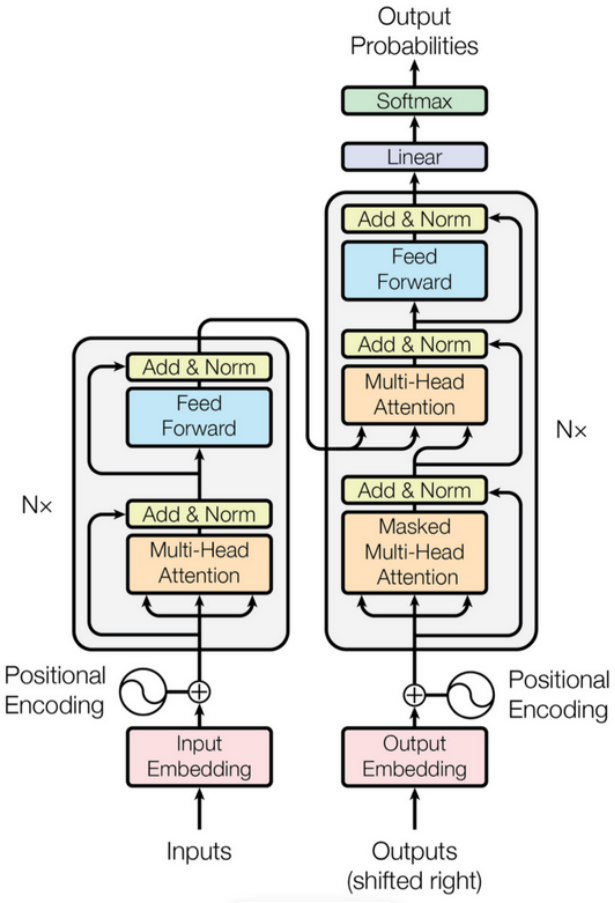


Figure 2.19: The transformer architecture, extracted from [260].

The authors justify the choice of these functions because they present linear properties and a linear offset $\mathbf{PE}(pos+k,.)$, where $k \in N$, can be reformulated as a linear function of $\mathbf{PE}(pos,.)$. It is noteworthy that other positional encoding strategies have been proposed in the literature. A detailed overview of these methods is provided in [89].

Attention Modules The fundamental idea that contributed to the development of Transformers is the *attention*. This module has mainly contributed to the extraction of *long-term dependencies*, a main challenge encountered by classical RNNs [128]. Let us introduce the key idea behind attention and the functional form of this layer.

Attention [25] was initially inspired by human biology. Indeed, the human visual system tends to selectively focus on some salient part of an image, and ignores all other less important parts, which are irrelevant for the perception task [280]. Likewise, in NLP, understanding the meaning of a world requires focusing on the adjacent context. Artificial attention was introduced to allow artificial neural networks to model this notion of relevance by paying more attention to relevant information of an input [64].

From a mathematical point of view, vanilla Transformers use Query-Key-Value (QKV) attention model. Given three data matrices, a Query matrix $\mathbf{Q} \in \mathbb{R}^{N \times d_{model}}$, a Key matrix $\mathbf{K} \in \mathbb{R}^{N \times d_{model}}$, and a Value matrix $\mathbf{V} \in \mathbb{R}^{N \times d_{model}}$, the attention score is computed with the scaled dot-product attention, given by:

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{model}}})\mathbf{V} = \mathbf{A}\mathbf{V}, \tag{2.42}$$

where $\mathbf{A} = \text{softmax}(\frac{\mathbf{QK}^T}{\sqrt{d_{model}}})$ is called the *attention matrix*, which quantifies the similarity between the key and the query matrices. In other words, for three matrices $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$, the attention computes the dot-product of the queries and the keys and rescales it with the softmax function. Then, the output is defined as the weighted sum of $\mathbf{V}$ with the attention scores.

Self-Attention Vanilla Transformers introduce the notion of *self-attention*, which is as an attention layer where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are defined using the same sequence: the input $\mathbf{X}$. Indeed, the data $\mathbf{X}$ are firstly projected with three matrices $\mathbf{W}^Q \in \mathbb{R}^{d_{model} \times d_k}$, $\mathbf{W}^K \in \mathbb{R}^{d_{model} \times d_k}$, and $\mathbf{W}^V \in \mathbb{R}^{d_{model} \times d_k}$, as follows:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^Q, \mathbf{K} = \mathbf{X}\mathbf{W}^K, \mathbf{V} = \mathbf{X}\mathbf{W}^V. \tag{2.43}$$

Then, the self-attention output is computed using Equation 2.42. Given an input sequence, the self-attention allows quantifying the relevance of each token with other tokens of the same sequence.

Multi-Head Self-Attention Instead of using a single attention function, vanilla Transformers use *multi-head* attention. The authors found that it is more optimal to learn $h > 1$ projections that transforms the d_{model}-dimensional query, key, and value matrices into a new space of dimension $d_k = \frac{d_{model}}{h}$. Each projection is then processed with an attention *head* according to Equation 2.42. The model finally concatenates all the head outputs into a single representation and projects it back into the original d_{model}-dimensional space. Formally, the multi-head attention process is defined as follows:

$$\text{MultiHeadAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, ..., \text{head}_h)\mathbf{w}^O, \tag{2.44}$$

$$\text{where head}_i = Attention(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{k}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V). \tag{2.45}$$

$\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, and $\mathbf{W}_i^V \in \mathbb{R}^{d_{model} \times d_k}$ are the respective head projection matrices, $\mathbf{W}^O \in \mathbb{R}^{hd_k \times d_{model}}$ is the output matrix, and Concat is the concatenation function. According to the authors, multi-head attention has the advantage of leveraging diverse information from the h projection sub-spaces, which results in a more informative representation. Vaswani et al. [260] use $h = 8$ and $d_{model} = 512$.

Masked Self-Attention The Transformer decoder introduces a layer called *masked multi-head self-attention*. Since the output sequence is autoregressively generated, this layer is used to prevent the decoder from attending to future tokens and restrict the attention to previous positions. For this reason, the authors propose a *look-ahead mask* function that hides the sequence of future tokens. Concretely, this is done by setting $[\frac{\mathbf{QK}^T}{\sqrt{d_{model}}}]_{ij} = -\infty$ when $i < j$, i refers to the matrix row and j to the matrix column. Consequently, applying the softmax function to this masked matrix outputs an upper triangular matrix, where the attention values of the upper diagonal elements, corresponding to future tokens, are equal to zero.

FFNs Multi-head attention layers are followed by FFN, containing two linear transformations and a ReLU function in between. Let $\mathbf{X} \in \mathbb{R}^{N \times d_{model}}$ a data matrix, the output of this layer is:

$$FFN(\mathbf{X}) = ReLU(\mathbf{X}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \tag{2.46}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d_{model} \times d_f}$ and $\mathbf{W}_2 \in \mathbb{R}^{d_f \times d_{model}}$ are the weight matrices of the linear projections, $\mathbf{b}_1 \in \mathbb{R}^d_f$ and $\mathbf{b}_2 \in \mathbb{R}^d_{model}$ are the biases, and d_f is the dimension of the latent layer. This dimension is set to $d_f = 2048$ in [260] and typically selected to be larger than d_{model}.

Residual Connection And Normalization The Transformer comprises a stack of multiple blocks with many layers. To mitigate a potential gradient vanishing problem, a residual connection [112] is employed after each layer. which is followed by Layer Normalization [24]. All in all, for an input matrix $\mathbf{X}$, the "Add Norm" performs the following computations:

$$AddNorm(\mathbf{X}) = \text{LayerNorm}(\text{MultiHeadAttention}(\mathbf{X}) + \mathbf{X}), \tag{2.47}$$

where LayerNorm is the conventional layer normalization strategy introduced in [24].

2.3.5.5 Anomaly Detection With Sequence-to-sequence Models And Transformers

Transformers have achieved astounding results in numerous artificial intelligence fields and application areas such as NLP, computer vision, and audio processing [152]. Recently, a wealth of studies has explored the efficacy of this model in time series AD [272].

Tan et al. [250] propose an approach called Attention for Network Intrusion Detection (ANID), which applies Transformers to network flow-based detection. The authors compare the performance of this model against two classical baselines: BLSTM-RNNs and conditional random fields, on the CICIDS17 benchmark dataset. The empirical evaluation revealed the advantage of Transformers, with higher precision and recall, and lower false positive rates. Besides the better detection performance, Transformers benefit from parallel computing and report shorter training and testing time, compared to BLSTM-RNNs. The former speed of training is 17 s/epoch while the latter is 22 s/epoch. Likewise, a series of recent studies have shown the advantage of using Transformers over classical methods in network intrusion detection [150, 250, 167]. Benefiting from the self-attention mechanism and parallel computations, Transformer-based anomaly detectors show a higher detection performance and a more efficient training process.

Besides vanilla Transformers, some publications, e.g., TranAD [258] and MT-RVAE [268], propose combining the Transformer-based architecture with common generative models, Generative Adversarial Networks (GANs) and VAEs, to further improve the model performance. For example, Tuli et al. [258] propose an adversarial training of a model constituted of two Transformers. While one sub-network learns to accurately reconstruct the time series data, the second sub-network goal is to distinguish between the original data and the artificial reconstructed ones by the first model. According to the authors, this strategy allows the model to be more robust to noisy data and to gain training stability. Furthermore, an ablation study shows that the use of Transformers as backbone models in this approach is key to extracting long-range dependencies and reliably detecting outliers. As far as we know, no previous research has investigated the efficacy of these variations in network intrusion detection. Additional studies are required to understand their potential in this field.

2.3.5.6 Strengths And Weaknesses Of Reconstruction-Based AD

Among the existing FFNs, AE-based AD has become ubiquitous thanks to multiple assets. To begin with, AEs are generic and can be applied to different types of data. The key intuition of this paradigm is elementary and straightforward, and thence the common use of these approaches in multiple applications. Then, multiple architectures have been proposed, which results in a wide choice of AE-based approaches available for researchers and industry players. In spite of that, there are still some important issues that should be addressed. The main limitation concerns their sensitivity to training anomalies. Indeed, these methods assume the availability of anomaly-free training data. However, even a few contaminants can completely skew the projection so that to minimize their reconstruction errors, a.k.a., the problem of anomaly masking [61]. In theory, and AEs are universal approximators, and with enough capacity, they can be flexible enough to model the minority of training contaminants. In this case, it is no longer possible to distinguish inliers and outliers on the basin of the reconstruction error. The robustness of these neural networks will be discussed in more detail later in the thesis. In short, it is of paramount importance to develop more robust AE-based approaches, which are not influenced by noisy data and training anomalies. This has been previously assessed only to a limited extent due to the peculiar difficulty of this task.

2.4 Conclusion

In this chapter, we presented a global overview of the AD problem. Three main aspects were investigated to provide an overall understanding of this challenging task: the data, the different anomaly categories, and the literature AD methods. We first discussed the IP traffic data structure and we explored the most common anomaly causes in this field. We advocated the design of generic models, where no prior knowledge about the anomalous class is required. Then, we presented a more generic taxonomy of AD. We provided the most popular generic definitions of an anomaly, we clarified the distinction between the different terminology used in the literature, between anomaly, outlier, novelty, and noise, and we discussed the main assumptions followed to detect anomalous data. Then, we provided a broad categorization of anomalies based on their inherent characteristics. In particular, three main categories were identified: punctual, contextual, and collective anomalies. Finally, we provided a global overview of the AD methods of the broad literature. We started by reviewing the methods tailored to punctual AD. We have grouped these approaches into different families. For each family, we introduced the overall theoretical principle and the most recent publications that apply these approaches to network traffic AD. More importantly, we meticulously discussed their strengths, weaknesses, and challenges. The same process has been repeated for contextual and collective AD, where we extended our review to time series data.

From our review, we shed the light on a central limitation that hinders the performance of the literature methods: the critical sensitivity to training contamination. This shortcoming appears when the training data contain an unknown ratio of outliers, and significantly degrades the model performances. In the following chapters, we tackle this robustness problem by the mean of robust reconstruction-based representation learning.

3

Robust Autoencoders for Unsupervised Anomaly Detection

"It is a part of probability that many improbable things will happen."
Aristotle

Contents

3.1 Problem Statement 61
3.1.1 Nature of Input Data and Anomalies 61
3.1.2 Unsupervised Anomaly Detection 62
3.1.3 Robustness to Training Contamination 63
3.2 Related Work 64
3.2.1 Principal Component Analysis (PCA) 64
3.2.1.1 PCA Overview 64
3.2.1.2 PCA-based Anomaly Detection 65
3.2.1.3 PCA Sensitivity to Training Outliers 66
3.2.2 Robust Principal Component Analysis (RPCA) 66
3.2.2.1 RPCA Overview 66
3.2.2.2 RPCA-based Anomaly Detection In Network Traffic 67
3.2.2.3 RPCA Limitations 68
3.2.3 Robust Deep Autoencoders (RDAs) 68
3.2.3.1 RDA Overview 68
3.2.3.2 RDA-based Anomaly Detection In Network Traffic 70
3.2.3.3 RDA Limitations 70
3.3 Contribution: RADON, Robust Autoencoder with Dynamic Outlier filteriNg 71
3.3.1 RADON Training Strategy 71
3.3.2 RADON Projection Strategy 72
3.3.3 Unimodal Thresholding of the Reconstruction Histogram 73
3.3.4 Hypotheses 74
3.4 Experiments and Results 74

3.4.1 NSL-KDD dataset 74
3.4.1.1 Dataset Description 74
3.4.1.2 Training Protocols 75
3.4.1.3 Training Parameter Settings and Evaluation Criteria 76
3.4.1.4 Results 77
3.4.2 MedBIoT dataset 78
3.4.2.1 Dataset Description 79
3.4.2.2 Training Protocols 79
3.4.2.3 Training Parameter Settings and Evaluation Criteria 79
3.4.2.4 Results 80
3.5 Conclusion 81

As we have seen in the previous chapter, anomaly detection is a tricky problem. Detecting anomalous data points helps researchers and industrials acquire more precise knowledge about the data and define an appropriate strategy to anticipate and prevent potential failures. As such, machine-learning-based anomaly detectors are nowadays widely used in multiple application areas, such as healthcare, cybersecurity, and autonomous vehicles [51]. However, most machine learning-based anomaly detectors make strong assumptions and assume ideal training conditions, where the training data are completely anomaly-free, i.e., only contain nominal data points. Such an assumption is unfortunately not guaranteed in a real-world uncontrolled environment, including LANs, where the collected data may be contaminated with various outliers. Guaranteeing that no anomalies are infiltrated in the training can be laborious, if not impossible, due to the large volume of data and the constant emergence of novel unknown anomalies. Therefore, it is advocated to develop robust unsupervised anomaly detectors that are insensitive to training contaminants [54, 92].

In this chapter, we focus on this robust detection of anomalous instances, under imperfect conditions, where the training data comprise an unknown ratio of outliers. We particularly propose RADON, a Robust Autoencoder with Dynamic Outlier filteriNg.

The present chapter is organized as follows. Section 3.1 formalizes the problem that we address: we identify the characteristics of the analyzed data (cf. Section 3.1.1), we define the underlying problem of unsupervised AD (cf. Section 3.1.2), and we focus on the precise definition of robustness retained for our contributions (cf. Section 3.1.3). Section 3.2 introduces the most relevant methods tailored to this problem. We mainly review three dimensionality-reduction-based anomaly detectors: PCA, RPCA [54], and RDA [10]. Then, we present in Section 3.3 the first contribution of the thesis: RADON, which seeks to improve the performance of the classical RDAs. Section 3.4 depicts the experimental protocols and results. Finally, conclusions and perspectives are drawn in the last section.

We highlight that the contributions of this chapter have been published in [183].

3.1 Problem Statement

In this chapter, we focus on unsupervised anomaly detection of multivariate non-sequence data. Let us firstly characterize the nature of the input data (cf. Section 3.1.1). We then formalize the problem of unsupervised AD (cf. Section 3.1.2) and define what "robustness" means in our study (cf. Section 3.1.3).

3.1.1 Nature of Input Data and Anomalies

The nature of the data significantly drives the choice of the adequate anomaly detector. Thus, data peculiarities must be formalized before defining the underlying algorithm. In the previous chapter (cf. Section 2.1), we thoroughly reviewed the different characteristics of network traffic metadata and we detailed the three categories of anomalies: punctual, contextual, and collective anomalies (cf. Figure 3.1). We limit the scope of this chapter to the analysis of multivariate non-sequence data. We do not consider temporality to make the problem more tractable. This simplifying assumption is followed by numerous studies to facilitate the downstream task of AD [201, 51, 222, 229]. Then, we will relax this restriction in the last chapter of the thesis, to extend our analysis to sequential data. Since we consider non-sequential data, this chapter particularly addresses punctual AD.

Non-sequence data are defined as a set of unordered records [119]. As we assume that the data are independent, we presume that permutating the order of training samples is inconsequential and

therefore, the expected values are not influenced by the contextual dependencies.

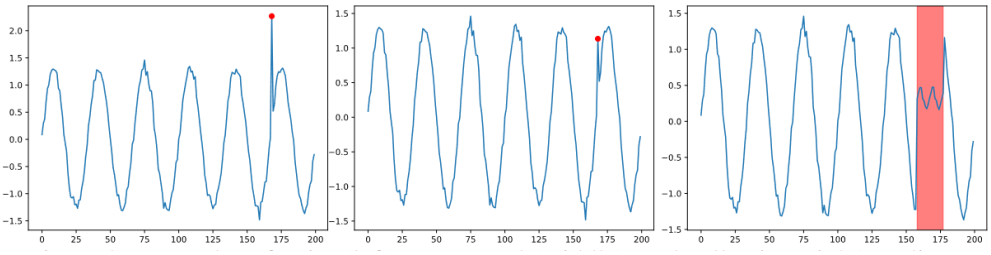


Figure 3.1: Illustration of punctual (left), contextual (middle), and collective (left) anomalies, extracted from [144]

Finally, the non-sequence data may be univariate or multivariate: each observation $\mathbf{x}_i$ can have one single attribute, i.e., $d = 1$, or several attributes, i.e., $d > 1$. For example, univariate data can be an unordered set of the lengths of network packets sent by a connected device. A multivariate set may contain multiple variables such as the length of the network packet, the IP version, and the TCP flags (e.g., *syn*, *urg*, *ack*). In this thesis, we consider the more general setting of multivariate AD, since we extract multiple network traffic metadata from each network flow. In other words, each network flow is encoded using a multidimensional vector, where each feature represents one metadata attribute. The comprehensive list of the extracted features is detailed in Section 2.1 and visualized in Table **??**.

3.1.2 Unsupervised Anomaly Detection

We provide in the present section the characteristics of the methods adapted to this problem of multivariate unsupervised anomaly detection.

First of all, the development of the adequate anomaly detector depends on the availability of data labels. We consider in this thesis unsupervised AD. Indeed, when data are collected from uncontrolled real-world environments, e.g., network traffic collected from a customer LAN, accurate labeling of each observation is not feasible. Continuously labeling this large set of network flows by experts is arduous and costly. Furthermore, new anomalies are constantly emerging, e.g., zero-day attacks, for which experts have no prior knowledge to manually annotate.

Unsupervised AD can be formalized into the following problem:

Problem 3.1 *Let $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_i, ..., \mathbf{x}_N)$ be $N \in \mathbb{N}^*$ **iid** observations of dimension $d \in \mathbb{N}^*$. Unsupervised anomaly detectors aim at learning a function that distinguishes inliers from outliers. This function f associates to each data point $\mathbf{x}_i$ an output s_i:*

$$f : \mathbb{R}^d \to \mathbb{R}, \quad \mathbf{x}_i \mapsto s_i = f(\mathbf{x}_i). \tag{3.1}$$

This output can be one of two types [6]:

- *Outlier score: the output of the function f is a score $s_i \in \mathbb{R}$ that quantifies the degree of anomalousness of the input observation $\mathbf{x}_i$. The higher the score s_i, the more likely that the corresponding observation is anomalous. The majority of AD methods output an outlier score [6].*
- *Binary label: some AD methods directly output a binary label $s_i \in \{0, 1\}$ indicating that $\mathbf{x}_i$ is nominal, i.e., $s_i = 0$, or anomalous, i.e., $s_i = 1$. It is noteworthy that the outlier score can also be transformed into a binary label output. Indeed, a threshold $\tau \in \mathbb{R}$ is defined according to the distribution of the data scores and all the observations with a score higher than the threshold τ are anomalous. In other words, this conversion can be modeled by the following rule:*

$$\textit{if } s_i > \tau \textit{ then } s_i = 1 \textit{ else } s_i = 0\,.$$

Binary labels provide less information than score outputs, because the data cannot be ranked according to their degree of abnormality, so that the most abnormal instances can be prioritized during the diagnosis process. In addition, one can study the outlier score distribution to characterize ambiguous samples, i.e., the samples with scores that are neither too low to be considered nominal nor too high to be rejected as anomalous. For this reason, and without loss of generality, we will mainly develop scoring-based anomaly detectors.

Unsupervised anomaly detection methods generally presuppose the following two assumptions [173]:

- **Assumption 3.1** *The anomalous data are statistically different and distinguishable from the nominal data.*
- **Assumption 3.2** *The majority of the training data are nominal. These data probably include a minority of anomalies, a.k.a., data contamination or pollution.*

The objective is to design a *robust* anomaly detector, which performance is not affected by the anomalies contaminating the training data. In the following section, we formalize the notion of *robustness*.

3.1.3 Robustness to Training Contamination

In the literature, there are several definitions of *robustness* depending on the application field and the underlying investigated problem. In this work, we narrow the connotation spectrum by adopting the definition from Huber [120] as “insensitivity to small deviations from the assumption". In our case, the assumption is that the training data are anomaly-free and we aim to develop an anomaly detector that is robust to training contamination. Indeed, we mentioned in Section 2.2.3.1 that most well-known anomaly detectors fall in the semi-supervised category, since they assume that the training data are fully clean and anomaly-free. In contrast, unsupervised methods assume that a small unknown fraction of the training data are contaminated with anomalies. The aim is to design robust methods that are insensitive to this corruption: minor data contamination should not significantly impact the model performance, which is equivalent to the insensitivity to small deviations from the anomaly-free data assumption.

In particular, a plethora of statistical methods has been developed in the literature [120]. These approaches are however mainly designed for *parametric* estimation. Parametric methods assume that the nominal data follow a known parametric distribution, e.g., a Gaussian distribution, and their objective is to robustly estimate the parameters of this idealized model, in the presence of noise.

Even though these methods have shown huge success in some applications, e.g., robust statistical regression [221], their applicability on real-world datasets and particularly network traffic data is limited, since real-world data do not always follow a predefined parametric distribution. The nominal data patterns are generally complex, or asymmetric with a heavy-tailed distribution. In this case, the inconsistent choice of the prior can significantly impact the AD performance and increase the false detection rate, as the nominal data may be poorly fitted to the model [6].

Alternative methods propose robust *non-parametric* anomaly detectors, which make no prior assumption about the distribution of the underlying data [248]. The most common methods that fall in this category are robust dimensionality-reduction-based approaches. They assume that the nominal data can be projected into a lower-dimensional space without a significant loss of information. Anomalies however present large reconstruction errors because they do not conform to the common properties of the norm and they can not be reliably fitted in the reduced subspace.

Unfortunately, some of the most common dimensionality-reduction-based anomaly detectors, i.e., PCA, AE (cf. Sections 2.3.5), are excessively sensitive to minor data contamination [120]. This

observation has been reported in numerous studies and a tremendous interest has been reported towards the design of robust methods in the last decades [54, 292, 10, 143].

In the following section, we review the three most common dimensionality reduction-based anomaly detectors. We mainly focus on PCA, RPCA, and AE. We present how one can apply these methods in the task of AD. Then, we outline the major limitations of these classical methods.

3.2 Related Work

Following up our formalization of the problem of robust, non-parametric, unsupervised anomaly detection, we review in this section the most relevant approaches tailored to this problem: PCA, RPCA, and AE.

As we discussed in Chapter 2, the main assumption of these approaches is that the nominal data can be encoded, i.e., projected, in a low-dimensional latent space. In contrast, anomalies present nonconforming properties and cannot fit this subspace without a significant loss of information. As such, the objective is to infer the optimal lower-dimensional subspace that captures most nominal data information and where anomalies are poorly fitted.

One of the most common methods is PCA. This approach linearly projects the nominal observations into a lower-dimensional subspace, defined using a few eigenvectors that capture most of the data variation. However, the efficacy of the projection depends on the availability of anomaly-free training data. Indeed, the selected eigenvectors may be biased toward these contaminants. This limitation motivated Candès et al. [54] to develop a more robust dimensionality reduction algorithm: RPCA. This approach can filter training outliers and model the remaining anomaly-free subset, using an additive sparse decomposition strategy, which will be reviewed later in this section.

Even though RPCA is significantly more robust than PCA, it makes the assumption that the data only present linear correlations, which can be invalid for numerous applications. To alleviate this limitation, Zhou and Paffenroth [292] extend RPCA and propose a robust AE that extracts non-linear correlations of the data, without being sensitive to training contaminants.

In the following, we review these three methods in more detail. Section 3.2.1 focuses on the classical linear dimensionality reduction approach: PCA. Section 3.2.2 explores the robust PCA: RPCA. The last section details the RDAs.

3.2.1 Principal Component Analysis (PCA)

PCA is a method that is widely used in numerous applications, including feature extraction, data compression, face recognition, and anomaly detection [38]. We describe in Section 3.2.1.1 the overall workflow of this approach. We then present how it can be applied to unsupervised AD. We particularly outline in Section 3.2.1.2 the most relevant research publications that propose PCA-based anomaly detectors. We conclude this part by providing the main limitations of these studies, namely their sensitivity to outliers that contaminate the training data.

3.2.1.1 PCA Overview

The main intuition of PCA is to reduce the dimensionality of multi-dimensional data that comprise many intercorrelated features, while retaining as much as possible of the data variation [125]. In fact, it linearly projects the data into a lower-dimensional latent space, where the utmost variance is captured by a few dimensions. To find the optimal projection, PCA relies on the eigenvalue decomposition of the data covariance matrix.

Let $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ be N training samples of dimension d, $\mathbf{x}_i \in \mathbb{R}^d$, and $\mathbf{C} \in \mathbb{R}^{d \times d}$ be the data

covariance matrix. In fact, the $(i,j)^{\text{th}}$ value of $\mathbf{C}$, denoted as c_{ij}, is the covariance between the i^{th} and j^{th} dimensions of $\mathbf{X}$. Suppose that the data $\mathbf{X}$ is mean-centered, the covariance matrix $\mathbf{C}$ can be defined as follows:

$$\mathbf{C} = \frac{1}{N-1}\mathbf{X}^T\mathbf{X} \tag{3.2}$$

This matrix is symmetric and positive semi-definite [6]. Thus, it can be orthogonally diagonalized as follows:

$$\mathbf{C} = \mathbf{P}\mathbf{Q}\mathbf{P}^T, \tag{3.3}$$

where $\mathbf{P}$ is a matrix that contains the eigenvectors of the covariance matrix $\mathbf{C}$ and $\mathbf{Q}$ is a diagonal matrix providing the corresponding eigenvalues. Indeed, the eigenvalues represent the variances of the data over the corresponding eigenvectors. The eigenvectors with the largest eigenvalues, a.k.a., principle components, retain the most important variations of the data. Therefore, we can consistently approximate the original data using a few uncorrelated dimensions $k \in \mathbb{N}$ such that $k << d$.

Suppose that $\mathbf{P}$ columns, i.e., eigenvectors, are ordered according to a decreasing eigenvalue tendency. The projected data matrix can be defined as:

$$\mathbf{X}^{'} = \mathbf{X}\mathbf{P} \tag{3.4}$$

Since the first k dimension of $\mathbf{X}^{'}$ captures the most significant data variation, the dimensionality of the data can be reduced without significant loss of information: only the first k dimension of $\mathbf{X}^{'}$ can be retained.

3.2.1.2 PCA-based Anomaly Detection

Several works have tailored PCA to semi-supervised anomaly detection in different application such as image, video, and network traffic AD [229, 78]. We limit the scope of this section to the application field of network traffic AD, our main research topic.

Ding and Tian [82] introduce a method that analyzes the network traffic flows and reports anomalous activities. They assume that the nominal flows are sparse and, unlike outliers, can be compressed into a compact representation. For this reason, PCA was used to extract the principal components of the nominal traffic flows. The variance of the data captured by these principal components is used to distinguish nominal and anomalous data. Indeed, they compute the ratio of the variance captured of the first principal component, which has the largest eigenvalue, according to the variance of the remaining components. As it is extremely hard to reliably represent anomalies with a single dimension, this ratio is large for nominal data and relatively low for outliers. They empirically demonstrate that their approach can detect network attacks with high accuracy and a low false-alarm rate.

Similarly, Lakhina et al.[145] apply PCA to detect network anomalous traffic. They use this method to extract the uncorrelated eigenvectors of the data covariance matrix. Then, they decompose this space into normal and anomalous subspaces using a threshold-based approach: the eigenvectors that capture more than three times the standard deviation of the data variance belong to the nominal subspace and the remaining components define the anomaly subspace. In this study, the nominal subspace is defined using the first four principal components. The authors propose to project the data into the anomalous subspace and use the squared norm of this projection as an anomaly score. This score is expected to be relatively low for nominal samples as the majority of the information is retained by the nominal subspace.

To reduce the quadratic complexity of the previous methods, Hoang and Nguyen [115] introduced a more efficient PCA-based anomaly detector, dedicated to IoT network intrusion detection. They particularly proposed a less-complex distance computation strategy, based on the Minkowski formula. We refer the reader to [115] for a detailed description of this strategy. Despite the slight

decrease in the anomaly detection accuracy, this method shows a significant reduction of the computation complexity of the classical PCA-based anomaly detectors.

Even though PCA shows good AD performance in some situations, numerous studies have criticized this approach limitations, namely its sensitivity to training contamination [54, 261].

3.2.1.3 PCA Sensitivity to Training Outliers

As mentioned in the previous section, PCA assumes that the data $\mathbf{X}$ can be consistently projected into a lower-dimensional subspace, defined using the principal components of the matrix $\mathbf{P}$. For this purpose, the optimal transformation is inferred through the least squares minimization, where the residual between the original data $\mathbf{X}$ and the reconstructed data $\widehat{\mathbf{X}} = \mathbf{P}\mathbf{P}^T\mathbf{X}$, a.k.a., the reconstruction error, is minimized [151]. We can formulate the optimization process of PCA as follows:

$$\begin{aligned} &\min_{\mathbf{P}} ||\mathbf{X} - \mathbf{P}\mathbf{P}^{\mathbf{T}}\mathbf{X}||_2^2 \\ &\text{subject to } \mathbf{P}^T\mathbf{P} = \mathbf{I} \end{aligned} \tag{3.5}$$

where $||.||_2$ is the Euclidean distance, i.e., the l_2 norm, and $\mathbf{I}$ is the identity matrix.

Unfortunately, this least-square optimization is not robust in the sense that training outliers can distort and skew the optimized subspace [78]. The large squared residuals of the outliers dominate and significantly influence the optimization. Even a single corrupted observation point may result in an approximate subspace that is arbitrarily far from the true data manifold [54]. Data contamination with unlabeled anomalies is common in real-world environments, particularly in network traffic analysis.

Parametric estimation methods were proposed to approximate the data covariance matrix with a robust estimator instead of the sensitive least-square optimization. These methods include multivariate trimming [105], alternating minimization [127], and robust M-estimators [53]. For example, Campbell [53] propose to use the robust M-estimators to infer the principal components. However, these methods have a polynomial complexity and are impractical for high-dimensional multivariate observations [54, 78].

In the next section, we develop an alternative approach for robust PCA which does not suffer from polynomial complexity, based on sparse representation learning.

3.2.2 Robust Principal Component Analysis (RPCA)

An alternative approach [54] leverages sparse representation learning. This is a promising candidate method that decomposes the contaminated training data into two matrices: a low-rank "data" matrix that contains the anomaly-free observations, plus a sparse "noise" matrix comprising the outliers. This approach is not restricted to low-dimensional data and outperforms previous competing methods. It has been shown that under certain reasonable conditions, the data decomposition task provides the "exact" solution to the underlying optimization problem. The purpose of this section is to present this method. We will formulate RPCA optimization, discuss its efficiency in network traffic anomaly detection, and conclude with its major weaknesses.

3.2.2.1 RPCA Overview

RPCA aims to reliably approximate the low-dimensional subspace of the nominal data, when the training data are contaminated with an unknown ratio of anomalies. For this reason, Candès et al. [54] propose to model the data through an additive decomposition strategy. Let $\mathbf{X} \in \mathbb{R}^{N \times d}$ be the data matrix that contain N d-dimensional data points. Since anomalies are rare, RPCA assumes that the data matrix $\mathbf{X}$ can be decomposed into two distinct matrices:

$$\mathbf{X} = \mathbf{L} + \mathbf{S}, \tag{3.6}$$

where $\mathbf{L}$ is a low-rank matrix that retain the variation of the majority of the data and $\mathbf{S}$ is a sparse matrix comprising few anomalies that cannot be efficiently captured by the low-dimensional subspace. This decomposition is formalized through the following constrained optimization:

$$\min_{L,S} \; rank(\mathbf{L}) + \lambda ||\mathbf{S}||_0 \quad \text{subject to } \mathbf{X} = \mathbf{L} + \mathbf{S}, \tag{3.7}$$

where $rank(\mathbf{L})$ is the rank of the matrix $\mathbf{L}$, i.e., the maximum number of the linearly independent dimensions of the matrix, and $||.||_0$ is the l_0 norm, which counts the number of non-zero entries in $\mathbf{S}$, and λ is a balancing hyperparameter. Unfortunately, this non-convex optimization is NP-hard and intractable [292]. An alternative convex optimization is proposed to relax this problem:

$$\min_{L,S} \; ||\mathbf{L}||_* + \lambda ||\mathbf{S}||_1 \quad \text{subject to } \mathbf{X} = \mathbf{L} + \mathbf{S}, \tag{3.8}$$

$||\mathbf{L}||_*$ is the nuclear norm, i.e., the sum of the singular values of $\mathbf{L}$, defined as follows:

$$||\mathbf{L}||_* = \sum_i \sigma_i(\mathbf{L}), \tag{3.9}$$

where σ_i is a singular value of $\mathbf{L}$ and $||.||_1$ is the l_1 norm, i.e., the sum of the absolute values of $\mathbf{S}$.

In fact, minimizing the nuclear norm of $\mathbf{L}$ penalizes the non-zero singular values and encourages the matrix $\mathbf{L}$ to have as few dimensions as possible. It has been shown that, under mild conditions, both low-rank and sparse matrices can be inferred exactly [218]. In addition, the l_1 norm regularizes the data rejected in $\mathbf{S}$ to reject as few as possible of the training data. λ is the hyperparameter that controls the trade-off between the low-ranking of $\mathbf{L}$ and the sparsity of S.

A myriad of methods have been introduced to solve this convex optimization problem. These methods include Singular Value Thresholding (SVT) [52] and augmented Lagrange multiplier-based approaches [153]. We refer the reader to [218] for a thorough survey of these optimization methods.

3.2.2.2 RPCA-based Anomaly Detection In Network Traffic

Numerous research studies have proposed RPCA-based methods tailored to network traffic anomaly detection. Paffenroth et al. [199] collect and process the network traffic metadata of a testbed network. Then, they apply RPCA to the processed data of 94 features and report any row of the extracted matrix $\mathbf{S}$ as an anomaly. They demonstrate the potential of this method on a public dataset: the Lincoln Labs DARPA Intrusion Detection dataset [1]. Finally, they propose a training strategy that enables the optimal selection of the RPCA hyperparameter λ (cf. Equation 3.7). Similarly, Vilaça et al. [262] leverage this robust method to detect network botnets. They propose a two-step method, called RPCA-MD. In the first step, RPCA is used to recover the anomaly-free low-rank matrix of the data. Then, the mean and covariance matrices of the nominal data are computed. The authors then use these two matrices to compute the Mahalanobis distance of test samples with respect to the norm, and large deviations are flagged as anomalies. Finally, Abdelkefi et al. [199] study the robustness of RPCA against data poisoning, i.e., the adversary data maliciously injected during the data collection to bias the modeling process. They empirically compare the performance of three models: PCA, RPCA, and Kalman-Loeve Expansion (KLE), i.e., an extension of PCA proposed in [47] to detect network traffic anomalies. These methods are compared on the Lincoln Labs DARPA Intrusion Detection dataset [1]. RPCA significantly outperforms the two other methods, with a less false positive rate of the poisoned data.

3.2.2.3 RPCA Limitations

Despite its popularity in the AD community, RPCA presents at least two major limitations.

Firstly, the optimization of the nuclear norm may result in biased estimation of the data rank [218]. Indeed, the nuclear norm penalizes all the singular values equally. Thus, eigenvectors with high single values are over-penalized compared to those with low single values. To put it differently, optimizing the nuclear norm results in learning a transformation that minimizes all the singular values, including the most important ones that correspond to the principal components. Consequently, it is no longer possible to capture the most data variation with few eigenvectors and the low-rank matrix may not well approximate the original data [197].

Secondly, RPCA assumes that the nominal observations can linearly be projected into a low-dimensional subspace. In various applications, the norm comprises complex patterns that cannot be reliably approximated with a linear projection. Data may present complex non-linear properties with uncorrelated features. An alternative and a more common dimensionality reduction methodology is to use AEs to mine nonlinear data correlations. The use of AEs for dimensionality reduction and AD has been discussed in Section 2.3.5.1.

As previously mentionned, AE AD performance is considerably impacted by the presence of training anomalies. Recently, Zhou and Paffenroth [292] proposed a novel approach that combines RPCA and AEs. Their method, called RDA, inherits both advantages of the classical AE and RPCA: while the AE extracts non-linear representation of the data, the RPCA filters out noise and corrupted observations. In the following, we will extensively develop this robust approach.

3.2.3 Robust Deep Autoencoders (RDAs)

RDAs aim is to alleviate the linear projection limitation of the RPCA thanks to the use of AEs. This is crucial for the modeling of complex real-world data, where the linear projection of the classical PCA and RPCA may fail in capturing the complex patterns of the nominal behavior.

In the following, we provide an overview of RDAs, with an emphasis on their robust optimization strategy. We then outline the most recent research studies that apply this robust method to network traffic AD. We conclude this section with the major limitations of RDAs, namely their sensibility to the hyperparameter selection.

3.2.3.1 RDA Overview

RDAs [292, 60] extend RPCA to nonlinear robust representation learning. RDA assumes that most of the training data is nominal along with a minority of corrupted observations. The main intuition is that outliers are unstructured and more difficult to compress than inliers. As such, they can not be compressed into a low-dimensional representation without a significant loss of information. In contrast, nominal data can be accurately condensed and reconstructed. Consequently, RDAs filter training instances that are difficult to reconstruct, and learn an accurate representation of the remaining training data.

Analogously to RPCA, RDA splits the input data matrix $\mathbf{X}$ into two parts: $\mathbf{X} = \mathbf{L} + \mathbf{S}$, where $\mathbf{L}$ is a low-rank matrix that contains nominal data, and $\mathbf{S}$ is a sparse matrix comprising the training anomalies. Thanks to this decomposition, the backbone AE can accurately reconstruct the anomaly-free training data $\mathbf{L}$ without being biaised by the corrupted data. This data decomposition is acheived by optimizing the following constrained objective function:

$$\min_{\theta,\phi} \left\| \mathbf{L} - D_\phi(E_\theta(\mathbf{L})) \right\|_2 + \lambda \left\| \mathbf{S} \right\|_0 \tag{3.10}$$
$$\text{subject to } \mathbf{X} = \mathbf{L} + \mathbf{S},$$

where $E_\theta(.)$ denotes the encoding function from the input to the hidden space, $D_\phi(.)$ is the decoding map from the hidden layer to the output (cf. Section 2.3.5.1), and λ is a hyperparameter that controls the trade-off between the reconstruction error and the sparsity of $\mathbf{S}$.

We note that RDAs replace the nuclear norm of RPCA (cf. Equation 3.8) by the reconstruction error of the AE. This first part trains the model to accurately reconstruct the majority of the training data, after a nonlinear projection into a lower-dimensional latent space. The second part is similar RPCA. Atypical training data are isolated in the matrix $\mathbf{S}$. Since anomalies are rare, this matrix must be sparse. The sparsity is induced by regularizing S with the l_0 norm, which represents the number of non-zero elements in this matrix.

Unfortunately, optimizing this objective function (cf. Equation 3.10) is not tractable, due to the presence of the l_0 norm. To cope with this problem, and following the RPCA literature, the authors propose two alternative relaxations for two distinct tasks: data denoising and outlier detection. Indeed, as the RDA was destined to process image data, the authors distinguish between noise and outliers. While an outlier refers to an observation that deviates too much from the remaining data, noise is defined as errors in the values of one observation attribute [230]. More precisely, for a data matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$ containing N data points of d dimensions, noise refers to element-wise corruption, while outlier refers to row-wise aberration [292]. In this study, the word *anomaly* has a broader meaning, representing both noise and outliers. The difference between the two notions is illustrated in Figure 3.2.

