DISSERTATION

A HYBRID FRAMEWORK FOR TIME-SERIES ANALYSIS

From Anomaly Detection to Uncertainty Estimation and Explainability

Thesis approved by the
Department of Computer Science
of the TU Kaiserslautern
for the award of the Doctoral Degree

DOCTOR OF ENGINEERING
(DR.-ING.)

to

Mohsin Munir

Date of the viva: 19-08-2021
Dean: Prof. Dr. Jens Schmitt

Reviewers:
Prof. Dr. Prof. h.c. Andreas Dengel
Prof. Dr. Seiichi Uchida (Distinguished Professor of Kyushu University, Japan)

TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

D 386

Mohsin Munir: *A Hybrid Framework for Time-series Analysis* – From Anomaly Detection to Uncertainty Estimation and Explainability

SUPERVISOR:
Prof. Dr. Prof. h.c. Andreas Dengel

CONTACT INFORMATION:
mohsin.munir@dfki.de

To my dearest parents,
Kouser Munir & Munir Ahmed

ABSTRACT

Data is the new gold and serves as a key to answer the five W's (Who, What, Where, When, Why) and How's of any business. Companies are now mining data more than ever and one of the most important aspects while analyzing this data is to detect anomalous patterns to identify critical patterns and points. To tackle the vital aspects of time-series analysis, this thesis presents a novel hybrid framework that stands on three pillars: Anomaly Detection, Uncertainty Estimation, and Interpretability and Explainability.

The first pillar is comprised of contributions in the area of time-series anomaly detection. *Deep Anomaly Detection for Time-series (DeepAnT)*, a novel deep learning-based anomaly detection method, lies at the foundation of the proposed hybrid framework and addresses the inadequacy of traditional anomaly detection methods. To the best of the author's knowledge, Convolutional Neural Network (CNN) was used for the first time in Deep Anomaly Detection for Time-series (DeepAnT) to robustly detect multiple types of anomalies in the tricky and continuously changing time-series data. To further improve the anomaly detection performance, a fusion-based method, Fusion of Statistical and Deep Learning for Anomaly Detection (FuseAD) is proposed. This method aims to combine the strengths of existing well-founded statistical methods and powerful data-driven methods.

In the second pillar of this framework, a hybrid approach that combines the high accuracy of the deterministic models with the posterior distribution approximation of Bayesian neural networks is proposed. Deterministic deep learning models are unarguably one of the best choices for classification. However, they don't provide any information that how confident they are about their decision. These confident decisions are required in the domains where critical decisions are based on network decisions.

In the third pillar of the proposed framework, mechanisms to enable both *HOW* and *WHY* parts are presented. Due to the black-box nature of deep learning methods, their acceptability is rather low. The trustworthiness of a system can be achieved by enabling it to answer *HOW* and *WHY* of a decision. *Time-series Visualization framework (TSViz)*, a visualization tool that demystifies convolutional deep learning models for time-series data contributes to the *HOW* part. This tool highlights the parts of the input that are influential to a network's decision. However, *Time-series Explanation framework (TSXplain)* contributes to the *WHY* part by generating natural language explanations of the decisions made by a Deep Neural Network (DNN).

ACKNOWLEDGMENTS

First of all, I would like to express my gratitude to the Almighty, who has enabled me to write this thesis and always helped me in this laborious journey.

A special thanks to my supervisor Prof. Dr. Andreas Dengel, you have always been very supportive and encouraging that I feel so blessed to be part of your research group. I always found you standing by my side through thick and thin. I would also like to thank you for giving me different opportunities and trusting in me, which eventually bought the best out of me and made me a better researcher and individual. You have always been a great source of inspiration for me.

This thesis would have not been possible without the continuous support of my mentors and elder brothers, Dr. Sheraz Ahmed and Dr. Muhammad Imran Malik. You have always guided and supported me through the most difficult times. I have learned a lot from the technical guidance and the discussions related to other life matters. I will always be in your debt. Thank you for being such firm support for me.