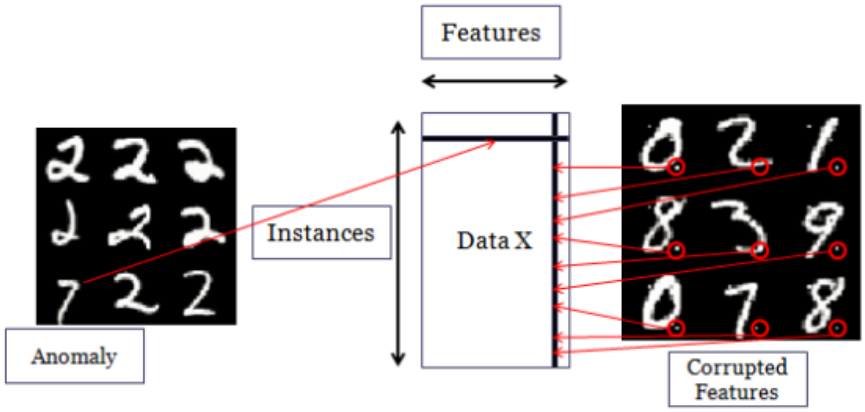


Figure 3.2: The main difference between noise and outlier according to RDA reference [292]. On the left, the image of the digit 7 is an outlier among 2 digit images. This outlier corresponds to a structured row that is corrupted in the data matrix. On the right, noisy data are visualized, where some pixels of the images are corrupted.

l_1 RDA for Data Denoising

To filter out the corrupted pixels of an input image, a.k.a., image denoising, Zhou et al. proposed to replace the l_0 norm with the l_1 norm for data denoising.

$$\min_{E_\theta, D_\phi} \left\| \mathbf{L} - D_\phi(E_\theta(\mathbf{L})) \right\|_2 + \lambda \left\| \mathbf{S} \right\|_1 \tag{3.11}$$
$$\text{subject to } \mathbf{X} = \mathbf{L} + \mathbf{S}.$$

This formulation of the training objective is similar to the relaxation of the RPCA, discussed in Section 3.2.2.1 (cf. Equation 3.8)). The l_1 norm favors $\mathbf{S}$ element-wise sparsity and filters out the data corrupted values.

$l_{2,1}$ RDA for Outlier Detection

To filter row-wise aberrations, i.e. outliers, the authors regularize $\mathbf{S}$ sparsity using the $l_{2,1}$ norm, defined as follows:

$$\|\mathbf{S}\|_{2,1} = \sum_j \|\mathbf{s}_j\| = \sum_j \sqrt{\sum_i |s_{ij}|^2}. \tag{3.12}$$

The $l_{2,1}$ norm consists in applying the l_2 norm over each column of the data matrix, and grouping all these results using the l_1 norm. As such, the $l_{2,1}$ RDA optimizes the following function:

$$\min_{E_\theta, D_\phi} \left\|\mathbf{L} - D_\phi(E_\theta(\mathbf{L}))\right\|_2 + \lambda \|\mathbf{S}\|_{2,1} \tag{3.13}$$
$$\text{subject to } \mathbf{X} = \mathbf{L} + \mathbf{S}.$$

In this thesis, we focus on outlier detection to flag anomalous network packets. For this purpose, we focus our study on the $l_{2,1}$ RDA which is more adapted to this task. In the remainder of this chapter, we refer to $l_{2,1}$ RDA as RDA.

3.2.3.2 RDA-based Anomaly Detection In Network Traffic

A series of recent studies have investigated the use of RDAs for network traffic anomaly detection. Kotani et al. [139] proposed an RDA-based approach for network flow intrusion detection, which proved to reduce the number of false positives on real-world traffic datasets. This method operates in two steps. It firstly trains a robust AE to recover an anomaly-free subset of a corrupted network traffic and model the nominal behavior of the network. Then, this trained RDA is used to detect any deviation, i.e., any observation with a large reconstruction error. The experimental results of RDA show fewer false positives compared to the classical AEs. A more recent study [200] demonstrates the efficacy of RDAs in the robust detection of network malicious attacks, particularly the well-known DDoS attacks. Along the same lines, Aboelwafa et al. [3] apply this robust method to identify network false data injection. These attacks are used to overcome the classical threat detectors by maliciously injecting falsified samples. The authors propose to clean the corrupted data and filter the injected points using RDAs. They empirically demonstrate the efficacy of this method in reliably recovering an anomaly-free subset, and show that this method outperforms SVM-based anomaly detectors, including linear, Radial Basic Function (RBF), and Gaussian kernels.

3.2.3.3 RDA Limitations

The performance of RDAs is highly dependent on the hyperparameter λ, which controls the trade-off between the reconstruction error and the sparsity of $\mathbf{S}$. A low λ results in rejecting too many training data in $\mathbf{S}$, while a high λ over-penalizes data isolated in $\mathbf{S}$, and consequently, enforces the model to reconstruct all training data. For this reason, the value of λ needs to be chosen carefully with a dedicated exploratory strategy that implicitly requires labelled anomalies. Thus, such RDA cannot deliver reliable results in uncontrolled environments that require unsupervised approaches. This is why we propose RADON, which addresses this important limitation, by redefining the optimization problem.

3.3 Contribution: RADON, Robust Autoencoder with Dynamic Outlier filteriNg

In this section, we introduce our first contribution RADON, a Robust Auto-encoder with Dynamic Outlier filteriNg that learns a robust representation of the nominal class. Our unsupervised method automatically filters training anomalies, with no access to any labeled samples. To further improve the AD performance, its training was enhanced to simultaneously minimize the reconstruction scores of nominal instances and maximize that of the filtered anomalies.

This work has been published in [183].

In the following, we detail the three pillars of our contribution. First, we introduce the new objective to optimize, based on the cosine function. Then, we revise the traditional projection constraint by dividing the set of training data into three subsets containing respectively the estimated normal, abnormal and undetermined samples. Finally, we explain how to constitute these subsets thanks to a unimodal triangular thresholding [220] on the reconstruction scores histogram.

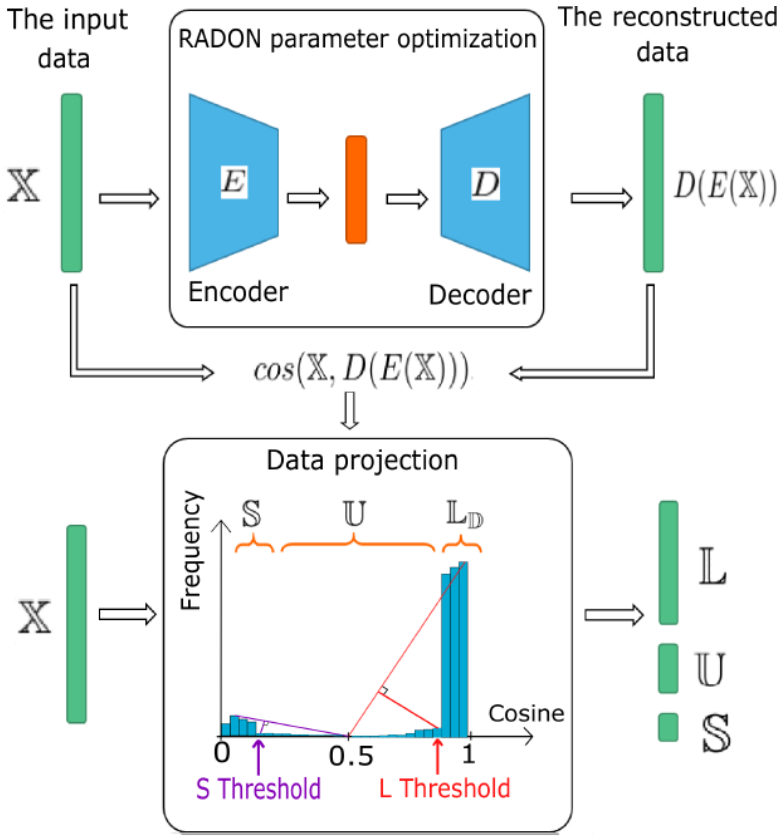


Figure 3.3: RADON training strategy, which alternates between updating the AE parameters (top) and updating the subsets $\mathbb{X} = \mathbb{L} \cup \mathbb{S} \cup \mathbb{U}$ (bottom).

3.3.1 RADON Training Strategy

The key insight of RDAs is that they filter training anomalies, and rely only on the remaining anomaly-free data to learn a non-linear representation of the nominal class. That is, this approach learns a non-linear representation of the nominal class, but does not leverage them to increase the relevance of this representation for the task of anomaly detection. We assume that the separation between nominal and abnormal instances can be maximized using these filtered instances as a prior for the negative class.

Inspired by some metric learning approaches [148, 34], our objective is to learn a robust projection, where nominal data are similar to their reconstruction, but also where anomalies are explicitly badly reconstructed. RDA reconstruction scores are based on the l_2 norm, which is not bounded. Instead of the Euclidean distances, we propose an objective based on the cosine similarity function. For two non-zero vectors $\mathbf{x}$ and $\mathbf{y}$, the cosine similarity function is defined as:

$$cos(\mathbf{x},\mathbf{y}) = \frac{\mathbf{x}.\mathbf{y}}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2} \tag{3.14}$$

The cosine metric normalizes the similarity with the magnitude of the vectors, which results in bounded values in the range $[0,1]$.

We propose to train the AE such that training inliers $\mathbb{L}$ are collinear to their reconstructions, and training anomalies $\mathbb{S}$ and their reconstructions are orthogonal. Concretely, let $E_\theta(.)$ denote the encoding function of the AE parameterized by θ, and $D_\phi(.)$ be the decoding map from the hidden layer to the output (cf. Figure 3.3). We formulates RADON optimization task as follows:

$$\min_{\theta,\phi} (1 - cos(\mathbb{L}, D_\phi(E_\theta(\mathbb{L}))))^2 + (0 - cos(\mathbb{S}, D_\phi(E_\theta(\mathbb{S}))))^2. \tag{3.15}$$

The question now is how to filter training outliers, i.e., how to define the two subsets $\mathbb{L}$ and $\mathbb{S}$? We will answer this question in the next section, by defining RADON projection strategy (cf. Section 3.3.2).

3.3.2 RADON Projection Strategy

We propose to adapt the projection strategy of the RDAs: instead of decomposing the input matrix into two matrices $\mathbf{L}$ and $\mathbf{S}$, we divide the input data into three disjoint subsets $\mathbb{X} = \mathbb{L} \cup \mathbb{S} \cup \mathbb{U}$ (cf. Figure 3.4).

- The first subset $\mathbb{L}$ represents data points that are colinear to their reconstructions, with $cos(\mathbb{L}, D_\phi(E_\theta(\mathbb{L})))$ close to 1.
- The second subset $\mathbb{S}$ represents data that cannot be reconstructed, with $cos(\mathbb{S}, D_\phi(E_\theta(\mathbb{S})))$ close to 0.
- The third subset $\mathbb{U}$ is formed of the remaining points, with a cosine score around 0.5. For these instances, it is not possible to confidently determine whether they should be correctly reconstructed or not. Indeed, as we are jointly learning to reconstruct inliers and to seclude outliers, we need to consider only confident accepted or rejected instances, with a cosine score very close to 1 or 0, respectively. Considering in-between instances may progressively destabilize the training.

Therefore, we reformulate the constrained optimization problem as follows:

$$\min_{\theta,\phi} (1 - cos(\mathbb{L}, D_\phi(E_\theta(\mathbb{L}))))^2 + (0 - cos(\mathbb{S}, D_\phi(E_\theta(\mathbb{S}))))^2 \tag{3.16}$$
$$subject\ to\ \ \mathbb{X} = \mathbb{L} \cup \mathbb{S} \cup \mathbb{U}$$

The training strategy is iterative and alternates between updating the AE parameters and updating the subsets $\mathbb{X} = \mathbb{L} \cup \mathbb{S} \cup \mathbb{U}$. We optimize our objective function with respect to ϕ and θ for a fixed number of epochs, we project the data according to the constraint, and we repeat this process using the updated values (see Figure 3.3).

The convergence criterion consists in that all training instances are classified whether in $\mathbb{S}$ or $\mathbb{L}$, and no instances remain unclassified in $\mathbb{U}$. We stop the training when $\frac{\|\mathbb{U}\|_2}{\|\mathbb{X}\|_2} < \varepsilon$, where $\varepsilon = 10^{-3}$. As $\{\mathbb{S} \cup \mathbb{L}\}$ may converge to a fixed set that is different from $\mathbb{X}$, we define a maximum number of epochs, after which we stop the training even if this inequality is not verified.

3.3.3 Unimodal Thresholding of the Reconstruction Histogram

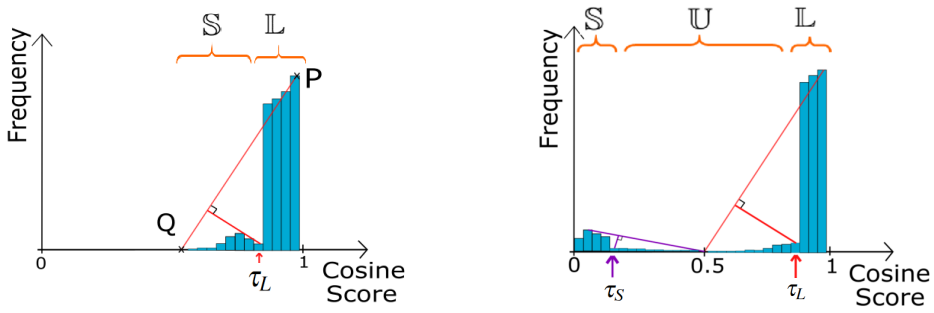


Figure 3.4: The thresholding strategy at the end of the first iteration (left), and at the end of the following ones (right).

To decompose $\mathbb{X}$ into the three subsets, we propose to form a histogram with reconstruction values $cos(\mathbb{X}, D_\phi(E_\theta(\mathbb{X})))$, enabling the use of image segmentation thresholding techniques [270]. We assume an unbalanced dataset, where normal data are over-represented compared to anomalies. The resulting reconstruction histogram should present a dominant peak, and may or may not have a second, much smaller peak. Consequently, we propose to use the unimodal thresholding [220], which specifically addresses this family of histograms. The unimodal threshold is performed by drawing a line between the peak of the histogram, P, with coordinates [center of the main bin; peak frequency]; and the first minimum, Q, with coordinates [center of the first empty bin, 0] (see Figure 3.4 (left)). We select the bin center whose peak is farthest from this line as the threshold.

Initially, $\mathbb{L} = \mathbb{X}$, and both $\mathbb{U}$ and $\mathbb{S}$ are empty. For the first iteration, we minimize the objective function (3.16). Then, we compute the reconstruction error of the input data, and we split it into two parts, according to the triangular threshold (see Figure 3.4 (left)): instances having a reconstruction score greater than the triangular threshold are put in $\mathbb{L}$, and the remaining data are placed in $\mathbb{S}$. Then, we continue RADON training using both positive and negative data. At the end of the next iteration, we have a reconstruction histogram similar to Figure 3.4 (right). We divide the histogram into two histograms: the first histogram, H_L, containing instances with a cosine score greater than 0.5, and the second histogram, H_S, containing instances with cosine scores lower than 0.5. We threshold both histograms using the unimodal threshold (see Figure 3.4 (right)). Instances with a cosine score greater than τ_L are classified in $\mathbb{L}$, instances with a cosine score less than τ_S are in $\mathbb{S}$, and the remaining are in $\mathbb{U}$.

Finally, as we threshold the reconstruction histogram to reject outliers, the number of bins is a hyperparameter that plays a key role in this approach. While a small number of bins may reduce the impact of noise caused by the randomness of sampling, a large number of bins provide more details about the original data. This hyperparameter can be fine-tuned, where the optimal value with the best performance on the validation sets is selected. However, this approach is computationally expensive, as it involves repeating the training several times for different bin number values.

Many heuristics were developed to estimate the optimal number of bins from the underlying data [246]. Let n be the number of data points in the original data, h the bin width, and k the optimal number of bins. Rice's rule estimates $k = 2\sqrt[3]{n}$. This estimator considers only the volume of the data n. Other rules not only depend on n but also on the data variability. Scott's rule [234] defines $h = 3.491\sigma n^{-\frac{1}{3}}$, where σ is the standard deviation of samples. As σ is not robust to outliers, Freedman and Diaconis [94] proposed a robust estimator, defined as $h = 2\text{IQR}n^{-\frac{1}{3}}$, where IQR refers to the interquartile range. However, these methods generally make strong assumptions about the underlying data distribution, and can be sub-optimal if these assumptions are not verified. Recently, Knuth [137] proposed a non-parametric data-based approach to determine the optimal number of bins, with no assumption on data distribution. This approach relies on the Bayesian framework to infer the posterior probability of the number of bins in a piecewise-constant density

function. In our experiments, we compare the performance of RADON with 3 different bin estimators: (1) the Rice rule, (2) the Freedman and Diaconis's (FD) estimator, and (3) Knuth's estimator.

3.3.4 Hypotheses

In this chapter, we conjecture the following hypotheses:

- **Hypothesis 3.1 (H3.1)** *: The first hypothesis is related to the rejection strategy. We hypothesize that the dynamic unimodal thresholding of training outliers is more robust than a predefined regularization hyperparameter. Indeed, we surmise that the dynamic variation of the threshold separating inliers and outliers during the training makes our contribution more robust to hyperparameter misspecification;*

- **Hypothesis 3.2 (H3.2)** *: The second hypothesis concerns the influence of the bin estimator on the AD performance. We conjecture that estimating the optimal number of bins from the data, with Knuth's estimator, is more flexible and yields better results than heuristic-based estimators.*

In the next Section 3.4, we aim to experimentally validate these hypotheses on the two public datasets: NSL-KDD and MedBIoT. Furthermore, we compare our contribution performance against various unsupervised anomaly detectors, to investigate the efficacy and the robustness of RADON compared to these competing methods.

3.4 Experiments and Results

In this section, we analyze the hypotheses defined in the previous section and we present an empirical evaluation of our approach against many competing methods, on two benchmark datasets: NSL-KDD and MedBIoT. Subsequently, we present these datasets, the training protocol, and the experimental results.

3.4.1 NSL-KDD dataset

3.4.1.1 Dataset Description

The first dataset used in our experiments is the NSL-KDD dataset [196]. The NSL-KDD is a benchmark dataset widely used to assess the performance of IDSs in distinguishing malicious network connections, also known as "intrusions", and normal ones. This dataset is divided into two subsets: the training subset with 125973 data points and the test subset with 22544 records. Normal records represent 53% of the training data. Each instance of this dataset contains 41 features extracted from the network traffic, e.g., protocol type, TCP flags, and labeled as normal or as a threat. This dataset encompasses 39 types of attacks, with 17 only present in the test subset. These attacks belong to 4 categories: Denial of Service (DoS), Remote to Local (R2L), User to Root (U2R), and Probes.

- Denial of Service (DoS) is a category of attacks that consists in overloading a system with a large number of queries, which exceeds its memory and capacity of processing. Consequently, this system becomes unavailable and the service is disrupted;

- Remote to Local (R2L) occurs when a remote attacker, which does not have access to a remote network or machine, illegally intrudes into it;

- User to Root (U2R) is an attack where a user takes advantage of system vulnerabilities to have super-user (a.k.a. root) permissions and privileges;
- Probe happens when an attacker steals private and personal information from a network.

Table 3.1 summarizes the volume of data in NSL-KDD per category of attacks.

Table 3.1: Statistics about NSL-KDD dataset

Subset		**Normal**	**DoS**	**R2L**	**U2R**	**Probe**	**Total**
Train	Records	67343	45927	11656	52	995	125973
	Percentage	53%	37%	9.11%	0.04%	0.85%	100%
Test	Records	9711	7458	2421	200	2654	22544
	Percentage	43%	33%	11%	0.9%	12.1%	100%

The NSL-KDD contains three categorical features: (1) the *protocol_type*, which presents the protocol of the flow, e.g., UDP and TCP, (2) the *service*, which provides the network service of L7 layer (cf. Section 2.1), e.g. http, https, ssh, and (3) the flag, which defines the status of the flow [91]. The remaining features are numerical, e.g., the number of packets included in the connection. These features belong to different scales. To prevent the largest scale features from biasing the optimization process, and to analyze all the attributes equally, we rescale all numeric features to be in the range $[0, 1]$. Categorical features are one-hot encoded. More details about this benchmark dataset can be found in [91].

3.4.1.2 Training Protocols

Here, we depict the three training protocols followed to investigate the two hypotheses defined in Section 3.3.4. We first explore the importance of the iterative dynamic thresholding strategy proposed in RADON. We compare the performance of this strategy with the classical $l_{2,1}$ regularization penalty of RDAs to analyze our first hypothesis H3.1. We then focus on the second hypothesis H3.2 to explore our contribution sensitivity to its hyperparameters, namely the number of bins of the histogram. we compare the performance of RADON with 3 different bin estimators: (1) the Rice rule, (2) the Freedman and Diaconis (FD) estimator, and (3) Knuth's estimator. Finally, we empirically compare our contribution to state-of-the-art methods of the literature.

Protocol 1: Iterative Unimodal Thresholding

This first protocol investigates the significance of our rejection strategy with the unimodal thresholding of the reconstruction error histogram. Unlike RDAs, we propose a dynamic threshold that varies during the training. The main intuition is that since the model iteratively learns the data distribution, the threshold that separates training inliers and outliers must also be iteratively adapted to the model learning evolution. To validate this hypothesis, we compare two RADON configurations. In the first configuration, we train our method using the unimodal thresholding strategy, as described in Section 3.3.3. In the second configuration, denoted as *RADON Static*, we use two fixed predefined thresholds to separate the three subsets $\mathbb{L}$, $\mathbb{S}$, and $\mathbb{U}$. These two thresholds are considered as two hyperparameters, which are fine-tuned on a dedicated validation subset. As such, the only difference between these configurations is the rejection strategy. While the former relies on a dynamic threshold that varies during the training, the latter is based on a static predefined threshold that remains constant. In addition, we compare RADON with the classical RDA to

contrast dynamic thresholding and the classical sparsity regularisation, i.e., the $l_{2,1}$ penalty applied on the rejected data $\mathbf{S}$.

Protocol 2: Bin Estimator

We presuppose that the number of bins of the reconstruction error histogram plays a critical role in this AD task. As mentioned in Section 3.3.3, a small number of bins may reduce the impact of noise caused by the randomness of sampling, while a large number of bins provide exhaustive details about the original data. To explore the influence of this hyperparameter on global performance, we compare three different configurations with three common bin estimators: (1) the Rice rule, (2) the Freedman and Diaconis (FD) estimator, and (3) the Knuth's estimator.

Protocol 3: Comparison with Competing Methods

Finally, we conduct an extensive set of experiments to assess the efficacy of our contribution compared to competing unsupervised anomaly detectors. These competing methods include OSVM with a Gaussian kernel, Isolation Forest (IF), AE, RVAE, and RDA. In line with prior works, performances are assessed using the AUROC metric.

To study the sensitivity of anomaly detectors concerning the ratio of anomaly contamination, we vary the training anomaly percentage. We prepare three training subsets, containing 1%, 5%, and 10% of outliers, respectively. Anomaly instances are selected randomly from all classes of NSL-KDD training anomalies. We corrupt training nominal data by mixing it with these outliers and we randomly shuffle all training data. To have a fair comparison and to use the same data for different model training, we fix a seed for data shuffle and training anomaly instance selection. To highlight the statistical significance of these results, we provide also the results of Welch's test, with a p-value = 0.05.

3.4.1.3 Training Parameter Settings and Evaluation Criteria

In this section, we describe the training parameters used in our experiments, as well as hyperparameter selection. The optimal hyperparameters selected in our first set of experiments are reported in Table 3.2.

Table 3.2: The hyperparamters used in the NSL-KDD experiments.

Hyperparamter	Value
Architecture	an 8-neuron single layer FFN
Learning rate	0.001
Maximum number of epochs	800
Number of epochs for each data projection	50
Batch size	256
Laptop	12-core Intel i7-9850H CPU clocked at 2.6GHz, with 16GB of RAM memory and with NVIDIA Quadro P2000 GPU

To limit the impact of random parameter initialization, we repeat each experiment 10 times and average the results over these five runs. In all experiments, we use symmetric autoencoders, with the standard MLP feed-forward architectures. The encoder is constituted of an eight-neuron single feedforward layer. For the competing methods, we fine-tune their hyperparameters on the

validation subset. We select the optimal hyperparameters that maximize validation AUROC. The data projection, i.e., the update of the three subsets $\mathbb{L}$, $\mathbb{S}$, and $\mathbb{U}$ with the unimodal histogram thresholding, occurs every 50 epochs. We use a learning rate of 0.001, an empirical maximum number of epochs of 800, and a batch size of 256.

All the experiments were run on a laptop equipped with a 12-core Intel i7-9850H CPU clocked at 2.6GHz, with 16GB of RAM memory, and with an NVIDIA Quadro P2000 GPU.

3.4.1.4 Results

In this section, we present the experimental results reported on the NSL-KDD. Table 3.3 summarizes these results.

*Table 3.3: NSL-KDD experimental results, with different percentages γ of training outliers. * indicates that the result is significant compared to RDA, according to Welch's test with a p-value*=0.05.

γ	Methods								
	IF	**OSVM**	**Vanilla AE**	**RVAE**	**$l_{2,1}$ RDA**	**RADON**			
						Static	**FD**	**Rice**	**Knuth**
1%	83.6 ± 0.5	81.6 ± 0.1	91.6 ± 1.0	90.8 ± 0.1	93.7 ± 0.7	93.9 ± 0.4	94.2 ± 0.9	$\mathbf{94.8 \pm 0.8}$*	94.7 ± 0.8
5%	84.1 ± 0.4	84.4 ± 0.1	89.5 ± 0.8	90.3 ± 1.3	91.9 ± 0.7	92.5 ± 0.5	94.5 ± 0.5	94.4 ± 0.6	$\mathbf{95.2 \pm 1.2}$*
10%	82.7 ± 0.5	83.2 ± 0.1	88.6 ± 0.7	90.1 ± 0.4	91.0 ± 0.6	92.0 ± 1.6	93.8 ± 1.4	$\mathbf{94.4 \pm 1.1}$*	$\mathbf{94.4 \pm 1.1}$*

Protocol 1: Iterative Unimodal Thresholding

As shown in Table 3.3, training RADON with a dynamic thresholding strategy, i.e., with FD, Rice, and Knuth estimators, yields more robust results than a static threshold, for the different percentages of training outliers. We observe that the greater the ratio γ of contamination, the larger the difference between the results of these strategies. For $\gamma = 1\%$, we report 0.8% points increase with Knuth RADON compared to Static RADON. This difference becomes more noticeable for larger values of γ and reaches 2.4% when $\gamma = 10\%$. The present results are further confirmed, with the larger improvement compared to RDAs, which reaches 3.4% points in favor of Knuth RADON, for $\gamma = 5\%$. These observations validate our first hypothesis (H3.1): the dynamic unimodal thresholding of training outliers is more robust than a predefined static hyperparameter.

To further inspect this interpretation, we report in Figure 3.5 the variation of the dynamic threshlods: $\mathbb{L}$ and $\mathbb{S}$ thresholds (cf. Section 3.3.3) during the training. After the first projection at 50 epochs, both $\mathbb{L}$ and $\mathbb{S}$ thresholds are equal. The filtering becomes more and more parsimonious and selective, as $\mathbb{L}$ thresholds are moving iteratively closer to 1, and $\mathbb{S}$ thresholds get closer to 0. Consequently, the discrepancy between both inliers and outliers reconstruction error is increasing after each projection, and is maximized at the end of the training.

Protocol 2: Bin Estimator

This part focuses on the second hypothesis, concerning the sensitivity of our contribution with the respect to the bin estimator. Table 3.3 summarises the results of three bin estimators: the Rice rule, the FD estimator, and Knuth's estimator. Globally, this hyperparameter does not significantly impact the performance, with similar AUROC around 94% for the three estimators. We observe that Knuth RADON slightly outperforms the two other configurations, particularly for $\gamma = 5\%$. This slight increase is however not statistically significant according to Welch's test with a p-value=0.05.

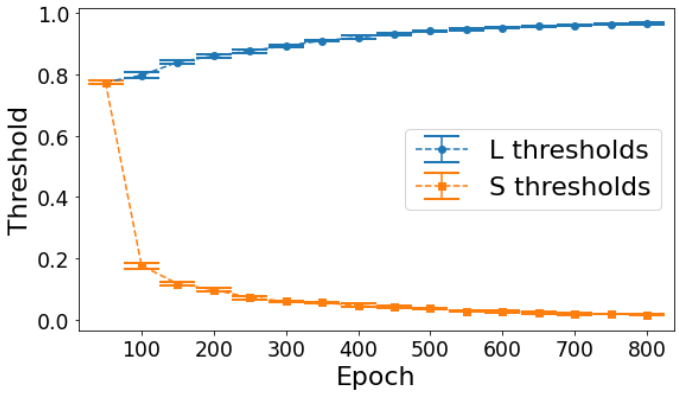


Figure 3.5: Variation of $\mathbb{L}$ and $\mathbb{S}$ thresholds during the training stage.

Thus, our second hypothesis is not confirmed: the three bin estimators show relatively similar results on this first dataset.

Protocol 3: Comparison with Competing Methods

We now focus on the empirical comparison between our contribution and the literature. As shown in Table 3.3, our contribution shows globally superior results for the different percentages γ of training outliers. We firstly report a significant improvement of RADON over the classical anomaly detectors IF and OSVM of around 10% points of mean AUROC. Furthermore, we highlight that our method is more robust than AEs. The difference increases with the ratio of contamination, to reach 6% when $\gamma = 10\%$. Indeed, we note that the performance of AEs gradually decreases with larger γ. In contrast, our method performances remain stable, around 94%, regardless of the contamination ratio. Finally, our method outperforms the literature robust AEs, i.e., RDA and RVAE, with a significant improvement of 3% and 4% points of AUROC, respectively.

To conclude, this first set of experiments has shown the robustness and effectiveness of our contribution to unsupervised network traffic anomaly detection on the NSL-KDD dataset. We have highlighted that the rejection strategy of our contribution, which rejects the training anomalies thanks to the dynamic thresholding of the reconstruction histogram, is significantly more robust than the conventional static rejection strategies. We demonstrated also that our method is not considerably sensitive to the histogram bin estimator, as the three tested bin estimators reported relatively similar results. Finally, RADON shows state-of-the-art results and outperforms the competing methods, for the three contamination ratio $\gamma \in \{1\%, 5\%, 10\%\}$.

3.4.2 MedBIoT dataset

In the previous part, we investigated the effectiveness of our contribution on a general, i.e., non-IoT, network traffic dataset. We particularly demonstrated RADON robustness to training contamination on one of the most popular datasets in network intrusion detection: the NSL-KDD dataset. To match the characteristics of our initial problem of IoT-focused anomaly detection, the second set of experiments is conducted on a larger and more recent IoT dataset: MedBIoT dataset [109]. In the following, we outline the peculiarities of this dataset, the training protocol, and the experimental results.

3.4.2.1 Dataset Description

The MedBIoT dataset [109] contains network traffic involving 83 IoT devices, belonging to four categories: switches, light bulbs, locks, and fans. Three botnet malwares were deployed, Mirai [17], Bashlite [170], and Torii [140] attacks. Both normal and malicious raw network traffic were collected for each device type and made public. Globally, this dataset comprises over 17 million network packets: around 30% of this traffic is anomalous and the remaining 70% is benign. Two reasons justify our choice of this dataset: (i) firstly, unlike other IoT datasets that focus on small-sized networks, MedBIoT provides raw traffic data collected from a large network containing 83 IoT devices, (ii) secondly, this dataset is recently published, in 2020, and contains up-to-date IoT botnet traffic. We extract network header metadata from these raw packets, using the open-source framework NFStream[190]. In particular, we extract 61 flow-based features, which are detailed in Appendix **??**.

We randomly split the benign data into 60% for the training, 20% for the validation, and 20% for testing. Then, we contaminate the benign training data, so that anomalies represent 10% of the final number of training instances. These contaminants are randomly selected from all anomalous classes of MedBIoT. The remaining anomalies are randomly split into 50% validation and 50% test. In this dataset, the cardinality of categorical features is high. To handle the problem of high dimensional categorical data encoding, and similar to [178], we use CountEncoder [75] which encodes categorical features with their frequency of occurrence. Finally, all the features are normalized using the Min-Max normalization method, to rescale them in the range $[0, 1]$.

3.4.2.2 Training Protocols

We aim at investigating the benefit of our contribution over the most relevant methods of the literature on an IoT dataset. The devices included in the MedBIoT dataset belong to four distinct families: switches, light bulbs, locks, and fans. The nominal behavior of an IoT device may significantly vary across these four categories. The Fan nominal network activity is dissimilar to that of a light bulb. As such, we propose a per-device anomaly detection strategy. This strategy consists in training one model per device family, to reliably capture the relative norm peculiarities. This results in four distinct nominal models that model the network activity of the switches, light bulbs, locks, and fans, respectively. Then, each model is used to flag any anomalous behavior of the corresponding devices. This per-device AD is in line with numerous recent studies [241, 242]. The surge of interest in this strategy can be explained by its several advantages. Besides the accurate and meticulous modeling of device type peculiarities, it allows to dynamically accommodate changes into a large IoT ecosystem without requiring the entire system retraining [241]. In fact, integrating a new device type or updating a device behavior requires training a new model or updating an existing model respectively.

For each device type, we follow the same Protocol 3 introduced in Section 3.4.1.2 and we compare our contribution RADON against the unsupervised anomaly detectors.

3.4.2.3 Training Parameter Settings and Evaluation Criteria

In this section, we describe the training parameters used in this second set of experiments, as well as hyperparameter selection. The optimal hyperparameters selected in our first set of experiments are reported in Table 3.4.

To limit the impact of random parameter initialization, we repeat each experiment 5 times and average the results over these five runs. In all experiments, we use symmetric autoencoders, with the standard Multilayer Perceptron (MLP) feed-forward architectures. The AE is a 5-layer MLP with 61-16-4-16-61 units. For the competing methods, we fine-tune their hyperparameters on the validation subset. We select the optimal hyperparameters that maximize validation AUROC. The data projection, i.e., the update of the three subsets $\mathbb{L}$, $\mathbb{S}$, and $\mathbb{U}$ with the unimodal histogram

Table 3.4: The hyperparameters used in the MedBIoT experiments.

Hyperparamter	Value
Architecture	a 5-layer FFN with 61-16-4-16-61 units.
Learning rate	0.001
Maximum number of epochs	800
Number of epochs for each data projection	100
Batch size	256
Laptop	12-core Intel i7-9850H CPU clocked at 2.6GHz, with 16GB of RAM memory and with NVIDIA Quadro P2000 GPU

thresholding, occurs every 100 epochs. We use a learning rate of 0.001, an empirical maximum number of epochs of 800, and a batch size of 256. Finally, all the experiments were run on a laptop equipped with a 12-core Intel i7-9850H CPU clocked at 2.6GHz, with 16GB of RAM memory, and with an NVIDIA Quadro P2000 GPU.

3.4.2.4 Results

*Table 3.5: MedBIoT experimental results, with $\gamma = 0.1$. * indicates that the result is significant, according to Welch's test with a p-value*=0.05.

Device	Methods						
	IF	OSVM	Vanilla AE	$l_{2,1}$ RDA	RADON		
					FD	Rice	Knuth
Lock	93.0 ± 1.5	92.0 ± 0.1	74.2 ± 2.0	95.6 ± 0.5	99.2 ± 2.8	**99.3 ± 0.3***	**$99.3 \pm 2.0e{-2}$***
Fan	$94.1 \pm 4.2e{-2}$	$94.0 \pm 9.2e{-2}$	98.9 ± 0.4	$99.5 \pm 2.5e{-2}$	**$99.9 \pm 1.3e{-2}$***	**$99.9 \pm 1.3e{-2}$***	**$99.9 \pm 2.8e{-2}$***
Bulbs	$94.0 \pm 5.3e{-2}$	$94.0 \pm 6.3e{-2}$	84.5 ± 0.6	$99.5 \pm 2.0e{-2}$	**$99.9 \pm 2.1e{-2}$***	**$99.9 \pm 2.1e{-2}$***	**$99.9 \pm 2.8e{-2}$***
Switch	$94.1 \pm 7.6e{-2}$	$94.1 \pm 7.1e{-2}$	92.5 ± 0.4	$99.4 \pm 1.2e{-2}$	**$99.8 \pm 0.9e{-2}$***	**$99.8 \pm 0.1e{-2}$***	**$99.8 \pm 0.7e{-2}$***

The results demonstrate a significant improvement in our proposed contribution compared to the other baselines. The three tested configurations of RADON show the best performance for all device types, and report an AUROC higher than 0.99. The difference between the three bin estimators is less noticeable on this second dataset. We note however the standard deviation of Knuth's estimator is reduced compared to FD and Rice estimators, for Lock devices. We can explain this observation by the fact that Knuth's estimator is more flexible, as it infers the optimal number of bins from the reconstruction errors, without any prior assumption about the data distribution.

Besides, these results confirm the robustness of our contribution compared to the conventional AEs. While the latter performance is considerably impacted by training contaminants, RADON does not learn to reconstruct outliers and yields good results. Indeed, our contribution exceeds AEs AUROC by 14% points on average.

Furthermore, RADON performs slightly better than $l_{2,1}$ RDA. The difference varies across the device types, from 0.4% points reported for Fans, to around 4.% with Locks. The advantage of our contribution is clearly observable compared to the classical methods IF and OSVM for the four device types.

All in all, the results of the experiments clearly support the robustness of our contribution in AD, all the more as the training data becomes more contaminated with anomalies.

3.5 Conclusion

In this chapter, we addressed the problem of the robust detection of punctual anomalies in non-sequence data. We first formalized this unsupervised problem and we introduced the aim of this first chapter, by highlighting the required characteristics of a robust anomaly detector. We defined the difference between literature *parametric* and *non-parametric* estimators and we justified the choice of the latter category. We then explored the most common methods of nonparametric robust estimation developed in the literature and we particularly reviewed three classical dimensionality reduction-based anomaly detectors: PCA, RPCA, and RDA. We identified the limitations of these methods, namely their static rejection of training anomalies and their sensitivity to hyperparameter selection. To address these major limitations, we proposed RADON, a robust AE for unsupervised punctual AD. Our approach filters training outliers and leverages them to learn a robust projection, where inliers and their reconstructions are similar, while outliers are poorly reconstructed. As such, we reformulated the training strategy as a metric learning problem. To effectively filter training outliers, we provided a dynamic rejection strategy based on the unimodal thresholding of the reconstruction error histogram. The relevance of this strategy and the robustness of our contribution were validated on two benchmark datasets the NSL-KDD, and the MedBIoT datasets. The experimental results showed competitive results compared to unsupervised anomaly detectors. These results clearly favor the proposed dynamic thresholding strategy compared to a static rejection of training anomalies. Furthermore, we investigate RADON sensitivity to its hyperparameter: the number of bins of the reconstruction error histogram. We particularly compared three configurations with three different bin estimators: FD, Rice, and Knuth estimators. The three configurations reported similar results, which show that RADON is insensitive to the bin estimator.

These promising findings were published in an international conference [183]. In addition, this algorithm was integrated in an industrial solution and protected in a patent [35]. This solution helps Orange finding dysfunctional IoT devices. Indeed, one customer faces a particular problem in the LAN and call his ISP for technical assistance, our solution uses the history of anomalies reported by RADON to target the potential devices originating this anomaly.

Despite the good performance of our first contribution RADON, there are still opportunities for further enhancements, mainly regarding the rejection of training anomalies. Indeed, RADON iteratively decomposes the noisy training data into three subsets: an inlier subset $\mathbb{L}$, an outlier subset $\mathbb{S}$, and an undefined subset $\mathbb{U}$. We trained RADON to well-reconstruct $\mathbb{L}$ samples and to poorly reconstruct $\mathbb{S}$ instances. However, the critical data placed in $\mathbb{U}$ are not leveraged during the training. Using a binary thresholding does not allow the model to confidently determine whether they should be correctly reconstructed or not. Consequently, they are excluded from the optimization task to avoid training destabilization. However, these instances may convey important information about the nominal behavior that, unless used, may result in a biased model of the true network behavior. A natural way of addressing the uncertainty of these observations is through probabilities. We propose to address this challenge with generative AEs in the next chapter.

4

Robust Variational Autoencoders for Unsupervised Anomaly Detection

"When one admits that nothing is certain, one must, I think, also add that some things are more nearly certain than others." Bertrand Russell

Contents

4.1 Problem Statement and Background 84
4.1.1 Problem Statement 84
4.1.2 Extreme Value Theory 85
4.2 Related Work: Robust Variational Autoencoder (RVAE) 86
4.2.1 Robust Variational Inference and the β-divergence 86
4.2.2 RVAE Training Strategy 87
4.3 Contribution: GRAnD, Robust VAEs and NFs for Unsupervised Network AD 89
4.3.1 Robust Rejection Strategy 89
4.3.2 Training Loss 91
4.3.3 Hypotheses 92
4.4 Experiments and Results 92
4.4.1 NSL-KDD dataset 92
4.4.1.1 Dataset Description 92
4.4.1.2 Training Protocols 92
4.4.1.3 Training Parameter Settings and Evaluation Criteria 93
4.4.1.4 Results 94
4.4.2 MedBIoT dataset 96
4.4.2.1 Dataset Description and Training Protocols 96
4.4.2.2 Training Parameter Settings and Evaluation Criteria 96
4.4.2.3 Results 96
4.5 Conclusion 98

As shown in the previous chapter, learning in the presence of corrupted data is a primordial step towards the development of robust anomaly detectors that can accurately flag any atypical observation. This major challenge defies the usefulness of many classical approaches, since they assume an ideal learning condition where the training data are completely clean and anomaly-free. Therefore, many recent studies [217, 292, 92] advocate for the development of robust models that are insensitive to training contamination, since these corrupted data are ubiquitous in almost every application field.

To prevent modeling the contaminated data, robust dimensionality reduction-based approaches have been developed and extensively used in the literature. We presented in the previous chapter two of the most popular approaches: RPCA and RDA. We discussed their main limitations and proposed a contribution, RADON, which has been shown to be more robust and not sensitive to the hyperparameter selection. Nonetheless, RADON cannot take into account the training samples in $\mathbb{U}$, i.e., the observation with an in-between outlier score where the model is not confident to retain or reject them. These training data points are completely ignored during the learning process to prevent a potential training destabilization.

In this chapter, we propose a new model designed to extend RADON learning capacity by leveraging these uncertain points. The main idea consists in associating to each sample of $\mathbb{U}$ a training weight which quantifies its degree of anomalousness. This measure quantifies the uncertainty of the model and allows to incorporate these mistrustful data points without introducing conflicting decisions. For this purpose, we propose to use generative AEs, and particularly VAEs and NFs. These two approaches can be seen as probabilistic versions of the classical AEs [103], as they adapt the encoder-decoder architecture to learn the probabilistic models that generate the underlying data. That is, the output of these models is not the predicted value of the input, but the parameters of its approximate probability distribution. The resulting probability can be leveraged to infer the confidence of the model decision. Here, the variational inference methodology plays a key role in the optimization of these probabilistic unsupervised models. This chapter is dedicated to providing the necessary background to understand and motivate the choice of these generative models.

The remainder of the chapter is organized as follows. Section 4.1 presents the problem statement of the variational-based unsupervised AD. We introduce in this part the theoretical background for for the uncertainty estimation, through EVT. Section 4.2 outlines the state-of-the-art robust generative AE: the RVAE [10]. Next, we present in Section 4.3 the second contribution of our thesis: GRAnD. Section 4.4 focuses on the experimental protocols and results. Finally, we provide in the last section the main conclusions and further research perspectives.

The contributions of this chapter have been published in [186].

4.1 Problem Statement and Background

This section presents the theoretical bases involved in our contribution.. Section 4.1.1 formalizes the problem of unsupervised AD under the variational inference setting. Section 2.3.5.2 introduces VAEs and NFs. Finally, Section 4.1.2 outlines the well-known EVT, which will be used later in our contribution robust rejection strategy.

4.1.1 Problem Statement

In this chapter, we consider the task of unsupervised non-sequence AD, introduced in Section 3.1, but under the variational inference setting. The nature of the input data, the detected anomalies, the underlying problem statement, and the motivation of the development of a robust model are

exactly similar to those of the previous chapter. The main difference is the backbone models, i.e., the scoring function, used to distinguish nominal and abnormal observations.

While we relied on discriminative AEs in the previous chapter, we focus now on two of the most popular generative AEs: VAE and NF. As mentioned previously, both VAE and NF allow the computation of the free energy, i.e., an upper bound for the negative log-likelihood, of the data. It is necessary to identify the atypical observations that have extreme negative log-likelihoods under these probabilistic models. This question will be addressed in the next section, through the introduction of the EVT.

4.1.2 Extreme Value Theory

The objective of EVT is to quantify the probability of occurrence of extreme values in a distribution function. In fact, EVT models extreme events with large quantiles, i.e., *the tail of the distribution*, rather than central statistics such as the mean or the median. Recently, EVT has been applied to detect anomalies in many applications including network traffic data streams [240] and financial risk measuring [102]. The Peaks-Over-Threshold (POT) is a typical approach used to model the extreme values of samples that exceed a specific high threshold. This approach is a result of the Picakands-Balkema-de-Han theorem of EVT [27].

Let $(x_1, x_2, \ldots, x_n)$ be n iid random variables. Let F_u be their conditional excess distribution function, i.e., $F_u(\mathbf{x}) = P(X - u > \mathbf{x} | X > u)$, where u is a high threshold. As shown in Figure 4.1,

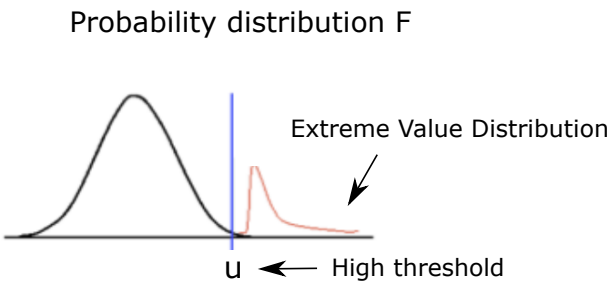


Figure 4.1: The POT method of EVT, adapted from [90].

the POT method models the extreme values that exceed the threshold u, using the Generalized Pareto Distribution (GPD) parametrized by two parameters, ξ and σ:

$$\tilde{F}_u(x) = P(X - u > x | X > u) = 1 - P(X - u < x | X > u) = 1 - F_u(x), \tag{4.1}$$

$$F_u(x) \to 1 - G_{\xi,\sigma}(x), \text{ as } u \to \infty \quad \text{where} \quad \begin{cases} G_{\xi,\sigma}(x) = 1 - (1 + \frac{\xi x}{\sigma})^{-\frac{1}{\xi}}, & \text{if } \xi \neq 0 \\ G_{\xi,\sigma}(x) = 1 - e^{-\frac{x}{\sigma}}, & \text{if } \xi = 0. \end{cases} \tag{4.2}$$

In practice, the two parameters of the GPD are empirically estimated by fitting the GPD to the data. The MLE can be used to find these optimal parameters $\tilde{\xi}$ and $\tilde{\sigma}$. Once the extreme values are modeled with the optimal GPD, $G_{\tilde{\xi},\tilde{\sigma}}$, we can identify rare extreme samples that have very low probabilities [240]. Given a small probability q, we can compute the threshold t_q such that, $P(X > t_q) < q$.

$$P(X - u > t_q | X > u) = \tilde{F}_u(t_q) \sim 1 - G_{\tilde{\xi},\tilde{\sigma}}(t_q). \tag{4.3}$$

$$\text{If } \xi \neq 0, \quad t_q \simeq u + \frac{\tilde{\xi}}{\tilde{\sigma}}((\frac{nq}{N})^{\tilde{\xi}} - 1), \tag{4.4}$$

where n is the total number of observations, and N is the number of x_i exceeding the threshold u, i.e., $x_i > u$. A key question arises as to how to choose the threshold u. Siffer et al. [240] state that "the value of u is not paramount except that it must be high enough." In practice, u is generally selected as a high empirical quantile of the data, e.g., 90% quantile. We refer the reader to this paper [240] for more details.

So far, we formalized the problem and we detailed the main theoretical background concerning VAE, NF, and EVT. Afterwards, we will explore the literature and focus on the most relevant robust generative AE designed for this problem: RVAE.

4.2 Related Work: Robust Variational Autoencoder (RVAE)

We review in this part a well-known robust generative model, particularly designed for unsupervised anomaly detection: RVAE [10]. In this work, Akrami et al. [10] leveraged the robust variational inference theory [98] to make the classical VAEs more robust to training outliers.

In the following sections, we will start by outlining the main theoretical concept of the robust variational inference used in RVAE: the β-divergence. We compare the robustness of the β-divergence against the classical KL-divergence. Then, we will present RVAE robust training strategy. We mainly review the objective function optimized by RVAE and the corresponding hyperparameter selection strategy. We conclude this section with the main limitations of this method.

4.2.1 Robust Variational Inference and the β-divergence

To begin with, the classical VAEs are trained to reconstruct the training data, based on stochastic encoder-decoder architecture. They specifically optimize the ELBO loss presented in Section 2.3.5.2 (see Equation 2.29). This function includes two parts: a reconstruction function that minimizes the data reconstruction error, and a regularizer term, computed using the KL-divergence, which forces the latent distribution to map a predefined prior distribution.

Nonetheless, VAE are sensitive to training contaminants, since the encoding process can be significantly impacted by these aberrations [10]. In fact, the classical MLE considers all training observations uniformly, so that they equally contribute to the optimization of the loss. Consequently, the training may be biased by training outliers.

More generally, let $\mathbf{x} = \{x_1, .., x_N\}$ be N iid samples, $p(\mathbf{x}, \theta)$ be a parametric model, parameterized by θ, and $\hat{p}(x)$ denote the empirical distribution:

$$\hat{p}(x) = \frac{1}{N} \sum_{i=1}^{N} \delta(x, x_i), \tag{4.5}$$

where δ is the Dirac delta function. MLE trains the parametric model to approximate the empirical distribution, $\hat{p}(x)$, by maximizing the data log-likelihood. This is equivalent to minimizing the KL-divergence, as shown in [11]:

$$D_{KL}(\hat{p}(x)||p(x;\theta)) = C - \frac{1}{N} \sum_{i=1}^{N} \ln p(x_i;\theta), \tag{4.6}$$

where $C \in \mathbb{R}$ is a constant. By minimizing the preceding KL-divergence, we aim to set the corresponding derivatives to 0, i.e.,

$$\frac{1}{N} \sum_{i=1}^{N} \frac{\partial}{\partial \theta} \ln p(x_i;\theta) = 0 \tag{4.7}$$

That is, all the derivatives weight equally in the MLE optimization, making it significantly sensitive to large errors of training outliers. To remediate this issue, several robust divergences were introduced in the literature, such as the β-divergence [31] and the γ-divergence [97]. Unlike the KL-divergence, robust divergences down-weight the likelihood of *unusual* training instances, to minimize their influence on the optimization. Even though the principle remains the same for all robust divergences, we focus here on the β-divergence.

The β-divergence between the empirical distribution $\hat{p}(x)$ and the parameterized distribution $p(x;\theta)$ is defined as follows:

$$D_\beta(\hat{p}(x)||p(x;\theta)) = \frac{1}{\beta}\int \hat{p}(x)^{\beta+1}dx - \frac{\beta+1}{\beta}\int \hat{p}(x)p(x;\theta)^\beta dx + \int p(x;\theta)^{\beta+1}dx \tag{4.8}$$

Minimizing the β-divergence results in the following equation:

$$\frac{1}{N}\sum_{i=1}^{N} p(x_i;\theta)^\beta \frac{\partial}{\partial\theta}\log p(x_i;\theta) - \mathbb{E}_{p(x;\theta)}[p(x;\theta)^\beta \frac{\partial}{\partial\theta}\log p(x_i;\theta)] = 0 \tag{4.9}$$

We refer the reader to [31, 98] for step-by-step description and proof. In the first part of Equation 4.9, we remark that the log-likelihood derivatives are weighted with the probability density of the training sample $p(x_i;\theta)^\beta$. As training anomalies are rare, their probability density is assumed to be lower than inliers. As such, this density down-weights the contribution of training outliers. It is noteworthy that all the weights $p(x;\theta)^\beta$ are equal to 1 when $\beta = 0$. This case ties in with the uniform training of the classical KL-divergence. Thus, β is an hyperparameter that controls the training robustness.

To compare the robustness of KL and β-divergences, let us consider this toy case, developed in [11]. Let $\mathbf{x} = \{x_1,...,x_N\}$ be N iid samples, generated from a mixture of two Gaussians:

$$\mathbf{x} \sim \alpha\mathcal{N}(\mu_1,\sigma_1) + \beta\mathcal{N}(\mu_2,\sigma_2) \tag{4.10}$$

We assume that the majority of the data are generated by the first dominant component and a minority of outliers are generated by the second component, i.e., $\alpha >> \beta$. The original data are plotted in Figure 4.2, which is extracted from [11] (see the black curve). We note that this toy data represent a typical situation in unsupervised AD, since it verifies the assumptions mentioned in Section 4.1.

We aim to robustly model this corrupted data and we fit a single-component Gaussian of mean μ and a standard deviation σ: $p(x;\theta) = \mathcal{N}(\mu,\sigma)$. This distribution is estimated using the KL and the β divergences, respectively. The optimized distributions are shown in Figure 4.2. The Gaussian estimated with the KL-divergence, which is visualized with the red curve, is significantly impacted by the outliers. This results in a flat Gaussian with a large standard deviation, to account for both Gaussian components. In contrast, outlier influence is down-weighted with the β-divergence. The resulting distribution only recovers the dominant component and ignores the outliers.

The robustness of the β-divergence compared to the classical KL-divergence motivates Akrami et al. [10] to design a novel robust VAE. They specifically propose a robust training strategy that reduces the influence of training contaminants. We will now detail this robust training strategy.

4.2.2 RVAE Training Strategy

In RVAE, Akrami et al. [10] propose to integrate the β-divergence into the classical VAE. For this reason, they firstly reformulate the training function of VAEs, i.e., the ELBO, by replacing the reconstruction log-likelihood with the KL-divergence, according to equation 4.6.

Let $\mathbf{x} = \{x_1,...,x_N\}$ be N iid samples, and $\mathbf{z} = \{z_1,...,z_N\}$ be their corresponding latent representations, extracted by the model. $p_\theta(x|z)$ denotes the distribution that maps back the latent encoding to the input space, via VAE stochastic decoder. $q_\phi(z|x)$ is the posterior distribution of the stochastic encoder, parametrized by ϕ. $\hat{p}(\mathbf{x})$ is the empirical distribution (see Equation 4.5).

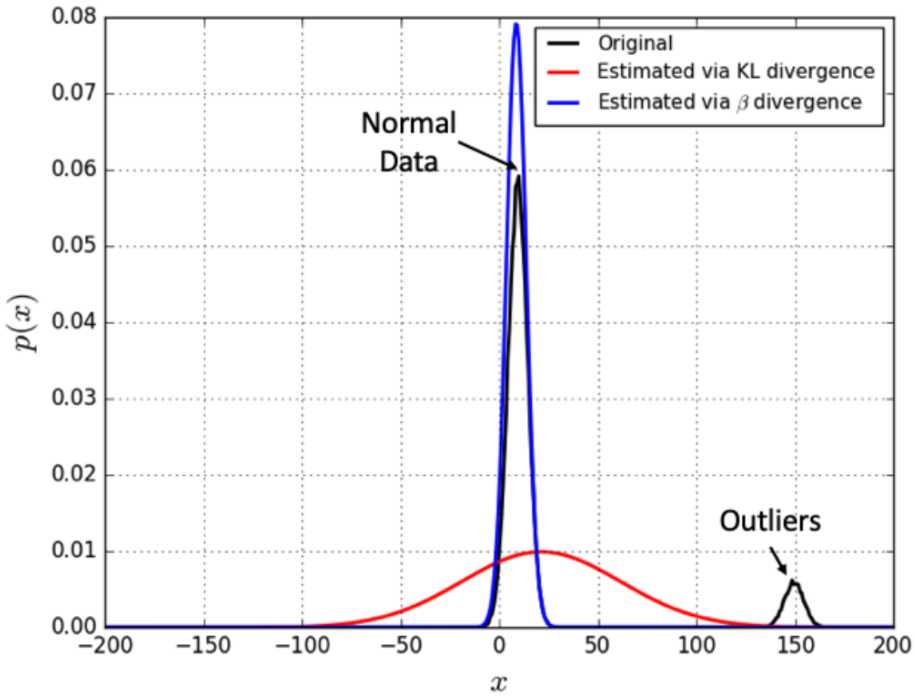


Figure 4.2: Comparison of the robustness of KL and β-divergences, extracted from [11].

$$L_{VAE} = L_{REC} + L_{KL} \tag{4.11}$$
$$= -\mathbb{E}_{z\sim q_\phi(z|x)} \log p_\theta(x|z) + D_{KL}(q_\phi(z|x)||p(z)) \tag{4.12}$$
$$= -N\mathbb{E}_{z\sim q_\phi(z|x)} D_{KL}(\hat{p}(\mathbf{x})||p_\theta(\mathbf{x}|\mathbf{z})) + D_{KL}(q_\phi(z|x)||p(z)) + C, \tag{4.13}$$

The authors propose to replace the reconstruction KL-divergence with a more robust divergence: the β-divergence. Thus, RVAE variational inference optimization is defined as follows:

$$L_\beta = -N\mathbb{E}_{z\sim q_\phi(z|x)} D_\beta(\hat{p}(\mathbf{x})||p_\theta(\mathbf{x}|\mathbf{z})) + D_{KL}(q_\phi(z|x)||p(z)). \tag{4.14}$$

Akrami et al. [11, 10] demonstrated the robustness of RVAE on multiple benchmark datasets from two different application fields: computer vision and network intrusion detection. They showed that RVAE is significantly more robust to training outliers than the classical VAEs. However, since RVAE training relies on the β-divergence, selecting the optimal β hyperparameter that balances the robustness and efficiency is central. A very small β results in tolerating too many training outliers, while a large β over-penalizes the majority of the data, since the weights $p(x;\theta)^\beta$ converges to 0 when $\beta >> 1$, regardless of the density $p(x;\theta)$. For this reason, and similar to RDAs, the value of β needs to be selected carefully with a dedicated exploratory strategy that implicitly requires labeled anomalies. In unsupervised anomaly detection, labeled anomalies are scarce and the hyperparameter selection may deliver unreliable results, a.k.a., hyperparameter misspecification.

To remediate this issue, we propose GRAnD, a robust generative autoencoder that is robust to the hyperparameter selection and outliers contaminating the training data. We introduce a novel training strategy that alternates between filtering outliers contaminating the training dataset and learning a robust representation of the norm. Our training strategy involves little architectural changes and can be integrated with VAEs [134] and NFs [215].

4.3 Contribution: GRAnD, Robust VAEs and NFs for Unsupervised Network AD

So far, we presented the state-of-the-art Robust Variational AE (RVAE). We reviewed the main limitations of this method, especially its sensitivity to hyperparameter selection. In this section, we aim at presenting our second contribution: GRAnD, a robust generative AE designed for unsupervised anomaly detection. Subsequently, we first introduce in Section 4.3.1 the EVT-based training strategy that allows GRAnD to reject training contaminants. Then, we formalize GRAnD training optimization strategy by defining the training loss.

Since we focus on unsupervised anomaly detection, we assume that the majority of the training data are nominal, along with a small ratio of contaminants, i.e. outliers. The ratio of these contaminants, which we call γ, is not known in advance.

As mentioned previously, we propose a new training strategy that enhances the robustness of two classical generative models: VAEs and NFs, with minimal changes to the original model architecture. Our contribution alternates between filtering training outliers and learning a robust distribution of the norm, as shown in Figure 4.3. Initially, all the training data are considered nominal and GRAnD incrementally down-weights the potential outliers, using the EVT-based rejection process. Even though these outliers are not well-sampled from the whole positive class, we find that they provide global insights that can be generalized to unseen anomalies. Accordingly, we optimize the model parameters to recover a robust distribution, where inliers are well-modeled and the filtered outliers are badly represented. Subsequently, we will first explain the rejection strategy that isolates training contaminants. Then, we will detail the objective function to optimize.

4.3.1 Robust Rejection Strategy

The rejection strategy aims to separate nominal training data points from potential outliers. The main idea consists in setting a relevant threshold to segment the reconstruction scores assigned to training samples, in order to reject outliers having extreme scores. Unlike traditional AD approaches, GRAnD relies on an EVT-based rejection strategy, which is more flexible and does not require any assumption about the underlying distribution of the data.

We hypothesize that, early in the training phase, contaminants have larger free energy (cf. Equation 2.29), compared to inliers. Consequently, we propose to isolate these extreme values by thresholding the energy with the POT approach, described in Section 4.1.2. The POT approach requires the selection of two parameters: the initial threshold u, and the risk parameter q.

In our experiments, we define u as follows :

$$u = Q_3(\mathscr{F}) + \alpha \, \mathrm{IQR}(\mathscr{F}) \tag{4.15}$$

where $\mathscr{F}$ is the free energy of the training instances, Q_3 is the third quartile, and the Inter-Quartile Range (IQR) is defined as the difference between the third and the first quartiles. α controls the scale of the decision rule. In all our experiments, we fixed $\alpha = 1.5$ and $q = 0.001$. In Sections 4.4.2 and 4.4.2.3, we study the sensitivity of our contribution with respect to α and q.

To extract an outlier-free subset from the unlabeled training data, we isolate the extreme instances having low likelihoods, at the end of each training epoch. Using the POT parameters, we propose to split the input data into three subsets $\mathbb{X} = \mathbb{L} \cup \mathbb{S} \cup \mathbb{U}$, as illustrated in Figure 4.4:

- The subset $\mathbb{L}$ contains nominal training samples, having energy lower than the initial threshold u of the POT method;
- $\mathbb{S}$ contains anomalous data points, with an energy higher than t_q, computed using equation 4.4;

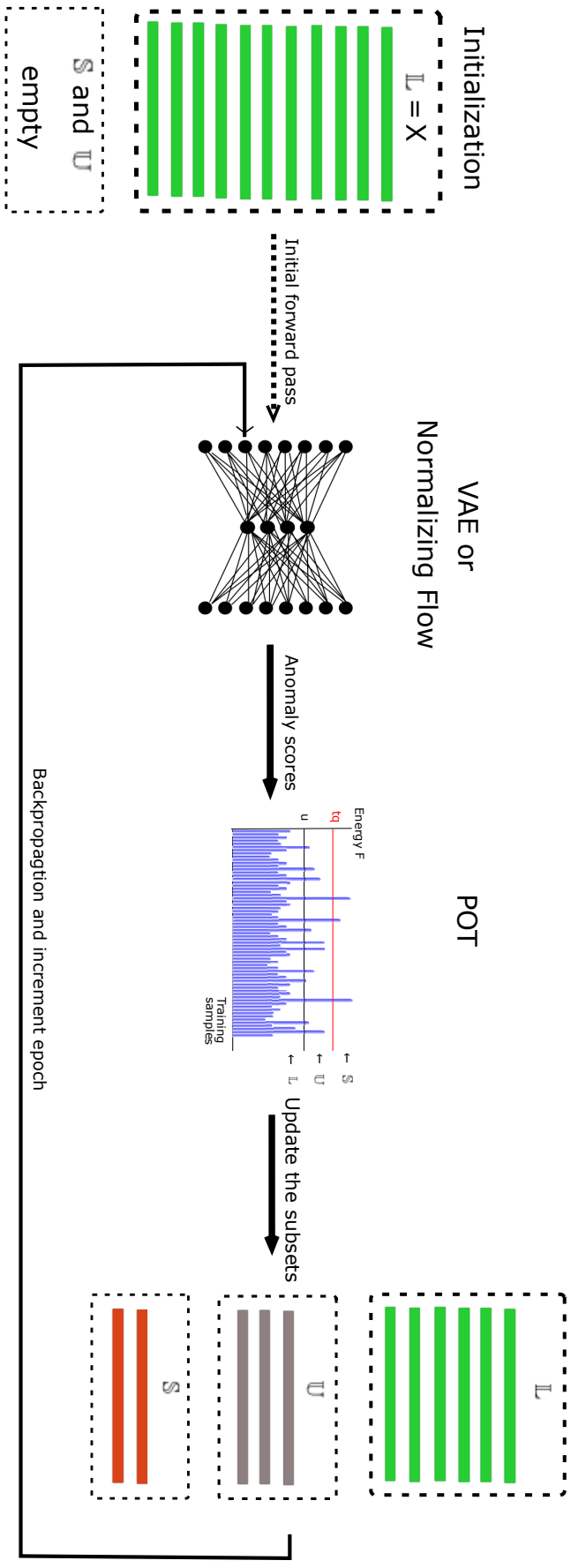


Figure 4.3: The workflow of the proposed approach, GRAnD.

- $\mathbb{U}$ comprises the remaining critical samples, with an energy higher than u and lower than t_q. These sample energies are neither low enough to be considered nominal, nor high enough to be rejected as anomalies.

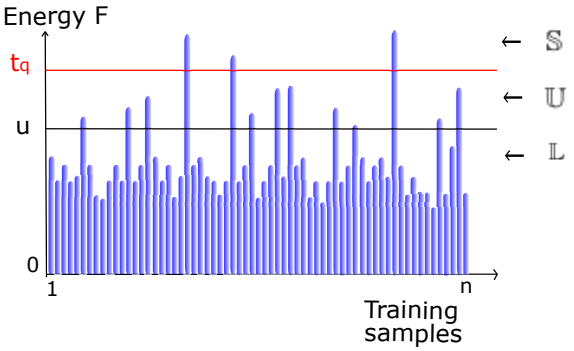


Figure 4.4: Illustration of the rejection strategy using the POT approach.

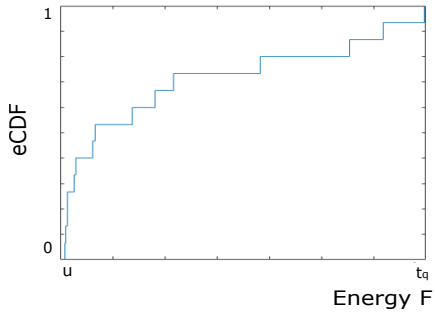


Figure 4.5: Empirical cumulative distribution function of $\mathbb{U}$ samples

4.3.2 Training Loss

The rejection strategy splits the training data into three subsets $\mathbb{L}$, $\mathbb{S}$, and $\mathbb{U}$. We train the autoencoder to jointly perform three tasks: (i) minimize $\mathbb{L}$-sample energies, (ii) badly reconstruct $\mathbb{S}$-samples by maximizing their energies, (iii) maximize a weighted energy function of $\mathbb{U}$ instances, which takes into account the classification uncertainty of these samples.

Let $\mathbb{U} = \{X_1, X_2, \ldots, X_n\}$ contain a sequence of n iid instances. We firstly sort these instances in increasing order according to their free-energies $(\mathscr{F}(X_1), \mathscr{F}(X_2), \ldots, \mathscr{F}(X_n))$, i.e., $\mathscr{F}(X_1) \leq \mathscr{F}(X_2) \leq \ldots \leq \mathscr{F}(X_n)$. We use the empirical Cumulative Distribution Function (eCDF) to define the anomalousness weight of each $X_i \in \mathbb{U}$.

$$P(X_i \in \mathbb{U} \text{ is anomalous}) = eCDF_n(\mathscr{F}(X_i)) = \frac{1}{n}\sum_{j=1}^{n} \mathbb{1}_{\mathscr{F}(X_j) \leq \mathscr{F}(X_i)} \tag{4.16}$$

where $\mathbb{1}$ is the indicator function. As illustrated in Figure 4.5, $\mathbb{U}$ samples with energies close to the threshold u have a small probability close to 0. Conversely, samples with high scores, i.e., close to t_q, have probabilities close to 1.

GRAnD Objective Function

Given the three subsets of data $\mathbb{L}$, $\mathbb{S}$, and $\mathbb{U}$, respectively generated from the three distributions, D_L, D_S, and D_U, GRAnD optimizes the following objective function:

$$\mathscr{L}(x) = \mathbb{E}_{x\sim D_L}[\mathscr{F}_{\mathbb{L}}(x)] + |m - \mathbb{E}_{x\sim D_S}[\mathscr{F}_{\mathbb{S}}(x)]| + eCDF_m(\mathscr{F}_{\mathbb{U}}(x))\,|m - \mathbb{E}_{x\sim D_U}[\mathscr{F}_{\mathbb{U}}(x)]| \tag{4.17}$$

The objective function comprises three components:

- $\mathbb{E}_{x\sim D_L}[\mathscr{F}_{\mathbb{L}}(x)]$ is the expectation of the free energy function of $\mathbb{L}$ samples, defined in Equation 2.29. This first component aims to minimize the energy of $\mathbb{L}$ samples.

- $\mathbb{E}_{x\sim D_S}[\mathscr{F}_{\mathbb{S}}(x)]$ is the expectation of the free energy function of $\mathbb{S}$ samples. $|.|$ is the absolute distance, and $m \in \mathbb{R}^+$ is a margin value. By maximizing this energy, we train the autoencoder to badly reconstruct the potential training contaminants. Since this energy function is positive and unbounded, we propose to fix an upper bound m, to prevent divergence during training.
- $\mathbb{E}_{x\sim D_U}[\mathscr{F}_{\mathbb{U}}(x)]$ is the expectation of the free-energy function of $\mathbb{U}$ samples. We weight the objective function of $\mathbb{U}$ instances according to their anomalousness probability, computed with the eCDF function. These weights account for the uncertainty of the classification of $\mathbb{U}$ instances.

4.3.3 Hypotheses

In this chapter, we conjecture the following hypothesis:

- **Hypothesis 4.1 (H4.1)** *: Thanks to the dedicated EVT-based rejection strategy and the proposed uncertainty-aware training process, we conjecture that GRAnD is robust to the hyperparamter selection. As such, a slight change of model hyperparameters does not impact the anomaly detection performance.*

In the next Section 4.4, we aim to experimentally validate our second contribution on the two public datasets: NSL-KDD and MedBIoT. Firstly, we compare GRAnD AD performance against the competing methods of the literature. We investigate the robustness of these methods in the presence of different ratio of contamination. Secondly, we explore the validity of the conjectured hypothesis regarding our contribution sensitivity regarding its three hyperparamters: the initial threshold u, the risk parameter q, and the margin m.

4.4 Experiments and Results

To inspect the robustness of our second contribution, we carry out two sets of experiments on the same benchmark datasets used in the previous chapter: NSL-KDD and MedBIoT.

4.4.1 NSL-KDD dataset

4.4.1.1 Dataset Description

Firstly, we conduct a first set of experiments on the NSL-KDD dataset [196], which was previously presented in Section 3.4.1.

4.4.1.2 Training Protocols

Protocol 1: GRAnD Robustness Regarding Training Contamination

This first protocol investigates the efficacy of our contribution compared to competing unsupervised anomaly detectors. The competing methods included in the present experiments are OSVM with a Gaussian kernel, IF, VAE, RVAE, DAGMM, NF, and RVAE. Similar to the training protocols of the previous chapter (cf. Section 3.4.1.2), we study the sensitivity of anomaly detectors concerning the ratio of anomaly contamination by varying the training anomaly percentage. We prepare four training subsets, containing 0%, 5%, 10%; and 15% of outliers, respectively. Anomaly instances are selected randomly from all classes of NSL-KDD training anomalies.

Protocol 2: GRAnD Sensitivity Analysis Regarding Hyperparameter Selection

The second protocol explores our contribution robustness with respect to its three hyperparameters: the initial threshold u, the risk parameter q, and the margin m. We train different models GRAnD with distinct hyperparameters to study the variation of the performance on the same test subset. Indeed, we ran multiple experiments where we only vary one hyperparameter and we keep the remaining ones fixed. These runs are performed with a contamination ratio $\gamma = 10\%$, repeated five times, and we report the average results over these five repetitions.

4.4.1.3 Training Parameter Settings and Evaluation Criteria

The optimal hyperparameters selected in our first set of experiments are reported in Table 4.1.

Table 4.1: The hyperparameters used in the NSL-KDD experiments.

Hyperparamter	Value
Architecture	a 3-layer FFN with 122-8-122 units
Learning rate	0.001
Maximum number of epochs	500
Number of epochs for each data projection	100
Batch size	256
The initial threshold u	1.5 IQR
The risk parameter q	0.001
The margin m	100
Laptop	12-core Intel i7-9850H CPU clocked at 2.6GHz, with 16GB of RAM memory and with NVIDIA Quadro P2000 GPU

In all experiments, we use the standard FFN architecture for all autoencoders. All autoencoder-based methods share the same architecture, composed of a 3-layer FFN with 122-8-122 units. The latent layers are followed by the ReLU activation function. The last layer of the decoder is followed by the Sigmoid function. We use an adaptive learning rate: initially, we use a learning rate of 0.001, which is divided by two if the training loss does not decrease after 20 consecutive epochs. We stop the training when the learning rate is lower than 10^{-6} or the number of epochs becomes higher than 500 epochs. We use a batch size of 256 in all experiments. We initialize model parameters randomly. To limit the impact of random parameter initialization, we repeat each experiment five times and average the results over these five runs.

Our approach comprises three specific hyperparameters: the rejection parameter α that controls the initial threshold u, the risk parameter q, and the margin m. In all experiments, q is fixed to 0.001, α to 1.5, and m to 100. We conduct a sensitivity analysis experiment in Sections 4.4.2 and 4.4.2.3, to assess our approach robustness regarding the hyperparameters.

We fine-tune competing methods hyperparameters with the greed search method. The experiments were run on a laptop equipped with a 12-core Intel i7-9850H CPU clocked at 2.6GHz and with NVIDIA Quadro P2000 GPU.

4.4.1.4 Results

In this section, investigate model sensitivity regarding training contaminants and hyperparameter misspecification.

Protocol 1: GRAnD Robustness Regarding Training Contamination

Figure 4.6 presents the results of the comparison between GRAnD and other competing methods on the NSL-KDD. We particularly investigate the performance variation for different outlier ratio $\gamma_p \in \{0\%, 5\%, 10\%, 15\%\}$.

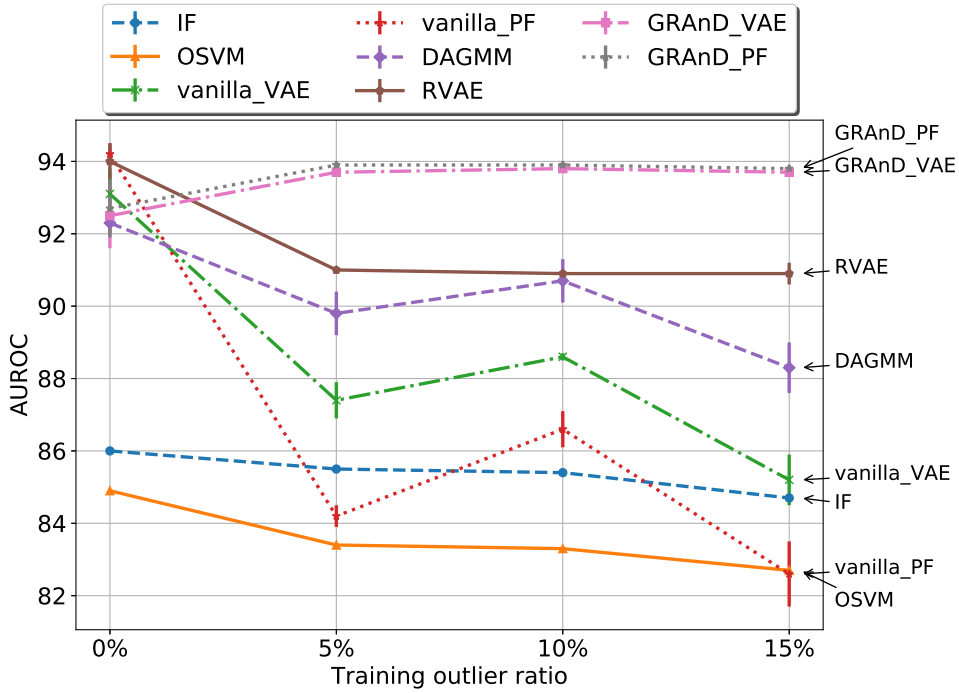


Figure 4.6: NSL-KDD experimental results: comparison of AD methods based on average AUROCs and deviations over five runs for multiple contamination ratios.

Firstly, when the training data are contaminated with anomalies, our approach significantly outperforms competing methods. While the performance of competing methods decreases with higher pollution ratios γ, our approach is more stable, with an average AUROC around 94% and very little deviation, for the three contamination ratios 5%, 10%, and 15%. Specifically, it is important to note that GRAnD is significantly more robust than basic VAEs and NFs. Moreover, GRAnD outperforms RVAE for the three contamination ratios $\{5\%, 10\%, 15\%\}$. We report a 3% points increase for mean AUROC, in the three cases. We mention that these results are statistically significant according to Welch's test with a p-value=0.05 These results mainly highlight the benefit of the robust rejection strategy, where no prior knowledge about the outlier ratio is required in advance.

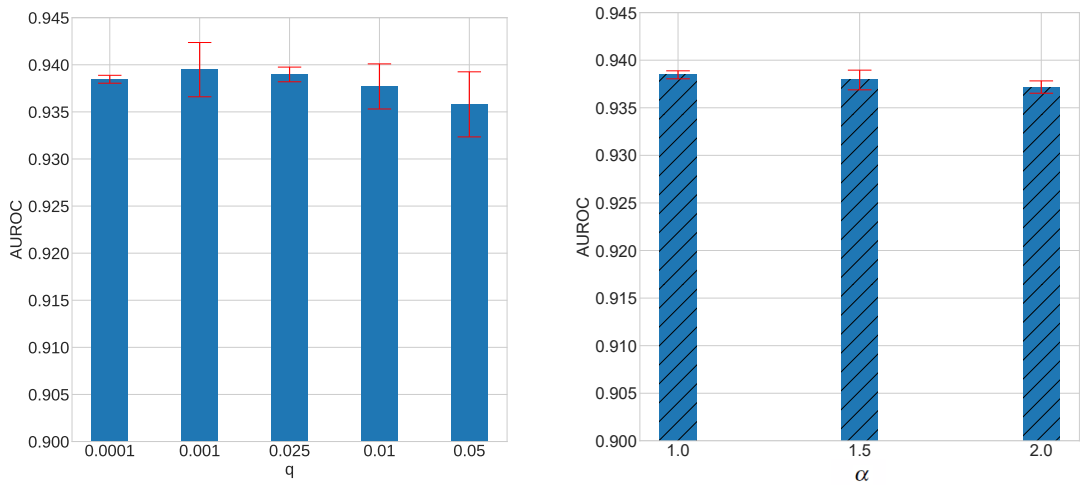
Secondly, when the training data contain only nominal data, i.e., when $\gamma = 0\%$, all neural network-based baselines report similar performance, with an AUROC around 93%. The two classical anomaly detectors, IF and OSVM, show poor results, in contrast, with an AUROC lower than 86%. This first observation favors neural network-based anomaly detectors, for this high-dimensional input data.

Thirdly, when the training data are anomaly-free, GRAnD performance slightly degrades, with an AUROC of 92.6% with a standard deviation of 0.8%. This observation can be explained by the fact that GRAnD leverages training outliers to learn a robust projection, where inliers are

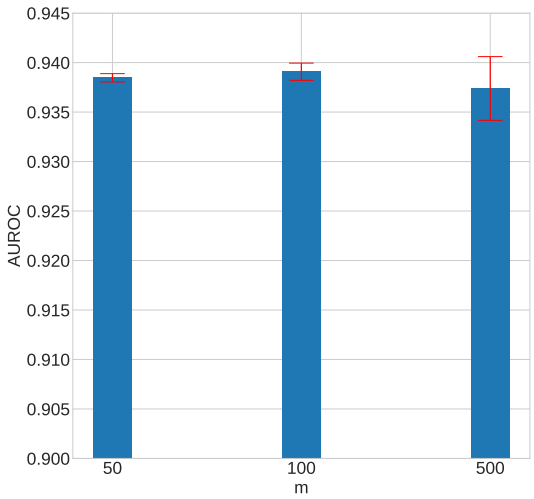
well reconstructed, while outliers are poorly reconstructed. When training data do not contain anomalies, GRAnD-PF and GRAnD-VAE performances are very similar to vanilla-PF and vanilla-VAE, respectively. Despite this slight decrease, GRAnD remains very competitive, with around 6% points better AUROC than IF.

Finally, for all contamination ratios, GRAnD-PF slightly outperforms GRAnD-VAE. In fact, since NFs propose a richer family of latent distributions, they are more flexible than VAEs. Consequently, this improves the generalization of the model and refines the downstream task of anomaly detection. In the conclusion, these first results show that GRAnD is robust to the ratio of training contaminants, on the NSL-KDD dataset.

Protocol 2: GRAnD Sensitivity Analysis Regarding Hyperparameter Selection



(a) GRAnD-PF Sensisitivity analysis regarding q. *(b) GRAnD-PF Sensisitivity analysis regarding α.*



(c) GRAnD-PF Sensisitivity analysis regarding m.

Figure 4.7: Sensitivity analysis of GRAnD-PF on the NSL-KDD dataset.

We conduct a second set of experiments to explore GRAnD sensitivity with respect to its hyper-

parameters, m, u, and q. As mentioned in numerous works in the anomaly detection community [217, 292, 92], it is advocated to develop robust anomaly detectors that do not depend on user-defined parameters. The sensitivity to hyperparameters is problematic in unsupervised AD, since outlier labels are scarce, and the selection of the optimal hyperparameters is not guaranteed. We conduct further experiments to assess the sensitivity of our approach regarding its hyperparameters. We train different models with distinct hyperparameters to study the variation of the performance on the same test subset. We report in Figure 4.7 the results of the sensibility analysis GRAnD-PF on the NSL-KDD dataset, with $\gamma_p = 10\%$.

Overall, a slight variation of GRAnD hyperparameters α, m, and q does not significantly the AD performance, with a mean AUROC as high as 93.8%. We firstly focus on the sensitivity analysis with respect to $q \in \{0.0001, 0.001, 0.025, 0.01, 0.05\}$. We remark that the result is globally stable for the five values of q. The best results are reported when $q = 0.001$, with an AUROC of $93.9\% \pm 0.4$. The result slightly decreases when q increases, and reaches $93.6\% \pm 0.3$. This 0.3% points decline of AUROC is however not significant according to Welch's test, with p-value = 0.05. For high values of q, the model may reject many training samples, including potential false positives, which results in a slight degradation of performance. The same observation is valid for the sensitivity analysis results regarding α and m. Even though we report a slight decrease with large hyperparameters, the overall performance is consistently stable, with an AUROC around 93.8%. This validate our hypothesis (H4.1): GRAnD is relatively insensitive to hyperparameter selection.