A special thanks to my best friend, Muhammad Ali Mubeen. In our life, we meet many people, but few stay for a lifetime, you are one of them. You are such a gem that I cannot afford to lose you. Keep teaching me different aspects of life. I would also like to thank my close friends: Zain, Mohsin, Adeel, Sajjid, Abdullah, Gulzar, Junaid, Imran, and Tahseen. Your company and support have helped me greatly.

A special thanks to all my colleagues at SDS group, DFKI Kaiserslautern for being nice and supportive to me. I would also like to thank Prof. Seiichi Uchida (Kyushu University, Japan), Takeshi Suzuki (RICOH, Japan), and Steffen Erkel (BOSCH, Germany) for giving me the opportunity to visit your labs, which helped me a lot to broaden my research horizon. The time I spent with you and in your labs, is very cherishable to me.

In last but not least, I would like to wholeheartedly thank my family: my parents, my wife, my sisters, my brother, and most importantly; my nephew, Muhammad Ibrahim – the most valuable assets of my life. I don’t have words to express my love to you. Your sincere prayers and support compelled me to where I am today. Without your love and support, this journey would not have been possible.

— Thank you, Mohsin Munir

GRANTS

This work has been supported by the TU Kaiserslautern CS department's PhD Program, DeFuseNN (Grant 01IW17002), IAV-FLaP, and the NVIDIA AI Lab program (NVAIL).

PUBLICATIONS AS PART OF THIS THESIS

Parts of the research and material (including figures, tables, and algorithms) in this thesis have already been published in:

M. Munir, S. Erkel, A. Dengel, and S. Ahmed. “Pattern-based contextual anomaly detection in hvac systems.” In: *IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE. 2017, pp. 1066–1073. DOI: `10.1109/ICDMW.2017.150`.

M. Munir, S. Baumbach, Y. Gu, A. Dengel, and S. Ahmed. “Data Analytics: Industrial Perspective & Solutions for Streaming Data.” In: *Data Mining in Time Series and Streaming Databases*. Vol. 83. Series in Machine Perception and Artificial Intelligence. World Scientific, Mar. 2018. Chap. 7, pp. 144–168. DOI: `10.1142/9789813228047_0007`.

M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed. “DeepAnT: A deep learning approach for unsupervised anomaly detection in time series.” In: *IEEE Access* 7 (2018), pp. 1991–2005. DOI: `10.1109/ACCESS.2018.2886457`.

S. A. Siddiqui, D. Mercier, M. Munir, A. Dengel, and S. Ahmed. “TSViz: Demystification of deep learning models for time-series analysis.” In: *IEEE Access* 7 (2019), pp. 67027–67040. DOI: `10.1109/ACCESS.2019.2912823`.

M. Munir, S. A. Siddiqui, M. A. Chattha, A. Dengel, and S. Ahmed. “FuseAD: unsupervised anomaly detection in streaming sensors data by fusing statistical and deep learning models.” In: *Sensors* 19.11 (2019), p. 2451. DOI: `10.3390/s19112451`.

M. Munir, S. A. Siddiqui, F. Küsters, D. Mercier, A. Dengel, and S. Ahmed. “TSXplain: Demystification of DNN Decisions for Time-Series using Natural Language and Statistical Features.” In: *International Conference on Artificial Neural Networks*. Springer. 2019, pp. 426–439. URL: `https://link.springer.com/chapter/10.1007/978-3-030-30493-5_43`.

M. A. Chattha, S. A. Siddiqui, M. Munir, M. I. Malik, L. van Elst, A. Dengel, and S. Ahmed. “DeepEX: Bridging the Gap Between Knowledge and Data Driven Techniques for Time Series Forecasting.” In: *International Conference on Artificial Neural Networks*. Springer. 2019, pp. 639–651. URL: `https://link.springer.com/chapter/10.1007/978-3-030-30484-3_51`.