In the following, we repeat the same set of experiments on an IoT dataset: the MedBIoT dataset, to explore the effectiveness of our contribution particularly in IoT network traffic analysis.

4.4.2 MedBIoT dataset

4.4.2.1 Dataset Description and Training Protocols

The second set of experiments are conducted on the IoT benchmark dataset: the MedBIoT. This dataset is introduced in Section 3.4.2. The same training protocols, as in Section 4.4.1.2, are followed in this part.

4.4.2.2 Training Parameter Settings and Evaluation Criteria

Training Parameter Settings are similar to the first set of experiments (see Section 4.4.1.3). The single difference is the autoencoder architecture. In these experiments, all autoencoders share the same architecture: a 5-layer FNN with 61-32-16-32-61 units. Indeed, this architecture is selected after a set of preliminary tests, and fixed for all baselines, to grant a fair performance comparison. In all experiments, q is equal to 0.001, α to 1.5 , and m to 100 . We conduct a sensitivity analysis experiment in Section 4.4.2.3 to assess our approach robustness regarding the hyperparameters. The hyperparameters of competing methods are fine-tuned on a dedicated validation subset.

The optimal hyperparameters selected in this second set of experiments are reported in Table 4.2.

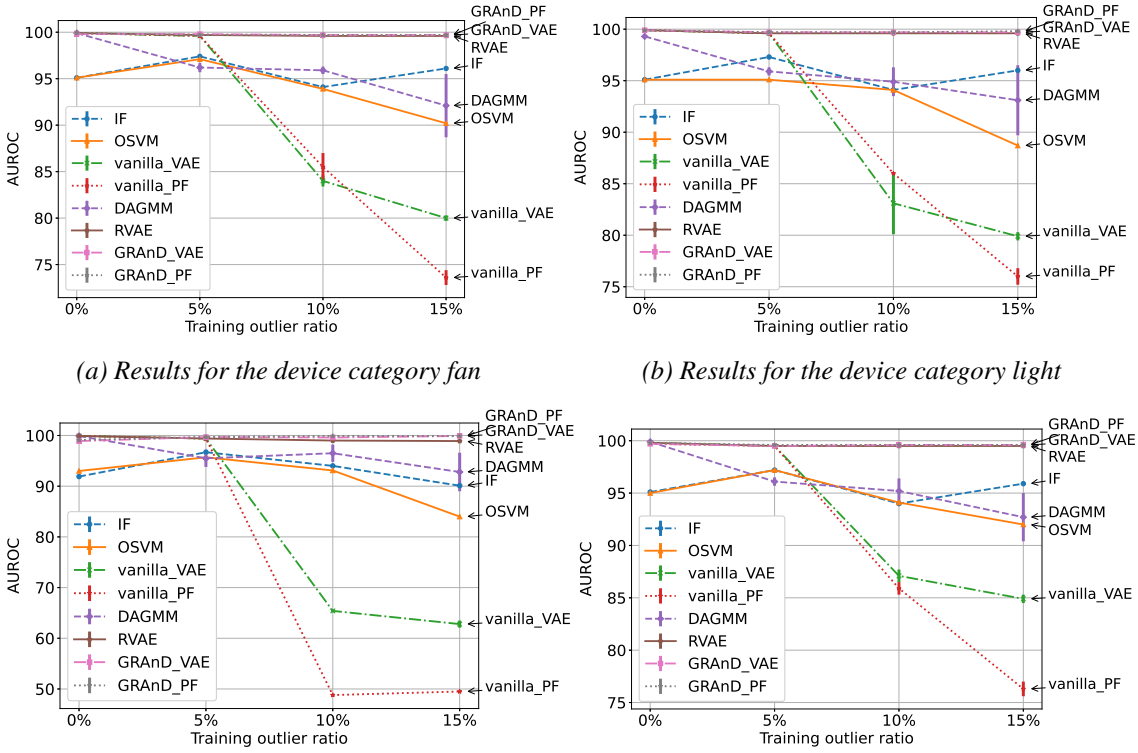
4.4.2.3 Results

Protocol 1: GRAnD Robustness Performance

We present the MedBIoT results in Figure 4.8. As mentioned previously, we train an anomaly detector for each device type. We obtain similar results for the four device types. Due to space constraints, we report the most representative results in Figure 4.8. For the four device types, and for all contamination ratios, GRAnD-PF, GRAnD-VAE, and RVAE outperform other anomaly detectors, with an AUROC of $99.9 \pm 0.1\%$. In particular, we highlight the robustness of our contribution compared to classical VAEs and . While the latter performances are considerably impacted when the

Table 4.2: The hyperparameters used in the MedBIoT experiments.

Hyperparamter	Value
Architecture	a 5-layer FNN with 61-32-16-32-61 units
Learning rate	0.001
Maximum number of epochs	500
Number of epochs for each data projection	100
Batch size	256
The initial threshold u	1.5 IQR
The risk parameter q	0.001
The margin m	100
Laptop	12-core Intel i7-9850H CPU clocked at 2.6GHz, with 16GB of RAM memory and with NVIDIA Quadro P2000 GPU



(a) Results for the device category fan

(b) Results for the device category light

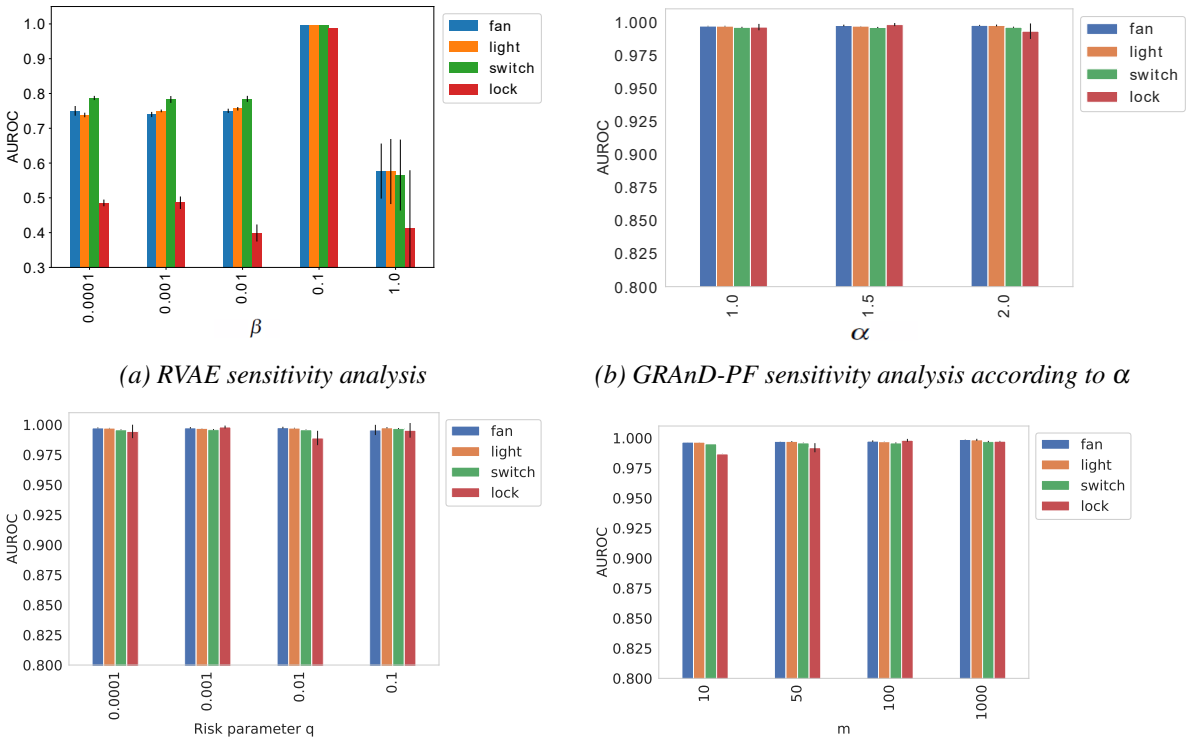
(c) Results for the device category lock

(d) Results for the device category switch

Figure 4.8: The MedBIoT experimental results, for the the four device categories: fan, light, lock, and switch. We report the average AUROC with the standard variation over five runs.

contamination ratio is higher than 10%, GRAnD yields stable results. For example, for $\gamma_p = 10\%$, GRAnD-PF and GRAnD-VAE exceed vanilla-VAE and vanilla-PF AUROCs by 19% and 23%, on average. Consequently, the robustness of GRAnD for IoT network traffic anomaly detection is validated on this dataset. Although GRAnD and RVAE yield close results, we will show in the next section that, unlike RVAE, GRAnD is robust to the hyperparameter selection.

Protocol 2: GRAnD Sensitivity Analysis Regarding Hyperparameter Selection



(a) RVAE sensitivity analysis

(b) GRAnD-PF sensitivity analysis according to α

(c) GRAnD-PF sensitivity analysis according to q

(d) GRAnD-PF sensitivity analysis according to m

Figure 4.9: Sensitivity analysis of RVAE and GRAnD-PF on MedBIoT dataset. We report the average AUROC with the standard variation over five runs.

In Figure 4.9a, we show RVAE performance for different $\beta \in \{0.0001, 0.001, 0.01, 0.1, 1\}$. Since GRAnD is defined using three hyperparameters, m, q, and α, we run three experiments, where we only vary one hyperparameter and we keep the remaining ones fixed. Figure 4.9a shows that RVAE is sensitive to the hyperparameter β. For all device types, RVAE AUROC drastically decreases, when β changes. In contrast, GRAnD-PF performance is not impacted by the variation of its hyperparameters, and the AUROC is stable around $99.8\% \pm 0.1$.

4.5 Conclusion

In this chapter, we proposed GRAnD, a robust generative method for unsupervised non-sequence AD. This contribution addresses one limitation of our first method RADON, developed in Chapter 3, thanks to the uncertainty modeling of the noisy training data. Our approach uses Extreme Value Theory to identify outliers contaminating the data and train a generative AE to learn a robust representation, where inliers can be accurately reconstructed, while outlier reconstructions are corrupted. Extensive experiments were conducted on benchmark datasets, and showed that our approach outperforms classical anomaly detection methods, all the while showing outstanding robustness to hyperparameter selection. This contribution has been published in an international conference research paper [187].

So far, our study has been focused on the punctual AD of non-sequence data. To make the underlying problem more tractable, we assumed that the data points are iid and their chronological

order was ignored during the modeling phase. This assumption restricts our first two contributions detection spectrum: only punctual anomalies can be detected and both contextual and collective ones pass under the radar. We therefore seek to propose a new approach that extends RADON and to sequential data analysis. We tackle this challenge in the next chapter, with the Transformers.

5

RESIST: Robust Transformer for Unsupervised Time Series Anomaly Detection

"In nature, we never see anything isolated, but everything in connection with something else which is before it, beside it, under it and over it."
Johann Wolfgang von Goethe

Contents

5.1 Problem Statement . . . 103
5.1.1 Nature of Input Data and Anomalies . . . 103
5.1.2 Unsupervised time Series Anomaly Detection . . . 104
5.1.3 Robustness to Training Contamination . . . 105
5.2 Related Work: AnomalyTransformer, Time Series Anomaly Detection with Association Discrepancy . . . 105
5.2.1 AnomalyAttention . . . 106
5.2.2 Series Association . . . 106
5.2.3 Prior Association . . . 107
5.2.4 Association Discrepancy . . . 107
5.2.5 AnomalyTransformer Global Architecture . . . 107
5.2.6 Minimax Training Strategy . . . 108
5.2.7 Test Point Anomaly Score . . . 109
5.3 Contribution: RESIST, Robust Transformer for Unsupervised Time Series Anomaly Detection . . . 109
5.3.1 Beyond Self-Sequence Dependencies . . . 109
5.3.2 Architecture Overview . . . 110
5.3.3 Siamese Encoder . . . 110
5.3.3.1 Self-attention and Co-attention Module . . . 112
5.3.3.2 FFN Layer . . . 113
5.3.4 Fusion Strategy . . . 113
5.3.5 RESIST Decoder . . . 114
5.3.6 Robust Training Loss . . . 114

5.3.7 Hypotheses . . . 115
5.4 Experiments and Results . . . 116
5.4.1 CICIDS2017 dataset . . . 116
5.4.1.1 Dataset Description . . . 116
5.4.1.2 Data Preprocessing . . . 117
5.4.1.3 Training and Testing Protocols . . . 117
5.4.1.4 Training Parameter Settings and Evaluation Criteria . . . 120
5.4.1.5 Results . . . 120
5.4.1.6 Synthesis and Discussion . . . 126
5.4.2 TON-IoT Dataset . . . 128
5.4.2.1 Dataset Description . . . 128
5.4.2.2 Data Preprocessing . . . 128
5.4.2.3 Training Protocol and Parameter Settings . . . 128
5.4.2.4 Results . . . 129
5.4.2.5 Synthesis and Discussion . . . 130
5.5 Comparison of the Three Contributions: RADON, GRAnD, and RESIST . 130
5.6 Conclusion and Perspectives . . . 131

We now propose to widen the scope of our study by exploring unsupervised AD in sequential data, a.k.a., time series. As mentioned previously, AD approaches are specific to the type of data. The two previous contributions are tailored to punctual AD of non-sequence data, as they do not consider the sequential arrangement of the input observations. However, the strong assumption that the data points are iid is not always valid in practice. Particularly, network traffic data may be interrelated to each other and can present contextual point-to-point dependencies. The presence of such dependencies significantly impacts the AD task and outliers are correspondingly detected based on the contextual anomalousness.

The temporal AD is a more laborious task with more challenges linked to the presence of temporal correlation in the noisy training data. A plethora of methods has been developed in the literature to address this problem [40, 144, 74, 69]. In particular, dimensionality reduction-based models have been extended to model the data temporal correlations and detect any deviation. Among these methods, Transformers [260] has received increasing attention in recent years in the machine learning and AD communities. Nonetheless, similar to classical AEs, these model performance depends on the availability of anomaly-free training data and they are sensitive to the presence of contaminants.

In this chapter, we introduce a novel method, designed to tackle the challenging problem of robust AD in sequential data. We introduce a robust learning strategy that trains a Transformer to model the nominal behavior of the network activity. Unlike most of the existing sequential anomaly detectors, our approach does not require the availability of an anomaly-free training subset. Relying on a contrastive learning-based robust loss function, our contribution automatically downweights atypical corrupted training data, to reduce their impact on training optimization.

This fifth chapter is organized as follows: Section 5.1 introduces the peculiarities of AD in time series. following up on this problem formalization, Section 5.2 surveys the state-of-the-art method tailored to solve the present problem. We identify the main limitations of this method and we then propose in Section 5.3 the third contribution of the thesis. We experimentally validate this contribution in Section 5.4 and we conclude this chapter with an empirical comparison between the three contributions of the thesis.

5.1 Problem Statement

This chapter explores unsupervised AD of multivariate sequential data. We begin with the description of the sequence data and the types of the detected anomalies (cf. Section 5.1.1). Then, we define the problem of unsupervised AD in sequential data (cf. Section 5.1.2). Finally, we present and justify the motivation for developing anomaly detectors to address this topic (cf. Section 5.1.3).

5.1.1 Nature of Input Data and Anomalies

This chapter analyzes sequence data. Sequence data are defined as *an ordered set* of data points, each of which is recorded at a time t [40]. A sequence can be either *continuous* or *discrete*. A *discrete* time series is a sequence where the observations are recorded at a discrete set of times, i.e., $t \in \mathbb{N}$. A *continuous* time series is a sequence where observations are continuously recorded over an interval of time T, i.e., $t \in [0, T]$. Most of the existing time series anomaly detectors [62, 40] analyze discrete sequences. This chapter is in line with these studies and focalizes on discrete sequences.

Similar to non-sequence data, sequences can be univariate or multivariate, depending on the dimension d sequence observation. While univariate sequences comprise a single attribute events, i.e., d = 1, multivariate sequences contain an ordered set of multidimensional vectors. The difference

between both types is illustrated in Figure 5.1, extracted from [40].

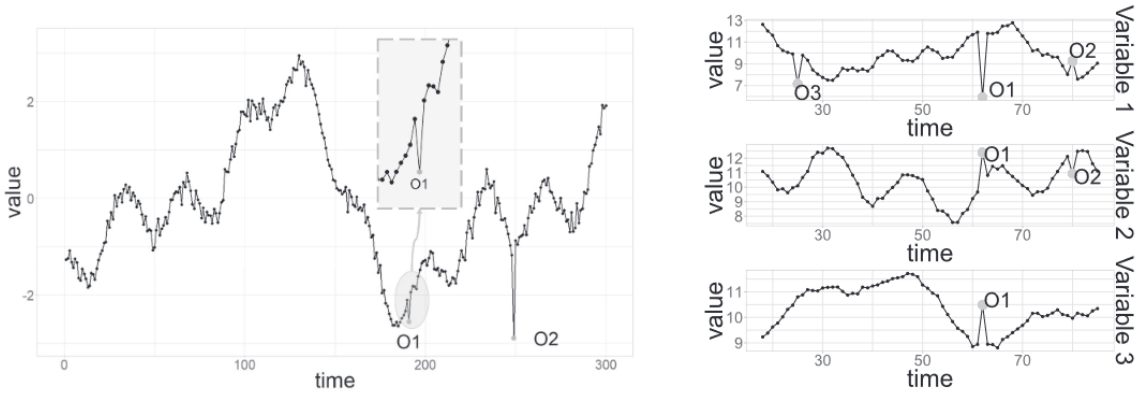


Figure 5.1: Univariate (left) and multivariate (right) time series, extracted from [40]. O1, O2, O3 are three anomalies.

Here, we focus on multivariate discrete time series. As mentioned in Chapter 2, multivariate time series outliers can be either univariate or multivariate, depending on whether they impact a single attribute, e.g., O3 in Figure 5.1, or multiple dimensions simultaneously, e.g., O1 and O2 in Figure 5.1, respectively. Moreover, temporal anomalies can be either punctual, contextual, or collective anomalies.

5.1.2 Unsupervised time Series Anomaly Detection

The problem that we address in this chapter is unsupervised time series AD. This is a critical problem that has been a point of interest for researchers in different application fields such as network and object monitoring, medical data analysis, fraud detection, and network intrusion detection [46]. In such fields, the outlier detection task mainly relies on the well-known *temporal continuity* assumption. Aggarwal [6] defines the temporal continuity assumption as "the fact that the patterns in the data are not expected to change abruptly unless there are abnormal processes at work." As such, a temporal outlier can be seen as an abrupt change in the data pattern, which results in a discontinuity of the data with its local context. This assumption makes temporal anomaly detection more challenging than the classical task, since considering the ordinal structure between the observations is of paramount importance.

In the following, we provide the common formal definition of unsupervised temporal anomaly detection.

Problem 5.1 *Let* $\mathbf{X}$ *be an* ***ordered set*** *of* T *successive observations, each of which is a vector of* d *dimensions, i.e.,* $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_t, ..., \mathbf{x}_T)$*, where each observation* $\mathbf{x}_t \in \mathbb{R}^d$ *is recorded at a timestamp* t*.* $T \in \mathbb{N}$*,* $d \in \mathbb{R}^+$*, and* $1 \leq t \leq T$*. An unsupervised anomaly detector aims at learning a function*

$$f_w : \mathbb{R}^d \to \mathbb{R}, \quad \mathbf{x}_t \mapsto s_t = f_w(\mathbf{x}_t) \tag{5.1}$$

f_w *models the local dependence of an observation* $\mathbf{x}_t$*, recorded at* t*, with the contextual adjacent points localized in the window* $\mathbf{X}_{t-w+1:t} = \{\mathbf{x}_{t-w+1}, ..., \mathbf{x}_t\}$*. For each* $\mathbf{x}_t$*,* f_w *can output either an anomaly score* $s_t \in \mathbb{R}$ *that quantifies the degree of anomalousness of the input observation or a binary label* $s_t \in \{0, 1\}$*. The difference between the two strategy was detailed in Chapter 3 (cf. Section 3.1.2).*

Finally, time series AD follow the same two assumptions defined in the non-sequence AD problem: [173]:

- **Assumption 5.1** *The anomalous data are statistically different and distinguishable from the nominal data.*
- **Assumption 5.2** *The majority of the training data are nominal. These data probably include a minority of anomalies, a.k.a., data contamination or pollution.*

The aim is to develop a *robust* time series anomaly detector, which is not sensitive to the presence of training contaminants.

5.1.3 Robustness to Training Contamination

The same definition of *robustness* that was introduced in Section 3.1.3 of Chapter 3 is followed in this part. Robustness means "insensitivity to small deviations from the assumption" [120].

in fact, unsupervised time series anomaly detectors assume that most of the training data is nominal. However, a small part of the data may be contaminated, a.k.a., polluted, with anomalous instances. The data distribution can be modeled in this case using a mixture of two distributions:

$$\mathbb{P} \equiv (1-\eta)\mathbb{P}^{+} + \eta\mathbb{P}^{-}, \tag{5.2}$$

where $\mathbb{P}^{+}$ is the nominal data distribution, $\mathbb{P}^{-}$ is the distribution of contaminants, and $\eta \in [0..1]$ is the contamination ratio. In general, η is unknown and assumed to be less than 0.5, since training anomalies represent the minority of the training data. Robust anomaly detection is particularly destined to design algorithms that are insensitive to polluted training samples and reduce their influence on the learning process.

In the following part, we present the most relevant and recent study that is designed for unsupervised time series anomaly detection: AnomalyTransformer [279].

5.2 Related Work: AnomalyTransformer, Time Series Anomaly Detection with Association Discrepancy

Before presenting our contribution, we review in detail a recent Transformer developed for time series anomaly detection: AnomalyTransformer. We mainly focus on this study because it introduces a novel and insightful observation about temporal anomalies: the Association Discrepancy (AssDis), which we will present later. The authors innovate the Transformer attention mechanism to improve the detection performance, and this model achieves competitive results on some classical benchmark datasets.

As we have outlined in Section 2.3, a series of recent studies [64, 272, 258] has shown the advantage of using Transformers in time series anomaly detection, over classical methods (e.g. LSTMs). Benefiting from the self-attention mechanism and parallel computations, Transformer-based anomaly detectors show a higher detection performance and a more efficient training process [272]. Some recent studies, e.g., TranAD [258] and MT-RVAE [269], propose to combine the Transformer architecture with common generative models, GANs [106] and VAEs [133], to further improve the model representation learning and to increase its robustness to training contamination.

Alternatively, Xu et al. [279] renovate the self-attention mechanism by introducing a new AnomalyAttention module, specifically tailored for unsupervised time series anomaly detection. Their method, called AnomalyTransformer, is based on the intuition that, due to the rarity of anomalies, it is harder to find an association spread over the whole sequence. Indeed, Xu et al. remark that the self-attention of anomalous points tends to be located generally in their adjacent data points. Consequently, AnomalyTransformer leverages this *adjacent concentration bias* to make anomalous points more distinguishable.

Subsequently, we first depict the novel AnomalyAttention proposed in AnomalyTransformer. We particularly shed the light on the adjacent concentration bias and its formal computation with the AssDis criterion. Secondly, we present the global overview of AnomalyTransformer. We mainly review the AnomalyTransformer min-max training strategy by analyzing the minimized objective function. Finally, we present how the authors leverage the AssDis for the inference of new unseen test points.

5.2.1 AnomalyAttention

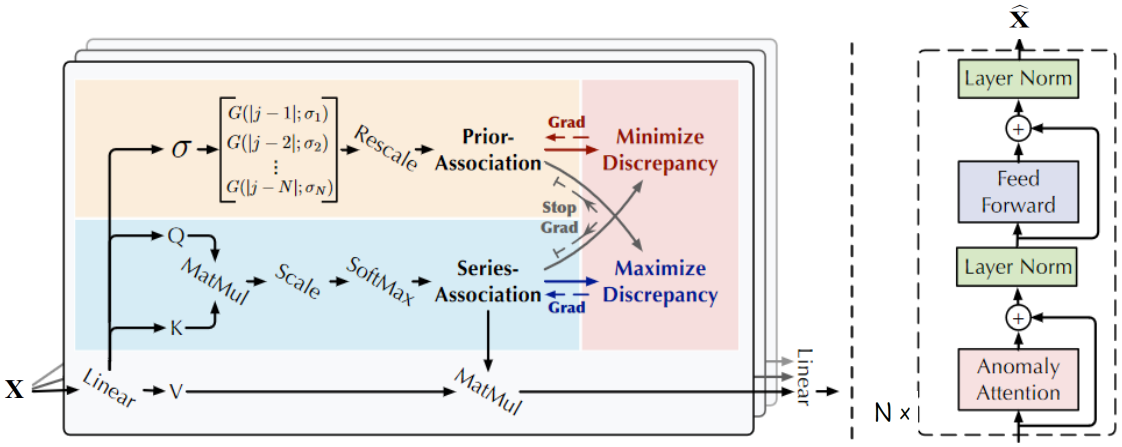


Figure 5.2: AnomalyTransformer architecture, adapted from [279].

Xu et al. [279] formalize the previously described *adjacent concentration bias* by defining the AssDis criterion. For each data point, the AssDis quantifies the disparity between the local attention relative to the adjacent points and the global attention with the whole series. To compute the AssDis, the authors introduced two novel concepts: the Series Association and the Prior Association. While the former quantifies the global association that maps each data point with the whole time series, the latter mostly focuses on the local attention covering the adjacent observations. Thus, the AssDis is formally determined as the divergence between both distributions.

The proposed Anomaly Attention module is a two-branch self-attention, as shown in Figure 5.2. AnomalyAttention two branches model the Prior Association and the Series Association, which we detail in the next paragraphs.

5.2.2 Series Association

The first branch of Anomaly Attention (cf. the blue layer of Figure 5.2) learns the temporal dependencies with respect to the whole sequence. This branch computes the *Series Association*, which finds the global point-wise dependencies linking each point with all other samples of the same input sequence.

Concretely, the Series Association is computed using the classical self-attention map (cf. Section 2.3.5.4 of Chapter 2). Let $\mathbf{Q} \in \mathbb{R}^{T \times d_{model}}$, $\mathbf{K} \in \mathbb{R}^{T \times d_{model}}$, and $\mathbf{V} \in \mathbb{R}^{T \times d_{model}}$ be the query, key, and value matrices of the AnomalyAttention, respectively. $T \in \mathbb{N}^*$ is the length of the sequence and $d_{model} \in \mathbb{N}^*$ is the dimension of the embedded data. Series Association is defined as the following:

$$S = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{model}}}\right) \tag{5.3}$$

Series Association uses the dot-product function to compute the similarity between a set of queries and a set of keys. Since we compute the self-attention map, the keys and the queries are the elements of the same sequence. Then this score is normalized using the Softmax(.) function.

5.2.3 Prior Association

The second branch (cf. the yellow layer of Figure 5.2) assesses the Prior Association, which quantifies the local self-attention distribution, relative to adjacent time points. Unlike Series Association, Prior Association focuses only on a small subset of the sequence: the surrounding data points. In this case, it is crucial to specify the window range of the local context defining the adjacent points. The authors propose to define the local context using a Gaussian Kernel, denoted as G in Figure 5.2, and parameterized by a learnable scale parameter σ. This learnable Gaussian kernel has the advantage to adapt the local context to input data characteristics.

For two data points $\mathbf{x_i}$ and $\mathbf{x_j}$, such that $0 \leq i \leq T$, $0 \leq j \leq T$, and $T \in \mathbb{R}^+$ is the sequence length, the Prior Association is defined as follows:

$$P_{ij} = \text{Rescale}\left(\frac{1}{\sqrt{2\pi}\sigma_i}\exp\left(-\frac{|j-i|^2}{2\sigma_i^2}\right)\right), \tag{5.4}$$

where $\sigma \in \mathbb{R}^{T\times 1}$ stands for the Gaussian learned scale, *Rescale* is used to normalize the rows of the matrix $\mathbf{P}$, i.e., to transform $\mathbf{P}$ rows to a discrete distribution. The training process of AnomalyTransformer will be described later in Section 5.2.6.

5.2.4 Association Discrepancy

After presenting both Prior and Series Associations, we define in this part the AssDis. The AssDis measures the difference between the adjacent concentration and the global attention. Formally, it is defined as the symmetric KL divergence between the Prior ($\mathbf{P}$) and the Series ($\mathbf{S}$) Associations. For the i^{th} data point of an input sequence, the AssDis, AssDis $\in \mathbb{R}^{T\times 1}$, is defined as:

$$AssDis_i(\mathbf{P},\mathbf{S}) = \text{KL}(\mathbf{P}_{i,:} \parallel \mathbf{S}_{i,:}) + \text{KL}(\mathbf{S}_{i,:} \parallel \mathbf{P}_{i,:}), \tag{5.5}$$

where $\text{KL}(\mathbf{P}_{i,:} \parallel \mathbf{S}_{i,:})$ is the KL divergence computed between two distributions: the i^{th} rows of $\mathbf{S}$ and $\mathbf{P}$.

As it is difficult to find a global mapping that links anomalous points with the whole sequence, both local and global self-attentions are mostly localized in the surrounding. The Prior and the Series associations of an anomaly are considerably similar, which results in a small AssDis. Thus, anomalies have smaller AssDis than nominal points.

5.2.5 AnomalyTransformer Global Architecture

Before presenting the training strategy, we depict in this Section the global overview of AnomalyTransformer architecture. As shown in Figure 5.2, AnomalyAttention is composed of a stack of N identical layers. Each layer comprises an AnomalyAttention module followed by FFN layer. The Prior and Series Associations are computed at the output of each layer and the global AssDis is the mean of layer-wise AssDis, defined in Equation 5.5.

$$AssDis_i(\mathbf{P},\mathbf{S}) = \frac{1}{N}\sum_{l=1}^{N} AssDis_i^l(\mathbf{P},\mathbf{S}) = \frac{1}{N}\sum_{l=1}^{N} \text{KL}(\mathbf{P^l}_{i,:} \parallel \mathbf{S^l}_{i,:}) + \text{KL}(\mathbf{S^l}_{i,:} \parallel \mathbf{P^l}_{i,:}) \tag{5.6}$$

This global AssDis combines the multiple AssDis extracted from the intermediate layers and aggregates this multi-level information into a single measure.

Finally, AnomalyTransformer learns to reconstruct the input data. Similar to the classical Transformer-based anomaly detectors, the output of this neural network $\widehat{\mathbf{X}}$, a.k.a., the reconstructed sequence, is defined as:

$$\widehat{\mathbf{X}} = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{model}}})\mathbf{V} = \mathbf{S}\mathbf{V} \tag{5.7}$$

5.2.6 Minimax Training Strategy

AnomalyTransformer has an encoder-decoder architecture. It is trained to reconstruct the input data. This learning phase consists in minimizing the reconstruction loss, defined using the Frobenius loss:

$$\text{Reconstruction Loss} = \left\|\widehat{\mathbf{X}} - \mathbf{X}\right\|_F^2, \tag{5.8}$$

where $\mathbf{X} \in \mathbb{R}^{T\times d}$ is the input sequence of T d-dimensional points, $\widehat{\mathbf{X}} \in \mathbb{R}^{T\times d}$ is the sequence reconstruction, and $\|.\|_F$ is the Frobenius norm.

Since the reconstruction is defined using the product of the Series Association and the value matrix, minimizing the reconstruction error guides $\mathbf{S}$ optimization and helps the model find the relevant global self-attention map. However, this classical objective function does not optimize the Prior Association.

To learn the optimal Gaussian scale parameter, the authors propose to constrain AnomalyTransformer training with an additional loss. Indeed, to make anomalies more distinguishable, the additional loss aims to maximize the Association Discrepancy of the training sample, which makes the task notoriously more difficult for anomalies. In fact, maximizing the AssDis encourages the model to learn a global point-wise mapping that is not restricted to adjacent points. However, due to the adjacent concentration bias, this constraint makes the anomaly reconstruction task harder. Clearly, the opposite is the case for inliers.

All in all, the loss function proposed in [279] is:

$$L(\mathbf{X},\mathbf{S},\mathbf{P},\lambda) = \left\|\widehat{\mathbf{X}} - \mathbf{X}\right\|_F^2 - \lambda \left\|AssDis(\mathbf{P},\mathbf{S})\right\|_1, \tag{5.9}$$

where $\lambda > 0$ is an hyperparameter used to trade off the reconstruction and the adjacent concentration constraint.

Nevertheless, the authors remarked that simultaneously minimizing the reconstruction error and maximizing the AssDis results in a meaningless Prior Association, with an infinitesimal Gaussian scale parameter σ. Indeed, with a very small σ, the contextual window of the adjacent points is very reduced so that the AssDis can be large for all training data, even anomalies. To avoid this collapse, the authors propose an alternative training strategy, where they disjointly optimize both associations in two separate steps: a minimize and a maximize phase.

In the minimize phase, the Series Association, denoted as $\mathbf{S}_{detach}$, is fixed, and the aim of this phase is to find the best Gaussian that approximates the Prior Association. The objective of this phase is to minimize the AssDis so that $\mathbf{P}$ approximates the fixed $\mathbf{S}_{detach}$. The loss of this phase is:

$$L_{min} = \left\|\widehat{\mathbf{X}} - \mathbf{X}\right\|_F^2 + \lambda \left\|AssDis(\mathbf{P},\mathbf{S}_{detach})\right\|_1 = L(\mathbf{X},\mathbf{S}_{detach},\mathbf{P},-\lambda). \tag{5.10}$$

Once the optimal σ is estimated, the Prior Association is now fixed, $\mathbf{S}_{detach}$, and the maximize phase optimizes the series association so that it focuses on the non-adjacent points. This phase, which maximizes the AssDis, optimizes the following function:

$$L_{max} = \left\|\widehat{\mathbf{X}} - \mathbf{X}\right\|_F^2 - \lambda \left\|AssDis(\mathbf{P}_{detach},\mathbf{S})\right\|_1 = L(\mathbf{X},\mathbf{S},\mathbf{P}_{detach},\lambda). \tag{5.11}$$

To conclude, the AnomalyTrabsformer minimax training strategy train the Transformer to disjointedly optimize the Series and the Prior Associations by alternating between these two min-max phases.

5.2.7 Test Point Anomaly Score

After the training, AnomalyTrasformer is used to assess the anomalousness of new samples. The author proposes a novel anomaly score that integrates the proposed AssDis. For a test data matrix $\mathbf{X} \in \mathbb{R}^{T\times d}$ and its reconstruction $\widehat{\mathbf{X}} \in \mathbb{R}^{T\times d}$, the anomaly score is computed as follows:

$$\text{AnomalyScore}(\mathbf{X}) = \text{Softmax}(-\text{AssDis}(\mathbf{P},\mathbf{S}))\left\|\mathbf{X}-\widehat{\mathbf{X}}\right\|_2^2. \tag{5.12}$$

The classical reconstruction error is amplified with a term inversely proportional to the AssDis. Since anomalies have smaller AssDis than inliers, their reconstruction error is amplified, which improves anomaly detection performance.

5.3 Contribution: RESIST, Robust Transformer for Unsupervised Time Series Anomaly Detection

5.3.1 Beyond Self-Sequence Dependencies

As mentioned in Section 5.2, Transformers have the potential to learn long-term relationships, thanks to the attention mechanism. In addition, AnomalyTransformer proposed by Xu et al. [279] shows that encouraging global attention spread over the entire sequence improves Transformer anomaly detection performance. While being more robust than basic Transformers, AnomalyTransformer attention is still restricted to the input sequence and lacks longer-term dependencies extracted from historical sequences. In fact, the time-series data are usually split into fixed-length consecutive segments using a sliding window (cf Figure 5.3). The reference of normality in AnomalyTransformer is bounded to a single segment and ignores all the previous windows. Even though anomalies are rare, the same anomaly may occur twice in the same window. In this situation, the adjacent concentration bias becomes invalid, as anomalous observation AssDis is no longer limited to its surroundings. Accordingly, we conjecture that outlier rejection must include an external reference of normality, mined from the history.

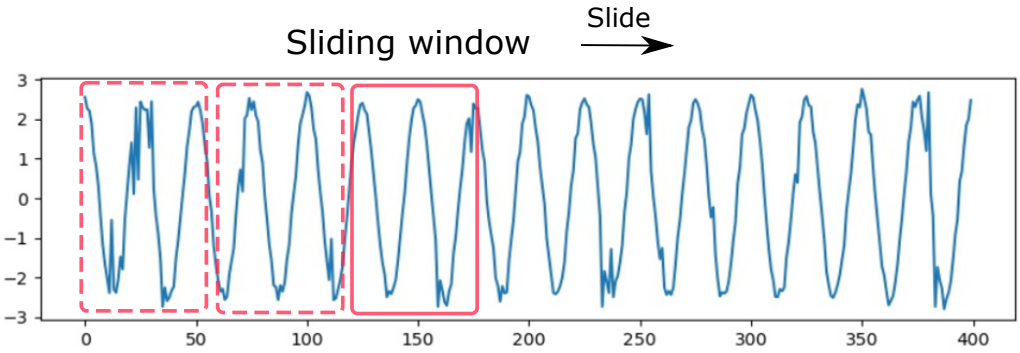


Figure 5.3: Non-overlapping sliding windows of a toy sequence.

Unlike AnomalyTransformer, we propose to extend the transformer attention to cover the historical data, in order to reject training unusual observations. We hypothesize that rejecting training contaminants requires building pairwise associations not only between data points of the same sequence but also with instances of previous segments. The main intuition is that nominal instances present a regular behavior shared across multiple segments. That is, reconstructing nominal sequences using either self-information extracted from the current input (i.e., self-reconstruction) or

using relevant information extracted from sequence adjacent neighbors (i.e., cross-reconstruction) would lead to similar results. In contrast, since anomalies are rare and different, building inter-sequence associations (or similarities) is more difficult and less informative. Therefore, anomaly self-reconstruction is much easier than cross-reconstruction.

Building on this insight, we propose RESIST. RESIST is trained to reconstruct input sequences using a hybrid representation that combines local intra-sequence information as well as global properties, shared between multiple segments. Accordingly, we introduce a Siamese training strategy that encourages the model to pay equal attention to the input sequence as well as to the previous ones. The sensitivity to training contaminants is significantly reduced since this reconstruction task becomes arduous for rare training outliers. Moreover, we train RESIST with a robust loss function to reduce the impact of large reconstruction errors caused by training outliers. Our experimental results show that this strategy significantly improves transformer robustness in AD (see Sections 5.4.1.5, 5.4.2.4).

In the following, we detail these contributions and the hypotheses that we will analyze in the experimental part. Firstly, we depict the global architecture overview of RESIST to present its main building blocks. Then, we present each component separately. We also focus on the different configurations that can influence RESIST performance. Finally, we state the presumed hypotheses, the training protocols followed to analyze these hypotheses, and the experimental results.

5.3.2 Architecture Overview

Our contribution presents an encoder-decoder architecture, comprised of four main components: a positional encoding and embedding layer, a siamese encoder, a fusion layer, and a decoder, as shown in Figure 5.4. Similar to Transformers [260], the original data is firstly encoded using the linear embedding and the positional encoding units. Both encoder and decoder are composed of stacked identical blocks, where each block contains a multi-head attention unit followed by a FFN layer.

RESIST takes as input K non-overlapping sequences $\mathbf{X}_t^w = (\mathbf{x}_{t-K+1}^w, ..., \mathbf{x}_t^w)$: an input sequence $\mathbf{x}_t^w$ and its $K-1$ previous sequences. In Figure 5.4, we illustrate our method for $K = 2$. Firstly, the linear embedding and the positional encoding units encode the input sequences $(\mathbf{x}_{t-K+1}^w, ..., \mathbf{x}_t^w)$ and output the K embedded sequences $(\mathbf{e}_{t-K+1}^w, ..., \mathbf{e}_t^w)$. Secondly, the encoder extracts from each embedded sequence $\mathbf{e}_t^w$ a low-dimensional latent encoding $\mathbf{z}_t^w$. Then, the fusion layer aggregates these encodings into a single representation. The decoder maps the fusion encoding to the input space in order to reconstruct the original sequence $\mathbf{x}_t$. Finally, RESIST minimizes the Geman-McClure robust function between the reconstructed sequence $\widehat{\mathbf{x}_t^w}$ and the original one $\mathbf{x}_t^w$.

After presenting the global architecture of our method, we will thoroughly review each component in the following Sections.

5.3.3 Siamese Encoder

RESIST encoder, illustrated in Figure 5.4, learns to project K consecutive sequences into K low-dimensional embeddings. The encoder receives a sequence $\mathbf{x}_t^w$ and its associated history, which contains the $K-1$ sequences preceding $\mathbf{x}_t^w$. It models the point-wise correlations between $\mathbf{x}_t^w$ and the history. Then, it learns to project these data into a common reduced space of dimension $d_{enc} \in \mathbb{N}^*$, where common data points share similar representations. This task is notoriously hard for anomalies, since they present non-representative uncommon patterns. For this reason, we propose an encoder with a Siamese architecture, with K identical sub-networks that share the same parameters. Input sequences are simultaneously processed using these networks. The sequences that share common proprieties have close encodings.

Unlike classical Siamese Neural Networks, our encoder is not trained to learn a similarity metric between input sequences. Its objective is to reduce the data dimensionality to only keep the most important information. Each siamese encoder sub-network is composed of a stack of $N = 2$

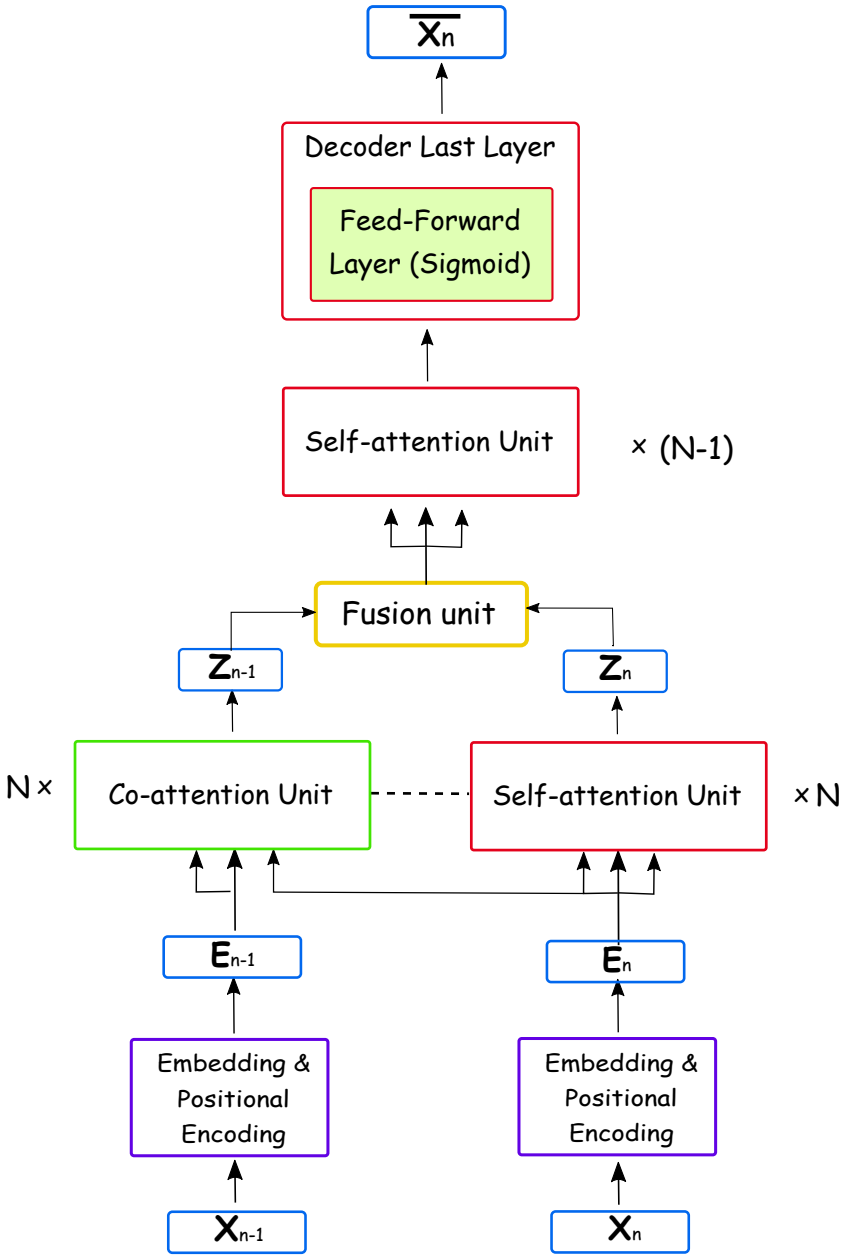


Figure 5.4: RESIST architecture.

identical blocks. Each block comprises two sub-modules: a multi-head attention unit followed by a FFN layer (cf. Figures 5.5 and 5.6). While the attention mines the temporal correlations in the data, the FFN layers are used for dimensionality reduction.

The Siamese encoder is a hybrid composition of both Self-Attention (SA) and Co-Attention (CA) units. While the SA units are used to extract the contextual properties of the current sequence, the CA unit is destined to extract inter-segment properties and only keep common relationships.

5.3.3.1 Self-attention and Co-attention Module

Attention modules are intended to mine pairwise interactions between data points. We propose to leverage the Self-Attention (SA) and Co-Attention (CA) layers, initially introduced in multimodal Visual Question Answering (VQA) [287], to our task of unsupervised AD.

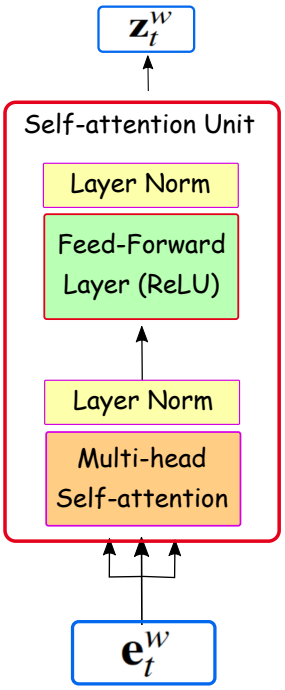


Figure 5.5: Self-attention unit

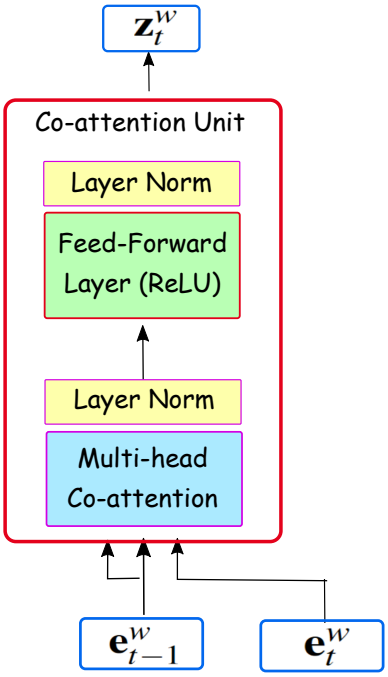


Figure 5.6: Co-attention unit

VQA is a visual reasoning task where we train a model to answer a question concerning an image. Identifying joint visual-linguistic representations is crucial in VQA. In [287], Yu et al. propose a Transformer-based VQA model where they introduce a co-attention layer, a.k.a., guided attention layer (see Figure 5.6). This layer is mainly designed to model multimodal interactions between a sentence and an image. The architecture of co-attention is the same as the self-attention layer. The main difference is that co-attention receives two different input sequences, a sentence and an image. It extracts the Query from the image and the pair (Key, Value) from the sentence. Recent studies [249] show the potential of co-attention to learn contextual representations and to improve model generalization performance.

We propose to extend CA to our task of unsupervised anomaly detection. CA can be seen as a module that filters similar data points between a sequence and the history. Then, it weights the current sequence observations with the relative normalized similarities. The aim is to guide the reconstruction with inter-sequence common information and to filter out sequence-specific rare patterns. This encourages the model to ignore unusual patterns that are only relevant for a single sequence. Different compositions of CA and SA may result in different configurations of RESIST. In Section 5.4.1.3, we will present these configurations and we will experimentally evaluate their impact on the AD performance.

5.3.3.2 FFN Layer

Each attention layer is followed by a connected FFN, composed of a single linear layer. These linear layers are used to reduce data dimensionality through a linear mapping from the input space to a space of fewer dimensions. Furthermore, we apply the activation function to these layer outputs. For a d_{in}-dimensional vector $\mathbf{x} \in \mathbb{R}^{d}_{in}$, the output of the FFN layer is:

$$FFN(\mathbf{x}) = \max(0, \mathbf{x}\mathbf{W} + \mathbf{b}) \tag{5.13}$$

where $\mathbf{W} \in \mathbb{R}^{d_{in} \times d_{out}}$ is the weight matrix, $\mathbf{b} \in \mathbb{R}^{d_{out}}$ is the bias vector, and d_{in} and d_{out} are the dimension of the input and the output, respectively. We have $d_{in} > d_{out}$ since the FFN main task is to map the input into a lower-dimensional latent space.

5.3.4 Fusion Strategy

Commonly, AE-based neural networks learn to reconstruct the input sequence using a single latent representation. Unlike these classical methods, we propose to leverage multiple data views for robust reconstruction. The fusion layer is destined to combine the multiple encodings extracted by RESIST encoder into a single vector representation, so that the decoder reconstructs the input sequence using this single compressed fusion.

Formally, fusion is defined as "a multi-level process dealing with the association, correlation, combination of data and information from single and multiple sources to achieve refined position, identify estimates and complete and timely assessments of situations, threats and their significance" [276]. Various data fusion strategies have been developed in different application fields of machine learning. Data and decision fusion strategies are extensively explored in intrusion detection [149], wireless sensor monitoring [55], image classification [209]. In this work, we compare two fusion strategies: concatenation and addition-based fusions. We will detail these strategies in the following paragraphs.

Regardless of the fusion strategy, the fusion module comprises a fusion layer (i.e., the concatenation or the average operations), followed by a single FFN layer. We use this FFN so that the output of the fusion module, i.e., the input of RESIST decoder, would have the same dimension, regardless of the applied fusion strategy.

Concatenation Strategy

The first strategy simply consists in concatenating the K outputs of the encoder. Let $(\mathbf{Z}^{w}_{n-K+1}, ..., \mathbf{Z}^{w}_{n})$ be K encodings, each of which contains w data points of dimension d_{enc}, i.e., $\mathbf{Z}^{w}_{n} \in \mathbb{R}^{w \times d_{enc}}$. d_{enc} is the dimension of RESIST latent space. The output $\mathbf{F}^{w}_{n}$ of the fusion module in this case is:

$$\mathbf{F}^{w}_{n}(\mathbf{Z}^{w}_{n-K+1}, ..., \mathbf{Z}^{w}_{n}) = ReLU([\mathbf{Z}^{w}_{n-K+1}, ..., \mathbf{Z}^{w}_{n}]\mathbf{W}_{f} + \mathbf{b}_{f}), \tag{5.14}$$

where $[.]$ is the concatenation function, $\mathbf{W}_{f} \in \mathbb{R}^{Kd_{enc} \times d_{f}}$ refers to the linear layer weights and $\mathbf{b}_{f} \in \mathbb{R}^{Kd_{f}}$ to the bias vector. d_{f} is the dimension of the fusion module outputs. In all experiments, we use $d_{f} = d_{enc} = 16$. The main drawback of this strategy is its sensitivity to the history length K. As such, concatenation fusion becomes computationally expensive with large K, since more and more learnable parameters are involved.

Mixup Strategy

The second strategy is inspired from the well-known manifold *mixup* method [290]. The original *mixup* method was initially proposed for data augmentation in supervised learning. For two training inputs x_i and x_j, having two labels y_i and y_j, respectively, *mixup* generates a new training instance, $\hat{x}$, using a linear interpolation:

$$\hat{x} = \delta x_i + (1-\delta)x_j. \tag{5.15}$$

$$\hat{y} = \delta y_i + (1-\delta)y_j. \tag{5.16}$$

$\hat{y}$ is the corresponding label of $\hat{x}$. The interpolation term $\delta \in [0,1]$ is an hyperparameter. In other words, *mixup* trains supervised classifiers to adapt a linear behavior in the boundaries between training classes. Recent publications show that *mixup* reduces classifier regularization error and makes classifiers more robust to corrupted labels [147]. We refer the reader to [132] for a comprehensive review of *mixup*-based strategies.

We extend the *mixup* method to robust unsupervised anomaly detection. Similar to the original *mixup* strategy, *mixup* fusion merges K instances into a single vector through linear interpolation. The fusion of K encodings $(\mathbf{Z}^w_{n-K+1}, ..., \mathbf{Z}^w_n)$ is defined as follow:

$$\mathbf{F}^w_n(\mathbf{Z}^w_{n-K+1}, ..., \mathbf{Z}^w_n) = ReLU(\widehat{\mathbf{Z}^w_n}\mathbf{W}_f + \mathbf{b}_f), \tag{5.17}$$

where

$$\widehat{\mathbf{Z}^w_n} = \frac{1}{K}\sum_{i=n-K+1}^{n} \mathbf{Z}^w_i \tag{5.18}$$

We propose a uniform contribution of all encodings. For example, for $K = 2$, $\widehat{\mathbf{Z}^w_n}$ is defined as,

$$\widehat{\mathbf{Z}^w_n} = \frac{1}{2}\mathbf{Z}^w_{n-1} + \frac{1}{2}\mathbf{Z}^w_n. \tag{5.19}$$

When the input sequence $\mathbf{X}^w_n$ share common properties with its history, represented by $\mathbf{X}^w_{n-1}$, we expect that the siamese encoder extracts close latent representations $\mathbf{Z}^w_n$ and $\mathbf{Z}^w_{n-1}$. In his case, the fused representation would be similar to the encoding of a classical Transformer, i.e., $\widehat{\mathbf{Z}^w_n} \approx \mathbf{Z}^w_{n-1}$. In contrast, when the current sequence comprises an uncommon pattern, the encoder self-attention and co-attention modules potentially extract different encodings. Therefore, the linear interpolation may generate an inconsistent and less informative encoding, and the reconstruction task becomes more difficult. In the experimental section 5.4.1.3, we compare RESIST performance with both strategies.

5.3.5 RESIST Decoder

Finally, the RESIST decoder learns to reconstruct the last sequence of the input using the compact representation that is the output of the fusion module. It is composed of a stack of $N = 2$ identical blocks. Each block comprises two sub-modules: a multi-head self-attention unit followed by a FFN layer. While activation function is used in the first block, the last block is followed by a Sigmoid function to ensure that the output has the same range as the input $[0,1]$.

5.3.6 Robust Training Loss

To hedge against training contaminants, we train RESIST using a robust loss function. Indeed, the commonly used Mean Squared Error (MSE) is sensitive to outliers, since squaring large deviations results in the dominance of anomalies during the training. In contrast, a robust loss can resist noise and anomalies by reducing the influence of their large reconstruction errors. There have been numerous studies to explore robust learning in the presence of outliers. The robust function list includes Charbonnier loss, Cauchy loss, Geman-McClure loss, and Welsch loss. Recently,

Barron [30] generalizes these common losses in a single parametric function, $\rho(x, \alpha, c)$, which is parameterized by two parameters: the scale c and the robustness parameter α.

$$\rho(x,\alpha,c) = \begin{cases} \frac{1}{2}(\frac{x}{c})^2 & if\ \alpha = 2 \\ \log\left(\frac{1}{2}(\frac{x}{c})^2 + 1\right) & if\ \alpha = 0 \\ 1 - \exp\left(-\frac{1}{2}(\frac{x}{c})^2\right) & if\ \alpha = -\infty \\ \frac{|\alpha-2|}{\alpha}\left(\left(\frac{(\frac{x}{c})^2}{|\alpha-2|} + 1\right)^{\frac{\alpha}{2}} - 1\right) & otherwise \end{cases} \tag{5.20}$$

Particular values of α define common robust losses: L2 loss ($\alpha = 2$), Charbonnier loss ($\alpha = 1$), Cauchy loss ($\alpha = 0$), Geman-McClure loss ($\alpha = -2$), and Welsch loss ($\alpha = -\infty$). These cases are visualized in Figure 5.7, extracted from [30]. We refer the reader to [30] for a detailed description of these losses.

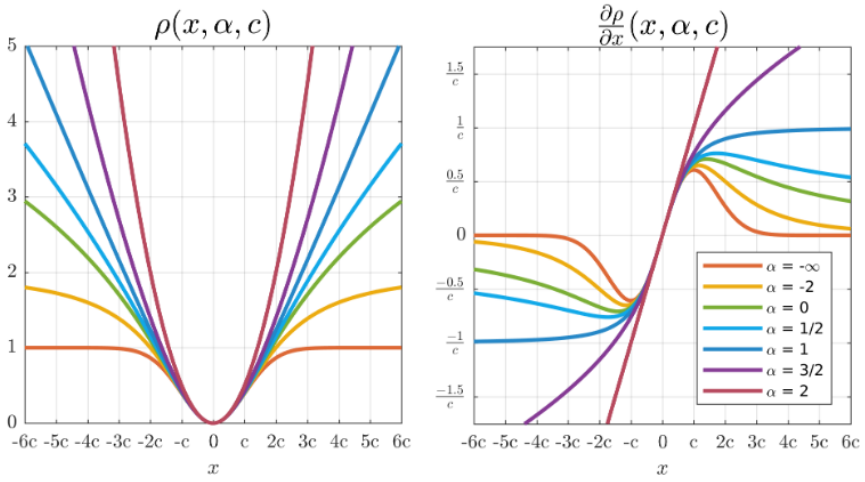


Figure 5.7: The general robust loss function proposed in [30].

In particular, we propose to train RESIST by minimizing the Geman-McClure robust function, i.e., $\rho(x, \alpha = -2, c)$. The conventional Geman-McClure function is defined as:

$$L(x) = \rho(x, \alpha = -2, c) = 2\frac{(\frac{x}{c})^2}{4 + (\frac{x}{c})^2} \tag{5.21}$$

where c is a scale parameter that modulates the loss robustness range. In all our experiments, we set $x = \lambda \text{IQR}$, where IQR is the interquartile range and $\lambda = 0.1$. We conduct a sensitivity analysis regarding this hyperparameter in Section 5.4.1.5.

5.3.7 Hypotheses

In the previous sections, we presented the global overview of RESIST, as well as a detailed description of each building block. To conclude the presentation of our method, we synthesize our contributions into the following hypotheses:

- **Hypothesis 5.1 (H5.1)** *We conjecture that guiding the Transformer reconstruction with both intra-sequence properties and inter-sequence pairwise interactions with the history results in a more robust anomaly detector. That is, fusionning local information extracted by the self-attention module and global information guided by the co-attention unit improves model performance in the presence of training anomalies;*

- **Hypothesis 5.2 (H5.2)** *We hypothesize that training RESIST with a robust loss function, and particularly the Geman-McClure loss, significantly reduces the impact of training noise and anomalies;*
- **Hypothesis 5.3 (H5.3)** *This hypothesis is related to the fusion of local and global information. We assume that the additive fusion with mixup method is more efficient and more robust than the classical concatenation strategy;*

In the next Sections, we investigate the validity of these hypotheses with a set of experiments performed on two public real-world datasets. In addition, we extensively compare our contribution against common unsupervised anomaly detectors.

5.4 Experiments and Results

In this section, we aim to explore the validity of the assumed hypotheses on two public datasets: the CICIDS17 evaluation dataset and the TON-IoT dataset. Firstly, we perform a set of experiments on the first benchmark dataset, CICIDS17 , to assess the performance of our approach in detecting anomalous flow-based network data. Then, the second set of experiments is particularly related to IoT traffic anomaly detection. These experiments explore the relevance and efficiency of our proposal for IoT data, on the recent IoT dataset, TON-IoT. In the following, we provide an overview of each dataset peculiarities. Then, we develop the main steps of the training and testing protocols. Finally, we present and analyze the empirical results of each experiment set.

5.4.1 CICIDS2017 dataset

5.4.1.1 Dataset Description

The first dataset used in our experiments is the CICIDS17 dataset [237]. CICIDS17 is a recent public dataset developed by the Canadian Institute of Cybersecurity (CIC) for IDS evaluation. Overall, this dataset comprises about 3 million labeled network flows collected over 5 days, starting from July 3, 2017, and ending on Friday, July 7, 2017. 83% of this traffic is benign and the remaining 17% is anomalous.

To collect the traffic, Sharafaldin et al. developed a testbed containing two networks: an Attack-Network and a Victim-Network. The Victim-Network comprises three servers, one firewall, two switches, and ten interconnected PCs. One switch was configured to mirror all the traffic passing through the network. The Attack-Network is a separate network that runs network attacks on the Victim-Network.

CICIDS17 provides full packet capture of the collected data in *pcap* files. In addition, the raw data are processed using CICFlowMeter [146], a flow-based feature extractor, to extract metadata from the packet traces. Each flow record is represented by 85 features: a flow ID, 83 flow metadata features, and a class label. A detailed description of the 83 flow-based features is presented in [142]. CICIDS17 comprises 15 classes: a nominal class and 14 attack types, including Denial of Service (DoS), Distributed DoS (DDoS), Web attacks, and Infiltration attacks. Table 5.1 summarizes the volume of data in CICIDS2017 per type of attack.

This dataset was extensively used in many recent publications [142], since it covers various recent attacks and comprises both punctual and collective anomalies. Furthermore, it has more complex types of attacks than NSL-KDD and MedBIoT datasets (cf. Sections 3.4.1 and 3.4.2).

Table 5.1: The label classes present in the CICIDS17 dataset

Attack	Number of flows	Percentage (%)
Benign	2358036	83.33
DoS Hulk	231073	8.16
Port Scan	158930	5.62
DDoS	41835	1.48
DoS GoldenEye	10293	0.36
FTPPatator	7938	0.28
SSHPatator	5897	0.21
DoS Slow Loris	5796	0.21
DoS Slow HTTP Test	5499	0.19
Botnet	1966	0.07
Web Attack: Brute Force	1507	0.05
Web Attack: XSS	652	0.02
Infiltration	36	0.001
Web Attack: SQL Injection	21	0.001
Heartbleed	11	0.0001

5.4.1.2 Data Preprocessing

We follow the same preprocessing steps proposed in [142, 144]. Since the original dataset is voluminous, and similar to [144], we focus on the data subset that is collected during one day: Thursday, July 6, 2017. This subset contains 170231 network flow and represents around 6% of the whole dataset. 98.7% of this traffic is benign and the remaining 1.3% is anomalous.

Firstly, we encode the categorical feature that describes the network protocol, i.e., 'TCP' or 'UDP' using the *Count encoder* [75]. The count encoder encodes the categorical features with their frequency of occurrence. As such, rare protocols have lower weight in flow vectors compared to frequent ones. Secondly, we rescale numerical features to be in the range $[0, 1]$, using the min-max normalization method. Finally, we randomly split the benign data into 40% for the training and 60% for testing. Similar to [224], 10% of testing data is selected to tune the hyperparameters of competing methods.

5.4.1.3 Training and Testing Protocols

Here, we describe the training and testing protocols used in our first set of experiments, to validate the conjectured hypotheses H5.1-H5.3. Firstly, we investigate the influence of RESIST different configurations on the global performance. We aim to analyze the importance of RESIST individual components: self-attention and co-attention modular composition, the robust loss function, and the fusion strategy. Afterward, we compare our proposal with relevant unsupervised methods developed for time series anomaly detection.

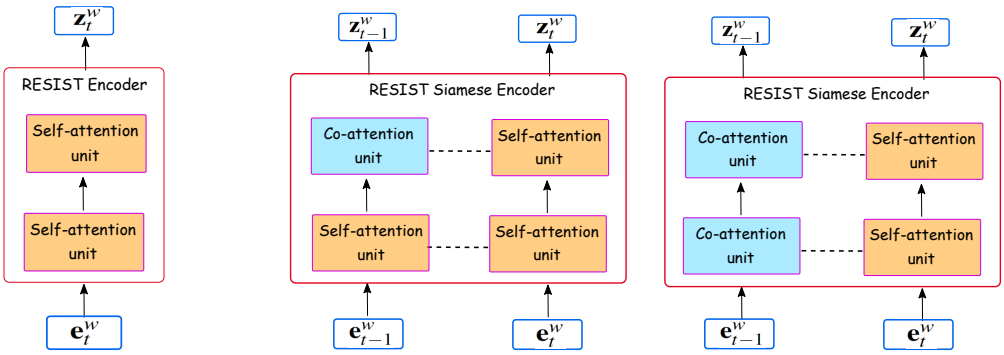
Protocol 1: Modular Composition of Co-attention and Self-attention Modules

Our method is composed of two attention-based components: the self-attention and the co-attention modules. Different combinations of these modules result in different variants. In this section, we analyze the first hypothesis (H5.1) by studying RESIST performance with three modular compositions of the attention-based units.

For ease of illustration, we only visualize in Figure 5.8, the RESIST encoder part of the three variants. The first variant, RESIST-SS that is illustrated in Figure 5.8a), is the baseline. This first configuration does not consider the history for data reconstruction. In this case, only the input sequence flows through self-attention units to gradually extract the intra-properties of each sequence Then, RESIST-SS decoder is trained to reconstruct the sequence-based only on this self-encoded representation.

The second configuration, RESIST-SC, represented in Figure 5.8b), considers inter-sequence similarities between the input and the history. Indeed, both sequences are processed using a first self-attention unit to model intra-sequence relationships. Then, the encoded representation of the current sequence is processed using a self-attention unit, while the historical representations are fed into a co-attention unit to introduce pairwise similarities between consecutive sequences. Afterwards, all encodings are aggregated and decoded.

Finally, the third variant is RESIST-CC (see Figure 5.8c). Here, the input sequence is encoded through cascaded self-attention units and adjacent sequences are encoded using co-attention units. The main difference between RESIST-CS and RESIST-CC is that the former encodes the history with a hybrid encoder that alternates CA and SA, while in the latter, only CA units are used to encode the previous segments.



(a) RESIST-SS *(b) RESIST-SC* *(c) RESIST-CC*

Figure 5.8: Three configurations of RESIST based on the modular composition of co-attention and self-attention modules.

Protocol 2: Fusion Strategy

In this section, we investigate the second hypothesis (H5.2), regarding the fusion strategy. In H5.2, we assume that the additive fusion with the *mixup* method is more efficient and more robust than the classical concatenation strategy.

Let K be the history length, i.e., the number of the consecutive training sequences fed to RESIST Siamese encoder, and d_{enc} be the dimension of each encoded sequence in RESIST latent space. The concatenation strategy merges these encodings into a single representation by joining all the vectors together. This results in a higher-dimensional vector of dimension $K \times d_{enc}$. Then, this large vector is projected back into a d_{enc}-dimensional latent space using a non-linear projection. This fusion projection selects the relevant information to be forwarded to the decode module. As such, this projection may discard all the information extracted from the history and keep only self-sequence

encoding. Formally, as presented in Section 5.3.4, the output of the concatenation module is :

$$\mathbf{F}_n^w(\mathbf{Z}_{n-i+1}^w,...,\mathbf{Z}_n^w) = ReLU(\sum_{i=1}^{K}\mathbf{Z}_{n-i+1}^w\mathbf{W}_i + \mathbf{b}_i), \tag{5.22}$$

Since each encoding is projected separately from others, with different parameters $\mathbf{W}_i$ and $\mathbf{b}_i$, the projection may discard historical information by setting all the corresponding parameters to 0. The second drawback of this strategy is that it becomes inefficient with more and more encoded sequences. The fusion layer parameters linearly increase with the history length. That is, increasing K results in a longer training phase.

In contrast, *mixup* fusion performs an element-wise summation of all K encoding. Consequently, the input and the output of the fusion layer have the same dimension. This summation constraints the fusion to consider all the latent representations equally, since the same parameters, $\mathbf{W}$, and $\mathbf{b}$, are involved in all projections:

$$\mathbf{F}_n^w(\mathbf{Z}_{n-i+1}^w,...,\mathbf{Z}_n^w) = ReLU(\frac{1}{K}\sum_{i=1}^{K}\mathbf{Z}_{n-i+1}^w\mathbf{W} + \mathbf{b}), \tag{5.23}$$

Finally, *mixup* fusion does not depend on history length. Increasing K would not impact the training efficiency of the fusion module.

Protocol 3: Robust Loss Function

Traditionally, encoder-decoder architectures are trained by minimizing the euclidean distance, a.k.a., the L2 loss, between the reconstruction and the input. However, this squared error is sensitive to training contaminants. In this section, we explore the importance of the robust loss function to reduce model sensitivity with respect to anomalies. As previously mentioned in Section 5.3.6, various robust losses are developed in the literature, such as Charbonnier loss, Cauchy loss, Geman-McClure loss, and Welsch loss.

In particular, we compare three different training losses. The first function is the classical L2 loss. Here, we train this first variant of RESIST with the common L2 loss to study its sensitivity to training outliers, in the absence of a robust training loss. Then, we compare three common robust functions: Charbonnier loss, Cauchy loss, and Geman-McClure loss. These losses are generally interchangeable and the performance may depend on the selected robust loss [30]. In the literature, researchers often experiment with different robust losses and select the best performing function. Here, we follow the same protocol and we investigate the three robust losses: Charbonnier loss, Cauchy loss, and Geman-McClure loss.

Protocol 4: Comparison with Competing Methods

Finally, after investigating the contribution of each component of RESIST to the global performance, we globally compare our contribution against common unsupervised time series anomaly detectors. In this experiment, we select the best performing configuration of RESIST: a siamese encoder that comprises a hybrid composition of self-attention and co-attention units, a *mixup* fusion strategy, and trained with the Geman-McClure loss.

The baseline models selected in our experiments belong to the different categories of unsupervised time series anomaly detection presented in Section 2.3 of Chapter 2. These baselines include one-class classifiers: IF [**liu2008isolations**], OSVM [251]; density-based methods: LOF [48]; reconstruction-based algorithms: OmniAnomaly [245], LSTM-AE [164], MSCRED [291], USAD [21], and Transformer [260]. In addition, we assess the performance of robust Transformers including TranAD [258] and AnomalyTransformer [279].

Table 5.2: The hyperparameters used in the experiments of this chapter.

Hyperparamter	Value
Architecture	a 5-layer MLP with 78-32-16-32-78 units
Learning rate	0.001
Maximum number of epochs	100
Batch size	64
The dimension of the embedding	128
The robust loss scale c	$c = 0.1$IQR
The history length K	$K = 2$
The window size w	$w = 100$
The number of heads h of the multi-head attention	$h = 2$
Laptop	12-core Intel i7-9850H CPU clocked at 2.6GHz, with 16GB of RAM memory and with NVIDIA Quadro P2000 GPU

5.4.1.4 Training Parameter Settings and Evaluation Criteria

The optimal hyperparameters selected in our experiments are reported in Table 5.2.

We follow the well-established protocol used by many recent papers [238]. We transform the input time series into consecutive sub-sequences using non-overlapped sliding windows of length $w = 100$. After preliminary tests, we use the same architecture for all autoencoder-based models. The autoencoders are a 5-layer MLP with 78-32-16-32-78 units. All latent layers are followed by activation function. The last layer is followed by a sigmoid function. We use the Adam optimizer to train all the neural networks, with an initial learning rate of 0.001, and a step-scheduler with a step of 0.5. All models are trained for 100 epochs, with a batch size of 64 in all experiments, and random parameter initialization. To limit the impact of random parameter initialization, we repeat each experiment five times and average the results over these five runs. Regarding Transformer-based anomaly detectors, we set the dimension of the embedding to 128 and we use 2-head attention units. In all our experiment, RESIST hyperparameter c is set as $c = 0.1$IQR (cf. Equation 5.21). Similar to the validation protocol adapted by Ruff et al. [224], the competing methods hyperparameters are tuned on the predefined validation subset. To minimize hyperparameter selection problems, we select the optimal hyperparameters that maximize their validation AUROC. This deliberately grants competing methods an advantage over RESIST. Lastly, all the experiments were run on a laptop equipped with a 12-core Intel i7-9850H CPU clocked at 2.6GHz and with NVIDIA Quadro P2000 GPU.

5.4.1.5 Results

In the following, we present and interpret the experimental results of our first set of experiments. We first validate each component of RESIST and assess their contribution to the performance, by analyzing the hypotheses (H5.1, H5.2, and H5.3). Then, we compare RESIST performance against common unsupervised time series anomaly detectors.

Protocol 1: Modular Composition of Co-attention and Self-attention Modules

For a fair comparison, the three variants have the same architecture and configuration. The three variants use the *mixup* fusion strategy and are trained with the same robust loss: Geman-McClure loss. The only difference between the three variants is the modular composition of co-attention and self-attention units. The experimental results of these 3 variants are shown in Figure 5.9. These first results highlight that the structure of RESIST encoder has a significant impact on the global performance, since varying the encoder composition of self and co-attention units is clearly reflected in the results.

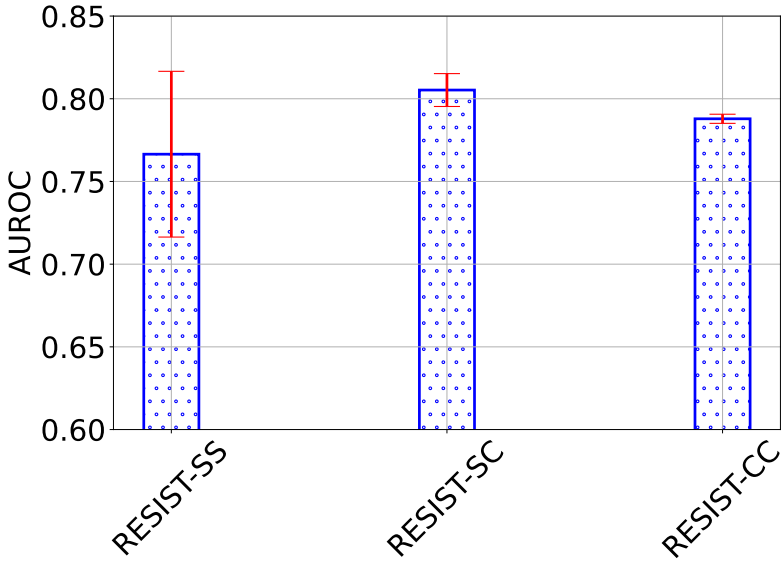


Figure 5.9: Comparison between RESIST three variants: RESIST-SS, RESIST-SC, and RESIST-CC, on the CICIDS17 dataset.

Firstly, RESIST-SS, whose encoder is purely composed of a cascade of self-attention units, performs poorly compared to the other variants. Indeed, RESIST-SS is similar to a Transformer trained to reconstruct the input, using the robust Geman-McClure loss, and without considering the historical data. This variant shows the lowest AUROCs in this first set of experiments, with a mean equal to 76.6%, and with a large standard variation of 5%. The other two variants, which integrate intra and inter-sequence properties with co-attention units, globally show better results with reduced standard variations. This confirms our first hypothesis (H5.1), in the sense that guiding the Transformer reconstruction with both intra-sequence properties and inter-sequence pairwise interactions with the history results in a more robust anomaly detector.

Furthermore, we note that the hybrid RESIST-SC reports higher AUROC, $80.5\% \pm 0.9$, compared to RESIST-SS, $78.8\% \pm 0.3$. This advantage is statistically significant, according to Welch's test with p-value $= 0.05$. This observation reveals that encoding the history with both self-attention and co-attention units is better than using only co-attention units. In RESIST-SC, the self-attention unit firstly extracts the intra-dependencies of the history. Then this first representation, which considers the history local context, is combined with the intermediate representation of the input. In contrast, RESIST-CC neglects history intra-sequence context and focuses only on inter-sequence properties. This result is consistent with other works in VQA [287]. In the following, we will use RESIST-SC encoder architecture as the basis for all next RESIST variants.

Protocol 2: Robust Loss Function

Similar to the protocol followed previously, all the variants share the same configuration, except for the training loss function. The three variants have a hybrid siamese encoder, similar to RESIST-SC encoder. The results are reported in Figure 5.10.

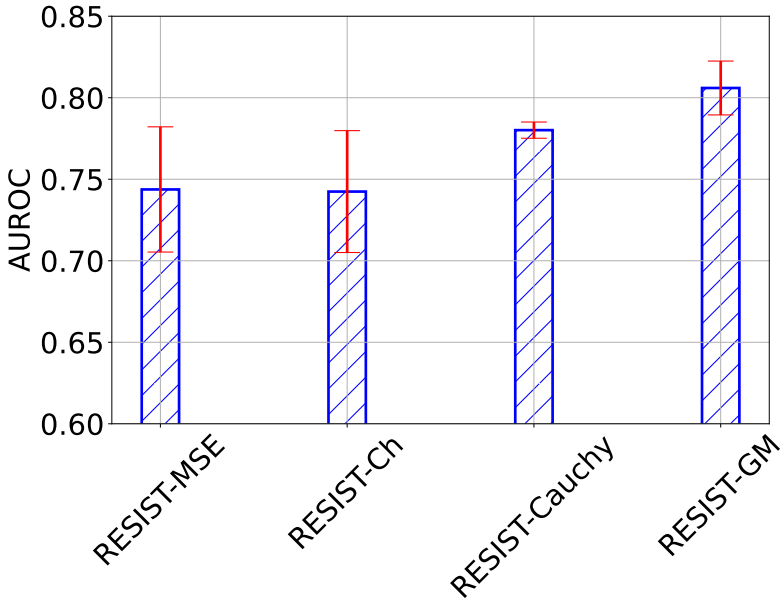


Figure 5.10: The experimental results for RESIST trained with different loss functions: L2 loss for RESIST-MSE, Charbonnier loss for RESIST-Ch, Cauchy loss for RESIST-Cauchy, and Geman-McClure loss for RESIST-GM.

From this figure, we can see that the training loss function has a significant influence on the performance. We note that the results steadily improve when decreasing the robustness parameter α of the loss function $\rho(x, \alpha, c)$, defined in Section 5.3.6. Firstly, RESIST-MSE, trained with the common Euclidean distance, i.e., $\alpha = 2$, shows the worst performance, with an AUROC around 74%. This result is in line with previous studies, which state that the mean-squared error is considerably influenced by outliers. Secondly, the Charbonnier loss, a.k.a, the pseudo-Huber loss, with $\alpha = 1$, does not improve the performance (cf. Figure 5.10). As shown in Figure 5.7 (left), even though the gradients of large error are reduced compared to the L2 loss, these gradients saturate to a non-zero value. That is, even though their contribution is slightly reduced, training contaminants still contribute to parameter optimization during the training. Nevertheless, when $\alpha \leq 0$, the gradient magnitude decreases and converges to 0, when the error is higher than the scale parameter c. As such, large errors are completely ignored and do not impact the training. The speed of converging to 0 clearly depends on the parameter α. The lower α, the higher the decreasing speed of large error gradients. Our results confirm this interpretation, in the sense that RESIST-GM, trained with Geman-McClure loss ($\alpha = -2$), exceeds RESIST-Cauchy, trained with Cauchy loss ($\alpha = 0$), by 2.6% on average. We can conclude that the second hypothesis (H5.2) is validated. Training RESIST with the Geman-McClure loss significantly reduces the impact of anomalies.

Protocol 3: Fusion Strategy

In this section, we explore the third hypothesis (H5.3), which is related to the fusion strategy. Here, we only vary RESIST fusion strategy: the concatenation or the *mixup* methods that are described in Section 5.3.6. In fact, the training of RESIST involves the input sequence and the previous historical segments. In all the previous experiments, we fixed $K = 2$. However, as

mentioned in Section 5.3.4, we assume that the fusion strategy may impact the performance, especially when we consider more and more sequences in the history, i.e., when K increases.

We propose to verify this hypothesis by running two sets of experiments. In the first experiment, we set $K = 2$ and we train two RESIST variants: a variant that uses the concatenation fusion, denoted as RESIST-Concat, and a second variant with the *mixup* strategy, denoted as RESIST-mixup. In the second set, we run the same tests with a higher $K = 5$. These two experiments aim to analyze the influence of the fusion strategy on the performance when the history length K increases. We report RESIST AUROC in Figure 5.11 for both $K = 2$ and $K = 5$.

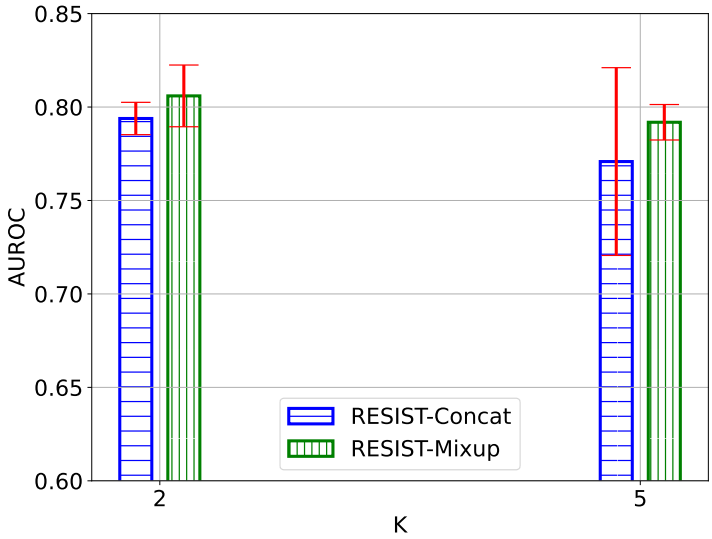


Figure 5.11: RESIST performances for different history length K and with different fusion strategies: RESIST-Concat denotes RESIST with the concatenation fusion, while RESIST-Mixup denotes RESIST with the mixup fusion.

We observe that RESIST-Mixup outperforms RESIST-Concat, for both $K = 2$ and $K = 5$. This advantage is particularly more noticeable with larger K, i.e., $K = 5$. The difference in mean AUROC increases from 1.2% to 2.1%. We also note an unstable performance of RESIST-Concat for $K = 5$, with a large standard variation of 5.0%. In contrast, RESIST-Mixup reports a stable result, with a smaller standard variation of 0.9%. This result can be explained by the fact that RESIST-Concat fusion layer depends on K. As we combine more and more encodings when K increases, RESIST-Concat involves more and more trainable parameters. This increase in capacity makes the model more prone to memorizing training contaminants. In contrast, RESIST-Mixup is agnostic to K, as the fusion, in this case, is an element-wise averaging of all the encodings. Therefore, increasing K does not increase the model capacity and has no significant impact on the performance. This observation partially confirms our hypothesis H5.3 that RESIST-Mixup is more robust than RESIST-Concat.