M. Munir, M. A. Chattha, A. Dengel, and S. Ahmed. “A Comparative Analysis of Traditional and Deep Learning-based Anomaly Detection Methods for Streaming Data.” In: *18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE. 2019, pp. 561–566. DOI: `10.1109/ICMLA.2019.00105`.

M. N. Bajwa, S. Khurram, M. Munir, S. A. Siddiqui, M. I. Malik, A. Dengel, and S. Ahmed. “Confident classification using a hybrid between deterministic and probabilistic convolutional neural networks.” In: *IEEE Access* 8 (2020), pp. 115476–115485. DOI: `10.1109/ACCESS.2020.3004409`.

CONTENTS

Publications as Part of this Thesis ix
List of Figures xv
List of Tables xix
Acronyms xxi

I INTRODUCTION

1 INTRODUCTION 3
- 1.1 Motivation 3
- 1.2 Research Questions and Goals 4
- 1.3 Contributions 5
- 1.4 Thesis Structure 8

2 DATA ANALYTICS 9
- 2.1 Industrial Perspective and Solutions for Streaming Data 10
  - 2.1.1 Data Analytics in Agriculture 11
  - 2.1.2 Data Analytics in Healthcare 13
  - 2.1.3 Data Analytics in Manufacturing 16
  - 2.1.4 Data Analytics in Connected Vehicles 21
  - 2.1.5 Data Analytics in Logistics 23
  - 2.1.6 Data Analytics in Milk Production and Cattle Monitoring 24
  - 2.1.7 Data Analytics in Smart Homes 27

II ANOMALY DETECTION

3 ANOMALY DETECTION 33
- 3.1 Terminology 34
  - 3.1.1 Time-series 34
  - 3.1.2 Time-series Forecasting 34
  - 3.1.3 Time-series Data Mining 34
  - 3.1.4 Anomaly Detection 34
  - 3.1.5 Interpretability 35
  - 3.1.6 Explainability 35
- 3.2 Foundation 35
  - 3.2.1 Local vs. Global Anomalies 35
  - 3.2.2 Categorization of Anomaly Detection Approaches 36
  - 3.2.3 Types of Anomalies 38
  - 3.2.4 Time-series Characteristics 39
- 3.3 Approaches for Traditional Anomaly Detection 40
  - 3.3.1 k-nearest-neighbor (kNN) Anomaly Detection 40
  - 3.3.2 Local Outlier Factor (LOF) 40
  - 3.3.3 Connectivity-based Outlier Factor (COF) 41
  - 3.3.4 Local Correlation Integral (LOCI) 41
  - 3.3.5 Cluster-based Local Outlier Factor (CBLOF) 41

3.3.6 Histogram-based Outlier Score (HBOS) 42
3.4 Approaches for Time-series Anomaly Detection 42
3.4.1 Statistical Approaches 42
3.4.2 Machine Learning Approaches 45
3.4.3 Deep Learning Approaches 48
3.5 Data sets 51
3.5.1 Types of Data 51
3.5.2 Data Sets Included in Thesis 52
3.6 Evaluation Metrics 56
3.6.1 Precision 56
3.6.2 Recall 57
3.6.3 F-score 57
3.6.4 False Positive Rate 57
3.6.5 ROC 58
3.6.6 Precision-Recall Curve 58
3.6.7 Inference Time 58
4 CONTEXTUAL ANOMALY DETECTION IN HVAC SYSTEMS 59
4.1 Literature Review 61
4.2 Data sets 62
4.3 Proposed Pipeline for Anomaly Detection 62
4.3.1 Pre-processing 62
4.3.2 Feature Generation 63
4.3.3 Anomaly Detection 64
4.4 Evaluation 65
4.4.1 Analysis 66
4.4.2 Problem of Misleading Anomaly Score 68
5 DEEP LEARNING FOR ANOMALY DETECTION 77
5.1 Proposed Approach for Time-series Anomaly Detection 79
5.1.1 Time-series Predictor 79
5.1.2 Anomaly Detector 83
5.2 Experimental Setting-I 84
5.2.1 Data set 84
5.2.2 Evaluation Metric and Experimental Setup . . . 84
5.2.3 Results 86
5.3 Experimental Setting-II 90
5.3.1 Data set 90
5.3.2 Experimental Setup and Evaluation Metric . . . 90
5.3.3 Results 91
5.4 Experimental Setting-III 91
5.4.1 Data set 91
5.4.2 Evaluation Metric and Experimental Setup . . . 94
5.4.3 Results 94
5.5 Experimental Setting-IV 95
5.5.1 Discord Detection 95
6 FUSION OF STATISTICAL AND DEEP LEARNING FOR ANOMALY DETECTION 97

6.1 Literature Review 98
6.2 Methodology 99
6.2.1 Statistical Model: ARIMA 100
6.2.2 Deep Learning-based Model: CNN 101
6.2.3 FuseAD: The Proposed Method 101
6.3 Experimental Setting-I 103
6.3.1 Data set and Experimental Setup 103
6.3.2 Results 105
6.4 Experimental Setting-II 106
6.4.1 Data set and Experimental Setup 106
6.4.2 Results 108
6.5 Ablation Study 109
7 COMPARATIVE STUDY OF TRADITIONAL AND DEEP LEARNING-BASED ANOMALY DETECTION 113
7.1 Literature Review 113
7.2 Experimental Setting 115
7.2.1 Data sets 115
7.2.2 Experimental Setup 116
7.2.3 Evaluation Metrics 117
7.3 Results and Analysis 117

III UNCERTAINTY ESTIMATION

8 UNCERTAINTY ESTIMATION OF DNN DECISION 123
8.1 Literature Review 124
8.2 Methodology 126
8.2.1 Time and space complexity analysis 129
8.3 Experimental Setup and Data set 130
8.3.1 Data sets 130
8.3.2 Data Pre-processing 131
8.3.3 Hyperparameter Selection 132
8.4 Evaluation and Discussion 132
8.4.1 Uncertainty Estimation 134

IV INTERPRETABILITY AND EXPLAINABILITY

9 INTERPRETABILITY AND EXPLAINABILITY OF DNN 139
9.1 Literature Review 141
9.2 Methodology 143
9.2.1 Influence Tracer (TSViz) 143
9.2.2 Statistical Feature Extractor 146
9.2.3 Textual Explanation Generator 146
9.2.4 Sanity Check 149
9.3 Experimental Setup and Data set 150
9.3.1 Machine Anomaly Detection 150
9.3.2 Mammography Data set 150
9.3.3 Shuttle Data set 151
9.4 Evaluation and Discussion 151

V ASSOCIATED RESEARCH

10 INCORPORATING EXPERT KNOWLEDGE IN DNN 157
10.1 Literature Review 158
10.2 Methodology 161
10.2.1 STL Decomposition 162
10.2.2 Expert Model 162
10.2.3 CNN Models 163
10.2.4 Knowledge Incorporation Scheme 163
10.3 Data sets 164
10.3.1 CIF2016 Data set 165
10.3.2 NN5 Data set 165
10.4 Results and Analysis 165

VI CONCLUSION

11 CONCLUSION AND OUTLOOK 173
11.1 Conclusion 173
11.2 Limitations 176
11.3 Future Work 177

Bibliography 179
Index 205
Curriculum Vitae: Mohsin Munir

LIST OF FIGURES

Figure 2.1 Advanced analytics can be used in streamlining manufacturing value chains by finding the core determinants of process performance [13]. 17

Figure 2.2 Simplified movement behavior of cows. Different conditions of cows can be classified based on their movement patterns. 25

Figure 3.1 An illustration of global anomalies (x_1 and x_2), local anomaly (x_3), and micro cluster (c_3) [90]. 36

Figure 3.2 Monthly temperature over time is plotted here over months. Temperature at t_1 and t_2 are same, but t_2 occurs in a different context that makes it a contextual anomaly [42]. 38

Figure 3.3 Human ECG output: Red data points show collective anomaly as the data points collectively occur for an undesired time period [42]. 39