Besides, we observe a slight decrease of both RESIST-Mixup and RESIST-Concat when K increases. However, this slight decrease is statistically insignificant according to Welch's test for p-value = 0.05. This sensitivity with respect to the hyperparameter k will be discussed in more detail later in the sensitivity analysis part 5.4.1.5.

We propose now to verify the second part of hypothesis H5.3, which states that RESIST-Mixup training is more efficient than RESIST-Concat, especially for large K. For this reason, we show in Figure 5.12 the training times of both variants. We note that the training times of both variants increase with larger K. This makes sense because RESIST siamese encoder processes more sequences when K is high. When $K = 2$, both training times are relatively similar: each training repetition lasts on average 140s. However, RESIST-Concat training becomes moderately longer when $K = 5$. In this case, the training time gap reaches 20s.

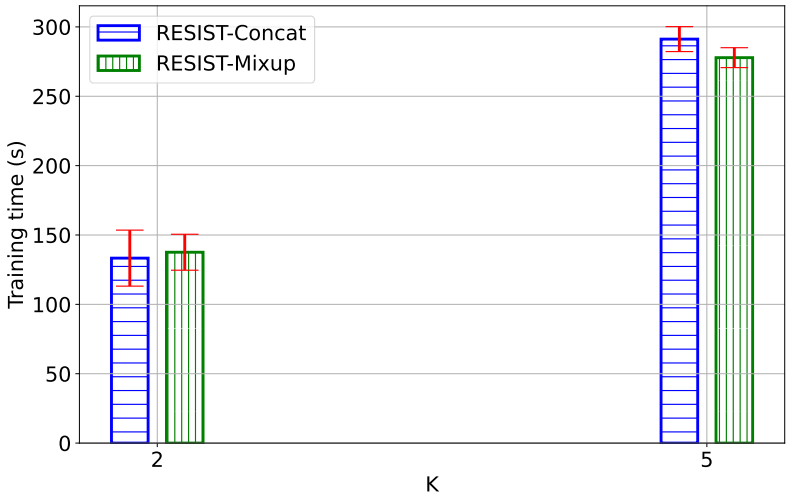


Figure 5.12: Evolution of RESIST-Concat and Resist-Mixup training times with the history length K.

All in all, we conclude that our hypothesis H5.3 is validated. The additive fusion with the *mixup* method is more efficient and more robust than the classical concatenation strategy, especially when K is large.

Protocol 4: Comparison with Competing Methods

In this section, we compare RESIST performance against common unsupervised anomaly detectors, presented in Section 5.4.1.3. We aim to demonstrate that RESIST outperforms these competing methods. The RESIST configuration used in this part is composed of the default architectures: a hybrid siamese encoder, i.e., the encoder of RESIST-SC, the *mixup* fusion layer, and the robust Geman-McClure loss, with $c = 0.1\text{IQR}$. The experiment results are reported in Figure 5.13.

Globally, RESIST achieves superior results compared to all the baselines, on the CICIDS17 dataset, with an average AUROC of $80.6\% \pm 1.3$. First, we note that RESIST is substantially more robust than Transformers. RESIST improves Transformer average AUROC by 10%. Second, the lowest results are reported with a density-based anomaly detector: LOF. Indeed, detecting contextual and collective outliers based on the local density of high-dimensional data is challenging. Surprisingly, Transformer-based anomaly detectors show poor performance on this dataset, even with careful tuning of these architectures. TranAD and AnomalyTranformer report AUROC of $50.7\% \pm 1.0$ and $52.4\% \pm 2.1$. This implies that these methods are significantly sensitive to training outliers, on this network traffic dataset. It is however difficult to explain such poor results, despite the careful fine-tuning of the hyperparameter on the dedicated validation subset. Third, classical anomaly detectors, i.e., IF and OSVM, give better results than deep neural network-based anomaly detectors, including OmniAnomaly, MSCRED, Vanilla Transformer, and LSTM-AE. This observation ties well with the previous study conducted by Lai et al. [144]. We speculate that this might be due to the fact that the latter are developed for semi-supervised AD. Indeed, they assume that the training data are anomaly free. In the case of data pollution with anomalies, this assumption is not respected and consequently, these methods fail to distinguish both classes. Fourth, RESIST exceeds IF AUROC by 4% and OSVM AUROC by 3%, on average. These results demonstrate that RESIST is more robust than these competing anomaly detectors on the CICIDS17 dataset.

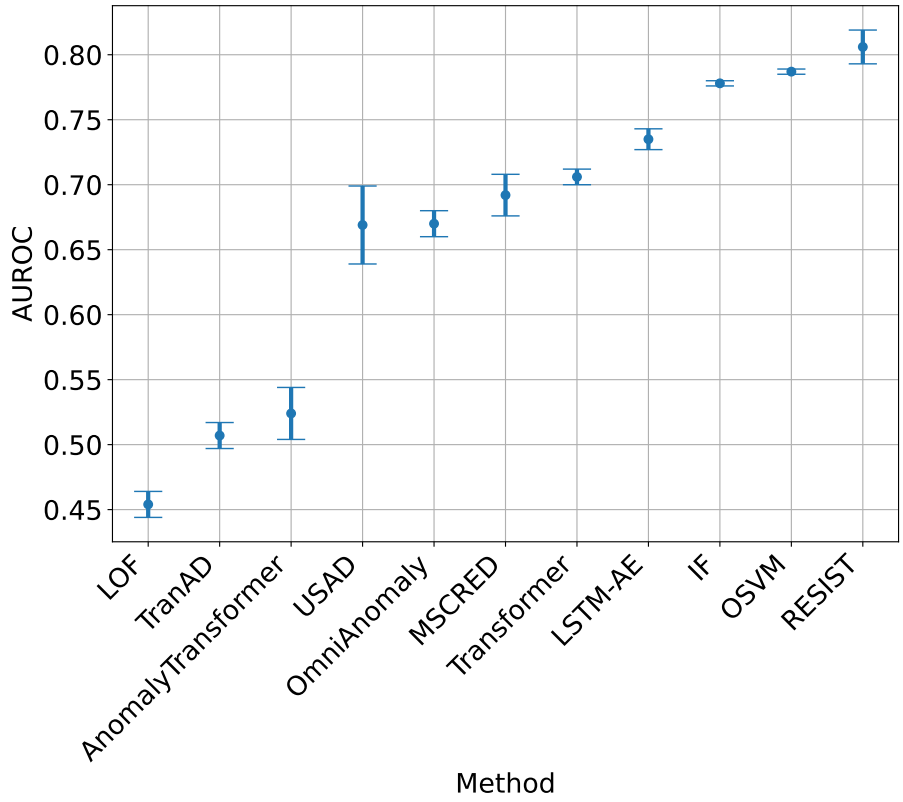


Figure 5.13: Comparison between RESIST and the competing methods on CICIDS17 dataset.

Sensitivity Analysis

Finally, we conduct further experiments to analyze RESIST sensitivity to the hyperparameter selection. In general, the hyperparameters are tuned on a dedicated validation subset to select the optimal ones that maximize model performances. This tuning requires the availability of ground truth data with labeled anomalies. However, in anomaly detection, labeled anomalies are scarce, and consequently, selecting the optimal hyperparameters is not guaranteed. As such, many studies advocate the design of robust anomaly detectors that are insensitive to hyperparameter selection [217, 292, 92]. A small variation of one hyperparameter should not significantly impact the model performance.

In this Section, we explore RESIST sensitivity to three hyperparameters: the history length K, the scale parameter c of the robust loss, and the number of heads of the multi-head attention units.

Sensitivity to the history length K We start by investigating RESIST sensitivity regarding the history length K. Figure 5.14 summarizes the sensitivity analysis results. In particular, we present RESIST performance for different $K \in \{2,3,5\}$. As shown in Figure 5.14, the hyperparameter K has no significant impact on RESIST performance. We report a stable average AUROC, around 80%, for all values of K. However, it is worth noting the slight decrease in performance with very large K. Indeed the performance decreases by 1.1%, when K increases from 3 to 5. This slight decrease can be explained by the increased difficulty to learn a mapping between a sequence and the far history. In time-series data, the local context is critical for defining the norm. The data patterns may evolve in time. Consequently, for a large value of K, it becomes harder for co-attention to find very long-range inter-sequence interactions, which makes the nominal data reconstruction task more difficult.

All in all, RESIST show competitive results, regardless of the selected history length K.

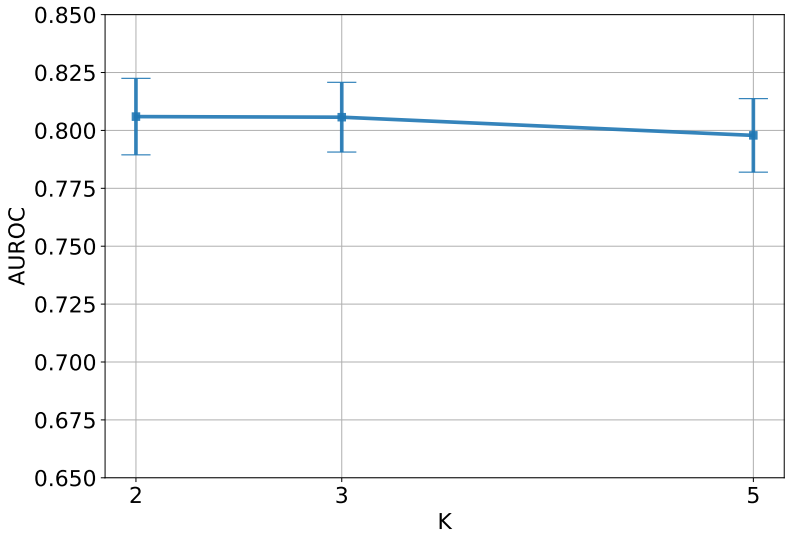


Figure 5.14: RESIST sensitivity analysis according to the history length hyperparameter K.

However, the performance slightly decreases with large K.

Sensitivity to the scale parameter c of the robust loss function In this section, we explore RESIST sensitivity with respect to the robust function scale hyperparameter c. c is defined as a multiple of the interquartile range of the reconstruction error: $c = \lambda \text{IQR}$, where $\lambda \in \mathbb{R}^+$. We train different models that share the same default architecture, we vary $\lambda \in \{0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4\}$, and we report the variation of the performance in Figure 5.15. We observe that RESIST performance depends on the hyperparameter c. The best performance is achieved with $c = 0.1\text{IQR}$ (i.e., with $\lambda = 0.1$). In this case, the average AUROC is equal to 81.3%. Unsurprisingly, the performance gradually degrades when we vary c. The lowest result is reported with a small $c = 0.05$. This case reflects an underfitting process, where the robust loss is very restrictive and over rejectitious. Indeed, since c is very small, RESIST ignores many nominal training data, and as a consequence, it fails to learn a representative mapping of inliers.

In contrast, the performance decrease is less sharp when c increases. The anomaly rejection becomes gradually less restrictive and RESIST models more and more anomalies with increasing c. Nonetheless, it is worth mentioning that, even with large c, e.g., $c = 0.4\text{IQR}$, RESIST is more robust than the classical Transformer and its performance is better than many competitive methods.

Sensitivity to head numbers h Lastly, we report in Figure 5.16 the sensitivity analysis results regarding the number of heads of the multi-head attention. In particular, we conduct a set of experiments where we vary $h \in \{1, 2, 4, 8\}$. Overall, the performance is stable and higher than 79.0%. For $h \in \{1, 2, 4\}$, the result is almost constant, around 81.1%. We report a slight decrease of 1% in mean AUROC, when $h = 8$.

To conclude, RESIST is insensitive to the number of heads used in the multi-head attention units.

5.4.1.6 Synthesis and Discussion

In this section, we analyzed the hypotheses H5.1-5.3. We have explored the impact of each building component of RESIST, by comparing the performance of different variants on the well-known CICIDS17 dataset. In the conclusion of this first set of experiments, we validated the three first hypotheses H5.1-5.3, and we selected the optimal architecture of RESIST. This architecture is

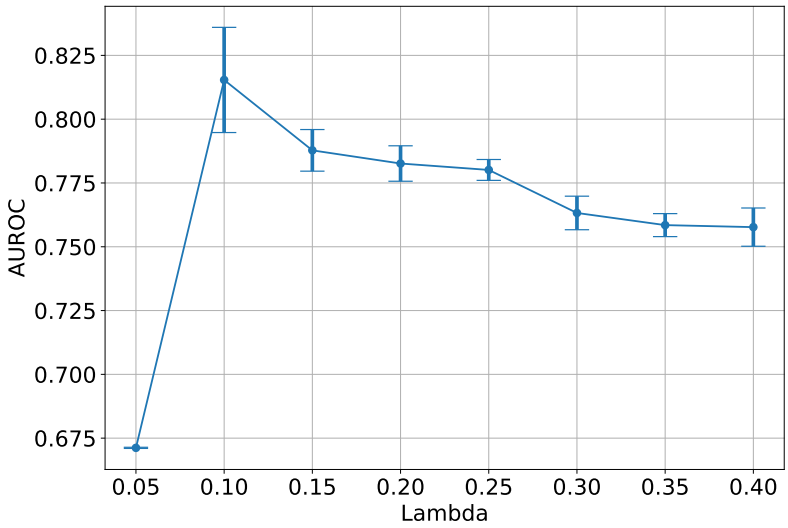


Figure 5.15: RESIST sensitivity analysis according to the scale hyperparameter c of training robust loss.

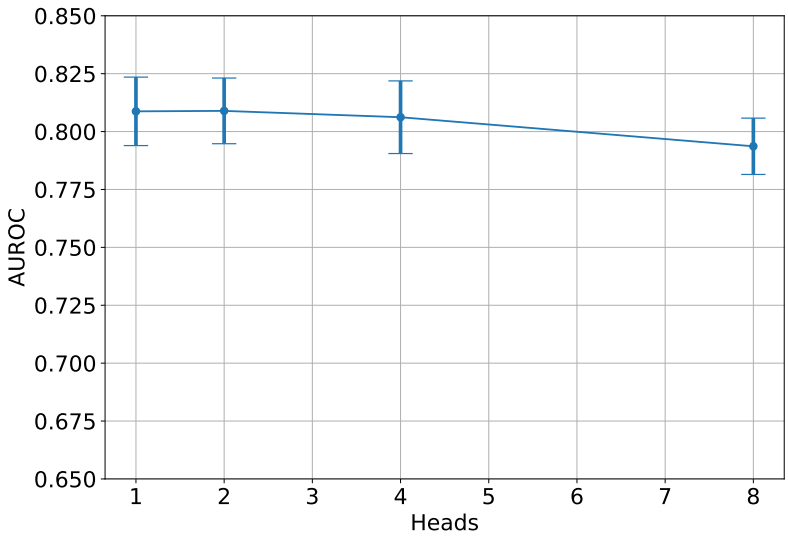


Figure 5.16: RESIST sensitivity analysis according to the number of heads h of multi-head attention units.

composed of a hybrid Siamese encoder, i.e., RESIST-SC encoder, the *mixup* fusion strategy, and the Geman-McClure robust loss, with $c = 0.1$IQR. Afterwards, we used this default architecture to compare RESIST robustness against common unsupervised anomaly detectors. We have proved that RESIST significantly outperforms these methods, which validates the last hypothesis H**??**. Finally, we investigated RESIST sensitivity with respect to its hyperparameters: the history length K, the robust loss scale parameter c, and the number of heads of the multi-head attention units. We have shown that RESIST is insensitive to both K and h. However, the performance depends on the scale parameter c. We explained this observation by the fact RESIST underfits inliers, when c is small, as the robust function becomes very rejectitious. In contrast, with high c, RESIST overfits a subpart of raining contaminants. We advocate to set $c = 0.1$IQR in the default configuration. Subsequently, we will validate these observations on a second public dataset: TON-IoT. This recent

IoT benchmark dataset is collected from a large-scale network involving real IoT devices and provides up-to-date IoT anomalous attacks.

5.4.2 TON-IoT Dataset

5.4.2.1 Dataset Description

The Telemetry dataset of IoT and IIoT sensors, Operating systems, and Network traffic, denoted as TON-IoT [179], is a collection of multiple datasets, collected from a large-scale network of IoT and Industrial IoT devices. Alsaedi et al. proposed this dataset in 2020, to address the lack of IoT-based datasets tailored for data-driven temporal anomaly detection. The Australian IoT Lab of UNSW Canberra Cyber collected this dataset from a large-scale network comprising multiple physical IoT sensors, virtual machines, and a router. TON-IoT testbed is shown in Figure **??**. This testbed mimics a classical network architecture used in IIoT and industry 4.0. We refer the reader to [179] for a detailed description of this architecture. We mainly focus here on the edge layer. The edge layer contains a set of real and simulated IoT devices, interconnected with a physical router. The physical devices include various IoT sensors such as weather sensors, temperature sensors, and pressure sensors. In addition, an NSX-VMware platform was installed to extend this testbed with virtual devices having similar behaviors.

Firstly, the authors collected the benign network traffic generated by these IoT devices during a typical configuration. Then, various hacking scenarios were designed to execute nine attack categories on these IoT devices and to generate malicious network traffic. The implemented attacks include, inter alia, Scanning attacks, Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks, Ransomware attacks, Backdoor attacks, and Cross-site Scripting (XSS) attacks. The raw network packets were stored in packet capture (pcap) files and made public. Later, Sarhan et al. [231] extracted network flow metadata from these raw files, using the CICFlowMeter tool [146]. Altogether, CICFlowMeter extracts 83 flow-based features, which were detailed in Section 5.4.1.1. A binary label of benign-anomaly was associated with each data flow. The new derived is publicly available for research purposes. Overall, TON-IoT comprises 5,351,760 labelled flows, where 2,836,524 samples are anomalous and the remaining 2,515,236 samples are benign. That is, benign data represents around 47% of the data.

5.4.2.2 Data Preprocessing

The same preprocessing steps as in Section 5.4.1.2 were applied to TON-IoT data. Indeed, the data is first sorted according to the flow timestamp. Then categorical features are count encoded and the data are rescaled to be in the range $[0, 1]$, with the min-max normalization method. Finally, the data is split into two subsets: the first 40% is dedicated for training and the remaining 60% is for the test. Similar to 5.4.1.2, 10% of the test subset is used to tune competing method hyperparameters.

Unlike CICIDS17 dataset, the ratio of anomalies in TON-IoT is large, i.e., 53%. This violated the common assumption of anomaly detection, which states that anomalies are rare and less frequent than the norm. To guarantee this assumption and to investigate algorithm sensitivity regarding the training anomaly ratio, we prepare four distinct training subsets containing 0%, 5%, 10%, and 15% of outliers. These anomalies are selected randomly from all TON-IoT anomalous training instances.

5.4.2.3 Training Protocol and Parameter Settings

In the following, we aim to validate RESIST robust performance on an IoT-based dataset. For this second dataset, we select the optimal default configuration of RESIST and we compare it against the competing methods. RESIST configuration is composed of the RESIST-CS encoder, the *mixup* fusion strategy, and Geman-McClure robust loss, with $c = 0.1\text{IQR}$. The history length K is equal to 2 and the multi-head attention units have two heads ($h = 2$).

The training parameter setting process and the evaluation criteria are identical to Section 5.4.1.4. We transform the data with a non-overlapped sliding window of length $w = 100$, we use the same 5-layer architecture, the same activation functions, and the same optimizer. All methods are trained for 100 epochs, with an initial learning rate of 0.001, a step-scheduler with a step of 0.5, and a batch size of 64. We repeat each experiment five times and average the results over these five runs.

5.4.2.4 Results

We present in Figure 5.17 the experimental results of the empirical comparison between RESIST and the other baselines, for different contamination ratios $\gamma \in \{0\%, 5\%, 10\%, 15\%\}$. This graph reveals that for all the methods, the performance trend is subjected to a gradual decline, with an increasing contamination ratio γ. However, the slope of the decline varies across methods.

Firstly, when the data are completely anomaly-free, i.e., $\gamma = 0$, the different methods report similar detection performance, with a mean AUROC around 79.5%. This case was specifically designed to acquire the optimal performance when the classical one-class assumption is verified. Secondly, it is interesting to notice that competing method performances considerably degrade, even with a moderate ratio of contamination $\gamma = 5\%$. We report a significant decline of the Vanilla Transformer and IF AUROC by 4% and 5%, respectively. Contrarily, RESIST retains a stable performance, equal to 79.6%. Thus, RESIST is robust against a low ratio of training outliers.

When the data contamination becomes relatively more important, with $\gamma \geq 10\%$, RESIST performance slightly decreases, to reach 76.8% with $\gamma = 15\%$. This decline is however more noticeable with competing methods, as IF AUROC stabilizes around 71.5%. The robustness of Vanilla Transformer is drastically degraded, to reach 62.9%.

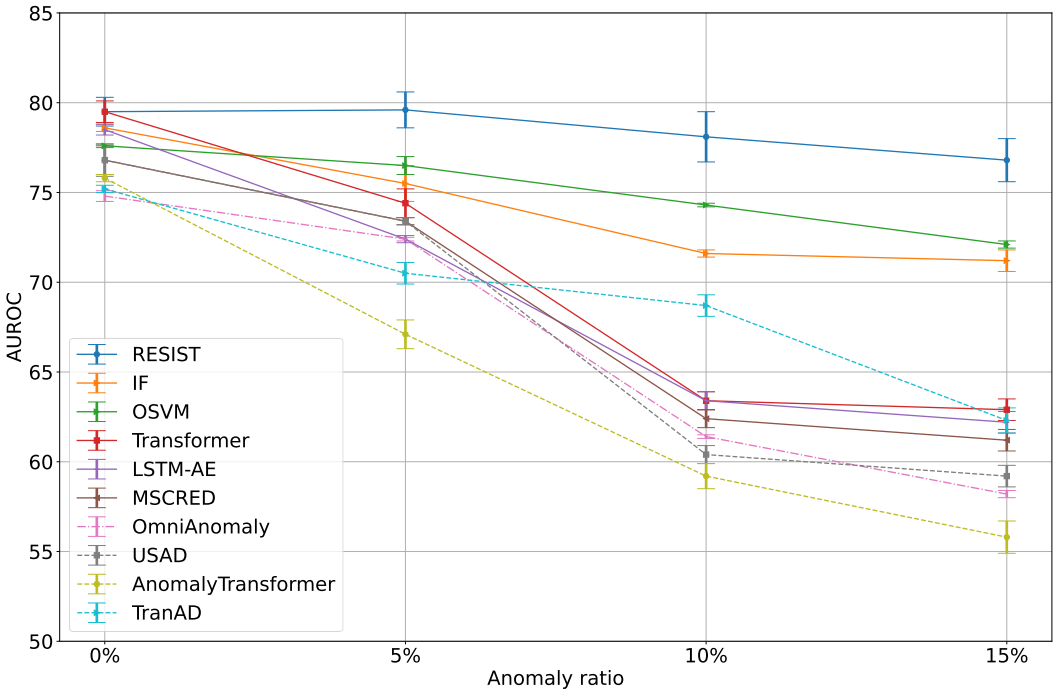


Figure 5.17: TON-IoT experimental results.

That is, RESIST is less sensitive to training contamination compared to competing methods, for the different pollution ratios. Despite the slight decline in performance with a higher contamination ratio, RESIST remains very competitive, with at least 5% points better AUROC than the baselines.

5.4.2.5 Synthesis and Discussion

In this section, we studied the potential of our contribution, RESIST, on the recent IoT dataset: TON-IoT. In essence, we have analyzed RESIST performance evolution with increasing contamination ratios $\gamma \in \{0\%, 5\%, 10\%, 15\%\}$. The main conclusion that can be drawn is that, even though the RESIST detection rate slightly declines with higher data pollution, our contribution shows competitive results compared to existing anomaly detectors. The present finding confirms that the same RESIST architecture achieves optimal performances and shows state-of-the-art results in anomaly detection, on two different datasets: CICIDS17 and TON-IoT. Therefore, we advocate for the use of this default configuration: the RESIST-SC hybrid encoder, the *mixup* fusion strategy, and the Geman-McClure robust loss, with $c = 0.1\text{IQR}$.

5.5 Comparison of the Three Contributions: RADON, GRAnD, and RESIST

This last section aims at experimentally comparing the three contribution performances on the benchmark dataset CICIDS17 (cf. Section 5.4.1.1). We particularly want to explore their detection performance and their training times on this dataset that contains the three types of anomalies: punctual, contextual, and collective.

The same data processing steps, training protocol, and training hyperparameters, which were defined in Section 5.4.1 are used here and the experimental results are reported in Table 5.3.

First, the best results are reported with our third contribution RESIST. RESIST AUROC exceeds GRAnD by around 14% and RADON by 17%. This significant advantage highlights the importance of modeling the contextual dependencies of the data for an accurate detection of temporal anomalies. In fact, since the iid assumption is violated in the CICIDS17 dataset, the performance of our two contributions is sub-optimal. However, we mention that RADON and GRAnD show relatively similar results compared to other baselines, such as USAD and OminAnomaly (cf. Section 5.4.1.5). Furthermore, one can notice the slight advantage of GRAnD over RADON, with a 3% points increase of AUROC. We explain this observation by the fact that GRAnD involves the uncertain samples in the optimization while RADON completely ignores them. This main distinction extends our contribution capacity to better represent the data.

Second, we compare the training time of the three algorithms. We note that our last contribution RESIST reports the fastest training, with lasts for 139s. GRAnD training is the slowest, around three times longer than RESIST. The training rejection strategy is the main cause of this difference. While the first two contributions explicitly reject training outliers with a dedicated strategy that studies the reconstruction scores of the AE, RESIST implicitly down-weights the impact of contaminants thanks to the robust loss. However, we mention that the cost of this implicit rejection strategy is its sensitivity to the hyperparameters, and mainly the robust loss scale parameter c (cf. Section 5.4.1.5). While RADON and GRAnD are robust to the hyperparameter selection, this is not the case with RESIST, with a clear sensitivity to c.

Table 5.3: The experimental results on the CICIDS17 dataset.

Method	AUROC (%)	Training time (s)
RADON	63.4 ± 0.4	258 ± 4
GRAnD	66.3 ± 1.0	416 ± 30
RESIST	80.6 ± 1.3	139 ± 11

5.6 Conclusion and Perspectives

In this chapter, we presented a proposal for robust anomaly detection in time series. We introduced RESIST, a Robust TransformEr tailored for unSupervISed Time series anomaly detection. Thanks to co-attention units, RESIST learns to reconstruct the input sequences using common properties shared with the history (i.e., the previous segments). Thus, both local information that is specific to the current input and global information shared with the history are leveraged in this downstream task. Moreover, we proposed a robust training strategy that minimizes the Geman-McClure loss function, to reduce the impact of training contaminants. We extensively study the contribution of RESIST components in the global performance, and the experimental evaluation on the two public datasets CICIDS17 and TON-IoT, confirmed that RESIST outperforms existing unsupervised anomaly detection. Finally, we substantially investigated RESIST sensitivity regarding its hyperparameters. This sensitivity analysis revealed RESIST sensitivity to the loss function scale parameter c.

Finally, an empirical comparison between the three contributions of the thesis is conducted on the CICIDS17 dataset. This comparison demonstrated the advantage of RESIST in detecting contextual and collective anomalies in time series data. RESIST implicit rejection strategy with the robust loss function resulted in faster training compared to the explicit rejection strategies of RADON and GRAnD.

However, one limitation reported with our last contribution RESIST is related to its sensitivity to the robust loss function scale hyperparameter. One potential solution to this problem consists in replacing the robust loss function with a more sophisticated rejection strategy that investigates the training data reconstruction scores and groups the data into disjoint subsets (i.e., similar to RADON or GRAnD rejection strategies). These strategy have show robust performance in Chapter 3 and 4. This strategy however must consider the contextual distribution of the reconstruction errors, as the context is key here.

6

Conclusion and Perspectives

"Normality is the ability to learn by experience, to be flexible, and to adapt to a changing environment." Lawrence S. Kubie

Contents

6.1 Conclusion . . . 134
6.1.1 RADON . . . 134
6.1.2 GRAnD . . . 135
6.1.3 RESIST . . . 135
6.2 Feasibility Study: End-to-end Monitoring Solution . . . 136
6.2.1 Proposition Of An Architecture For An End-to-end Monitoring Tool . . . 136
6.2.1.1 Data Collection . . . 136
6.2.1.2 Data Processing And Anomaly detection . . . 136
6.2.1.3 Data Visualization . . . 138
6.2.2 Discussion On Ethics . . . 138
6.2.2.1 Data Collection . . . 138
6.2.2.2 Data Processing And Anomaly Detection . . . 138
6.2.2.3 Data Visualization . . . 143
6.3 Research Limitations And Perspectives . . . 143
6.3.1 Data Discussion . . . 143
6.3.1.1 Validation Of Anomaly Detectors . . . 143
6.3.1.2 Privacy Discussion . . . 144
6.3.2 Model Discussion . . . 144
6.3.2.1 Sensitivity of RESIST . . . 144
6.3.2.2 Static Models . . . 144
6.3.2.3 One Model Per Device Family . . . 145
6.3.3 Explainable Anomaly Detection Discussion . . . 145

This chapter synthesizes the three contributions of the thesis, presents the main limitations of our research, and develops the potential perspectives toward the development of robust unsupervised anomaly detectors that can be applied to network traffic analysis.

6.1 Conclusion

The objective of our thesis consists in developing robust unsupervised anomaly detectors, that can infer an accurate representation of the norm from contaminated, noisy data. These models should be adequate for IoT device monitoring on the basis of network traffic analysis. This task is challenging as it involves not only optimizing one model to represent the input data, but also an additional step of rejecting potential corrupted data points that may skew the underlying representation learning task.

To this end, we developed three contributions. To simplify this arduous problem, we started by assuming that the data points are iid and hence, focused on punctual AD. Then, we relaxed this ideal hypothesis in our last contribution to study contextual and collective AD. In the following, we will present an overall summary of our three contributions, by developing two main aspects: (1) the training anomaly rejection strategy and (2) the optimization process.

6.1.1 RADON

Our first contribution leverages the classical vanilla AE to model the nominal behavior from the data. To mitigate the sensitivity of these conventional models with respect to training contamination, we proposed a novel training strategy that alternates between filtering training outliers and optimizing the model free-parameters.

To effectively filter training outliers, we performed unimodal thresholding of the reconstruction error histogram. We assume that most of the training data is nominal along with a minority of corrupted observations. As such, outliers are unstructured and more difficult to compress than inliers, particularly early in the training, and they present large reconstruction errors than inliers. We proposed to analyze the reconstruction error distribution to identify potential contaminants. Accordingly, we split the training data into three disjoint subsets containing respectively potential inliers with low errors, potential outliers presenting extremely high errors, and undetermined samples that have in-between reconstruction errors. The thresholds that separate these three subsets are automatically inferred from the distribution of errors by means of unimodal thresholding.

We propose to leverage these filtered anomalies to improve the distinction between both classes, acting as a self-supervision for the model: we learn a robust projection mapping, where inliers and their reconstructions are similar, while outliers are poorly reconstructed. For this purpose, we reformulated the training strategy as a metric learning problem. Indeed, the parameters of the backbone AE are optimized such that inliers are collinear to their reconstructions and outliers are orthogonal to their reconstructions. The undetermined instance subset is not involved in the optimization process to prevent training destabilization.

Excluding some training data from the learning process is the main shortcoming of our first contribution. Even though binary thresholding does not allow the model to determine their class memberships, one can address the uncertainty of these observations through probabilities, i.e., soft probabilistic membership instead of categorical one. We proposed to alleviate this limitation in our second contribution.

6.1.2 GRAnD

Our second contribution is designed to further enhance RADON learning capacity by leveraging the uncertain training points. The key intuition is to associate to each undetermined sample a probabilistic weight that quantifies its degree of anomalousness, i.e., the probability that the underlying data point belongs to the anomalous class. This allows to incorporate these uncertain data points in the optimization task without introducing conflicting steps.

For this purpose, we replaced vanilla AEs with VAEs and NFs, to introduce probabilistic modeling of the reconstruction errors. The resulting probability can be leveraged to infer the confidence of the model decision.

Furthermore, we replaced the histogram unimodal thresholding strategy with an EVT-based approach, which is more adequate for such settings. Similar to RADON, the training data are divided into inliers, outliers, and undetermined points. The distinction of GRAnD consists in estimating a set of fuzzy weights for the undetermined samples with the eCDF function. Therefore, we defined a weighted objective function that optimizes the model parameters, taking into account these uncertainty measures.

Both RADON an GRAnD are limited to punctual AD in non-sequential data. We extended the horizon of our study to contextual and collective AD in time series, with our third contribution RESIST.

6.1.3 RESIST

Our third contribution RESIST relies on Transformers to model the data temporal correlations and detect any deviation. Similar to AEs, these sequence-to-sequence models are trained to reconstruct the training data and to learn a latent representation where inliers and outliers are distinguishable.

Since vanilla Transformers are not robust to training contamination by design, we introduced a learning strategy to mitigate this sensitivity limitation. We modified the architecture of these models by incorporating a Siamese architecture in the encoder. Benefitting from the self-attention and co-attention mechanisms, this model can learn the point-wise correlations between each sequence data point and its history. In other words, the encoder is encouraged to find a low-dimensional space where common data points share similar representations. This task is notoriously hard for anomalies, since they present non-representative uncommon patterns.

Unlike the first two contributions, the rejection strategy of RESIST is done implicitly with a robust loss function. Unlike the traditional least-square minimization, this robust loss can resist noise and anomalies by reducing the influence of their large reconstruction errors. We empirically compared RESIST performance with different robust functions of the literature: Charbonnier loss, Cauchy loss, Geman-McClure loss, and Welsch loss.

We extensively studied the contribution of RESIST components in the global performance, with an ablation study, and the experimental evaluation on the two public datasets CICIDS17 and TON-IoT, confirmed that RESIST outperforms existing unsupervised anomaly detection. The sensitivity analysis showed however that RESIST is sensitive to one hyperparameter: the scale of the robust loss function.

Finally, an empirical comparison between the three contributions of the thesis was conducted on the CICIDS17 dataset. This comparison demonstrated the advantage of RESIST in detecting contextual and collective anomalies in time series data. RESIST implicit rejection strategy with the robust loss function resulted in faster training compared to the explicit rejection strategies of RADON and GRAnD. GRAnD performance was slightly better in these experiments than RADON, highlighting the advantage of our probabilistic modeling of the problem.

In the present thesis, we assessed the performance of our models on multiple public datasets. Now, we must ensure that these algorithms are operational in real-world environments and can

monitor physical IoT devices. The following section presents a feasibility study of an end-to-end monitoring tool integrating our robust models.

6.2 Feasibility Study: End-to-end Monitoring Solution

This section aims at investigating the efficiency of our models in real-world environments. Section 6.2.1 presents an architecture description of our implemented end-to-end monitoring tools that collects the data flow from a network of IoT devices, process and process the data, and visualize the detected anomalies. Then, Section 6.2.2 discusses the ethical-related aspects of our tool and ensures that our baseline ethical principles (cf. Section 1.1.5 of Chapter 1) are respected.

6.2.1 Proposition Of An Architecture For An End-to-end Monitoring Tool

We developed an end-to-end monitoring tool that collects the network traffic of a testbed comprising real IoT devices, extracts and processes the flow metadata, stores these data into a monitoring database, detects any potential anomalous flow using one of our models, and visualize the results in a user-friendly dashboard. The overall architecture of our tool is illustrated in Figure 6.1. We mention that this implementation is based on open-source software and libraries.

We present here a brief description of each building block. We install a testbed that simulates customer LANs. This testbed is composed of a router/switch and a set of IoT that are commonly used in end-user LANs. We configured the router to mirror all the in-going and out-going traffic and extract the flow data. These data are sent to another analysis PC. On this machine, we have implemented our monitoring solution. As illustrated, the pipeline involves 3 main stages (1) data collection, (2) data processing and AD, and (3) data visualization.

6.2.1.1 Data Collection

The first stage consists in collecting the raw flow data sent by the router, and prepossessing and storing it in a local database. To this end, *nProbe*[1], which is an open-source flow exporter, was installed to transform the testbed network packets into a set of timestamped flows. These data are streamed to a flow collector installed on a monitoring computer. We selected this popular software because it is flexible, with many possible configurations, and provides a large list of the exported IEs. It is also efficient for real-time processing, with optimized packet capture techniques [18].

Afterward, the collected flows are parsed and stored in a monitoring database. For this task, we use the *Elasticsearch, Logstash, and Kibana (ELK) stack*. It is a collection of three open source software: Logstash[2], Elasticsearch[3], and Kibana[4]. Network flows are parsed with Logstash and sent to Elasticsearch for storage in a local database. Elasticsearch is optimized to index large volumes of data in an efficient way that allows quick search and retrieval with a set of queries.

6.2.1.2 Data Processing And Anomaly detection

The second step is to analyze the network flows indexed in the Elasticsearch database. A python connector[5] is installed to retrieve the indexed data and process it according to the processing

[1] https://www.ntop.org/products/netflow/nprobe/
[2] https://www.elastic.co/fr/logstash/
[3] https://www.elastic.co/fr/elasticsearch/
[4] https://www.elastic.co/fr/kibana/
[5] https://pyelasticsearch.readthedocs.io/en/latest/

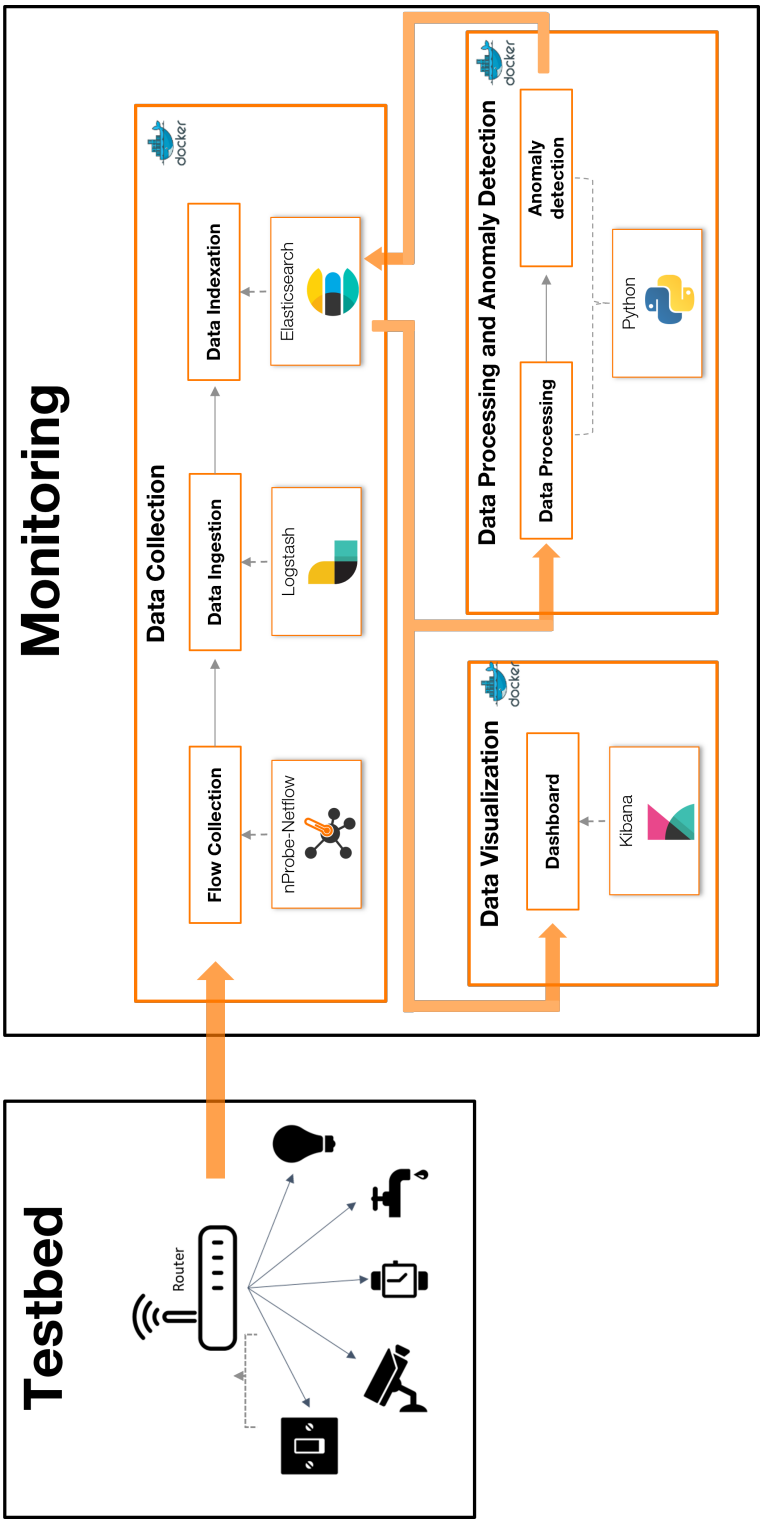


Figure 6.1: Architecture of the end-to-end monitoring tool.

protocol described in the thesis. Then, our models are trained on a subset of the processed data flows until convergence. Finally, these models are used to output an anomaly score for each new indexed network flow, and these scores are fed back into Elasticsearch.