Figure 4.1 Share of internet-connectable consumer household devices 60

Figure 4.2 Examples of HVAC time-series data set. This data set contains both point and contextual anomalies. 70

Figure 4.3 Proposed pipeline for anomaly detection on HVAC data. 71

Figure 4.4 ROC plot comparing the proposed technique with the state-of-the-art anomaly detection techniques. 72

Figure 4.5 Color coded anomalies show anomalies detected by respective methods. Abnormal patterns exit in the highlighted area. Only the proposed method is able to detect these contextual and point anomalies. 73

Figure 4.6 Comparison of precision (P) and recall (R) of different anomaly detection techniques and the presented technique on a specific HVAC system. Here, the precision and recall are calculated for a single HVAC system. 74

Figure 4.7 Color-coded anomaly scores given by HBOS and the presented pattern-based technique. Anomalies toward the bottom of the curve are given relatively low anomaly score by HBOS. Whereas, correct anomaly score behavior is given by the proposed technique. 75

Figure 5.1 Performance comparison of traditional machine learning and DNN. 78
Figure 5.2 DeepAnT architecture for time-series prediction: A convolutional neural network with two convolutional layers, two max pooling, and a fully connected layer. 80
Figure 5.3 One time-series from all of the four Yahoo Webscope data set is shown here. Actual streaming data are shown in blue, whereas red vertical lines highlight anomalous data points based on the provided labels. 85
Figure 5.4 Average F-score of each Yahoo Webscope data set is plotted per history window used in DeepAnT. 87
Figure 5.5 An example of time-series prediction and anomaly detection using DeepAnT 88
Figure 5.6 Time-series ground truth values at $(t+1)$ are plotted against predictions at $(x+1)$ to show the accuracy of the prediction model. The data points away from the diagonal line show anomalies. 89
Figure 5.7 DeepAnT can also be used to detect time-series discords 96
Figure 6.1 FuseAD overview: The system consists of two modules, Forecasting Pipeline and an Anomaly Detector. 100
Figure 6.2 FuseAD forecasting pipeline. 102
Figure 6.3 Forecasting and anomaly detection results of FuseAD on *TS11* time-series from Yahoo *A3* data set. The upper plot shows actual time-series and forecasting results on test data, whereas the lower plot shows anomaly score at each timestamp. Anomaly label (i.e. 1) is assigned to data points that have a high anomaly score. 104
Figure 6.4 Zoomed-in plots of two out of three anomalies detected in Figure 6.3. It shows that FuseAD is capable of correctly detecting point anomalies in streaming data where traditional anomaly detection methods fail normally. 105
Figure 6.5 Snippets of NAB data sets from different time-series. 107
Figure 6.6 Comparative analysis of FuseAD and anomalies detected by ARIMA and CNN models on two yahoo time-series 111

Figure 7.1 Sample time-series from Yahoo Webscope data sets. Actual streaming data are shown in blue, whereas red vertical lines are anomalous data points based on the provided labels. 115
Figure 7.2 Sample time series from different NAB data sets. Actual streaming data are shown in blue, whereas the highlighted area shows anomaly window. 116
Figure 8.1 Proposed Hybrid Model. Convolutional Layers are trained separately using point estimates. The parameters of the convolutional layers are then frozen and Bayesian classifier is trained. 126
Figure 8.2 An analysis of confidence comparison for all three approaches on various samples of CIFAR10 and ORIGA data sets. The actual class is mentioned on left side of each image in bold vertical text. 134
Figure 8.3 Trade-off between number of uncertain samples and the accuracy on remaining predictions. The threshold on x-axis is calculated using *percentile* parameter as shown in Algorithm 1. 136
Figure 9.1 TSXplain system diagram. 144
Figure 9.2 Visualization of the saliency information along with the raw signal values. Influential data points are highlighted in the shades of red, dark being more influential. 145
Figure 9.3 An example of rules and statistical features used to generate textual explanations for a given anomalous time-series. 147
Figure 9.4 Channel-wise natural language explanations generated for the expert users. The second time-series in this figure represents a failure-case where the explanations are not accurate, which is also detected in the sanity check. . . 148
Figure 9.5 As a results of sanity check, the anomalous peak is suppressed and sequence in (a) is classified as normal in (b). 149
Figure 9.6 Summary of evaluation done by expert (left) and novice (right) users on Machine Anomaly Detection data set [221] explanations. 152
Figure 9.7 Sample explanations generated for Mammography (top) and NASA Shuttle (bottom) time-series. 153
Figure 10.1 DeepEX pipeline. 161