6.2.1.3 Data Visualization

The final module role is to visualize the network flows as well as the anomaly scores in interactive user-friendly dashboards. This task is performed by Kibana, which provides dynamic dashboards that allow network administrators to explore the overall network anomalies and identify anomalous connections. We adapted *Elastiflow*[6] open-source dashboards to our situation. Figures 6.2, 6.3, 6.4, and 6.5 illustrate some interfaces of our monitoring tool.

6.2.2 Discussion On Ethics

Our research involves the collection of end-users data to train and evaluate the anomaly detectors. We described in the introduction of the thesis a set of ethical principles that that Orange is committed to in order to protect the personal data and privacy of its customers. This section discusses the ethical-related aspects of our aforementioned monitoring solution and algorithms.

6.2.2.1 Data Collection

Firstly, since the collected data may include personal en-user information, our study focused on flow-based AD, which is less intrusive than full packet analysis. In our study, only network traffic header-based metadata are analyzed and the data contents, i.e., the payloads, are excluded and never examined. Secondly, the collected data are preprocessed and stored in a local database in the LAN. Indeed, network flows disclose a lot of information on users' presence, habits and services. Thus, we advocate for an on-site monitoring process where the user data are kept in the LAN and never shared with external company servers.

6.2.2.2 Data Processing And Anomaly Detection

Since the thesis proposed model-based anomaly detectors, two main aspects should be studied: (1) the training and (2) the inference, i.e., AD.

Firstly, the training of artificial neural networks requires the availability of advanced computational resources, i.e., CPU, Graphics Processing Units (GPUs), and memory. A first intuitive idea consists in leveraging the customer *gateway*, i.e., *Orange LiveBox* for the training of our models. However, classical customer gateways have constrained resources and hardware [18]. A potential solution consists in introducing novel equipment in the LAN, equipped with the required hardware resources and serving as a server for the training of our anomaly detectors. Even though this option may be expensive, it guarantees that the training phase is locally performed in customer premises and no data are transferred outside the home. Rather than introducing novel equipment, another perspective consists in integrating add-on tools, e.g., Google Edge TPUs[7] and NVIDIA Nano[8], into existing LAN equipment. These AI accelerators can be flexibly integrated into existing System On Chip (SoC) that Orange uses in its smart home devices. For example, these add-on AI accelerators are well-suitable for Set-top-boxes and other media products, e.g., Mediatek and Broadcom devices [8, 81].

Secondly, once the models are trained on user IoT device data, they can be deployed to detect novel observation anomalies. Unlike the first stage of training, the inference is less resource-

[6] https://www.elastiflow.com/
[7] https://cloud.google.com/edge-tpu?hl=fr
[8] https://www.nvidia.com/fr-fr/autonomous-machines/embedded-systems/jetson-nano/

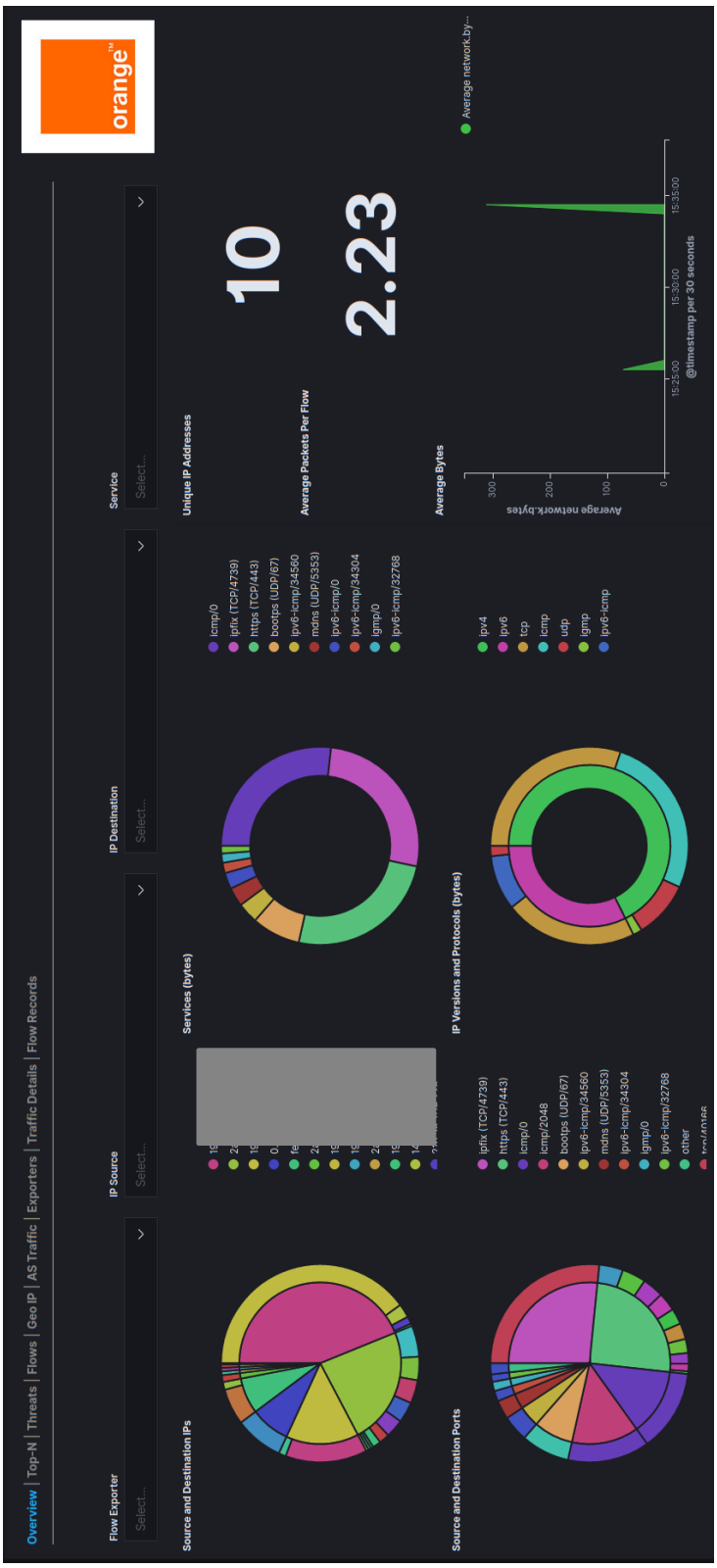


Figure 6.2: The overview interface of our monitoring solution.

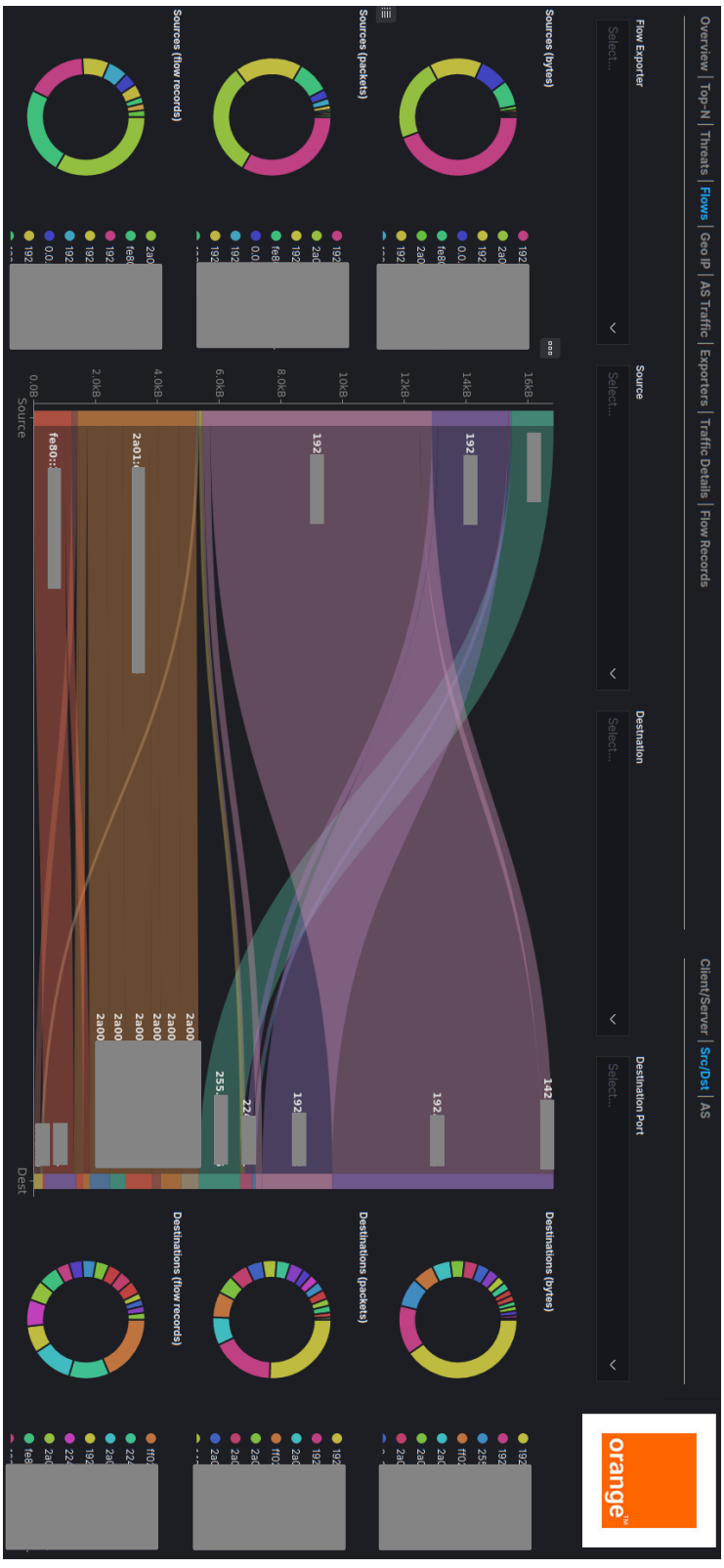


Figure 6.3: The flow interface of our monitoring solution.

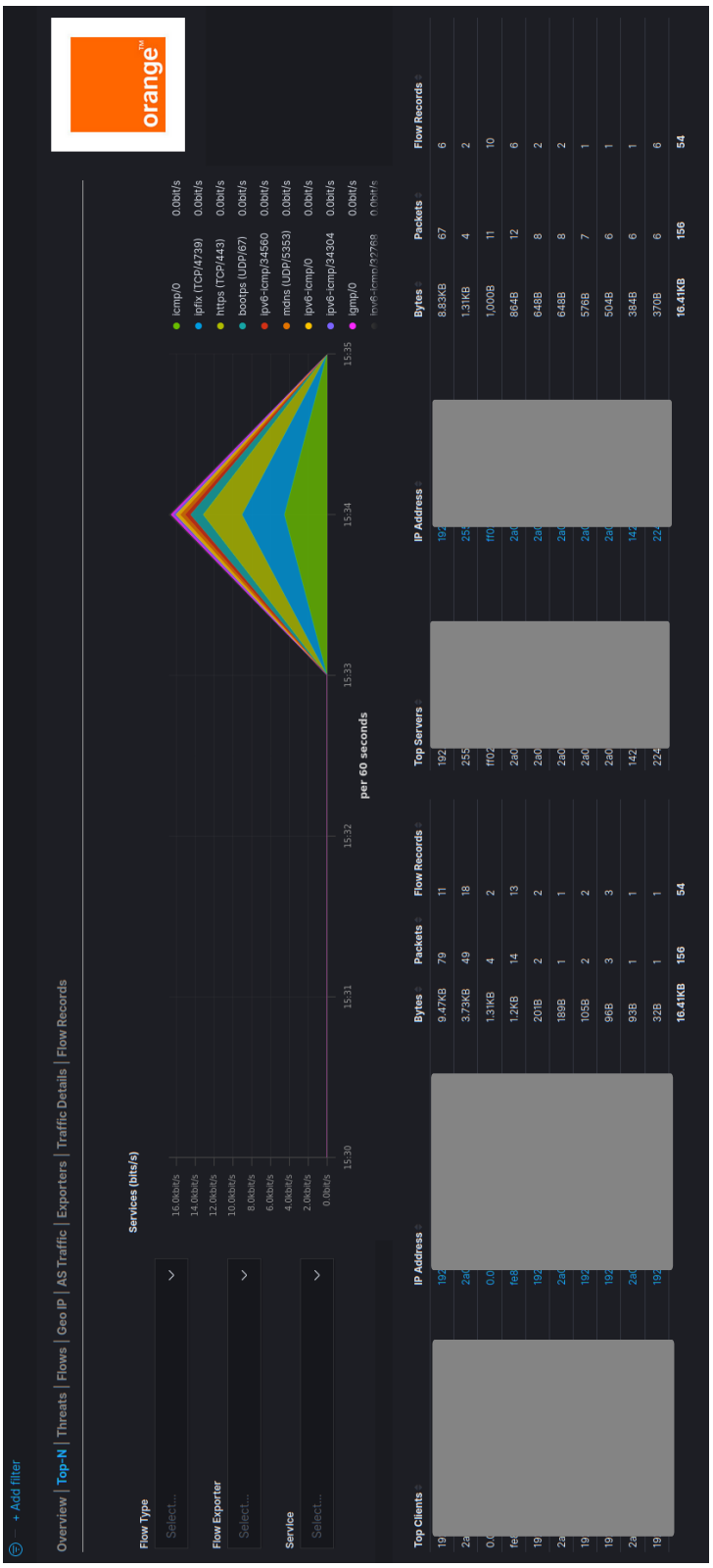


Figure 6.4: The service interface of our monitoring solution.

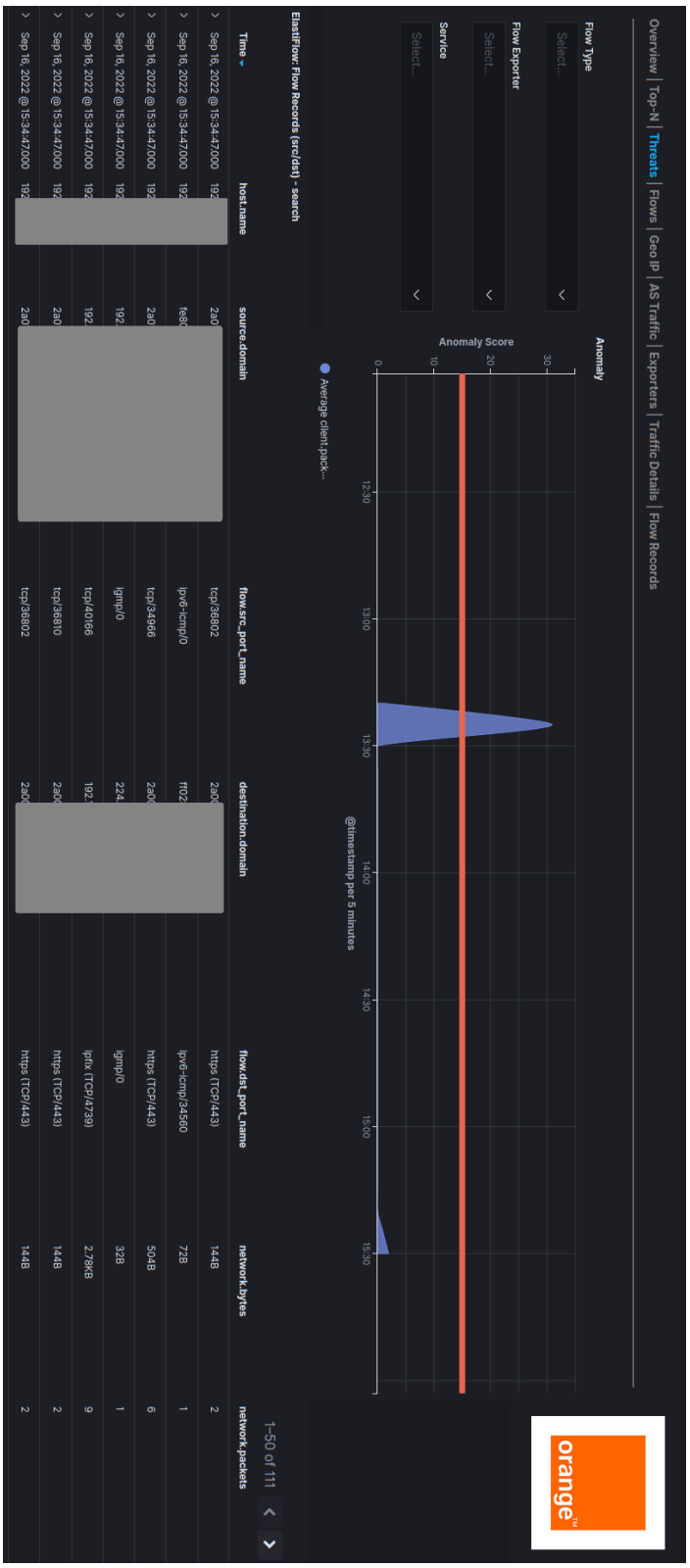


Figure 6.5: The anomaly interface of our monitoring solution.

consuming and faster. We can rely on edge computing strategies, with edge machine learning software, e.g., TensorFlow Lite[9], to deploy the trained model on an edge device such as gateways and routers. In our proposed implementation, both training and inference stages are performed on the same monitoring computer: a laptop equipped with a 12-core Intel i7-9850H CPU clocked at 2.6GHz and with NVIDIA Quadro P2000 GPU.

6.2.2.3 Data Visualization

Finally, data visualization aims to provide insights into the detected anomalies, to ease their diagnosis and troubleshooting. Indeed, the potential anomalous flows are tagged in the local Elasticsearch database. A set of queries can be executed locally or remotely by network administrators in case of issues to provide the customer with information on the potential equipment causing the problem, so that he can be redirected to adequate support services.

To summarize, our final proposition respects the following principles: data privacy, thanks to in-premise AD and SPI, i.e., header-metadata-based flow analysis; and model privacy, thanks to our robust unsupervised training strategies and contributions.

We now discuss the limitations of our research and we present the potential perspectives on these shortcomings.

6.3 Research Limitations And Perspectives

We articulate our discussion around three key aspects: (1) the data (cf. Section 6.3.1), (2) the model (cf. Section 6.3.2), and (3) the post-processing of the detected anomalies towards an explainable AD (cf. Section 6.3.3).

6.3.1 Data Discussion

This section discusses two data-related limitations: (1) the data required to validate anomaly detectors and (2) and the privacy-based issues related to the corresponding data analysis. For each limitation, we present a set of potential perspectives that can be addressed in future studies.

6.3.1.1 Validation Of Anomaly Detectors

In our thesis, we carefully evaluated the performance of our approaches on four public benchmark network traffic datasets. Furthermore, we tested these contributions in near-real conditions, with a testbed comprising multiple IoT devices. However, all the anomalies included in these experiments are related to malicious network attacks. The lack of public datasets comprising labeled traffic of non-malicious anomalies is the main limitation to a realistic evaluation of our anomaly detectors. The scarcity of adequate datasets has been highlighted by numerous publications in the network monitoring community [201].

Given the promising results reported in our study, the next step consists in expanding our testbed configuration to include more heterogeneous IoT devices and developing more sophisticated non-malicious anomalies. A partially labeled network traffic dataset could be generated and made public to the community.

[9] https://www.tensorflow.org/lite

6.3.1.2 Privacy Discussion

We presented at the beginning of the thesis a set of ethical guidelines that motivated our choice of the data features, i.e., SPI and home-centric monitoring solution, that is models specific to a single LAN. Nonetheless, some constrained IoT devices generate few data during their nominal activity. Collecting enough nominal data to train neural network-based anomaly detectors, relying on one single home frugal IoT device may be time-consuming and inefficient [232]. One potential solution to this problem is to aggregate the data generated from multiple LANs that share similar characteristics. This solution introduces however privacy problems, since sharing the customer data may violate their secrecy. Instead of sharing the data and centralizing the model training, a less intrusive solution consists in keeping the customer data in the home and federating the training of the model thanks to federative learning strategies [195].

The following section discusses the perspectives on model training and inference.

6.3.2 Model Discussion

6.3.2.1 Sensitivity of RESIST

While our two first contributions are robust to hyperparameter selection, the principal limitation of our last contribution is the sensitivity with respect to the scale parameter of the robust loss function.

This could probably be improved by replacing the robust loss function with a more sophisticated rejection strategy that investigates the reconstruction scores and explicitly rejects extreme values, similar to RADON and GRAnD rejection strategies. This potential perspective is justified by the robust performance shown by these strategies in Chapters 2 and 3, despite the slower training time (cf. Section 5.5 of Chapter 5). It is a question of future research to adapt the threshold selection strategy to the temporal context of the data. We particularly speculate that local thresholds defined over data point history, i.e., adjacent sliding windows, are crucial to accurately filter contextual outliers. This intuition is justified by the evolution of the nominal pattern over time and the extreme reconstruction errors in one context may be nominal in another one. Developing the optimal explicit rejection strategy with local threshold selection may constitute the object of future studies.

6.3.2.2 Static Models

In this section, we discuss a broader challenge that concerns our three contributions, as well as the majority of literature anomaly detectors. This challenge concerns the static modeling of real-world phenomena, e.g., the network behavior of one IoT device, with artificial neural networks. Indeed, most machine learning-based models are designed to learn a static model from the input data [111]. The first phase of *offline* training is conducted to estimate the optimal parameters of the model and then the model is deployed for inference. However, the real world is continuously changing, e.g., the nominal behavior of one device may evolve after firmware updates. The underlying model must be flexible enough to integrate these novelties into consideration without forgetting previously learned aspects. This point is particularly important in our task of AD. If the nominal model does not continuously integrate the novel characteristics introduced after an update, the false alarm rate may increase as the model considers such novelties as anomalies. The model should also be able to integrate expert feedback on false detection in an efficient way.

One possible solution to address this challenge is by continuously re-training the model with newly collected data. This is however not straightforward as the update must be carefully performed without completely impacting the learned patterns, a.k.a., catastrophic forgetting. Although some studies are emerging to address this problem of lifelong learning applied to AD [88, 84], it remains an open research issue where numerous questions remain to be addressed.

6.3.2.3 One Model Per Device Family

Our study advocates LAN-specific AD approaches, with one nominal model per device family and per LAN. This results in an expensive data collection and processing process. One can envision sharing models between similar devices of multiple LANs, which share a close overall activity. For example, one trained model that represents the network activity of an alarm sensor or a connected thermometer may be adequate for many customer networks. Besides, a shared vision of different LANs may facilitate the diagnosis, by correlating detecting anomalies. For example, the same anomaly reported in many LANs after a firmware update or a specific device configuration, may be caused by misconfiguration errors.

We note however that sharing model between LANs is not straightforward. Even though IoT devices are designed to perform specific tasks without human interaction, their nominal activity may significantly depend on the contextual behaviors of householders. A light bulb nominal behavior may differ according to the lifestyle of home occupants. Sharing models may require considering each LAN supercities to minimize false alarms. There exist an increasing interest in federated learning approaches, which allow synchronizing model learning without sharing the training data [195]. In future work, investigating such a paradigm might prove important.

6.3.3 Explainable Anomaly Detection Discussion

In this thesis, we investigated network traffic AD based on the unsupervised learning paradigm. In particular, the studied models detect anomalies according to the difference between the expectation, i.e., the norm, and the reality. This makes such models, and particularly AEs a good fit for unsupervised AD, as their training does not require labeled outliers and predefined knowledge about the anomalous class. However, such a paradigm introduces one critical challenge: the explainability of the detected anomalies. In fact, understanding the root cause that triggers the alarm relying only on the reconstruction error is challenging. It is difficult to precisely localize the anomalous attributes in the reconstructed high-dimensional data and more explainable solutions may be required to save the time-consuming diagnosis task by experts [131].

There have been numerous studies to investigate the topics of eXplainable Artificial Intelligence (XAI), and more specifically eXplainable Anomaly Detection (XAD) [235, 212]. A wealth of publications has mainly focused on supervised learning to explain the raw model outputs and provide useful insights for decision-makers and domain experts. Fewer studies has investigated XAI and XAD under the unsupervised setting [235, 213]. One interesting idea that can be applied to AE detected anomalies consists in studying the influence of each data feature on the optimization, for example using influence functions developed in statistics [138]. If a small variation of one feature resulted in a significant variation of the loss function, the corresponding feature is probably anomalous. A similar idea has been explored in [193] where the gradient, i.e., derivatives of the variational lower bound of a vanilla VAE are studied to precisely identify the features causing the anomaly. This method may become ineffective for high-dimensional data, as it requires analyzing the gradient of each data feature. One second perspective is to apply some generic XAI methods to AE-based AD. For example, Ravi et al.[213] compare the AD performance of AEs with four different XAI methods: Layer Relevance Propagation, Local Interpretable Model-agnostic Explanation, SHapley Additive explanations, Counterfactual approaches. We refer the reader to this study for further details [213].

All in all, post-process the detected anomalies is of paramount importance to diagnose the root cause, filter potential false alarms, and fine-tune the underlying model. Future research should examine strategically this topic to further improve network traffic-based AD.

Bibliography

[1] *1998 DARPA Intrusion Detection Evaluation Dataset | MIT Lincoln Laboratory*. URL: https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset.

[2] Janet Ellen Abbate. *From ARPANET to INTERNET: A history of ARPA-sponsored computer networks, 1966-1988*. 1994.

[3] Mariam M. N. Aboelwafa, Karim G. Seddik, Mohamed H. Eldefrawy, Yasser Gadallah, and Mikael Gidlund. "A Machine-Learning-Based Technique for False Data Injection Attacks Detection in Industrial IoT". In: *IEEE Internet of Things Journal* (2020), pp. 8462–8471.

[4] Admin. *That 'Internet of Things' Thing*. 2009. URL: https://www.rfidjournal.com/that-internet-of-things-thing.

[5] Charu C. Aggarwal. "Data Mining: The Textbook". In: 2015.

[6] Charu C. Aggarwal. *Outlier Analysis*. 2017.

[7] Shikha Agrawal and Jitendra Agrawal. "Survey on Anomaly Detection using Data Mining Techniques". In: *Procedia Computer Science* (2015), pp. 708–713.

[8] *AI Based Hand-held Device*. MediaTek. URL: https://www.mediatek.com/aiot/solutions/case-studies/ai-based-handheld-device.

[9] François Aıssaoui, Samuel Berlemont, Marc Douet, and Emna Mezghani. "A semantic model toward smart iot device management". In: *Workshops of the International Conference on Advanced Information Networking and Applications*. Springer. 2020, pp. 640–650.

[10] Haleh Akrami, Sergul Aydore, Richard M. Leahy, and Anand A. Joshi. "Robust Variational Autoencoder for Tabular Data with Beta Divergence". In: *arXiv:2006.08204 [cs, eess, stat]* (2020).

[11] Haleh Akrami, Anand A. Joshi, Jian Li, Sergul Aydore, and Richard M. Leahy. "Robust Variational Autoencoder". In: *arXiv:1905.09961 [cs, eess, stat]* (2019).

[12] Hashem Alaidaros, Massudi Mahmuddin, Ali Al Mazari, et al. "An overview of flow-based and packet-based intrusion detection performance in high speed networks". In: *Proceedings of the International Arab Conference on Information Technology*. 2011, pp. 1–9.

[13] Omar Alghushairy, Raed Alsini, Terence Soule, and Xiaogang Ma. "A Review of Local Outlier Factor Algorithms for Outlier Detection in Big Data Streams". In: *Big Data and Cognitive Computing* (2020), p. 1.

[14] Mustafa Altaha, Jae-Myeong Lee, Muhammad Aslam, and Sugwon Hong. "An Autoencoder-Based Network Intrusion Detection System for the SCADA System." In: *J. Commun.* (2021), pp. 210–216.

[15] K Amirthalingam and Robert J Moorhead. "SNMP-an overview of its merits and demerits". In: *Proceedings of the Twenty-Seventh Southeastern Symposium on System Theory*. IEEE. 1995, pp. 180–183.

[16] Eirini Anthi, Lowri Williams, Małgorzata Słowińska, George Theodorakopoulos, and Pete Burnap. "A Supervised Intrusion Detection System for Smart Home IoT Devices". In: *IEEE Internet of Things Journal* (2019), pp. 9042–9053.

[17] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. "Understanding the mirai botnet". In: *26th USENIX security symposium (USENIX Security 17)*. 2017, pp. 1093–1110.

[18] Z. Aouini, A. Kortebi, and Y. Ghamri-Doudane. "Traffic monitoring in home networks: Enhancing diagnosis and performance tracking". In: *2015 International Wireless Communications and Mobile Computing Conference (IWCMC)*. 2015, pp. 545–550.

[19] Nari Sivanandam Arunraj, Robert Hable, Michael Fernandes, and Michael Heigl. "Comparison of Supervised , Semi-supervised and Unsupervised Learning Methods in Network Intrusion Detection System (NIDS) Application". In: 2017.

[20] Luigi Atzori, Antonio Iera, and Giacomo Morabito. "The internet of things: A survey". In: *Computer networks* (2010), pp. 2787–2805.

[21] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A. Zuluaga. "USAD: UnSupervised Anomaly Detection on Multivariate Time Series". In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020, pp. 3395–3404.

[22] Juozas Auskalnis, Nerijus Paulauskas, and Algirdas Baskys. "Application of local outlier factor algorithm to detect anomalies in computer network". In: *Elektronika ir Elektrotechnika* (2018), pp. 96–99.

[23] R. Can Aygun and A. Gokhan Yavuz. "Network Anomaly Detection with Stochastically Improved Autoencoder Based Models". In: *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*. 2017, pp. 193–198.

[24] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. "Layer normalization". In: *CoRR* (2016).

[25] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473* (2014).

[26] M. Bahrololum and M. Khaleghi. "Anomaly Intrusion Detection System Using Gaussian Mixture Model". In: *2008 Third International Conference on Convergence and Hybrid Information Technology*. 2008, pp. 1162–1167.

[27] A. A. Balkema and L. de Haan. "Residual Life Time at Great Age". In: *The Annals of Probability* (1974), pp. 792–804.

[28] Dor Bank, Noam Koenigstein, and Raja Giryes. "Autoencoders". In: *arXiv:2003.05991 [cs, stat]* (2020).

[29] Paul Barford, Jeffery Kline, David Plonka, and Amos Ron. "A Signal Analysis of Network Traffic Anomalies". In: *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment*. IMW '02. 2002, pp. 71–82.

[30] Jonathan T. Barron. "A General and Adaptive Robust Loss Function". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 4326–4334.

[31] A Basu. "Robust and efficient estimation by minimising a density power divergence". In: *Biometrika* (1998), pp. 549–559.

[32] Oladayo Bello and Sherali Zeadally. "Intelligent device-to-device communication in the internet of things". In: *IEEE Systems Journal* (2014), pp. 1172–1182.

[33] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation Learning: A Review and New Perspectives". In: *arXiv:1206.5538 [cs]* (2014).

[34] Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "Class-balanced siamese neural networks". In: *Neurocomputing* (2018), pp. 47–56.

[35] Samuel Berlemont, Grégoire Lefebvre, and Naji Najari. *Remote support solution to aid in the diagnosis of identification of dysfunctional equipment not administered by the service provider.* FR2012433, 2021.

[36] Gautam Bhattacharya, Koushik Ghosh, and Ananda S. Chowdhury. "Outlier Detection Using Neighborhood Rank Difference". In: *Pattern Recogn. Lett.* (2015), pp. 24–31.

[37] Christopher M. Bishop. *Pattern recognition and machine learning*. Information science and statistics. 2006.

[38] Christopher M. Bishop. *Pattern recognition and machine learning*. Corrected at 8th printing 2009. Information science and statistics. 2009.

[39] Marzieh Bitaab and Sattar Hashemi. "Hybrid Intrusion Detection: Combining Decision Tree and Gaussian Mixture Model". In: *2017 14th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology (ISCISC)*. 2017, pp. 8–12.

[40] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A. Lozano. "A Review on Outlier/Anomaly Detection in Time Series Data". In: *ACM Computing Surveys* (2021), pp. 1–33.

[41] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A. Lozano. "A review on outlier/anomaly detection in time series data". In: *arXiv:2002.04236 [cs, stat]* (2020).

[42] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. "A training algorithm for optimal margin classifiers". In: *Proceedings of the fifth annual workshop on Computational learning theory*. 1992, pp. 144–152.

[43] Raouf Boutaba, Mohammad A. Salahuddin, Noura Limam, Sara Ayoubi, Nashid Shahriar, Felipe Estrada-Solano, and Oscar M. Caicedo. "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities". In: *Journal of Internet Services and Applications* (2018), p. 16.

[44] Benamar Bouyeddou, Fouzi Harrou, Ying Sun, and Benamar Kadri. "An Effective Network Intrusion Detection Using Hellinger Distance-Based Monitoring Mechanism". In: *2018 International Conference on Applied Smart Systems (ICASS)*. 2018, pp. 1–6.

[45] Benamar Bouyeddou, Benamar Kadri, Fouzi Harrou, and Ying Sun. "Nonparametric Kullback-Leibler distance-based method for networks intrusion detection". In: *2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI)*. 2020, pp. 1–5.

[46] Mohammad Braei and Sebastian Wagner. "Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art". In: *arXiv:2004.00433 [cs, stat]* (2020).

[47] D. Brauckhoff, K. Salamatian, and M. May. "Applying PCA for Traffic Anomaly Detection: Problems and Solutions". In: *IEEE INFOCOM 2009*. 2009, pp. 2866–2870.

[48] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. "LOF: Identifying Density-Based Local Outliers". In: (), p. 12.

[49] Olivier Brun, Yonghua Yin, Javier Augusto-Gonzalez, Manuel Ramos, and Erol Gelenbe. "IoT Attack Detection with Deep Learning". In: (), p. 11.

[50] Olivier Brun, Yonghua Yin, and Erol Gelenbe. "Deep Learning with Dense Random Neural Network for Detecting Attacks against IoT-connected Home Environments". In: *FNC/MobiSPC*. 2018.

[51] Saikiran Bulusu, Bhavya Kailkhura, Bo Li, Pramod K. Varshney, and Dawn Song. "Anomalous Example Detection in Deep Learning: A Survey". In: *arXiv:2003.06979 [cs, stat]* (2021).

[52] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. “A Singular Value Thresholding Algorithm for Matrix Completion”. In: *SIAM Journal on Optimization* (2010), pp. 1956–1982.

[53] N. A. Campbell. “Robust Procedures in Multivariate Analysis I: Robust Covariance Estimation”. In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* (1980), pp. 231–237.

[54] Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. “Robust principal component analysis?” In: *Journal of the ACM* (2011), pp. 1–37.

[55] Li Cao, Yong Cai, and Yinggao Yue. “Data Fusion Algorithm for Heterogeneous Wireless Sensor Networks Based on Extreme Learning Machine Optimized by Particle Swarm Optimization”. In: *Journal of Sensors* (2020), pp. 1–17.

[56] Ander Carreño, Iñaki Inza, and Jose A. Lozano. “Analyzing rare event, anomaly, novelty and outlier detection terms under the supervised classification framework”. In: *Artificial Intelligence Review* (2019).

[57] Raghavendra Chalapathy and Sanjay Chawla. “Deep Learning for Anomaly Detection: A Survey”. In: *arXiv:1901.03407 [cs, stat]* (2019).

[58] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. “Anomaly Detection using One-Class Neural Networks”. In: (2018), p. 10.

[59] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. “Anomaly Detection using One-Class Neural Networks”. In: *arXiv:1802.06360 [cs, stat]* (2019).

[60] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. “Robust, Deep and Inductive Anomaly Detection”. In: *Machine Learning and Knowledge Discovery in Databases*. 2017, pp. 36–51.

[61] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. “Robust, deep and inductive anomaly detection”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2017, pp. 36–51.

[62] V. Chandola, A. Banerjee, and V. Kumar. “Anomaly Detection for Discrete Sequences: A Survey”. In: *IEEE Transactions on Knowledge and Data Engineering* (2012), pp. 823–839.

[63] Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Anomaly detection: A survey”. In: *ACM Computing Surveys* (2009), pp. 1–58.

[64] Sneha Chaudhari, Varun Mithal, Gungor Polatkan, and Rohan Ramanath. “An Attentive Survey of Attention Models”. In: *ACM Transactions on Intelligent Systems and Technology* (2021), pp. 1–32.

[65] Yunhong Chen and Shuming Li. “A Lightweight Anomaly Detection Method Based on SVDD for Wireless Sensor Networks”. In: *Wireless Personal Communications* (2019), pp. 1235–1256.

[66] Zhaomin Chen, Chai Kiat Yeo, Bu Sung Lee, and Chiew Tong Lau. “Autoencoder-based network anomaly detection”. In: *2018 Wireless Telecommunications Symposium (WTS)*. 2018, pp. 1–5.

[67] Clarence Chio and David Freeman. *Machine Learning and Security: Protecting Systems With Data and Algorithms*. 2018.

[68] Hyunseung Choi, Mintae Kim, Gyubok Lee, and Wooju Kim. “Unsupervised Learning Approach for Network Intrusion Detection System Using Autoencoders”. In: *J. Supercomput.* (2019), pp. 5597–5621.

[69] Kukjin Choi, Jihun Yi, Changhwa Park, and Sungroh Yoon. "Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines". In: *IEEE Access* (2021), pp. 120043–120065.

[70] Penny Chong, Lukas Ruff, Marius Kloft, and Alexander Binder. "Simple and effective prevention of mode collapse in deep one-class classification". In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–9.

[71] Benoit Claise, Brian Trammell, and Paul Aitken. "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information". In: *RFC* (2013), pp. 1–76.

[72] *Commtouch Antivirus for Embedded OS Datasheet*. URL: http://www.commtouch.com/uploads/pdf/CommtouchAntivirus-%20for-%20Embedded-%20OS-%20Datasheet.pdf.

[73] Paul Compagnon. "Sequence metric learning: Application to human activity recognition". PhD thesis. Université de Lyon, 2021.

[74] Andrew A. Cook, Göksel Mısırlı, and Zhong Fan. "Anomaly Detection for IoT Time-Series Data: A Survey". In: *IEEE Internet of Things Journal* (2020), pp. 6481–6494.

[75] *Count Encoder — Category Encoders 2.4.1 documentation*. URL: https://contrib.scikit-learn.org/category_encoders/count.html.

[76] KDD Cup. "http://kdd. ics. uci. edu/databases/kddcup99/kddcup99. html". In: *The UCI KDD Archive* (1999).

[77] Angela Daly. "The legality of deep packet inspection". In: *International Journal of Communications Law & Policy* (2011).

[78] F. De la Torre and M.J. Black. "Robust principal component analysis for computer vision". In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. 2001, 362–369 vol.1.

[79] Luca Deri and Daniele Sartiano. "Using DPI and Statistical Analysis in Encrypted Network Traffic Monitoring". In: *International Journal for Information Security Research* (2020).

[80] *Détection de Fraudes par Autoencodeur Variationnel entraîné sur Google AI Platform - Publicis Sapient Engineering - Engineering Done Right*. URL: https://blog.engineering.publicissapient.fr/2020/12/09/detection-de-fraudes-par-autoencodeur-variationnel-entraine-sur-google-ai-platform/.

[81] Dandan Ding, Zhan Ma, Di Chen, Qingshuang Chen, Zoe Liu, and Fengqing Zhu. "Advances in Video Compression System Using Deep Neural Network: A Review and Case Studies". In: *Proceedings of the IEEE* (2021), pp. 1494–1520.

[82] Meimei Ding and Hui Tian. "PCA-based network Traffic anomaly detection". In: *Tsinghua Science and Technology* (2016), pp. 500–509.

[83] Laurent Dinh, David Krueger, and Yoshua Bengio. "Nice: Non-linear independent components estimation". In: *arXiv preprint arXiv:1410.8516* (2014).

[84] Keval Doshi and Yasin Yilmaz. "Rethinking video anomaly detection-a continual learning approach". In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2022, pp. 3961–3970.

[85] R. Doshi, N. Apthorpe, and N. Feamster. "Machine Learning DDoS Detection for Consumer Internet of Things Devices". In: *2018 IEEE Security and Privacy Workshops (SPW)*. 2018, pp. 29–35.

[86] Rohan Doshi, Noah Apthorpe, and Nick Feamster. "Machine Learning DDoS Detection for Consumer Internet of Things Devices". In: *2018 IEEE Security and Privacy Workshops (SPW)*. 2018, pp. 29–35.

[87] Juliette Dromard and Philippe Owezarski. “Study and Evaluation of Unsupervised Algorithms Used in Network Anomaly Detection”. In: *Proceedings of the Future Technologies Conference (FTC) 2019*. 2020, pp. 397–416.

[88] Min Du, Zhi Chen, Chang Liu, Rajvardhan Oak, and Dawn Song. “Lifelong anomaly detection through unlearning”. In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 2019, pp. 1283–1297.

[89] Philipp Dufter, Martin Schmitt, and Hinrich Schütze. “Position information in transformers: An overview”. In: *Computational Linguistics* (2021), pp. 1–31.

[90] Sandra G Dykes. “Poster: An Extreme Value Theory Approach to Anomaly Detection (EVT-AD)”. In: (), p. 2.

[91] Reda Elbasiony, Elsayed A. Sallam, Tarek E. Eltobely, and Mahmoud M. Fahmy. “A hybrid network intrusion detection framework based on random forests and weighted k-means”. In: *Ain Shams Engineering Journal* (2013), pp. 753–762.

[92] Gilberto Fernandes, Joel J. P. C. Rodrigues, Luiz Fernando Carvalho, Jalal F. Al-Muhtadi, and Mario Lemes Proença. “A comprehensive survey on network anomaly detection”. In: *Telecommunication Systems* (2019), pp. 447–489.

[93] Tharindu Fernando, Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. “Deep Learning for Medical Anomaly Detection – A Survey”. In: *arXiv:2012.02364 [cs, eess, stat]* (2021).

[94] David Freedman and Persi Diaconis. “On the histogram as a density estimator: L 2 theory”. In: *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete* (1981), pp. 453–476.

[95] Jim Freeman. “Outliers in Statistical Data (3rd edition)”. In: *Journal of the Operational Research Society* (1994), pp. 1034–1035.

[96] Hamza Frihia, Halima Bahi, and Djamel Eddine Mahrougui. “Combination of a DAE-CNN and OC-SVDD for intrusion detection”. In: *International Journal of Computational Systems Engineering* (2021), pp. 239–245.

[97] Hironori Fujisawa and Shinto Eguchi. “Robust parameter estimation with a small bias against heavy contamination”. In: *Journal of Multivariate Analysis* (2008), p. 29.

[98] Futoshi Futami, Issei Sato, and Masashi Sugiyama. “Variational Inference based on Robust Divergences”. In: (), p. 10.

[99] Ibrahim Ghafir, Vaclav Prenosil, Jakub Svoboda, and Mohammad Hammoudeh. “A Survey on Network Security Monitoring Systems”. In: *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*. 2016, pp. 77–82.

[100] Ali A Ghorbani, Wei Lu, and Mahbod Tavallaee. *Network intrusion detection and prevention: concepts and techniques*. 2009.

[101] Amol Ghoting, Srinivasan Parthasarathy, and Matthew Eric Otey. “Fast Mining of Distance-Based Outliers in High-Dimensional Datasets”. In: *Data Min. Knowl. Discov.* (2008), pp. 349–364.

[102] Manfred Gilli and Evis këllezi. “An Application of Extreme Value Theory for Measuring Financial Risk”. In: *Computational Economics* (2006), pp. 207–228.

[103] Laurent Girin, Simon Leglaive, Xiaoyu Bie, Julien Diard, Thomas Hueber, and Xavier Alameda-Pineda. “Dynamical Variational Autoencoders: A Comprehensive Review”. In: *arXiv:2008.12595 [cs, stat]* (2020).

[104] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 580–587.

[105] R. Gnanadesikan and J. R. Kettenring. “Robust Estimates, Residuals, and Outlier Detection with Multiresponse Data”. In: *Biometrics* (1972), pp. 81–124.

[106] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. “Generative Adversarial Nets”. In: (), p. 9.

[107] Rebecca E Grinter, W Keith Edwards, Marshini Chetty, Erika S Poole, Ja-Young Sung, Jeonghwa Yang, Andy Crabtree, Peter Tolmie, Tom Rodden, Chris Greenhalgh, et al. “The ins and outs of home networking: The case for useful and usable domestic networking”. In: *ACM Transactions on Computer-Human Interaction (TOCHI)* (2009), pp. 1–28.

[108] Frank E. Grubbs. “Procedures for Detecting Outlying Observations in Samples”. In: *Technometrics* (1969), pp. 1–21.

[109] Alejandro Guerra-Manzanares, Jorge Medina-Galindo, Hayretdin Bahsi, and Sven Nõmm. “MedBIoT: Generation of an IoT Botnet Dataset in a Medium-sized IoT Network:” in: *Proceedings of the 6th International Conference on Information Systems Security and Privacy*. 2020, pp. 207–218.

[110] D.M. Hawkins. *Identification of Outliers*. Monographs on applied probability and statistics. 1980.

[111] Jeff Hawkins. *A thousand brains: A new theory of intelligence*. 2021.

[112] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[113] Mingshu He, Xiaojuan Wang, Junhua Zhou, Yuanyuan Xi, Lei Jin, and Xinlei Wang. “Deep-feature-based autoencoder network for few-shot malicious traffic detection”. In: *Security and Communication Networks* (2021).

[114] Paul HERMAN. “tcpstat home page”. In: *http://www. frenchfries. net/paul/tcpstat/* (2001).

[115] Dang Hai Hoang and Ha Duong Nguyen. “A PCA-based method for IoT network traffic anomaly detection”. In: *2018 20th International Conference on Advanced Communication Technology (ICACT)*. 2018, pp. 381–386.

[116] R. Hofstede, P. Čeleda, B. Trammell, I. Drago, R. Sadre, A. Sperotto, and A. Pras. “Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX”. In: *IEEE Communications Surveys Tutorials* (2014), pp. 2037–2064.

[117] Md. Ariful Hoque and Barnali Chakraborty. “Anomaly Based Intrusion Detection Systems Using SNMP Data”. In: 2015.

[118] Hong Huang, Hussein Al-Azzawi, and Hajar Barani. “Network Traffic Anomaly Detection”. In: *CoRR* (2014).

[119] Tzu-Kuo Huang and Jeff Schneider. “Learning Hidden Markov Models from Non-sequence Data via Tensor Decomposition”. In: (2013), p. 9.

[120] Peter J. Huber. *Robust Statistics*. 1981.

[121] Peter Hustinx. “Opinion of the European Data Protection Supervisor”. In: *Hentet fra https://edps. europa. eu/sites/edp/files/publication/05-09-26_data_retention_en. pdf* (2005).

[122] Abdalrahman Hwoij, Mouhammd Al-Kasassbeh, and Mustafa A. Al-Fayoumi. “Detecting Network Anomalies using Rule-based machine learning within SNMP-MIB dataset”. In: *ArXiv* (2020).

[123] *IoT under fire: Kaspersky detects more than 100 million attacks on smart devices in H1 2019 | Kaspersky*. URL: https://www.kaspersky.com/about/press-releases/2019_iot-under-fire-kaspersky-detects-more-than-100-million-attacks-on-smart-devices-in-h1-2019.

[124] Jayshree Jha and Leena Ragha. “Intrusion detection system using support vector machine”. In: *International Journal of Applied Information Systems (IJAIS)* (2013), pp. 25–30.

[125] Ian T. Jolliffe. “Principal Component Analysis”. In: *International Encyclopedia of Statistical Science*. 1986.

[126] Jaeyeon Jung, Balachander Krishnamurthy, and Michael Rabinovich. “Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites”. In: *Proceedings of the 11th International Conference on World Wide Web*. 2002, pp. 293–304.

[127] Qifa Ke and Takeo Kanade. “Robust L1 Norm Factorization in the Presence of Outliers and Missing Data by Alternative Convex Programming”. In: *Proceedings of (CVPR) Computer Vision and Pattern Recognition*. 2005, pp. 739–746.

[128] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. “Transformers in vision: A survey”. In: *ACM Computing Surveys (CSUR)* (2021).

[129] Shehroz S. Khan and Michael G. Madden. “One-Class Classification: Taxonomy of Study and Review of Techniques”. In: *The Knowledge Engineering Review* (2014), pp. 345–374.

[130] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. “Survey of intrusion detection systems: techniques, datasets and challenges”. In: *Cybersecur.* (2019), p. 20.

[131] Tung Kieu, Bin Yang, Chenjuan Guo, Christian S Jensen, Yan Zhao, Feiteng Huang, and Kai Zheng. “Robust and Explainable Autoencoders for Unsupervised Time Series Outlier Detection—Extended Version”. In: *arXiv preprint arXiv:2204.03341* (2022).

[132] Masanari Kimura. “Why Mixup Improves the Model Performance”. In: *Artificial Neural Networks and Machine Learning – ICANN 2021*. 2021, pp. 275–286.

[133] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *arXiv:1312.6114 [cs, stat]* (2014).

[134] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *CoRR* (2014).

[135] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. “Improved variational inference with inverse autoregressive flow”. In: *Advances in neural information processing systems* (2016).

[136] Edwin M. Knorr and Raymond T. Ng. “Algorithms for Mining Distance-Based Outliers in Large Datasets”. In: *VLDB*. 1998.

[137] Kevin H. Knuth. “Optimal data-based binning for histograms and histogram-based probability density models”. In: *Digital Signal Processing* (2019), p. 102581.

[138] Pang Wei Koh and Percy Liang. “Understanding black-box predictions via influence functions”. In: *International conference on machine learning*. PMLR. 2017, pp. 1885–1894.

[139] Gaku Kotani and Yuji Sekiya. “Unsupervised Scanning Behavior Detection Based on Distribution of Network Traffic Features Using Robust Autoencoders”. In: *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. 2018, pp. 35–38.

[140] Jakub Kroustek, Vladislav Iliushin, Anna Shirokova, Jan Neduchal, and Martin Hron. “Torii botnet-Not another Mirai variant”. In: *URL: https://blog. avast. com/new-torii-botnet-threat-research* (2018).

[141] Thomas S Kuhn. *The structure of scientific revolutions*. 1970.

[142] Kurniabudi, Deris Stiawan, Darmawijoyo, Mohd Yazid Bin Idris, Alwi M. Bamhdi, and Rahmat Budiarto. "CICIDS-2017 Dataset Feature Analysis With Information Gain for Anomaly Detection". In: *IEEE Access* (2020), pp. 132911–132921.

[143] Chieh-Hsin Lai, Dongmian Zou, and Gilad Lerman. "Robust Subspace Recovery Layer for Unsupervised Anomaly Detection". In: *arXiv:1904.00152 [cs, stat]* (2019).

[144] Kwei-Herng Lai, Daochen Zha, Yue Zhao, Guanchu Wang, Junjie Xu, and Xia Hu. "Revisiting Time Series Outlier Detection: Definitions and Benchmarks". In: (2021), p. 13.

[145] Anukool Lakhina, Mark Crovella, and Christophe Diot. "Diagnosing Network-Wide Traffic Anomalies". In: *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. SIGCOMM '04. Portland, Oregon, USA, 2004, pp. 219–230.

[146] Arash Habibi Lashkari. *ahlashkari/CICFlowMeter*. 2022. URL: https://github.com/ahlashkari/CICFlowMeter.

[147] Alfred Laugros, Alice Caplier, and Matthieu Ospici. "Addressing Neural Network Robustness with Mixup and Targeted Labeling Adversarial Training". In: *Computer Vision – ECCV 2020 Workshops*. 2020, pp. 178–195.

[148] Grégoire Lefebvre and Christophe Garcia. "Learning a bag of features based nonlinear metric for facial similarity". In: *2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance*. 2013, pp. 238–243.

[149] Guoquan Li, Zheng Yan, Yulong Fu, and Hanlu Chen. "Data Fusion for Network Intrusion Detection: A Review". In: *Security and Communication Networks* (2018), pp. 1–16.

[150] Ming Li, Dezhi Han, Dun Li, Han Liu, and Chin-Chen Chang. "MFVT: an anomaly traffic detection method merging feature fusion network and vision transformer architecture". In: *EURASIP Journal on Wireless Communications and Networking* (2022), pp. 1–22.

[151] Shuangli Liao, Jin Li, Yang Liu, Quanxue Gao, and Xinbo Gao. "Robust Formulation for PCA: Avoiding Mean Calculation With L_{2,p}-norm Maximization". In: *Proceedings of the AAAI Conference on Artificial Intelligence* (2018).

[152] Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. "A survey of transformers". In: *CoRR* (2021).

[153] Zhouchen Lin, Minming Chen, and Yi Ma. "The Augmented Lagrange Multiplier Method for Exact Recovery of Corrupted Low-Rank Matrices". In: *ArXiv* (2010).

[154] Duo Liu, Chung-Horng Lung, Ioannis Lambadaris, and Nabil Seddigh. "Network traffic anomaly detection using clustering techniques and performance comparison". In: *2013 26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*. 2013, pp. 1–4.

[155] Ophtek LLC. *4 Real Life Examples of the IoT Being Hacked*. 2018. URL: https://ophtek.com/4-real-life-examples-iot-hacked/.

[156] Weining Lu, Yu Cheng, Cao Xiao, Shiyu Chang, Shuai Huang, Bin Liang, and Thomas Huang. "Unsupervised Sequential Outlier Detection With Deep Architectures". In: *IEEE Transactions on Image Processing* (2017), pp. 4321–4330.

[157] Z Lu and N Sedransk. "Generalized Pareto Mixture Distribution Approach to Network Modeling and Performance Evaluation 1". In: (2021).

[158] Minzhao Lyu, Dainel Sherratt, Arunan Sivanathan, Hassan Habibi Gharakheili, Adam Radford, and Vijay Sivaraman. "Quantifying the reflective DDoS attack capability of household IoT devices". In: *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. 2017, pp. 46–51.

[159] F.Y. Edgeworth M.A. “XLI. On discordant observations”. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* (1887), pp. 364–375.

[160] Pooria Madani and Natalija Vlajic. “Robustness of Deep Autoencoder in Intrusion Detection under Adversarial Contamination”. In: *Proceedings of the 5th Annual Symposium and Bootcamp on Hot Topics in the Science of Security*. HoTSoS ’18. Raleigh, North Carolina, 2018.

[161] Reham Taher El-Maghraby, Nada Mostafa Abd Elazim, and Ayman M. Bahaa-Eldin. “A survey on deep packet inspection”. In: *2017 12th International Conference on Computer Engineering and Systems (ICCES)*. 2017, pp. 188–197.

[162] Abdun Naser Mahmood, Christopher Leckie, Jiankun Hu, Zahir Tari, and Mohammed Atiquzzaman. “Network Traffic Analysis and SCADA Security”. In: *Handbook of Information and Communication Security* (), pp. 383–405.

[163] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. “Adversarial Autoencoders”. In: *arXiv:1511.05644 [cs]* (2016).

[164] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. “LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection”. In: *arXiv:1607.00148 [cs, stat]* (2016).

[165] Larry M Manevitz and Malik Yousef. “One-class SVMs for document classification”. In: *Journal of machine Learning research* (2001), pp. 139–154.

[166] Marie-Helen Maras. “Internet of Things: security and privacy implications”. In: *International Data Privacy Law* (2015), p. 99.

[167] Daniel L Marino, Chathurika S Wickramasinghe, Craig Rieger, and Milos Manic. “Self-Supervised and Interpretable Anomaly Detection using Network Transformers”. In: *arXiv preprint arXiv:2202.12997* (2022).

[168] Daniel L. Marino, Chathurika S. Wickramasinghe, Craig Rieger, and Milos Manic. “Data-driven Stochastic Anomaly Detection on Smart-Grid communications using Mixture Poisson Distributions”. In: *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*. 2019, pp. 5855–5861.

[169] A.K. Marnerides, A. Schaeffer-Filho, and A. Mauthe. “Traffic anomaly diagnosis in Internet backbone networks: A survey”. In: *Computer Networks* (2014), pp. 224–243.

[170] Artur Marzano, David Alexander, Osvaldo Fonseca, Elverton Fazzion, Cristine Hoepers, Klaus Steding-Jessen, Marcelo HPC Chaves, Ítalo Cunha, Dorgival Guedes, and Wagner Meira. “The evolution of bashlite and mirai iot botnets”. In: *2018 IEEE Symposium on Computers and Communications (ISCC)*. IEEE. 2018, pp. 00813–00818.

[171] Marc Masana, Idoia Ruiz, Joan Serrat, Joost van de Weijer, and Antonio M. Lopez. “Metric Learning for Novelty and Anomaly Detection”. In: *arXiv:1808.05492 [cs]* (2018).

[172] Douglas Mauro and Kevin Schmidt. *Essential SNMP: Help for System and Network Administrators*. 2005.

[173] Johan Mazel. “Unsupervised network anomaly detection”. In: (), p. 131.

[174] Warren S McCulloch and Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. In: *The bulletin of mathematical biophysics* (1943), pp. 115–133.

[175] Christopher D. McDermott, Farzan Majdani, and Andrei V. Petrovski. “Botnet Detection in the Internet of Things using Deep Learning Approaches”. In: *2018 International Joint Conference on Neural Networks (IJCNN)*. 2018, pp. 1–8.

[176] Yair Meidan, Michael Bohadana, Yael Mathov, Yisroel Mirsky, Asaf Shabtai, Dominik Breitenbacher, and Yuval Elovici. "N-baiot—network-based detection of iot botnet attacks using deep autoencoders". In: *IEEE Pervasive Computing* (2018), pp. 12–22.

[177] Lotfi Mhamdi, Desmond McLernon, Fadi El-moussa, Syed Ali Raza Zaidi, Mounir Ghogho, and Tuan Tang. "A Deep Learning Approach Combining Autoencoder with One-class SVM for DDoS Attack Detection in SDNs". In: *2020 IEEE Eighth International Conference on Communications and Networking (ComNet)*. 2020, pp. 1–6.

[178] Carlos Mougan, David Masip, Jordi Nin, and Oriol Pujol. "Quantile Encoder: Tackling High Cardinality Categorical Features in Regression Problems". In: *Modeling Decisions for Artificial Intelligence*. 2021, pp. 168–180.

[179] Nour Moustafa, Marwa Keshky, Essam Debiez, and Helge Janicke. "Federated TON_IoT Windows Datasets for Evaluating AI-Based Security Applications". In: *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. 2020, pp. 848–855.

[180] Mary M. Moya and Don R. Hush. "Network constraints and multi-objective optimization for one-class classification". In: *Neural Networks* (1996), pp. 463–474.

[181] Naji Najari, Samuel Berlemont, and Grégoire Lefebvre. *Anomaly Detection for the Applicative Maintenance of IoT Devices.* FR2205936, 2022.

[182] Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "Network Traffic Modeling For IoT-device Re-identification". In: *2020 International Conference on Omni-layer Intelligent Systems (COINS)*. 2020, pp. 1–6.

[183] Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "RADON: Robust Autoencoder for Unsupervised Anomaly Detection". In: *2021 14th International Conference on Security of Information and Networks (SIN)*. 2021, pp. 1–8.

[184] Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "RESIST: Robust Transformer for Unsupervised Time Series Anomaly Detection". In: *7th Workshop on Advanced Analytics and Learning on Temporal Data (AALTD) of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*. 2022, p. 16.

[185] Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "Robust Unsupervised Time Series Anomaly Detection with Transformers". In: *Neurocomputing (under submission)*. 2022.

[186] Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "Robust Variational Autoencoders and Normalizing Flows for Unsupervised Network Anomaly Detection". In: *International Conference on Advanced Information Networking and Applications (AINA)*. Springer. 2022, pp. 281–292.

[187] Naji Najari, Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. "Robust Variational Autoencoders and Normalizing Flows for Unsupervised Network Anomaly Detection". In: *Advanced Information Networking and Applications*. 2022, pp. 281–292.

[188] Ghazi Al-Naymat, Mouhammd Al-Kasassbeh, and Eshraq Al-Hawari. "Exploiting SNMP-MIB Data to Detect Network Anomalies using Machine Learning Techniques". In: *ArXiv* (2018).

[189] *New report finds over 90% of IoT device transactions are unencrypted*. URL: https://www.perle.com/articles/new-report-finds-over-90-of-iot-device-transactions-are-unencrypted-40185159.shtml.

[190] *NFStream - a Flexible Network Data Analysis Framework*. URL: https://nfstream.org/.

[191] Andrew Ng et al. "Sparse autoencoder". In: *CS294A Lecture notes* (2011), pp. 1–19.

[192] Huy Nguyen and Deokjai Choi. "Network anomaly detection: Flow-based or packet-based approach?" In: *CoRR* (2010).

[193] Quoc Phong Nguyen, Kar Wai Lim, Dinil Mon Divakaran, Kian Hsiang Low, and Mun Choon Chan. "GEE: A Gradient-based Explainable Variational Autoencoder for Network Anomaly Detection". In: *2019 IEEE Conference on Communications and Network Security (CNS)*. 2019, pp. 91–99.

[194] Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Hossein Fereidooni, N. Asokan, and Ahmad-Reza Sadeghi. "D\"IoT: A Federated Self-learning Anomaly Detection System for IoT". In: *arXiv:1804.07474 [cs]* (2019).

[195] Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Hossein Fereidooni, N. Asokan, and Ahmad-Reza Sadeghi. "DÏoT: A Federated Self-learning Anomaly Detection System for IoT". In: *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. 2019, pp. 756–767.

[196] *NSL-KDD | Datasets | Research | Canadian Institute for Cybersecurity | UNB*. URL: https://www.unb.ca/cic/datasets/nsl.html.

[197] Tae-Hyun Oh, Yu-Wing Tai, Jean Charles Bazin, Hyeongwoo Kim, and In-So Kweon. "Partial Sum Minimization of Singular Values in Robust PCA: Algorithm and Applications". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2016), pp. 744–758.

[198] Poojan Oza and Vishal M. Patel. "One-Class Convolutional Neural Network". In: *IEEE Signal Processing Letters* (2019), pp. 277–281.

[199] Randy Clinton Paffenroth, Kathleen Kay, and Leslie D. Servi. "Robust PCA for Anomaly Detection in Cyber Networks". In: *ArXiv* (2018).

[200] Randy Clinton Paffenroth and Chong Zhou. "Modern Machine Learning for Cyber-Defense and Distributed Denial-of-Service Attacks". In: *IEEE Engineering Management Review* (2019), pp. 80–85.

[201] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. "Deep Learning for Anomaly Detection: A Review". In: *ACM Computing Surveys* (2021), pp. 1–38.

[202] Guansong Pang, Cheng Yan, Chunhua Shen, Anton van den Hengel, and Xiao Bai. "Self-Trained Deep Ordinal Regression for End-to-End Video Anomaly Detection". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 12170–12179.

[203] Emanuel Parzen. "On Estimation of a Probability Density Function and Mode". In: *Annals of Mathematical Statistics* (1962), pp. 1065–1076.

[204] Nerijus Paulauskas and Azuolas Faustas Bagdonas. "Local outlier factor use for the network flow anomaly detection". In: *Secur. Commun. Networks* (2015), pp. 4203–4212.

[205] Felisberto Pereira, Ricardo Correia, Pedro Pinho, Sérgio I Lopes, and Nuno Borges Carvalho. "Challenges in resource-constrained IoT devices: Energy and communication as critical success factors for future IoT deployment". In: *Sensors* (2020), p. 6420.

[206] Pramuditha Perera, Poojan Oza, and Vishal M. Patel. "One-Class Classification: A Survey". In: *CoRR* (2021).

[207] Adrian Alan Pol, Victor Berger, Cecile Germain, Gianluca Cerminara, and Maurizio Pierini. "Anomaly Detection with Conditional Variational Autoencoders". In: *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. 2019, pp. 1651–1657.

[208] *Proofpoint Uncovers Internet of Things (IoT) Cyberattack | Proofpoint US*. 2014. URL: https://www.proofpoint.com/us/proofpoint-uncovers-internet-things-iot-cyberattack.

[209] Yinghui Quan, Yingping Tong, Wei Feng, Gabriel Dauphin, Wenjiang Huang, and Mengdao Xing. "A Novel Image Fusion Method of Multi-Spectral and SAR Images for Land Cover Classification". In: *Remote Sensing* (2020), p. 3801.

[210] J. Ross Quinlan. "Induction of Decision Trees". In: *Machine Learning* (2004), pp. 81–106.

[211] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. "Efficient Algorithms for Mining Outliers from Large Data Sets". In: *SIGMOD Rec.* (2000), pp. 427–438.

[212] Ambareesh Ravi, Xiaozhuo Yu, Iara Santelices, Fakhri Karray, and Baris Fidan. "General Frameworks for Anomaly Detection Explainability: Comparative Study". In: *2021 IEEE International Conference on Autonomous Systems (ICAS)*. 2021, pp. 1–5.

[213] Ambareesh Ravi, Xiaozhuo Yu, Iara Santelices, Fakhri Karray, and Baris Fidan. "General Frameworks for Anomaly Detection Explainability: Comparative Study". In: *2021 IEEE International Conference on Autonomous Systems (ICAS)*. IEEE. 2021, pp. 1–5.

[214] Francesco Restuccia, Salvatore D'oro, and Tommaso Melodia. "Securing the Internet of Things in the Age of Machine Learning and Software-Defined Networking". In: *IEEE Internet of Things Journal* (2018), pp. 4829–4842.

[215] Danilo Jimenez Rezende and Shakir Mohamed. "Variational Inference with Normalizing Flows". In: *ICML*. 2015.

[216] Danilo Jimenez Rezende and Shakir Mohamed. "Variational Inference with Normalizing Flows". In: *arXiv:1505.05770 [cs, stat]* (2016).

[217] Haakon Ringberg, Augustin Soule, Jennifer Rexford, and Christophe Diot. "Sensitivity of PCA for traffic anomaly detection". In: *ACM SIGMETRICS Performance Evaluation Review* (2007), pp. 109–120.

[218] "Robust principal component analysis: A factorization-based approach with linear complexity". In: *Information Sciences* (2020), pp. 581–599.

[219] Paula Roquero and Javier Aracil. "On Performance and Scalability of Cost-Effective SNMP Managers for Large-Scale Polling". In: *IEEE Access* (2021), pp. 7374–7383.

[220] Paul L. Rosin. "Unimodal thresholding". In: *Pattern Recognition* (2001), pp. 2083–2096.

[221] Peter J. Rousseeuw and Annick M. Leroy. *Robust regression and outlier detection*. Wiley series in probability and mathematical statistics. 1987.

[222] Lukas Ruff, Jacob R. Kauffmann, Robert A. Vandermeulen, Gregoire Montavon, Wojciech Samek, Marius Kloft, Thomas G. Dietterich, and Klaus-Robert Muller. "A Unifying Review of Deep and Shallow Anomaly Detection". In: *Proceedings of the IEEE* (2021), pp. 1–40.

[223] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. "Deep one-class classification". In: *International conference on machine learning*. PMLR. 2018, pp. 4393–4402.

[224] Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. "DEEP SEMI-SUPERVISED ANOMALY DETECTION". In: (2020), p. 23.

[225] Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Lucas Deecke, Shoaib A Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. "Deep One-Class Classification". In: (), p. 10.

[226] Bardia Safaei, Amir Mahdi Hosseini Monazzah, Milad Barzegar Bafroei, and Alireza Ejlali. "Reliability side-effects in Internet of Things application layer protocols". In: *2017 2nd International Conference on System Reliability and Safety (ICSRS)*. IEEE. 2017, pp. 207–212.

[227] Mahdi Saleh. "Internet Protocol (IP) Addressing". In: (), p. 96.

[228] Mahsa Salehi and Lida Rashidi. "A Survey on Anomaly detection in Evolving Data: [with Application to Forest Fire Risk Prediction]". In: *ACM SIGKDD Explorations Newsletter* (2018), pp. 13–23.

[229] Mohammadreza Salehi, Hossein Mirzaei, Dan Hendrycks, Yixuan Li, Mohammad Hossein Rohban, and Mohammad Sabokrou. "A Unified Survey on Anomaly, Novelty, Open-Set, and Out-of-Distribution Detection: Solutions and Future Challenges". In: *arXiv:2110.14051 [cs]* (2021).

[230] Cátia M. Salgado, Carlos Azevedo, Hugo Proença, and Susana M. Vieira. "Noise Versus Outliers". In: *Secondary Analysis of Electronic Health Records*. 2016, pp. 163–183.

[231] Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. *Evaluating Standard Feature Sets Towards Increased Generalisability and Explainability of ML-based Network Intrusion Detection*. 2021. URL: http://arxiv.org/abs/2104.07183.

[232] Florian Scheidegger, Luca Benini, Costas Bekas, and A Cristiano I Malossi. "Constrained deep neural network architecture search for IoT devices accounting for hardware calibration". In: *Advances in Neural Information Processing Systems* (2019).

[233] Bernhard Scholkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. "Support Vector Method for Novelty Detection". In: (), p. 8.

[234] David W Scott. "On optimal and data-based histograms". In: *Biometrika* (1979), pp. 605–610.

[235] Jonas Herskind Sejr and Anna Schneider-Kamp. "Explainable outlier detection: What, for Whom and Why?" In: *Machine Learning with Applications* (2021), p. 100172.

[236] Naeem Seliya, Azadeh Abdollah Zadeh, and Taghi M Khoshgoftaar. "A literature review on one-class classification and its potential applications in big data". In: *Journal of Big Data* (2021), pp. 1–31.

[237] Iman Sharafaldin, Arash Habib Lashkar, and Ali Ghorbani. *Intrusion Detection Evaluation Dataset (CICIDS2017), Canadian Institute for Cybersecurity | UNB*. 2017. URL: https://www.unb.ca/cic/datasets/ids-2017.html.

[238] Lifeng Shen, Zhuocong Li, and James T Kwok. "Timeseries Anomaly Detection using Temporal Hierarchical One-Class Network". In: (), p. 11.

[239] Xiaoping Shen and Sonali Agrawal. "Kernel Density Estimation for An Anomaly Based Intrusion Detection System." In: 2006, pp. 161–167.

[240] Alban Siffer, Pierre-Alain Fouque, Alexandre Termier, and Christine Largouet. "Anomaly Detection in Streams with Extreme Value Theory". In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2017, pp. 1067–1075.

[241] Arunan Sivanathan. "IoT Behavioral Monitoring via Network Traffic Analysis". In: *CoRR* (2020).

[242] Arunan Sivanathan, Hassan Habibi Gharakheili, Franco Loi, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics". In: *IEEE Transactions on Mobile Computing* (2019), pp. 1745–1759.

[243] *Snort - Network Intrusion Detection & Prevention System*. URL: https://www.snort.org/.

[244] Youngrok Song, Sangwon Hyun, and Yun-Gyung Cheong. "Analysis of autoencoders for network intrusion detection". In: *Sensors* (2021), p. 4294.

[245] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. "Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 2828–2837.

[246] Piotr Sulewski. "Equal-bin-width histogram versus equal-bin-count histogram". In: *Journal of Applied Statistics* (2020), pp. 1–20.

[247] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems* (2014).

[248] K. Takeuchi. "Nonparametric Statistics: Asymptotics". In: *International Encyclopedia of the Social & Behavioral Sciences*. 2001, pp. 10667–10673.

[249] Hao Tan and Mohit Bansal. "LXMERT: Learning Cross-Modality Encoder Representations from Transformers". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 2019, pp. 5099–5110.

[250] Mengxuan Tan, Alfonso Iacovazzi, Ngai-Man Man Cheung, and Yuval Elovici. "A Neural Attention Model for Real-Time Network Intrusion Detection". In: *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. 2019, pp. 291–299.

[251] David M.J. Tax and Robert P.W. Duin. "Support Vector Data Description". In: *Machine Learning* (2004), pp. 45–66.

[252] David Martinus Johannes Tax. "One-class classification: Concept learning in the absence of counter-examples." In: (2002).

[253] *The Internet of Things and the consequences of downtime*. 2016. URL: https://www.information-age.com/internet-things-consequences-downtime-123463189/.

[254] *The Internet of Things at the IETF*. URL: https://www.ietf.org/topics/iot/.

[255] *The Zeek Network Security Monitor*. URL: https://zeek.org/.

[256] Quang-Anh Tran, Haixin Duan, and Xing Li. "One-class support vector machine for anomaly network traffic detection". In: *China Education and Research Network (CERNET), Tsinghua University, Main Building* (2004).

[257] Quang-Anh Tran, Qianli Zhang, and Xing Li. "Evolving training model method for one-class SVM". In: *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme-System Security and Assurance (Cat. No. 03CH37483)*. IEEE. 2003, pp. 2388–2393.

[258] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. *TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data*. 2022. URL: http://arxiv.org/abs/2201.07284.

[259] Imtiaz Ullah and Qusay H Mahmoud. "A scheme for generating a dataset for anomalous activity detection in iot networks". In: *Canadian Conference on Artificial Intelligence*. Springer. 2020, pp. 508–520.

[260] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is All you Need". In: (), p. 11.

[261] Namrata Vaswani and Praneeth Narayanamurthy. "Static and Dynamic Robust PCA and Matrix Completion: A Review". In: *Proceedings of the IEEE* (2018), pp. 1359–1379.

[262] Eduardo S. C. Vilaça, Thiago Pereira de Brito Vieira, Rafael Timóteo de Sousa, and João Paulo Carvalho Lustosa Da Costa. "Botnet traffic detection using RPCA and Mahalanobis Distance". In: *2019 Workshop on Communication Networks and Power Systems (WCNPS)* (2019), pp. 1–6.

[263] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. "Extracting and composing robust features with denoising autoencoders". In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 1096–1103.

[264] Hongzhi Wang, Mohamed Jaward Bah, and Mohamed Hammad. "Progress in Outlier Detection Techniques: A Survey". In: (2019), p. 38.

[265] Mowei Wang, Yong Cui, Xin Wang, Shihan Xiao, and Junchen Jiang. "Machine Learning for Networking: Workflow, Advances and Opportunities". In: *IEEE Network* (2018), pp. 92–99.

[266] Wei Wang and Sylvain Gombault. "Distance Measures for Anomaly Intrusion Detection". In: *SAM'07: the 2007 International Conference on Security and Management*. 2007, pp. 17–23.

[267] Xiaochun Wang, Xia Li Wang, Yongqiang Ma, and D Mitchell Wilkes. "A fast MST-inspired kNN-based outlier detection method". In: *Information Systems* (2015), pp. 89–112.

[268] Xixuan Wang, Dechang Pi, Xiangyan Zhang, Hao Liu, and Chang Guo. "Variational transformer-based anomaly detection approach for multivariate time series". In: *Measurement* (2022), p. 110791.

[269] Xixuan Wang, Dechang Pi, Xiangyan Zhang, Hao Liu, and Chang Guo. "Variational transformer-based anomaly detection approach for multivariate time series". In: *Measurement* (2022), p. 110791.

[270] Muhammad Waseem Khan. "A Survey: Image Segmentation Techniques". In: *International Journal of Future Computer and Communication* (2014), pp. 89–93.

[271] Barry Wellman and Caroline Haythornthwaite. *The Internet in everyday life*. 2008.

[272] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. *Transformers in Time Series: A Survey*. 2022. URL: http://arxiv.org/abs/2202.07125.

[273] Lilian Weng. *Flow-based Deep Generative Models*. 2018. URL: https://lilianweng.github.io/posts/2018-10-13-flow-models/.

[274] *What is OSI Model | 7 Layers Explained | Imperva*. URL: https://www.imperva.com/learn/application-security/osi-model/.

[275] *What's new with the Internet of Things? | McKinsey*. URL: https://www.mckinsey.com/industries/semiconductors/our-insights/whats-new-with-the-internet-of-things.

[276] Frank White. "Data Fusion Lexicon". In: 1991.

[277] Daniel Wood, Noah Apthorpe, and Nick Feamster. "Cleartext Data Transmissions in Consumer IoT Medical Devices". In: *arXiv:1803.10147 [cs]* (2018).

[278] Guest Writer. *The 5 Worst Examples of IoT Hacking and Vulnerabilities in Recorded History*. 2020. URL: https://www.iotforall.com/5-worst-iot-hacking-vulnerabilities/.

[279] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. *Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy*. 2022. URL: http://arxiv.org/abs/2110.02642.

[280] Ke Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention". In: *ICML*. 2015.

[281] Jeonghwa Yang and W Keith Edwards. "A study on network management tools of householders". In: *Proceedings of the 2010 ACM SIGCOMM workshop on Home networks*. 2010, pp. 1–6.

[282] Jeonghwa Yang, W Keith Edwards, and David Haslem. "Eden: supporting home network management through interactive visual tools". In: *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. 2010, pp. 109–118.

[283] Nong Ye and Qiang Chen. "An anomaly detection technique based on a chi-square statistic for detecting intrusions into information systems". In: *Quality and Reliability Engineering International* (2001).

[284] Dit-Yan Yeung and C. Chow. "Parzen-window network intrusion detectors". In: *2002 International Conference on Pattern Recognition*. 2002, 385–388 vol.4.

[285] Dit-Yan Yeung and Calvin Chow. "Parzen-window network intrusion detectors". In: *2002 International Conference on Pattern Recognition*. IEEE. 2002, pp. 385–388.

[286] Tianlong Yu, Vyas Sekar, Srinivasan Seshan, Yuvraj Agarwal, and Chenren Xu. "Handling a trillion (unfixable) flaws on a billion devices: Rethinking network security for the Internet-of-Things". In: *Proceedings of the 14th ACM Workshop on Hot Topics in Networks - HotNets-XIV*. 2015, pp. 1–7.

[287] Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. "Deep Modular Co-Attention Networks for Visual Question Answering". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 6274–6283.

[288] Sultan Zavrak and Murat İskefiyeli. "Anomaly-based intrusion detection from network flow features using variational autoencoder". In: *IEEE Access* (2020), pp. 108346–108358.

[289] *Zero Day Attacks*. URL: https://www.micro.ai/wp-content/uploads/2022/01/MicroAI-zero-day-attacks-2.0.pdf.

[290] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. "Mixup: Beyond Empirical Risk Minimization". In: *ICLR* (2018), p. 13.

[291] Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. "Deep Mutual Learning". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4320–4328.

[292] Chong Zhou and Randy C. Paffenroth. "Anomaly Detection with Robust Deep Autoencoders". In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '17*. 2017, pp. 665–674.

[293] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. "Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection". In: *ICLR* (2018), p. 19.

FOLIO ADMINISTRATIF

THESE DE L'INSA LYON, MEMBRE DE L'UNIVERSITE DE LYON

NOM : NAJARI DATE de SOUTENANCE : 13/12/2022
(avec précision du nom de jeune fille, le cas échéant)

Prénoms : Naji

TITRE : Détection D'anomalies Robuste et Non-supervisée, Appliquée à la Supervision du Trafic Réseau

NATURE : Doctorat Numéro d'ordre : 2022ISAL0124

Ecole doctorale : Infomaths

Spécialité : Informatique

RESUME : Cette thèse étudie la détection non-supervisée et robuste des anomalies à partir du trafic réseau des équipements connectés (Internet of Things, IoT), appliquée à la supervision des objets IoT, cibles de dysfonctionnements et d'attaques généralement inconnues. On explore en particulier l'apprentissage de représentations pour la modélisation de la norme à l'aide des réseaux de neurones artificiels, et en particulier l'architecture autoencodeurs. Pour modéliser la norme, les approches classiques de détection d'anomalies entrainent un réseau autoencodeur à reconstruire les données nominales, en minimisant un critère d'erreur entre les données initiales et celles produites en sortie du réseau. Comme les nouvelles observations anormales sont structurellement différentes, leur reconstruction est accompagnée par une significative perte d'information, avec une large erreur de reconstruction. Toutefois, la constitution d'une base d'apprentissage qui ne contient que des données nominales reste coûteuse, chronophage, et parfois infaisable lorsque l'anomalie n'a pas encore été identifiée par les experts. Ainsi, nous avons cherché à développer des autoencodeurs robustes, c'est-à-dire capables de modéliser la norme, même si la base d'apprentissage est contaminée par des anomalies. En particulier, nous proposons trois contributions. Dans un premier temps, nous proposons d'analyser la distribution des scores de reconstruction afin d'opérer une auto -supervision. Nous estimons dynamiquement des seuils à partir de l'histogramme de reconstruction, afin d'identifier les anomalies de l'ensemble d'apprentissage. Enfin, nous les exploitons pour renforcer le potentiel de discrimination du modèle. Cette contribution a été concrétisée avec RADON (Robust Autoencoder with Dynamic Outlier filteriNg). Dans un deuxième temps, nous proposons d'exploiter la puissance des autoencodeurs variationnels et des normalizing fows pour améliorer le processus d'estimation des anomalies. Le critère de seuillage sur l'histogramme des scores de reconstruction est remplacé par une modélisation statistique grâce à la théorie des valeurs extrêmes. La combinaison de ces contributions aboutit à notre modèle GRAnD (Generative Robust autoencoder for unsupervised Anomaly Detection). Enfin, nous proposons une troisième contribution, RESIST (Robust transformEr developed for unSupervised tIme Series anomaly deTection), s'appuie sur les modèles sequence-to-sequence, et en particulier les Transformeurs, pour modéliser les dépendances temporelles entres les tokens d'une séquence de flux réseaux et détecter toute déviation contextuelle et collective. L'impact des contaminants lors de l'apprentissage est significativement atténué grâce à une architecture Siamoise et une fonction objective robuste.

MOTS-CLÉS : Détection d'Anomalies, Supervision des Equipements Connectés, Analyse du Trafic Réseau, Apprentissage Automatique, Apprentissage Robuste et Non-supervisé, Apprentissage Auto-supervisé, Réseaux de Neurones Artficiels, AutoEncodeurs, Autoencodeurs Variationnels, Normalizing Flows, Transformeurs, Seuillage Unimodal, Théorie des Valeurs Extrêmes.

Laboratoire (s) de recherche : LIRIS

Directeur de thèse: Christophe GARCIA

Président de jury :

Composition du jury :

Rapporteur	OWEZARSKI, Philippe	Directeur de Recherche - HDR, LAAS-CNRS
Rapporteure	VINCENT, Nicole	Professeur, Université de Paris
Examinateur	MALLAT, Stéphane	Professeur, Collège de France, Président
Examinateur	CHATEAU, Thierry	Professeur, Université de Clermont Auvergne
Examinatrice	DEVIJVER, Emilie	Chercheure, CNRS
Directeur	GARCIA, Christophe	Professeur, INSA de Lyon
Co-directeur	DUFFNER, Stefan	Maître de Conférences, INSA de Lyon
Co- encadrant	LEFEBVRE, Grégoire	Chercheur, Orange Labs
Co- encadrant	BERLEMONT, Samuel	Chercheur, Orange Labs