Figure 10.2 Prediction of DeepEX and other networks on a randomly selected time-series from the NN5 data set with a horizon 1 166

LIST OF TABLES

Table 4.1 Precision and recall of proposed and other observed techniques on HVAC data set. The best threshold is selected by equal error rate. 66
Table 5.1 This table shows the selected history window and thresholds which are used to evaluate DeepAnT on Yahoo data set. 86
Table 5.2 Average F-scores of Twitter AD, Yahoo EGADS, DeepAnT, and LSTM (DeepAnT using LSTM as time-series predictor) on Yahoo data set are given in this table. Bold F-scores are the best scores for corresponding Yahoo Webscope data set. 89
Table 5.3 Comparative evaluation of different state-of-the-art algorithms and the proposed algorithm on 20 NAB time-series from different domains. Precision and recall are reported in this table. 92
Table 5.4 Comparative evaluation of the state-of-the-art anomaly detection methods on the NAB data set. 93
Table 5.5 Data properties of the used anomaly detection benchmarks. 94
Table 5.6 Comparison of the state-of-the-art anomaly detection methods in semi-supervised (novelty detection) setting. DeepAnT performs best in most of the cases (best AUC in bold). 95
Table 6.1 Comparative evaluation of state-of-the-art anomaly detection methods on the Yahoo Webscope data set. Average AUC per sub-benchmark is shown in this table. 106
Table 6.2 Comparative evaluation of anomaly detection methods on the NAB data set. Average AUC per domain is reported here. Bold numbers show highest AUC in a particular domain. . . 108
Table 6.3 Ablation study on the Yahoo Webscope data set. 109
Table 6.4 Ablation study on the NAB data set. 110
Table 7.1 Comparative evaluation of different anomaly detection methods on Yahoo Webscope and NAB data sets. 119

Table 7.2 Comparative evaluation of different anomaly detection methods on the basis of Precision @ Rank n (P) and the inference time (T). The time is reported in seconds. For each data set, the maximum P@n and the minimum inference time are shown in bold. 120
Table 8.1 Time and space requirement of deterministic, Bayesian and hybrid models for some data sets. 129
Table 8.2 Distribution of data sets used to evaluate proposed architecture 130
Table 8.3 Comparison of deterministic, Bayesian, and proposed hybrid models on different data sets without using uncertainty estimation 133
Table 8.4 Comparison of Bayesian and proposed hybrid models on different data sets with uncertainty estimation 135
Table 9.1 Effect of masking the data points in *Machine Anomaly Detection* data set which are relevant for the explanation. 154
Table 10.1 Results of CIF2016 and NN5 data sets when applied different techniques. Minimum SMAPE is highlighted. 166
Table 10.2 Results of CIF2016 data set of different techniques ordered by mean SMAPE. DeepEX result is highlighted. 167
Table 10.3 Results of NN5 data set of different techniques ordered by Mean SMAPE. DeepEX results are highlighted. 169

ACRONYMS

ECG	Electrocardiography
DNN	Deep Neural Network
ANN	Artificial Neural Network
AI	Artificial Intelligence
XAI	Explainable Artificial Intelligence
LSTM	Long short-term memory
HVAC	Heating, Ventilation and Air Conditioning
IoT	Internet of Things
AUC	Area Under the Curve
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
MC-DCNN	Multi Channel Deep Convolutional Neural Network
MCNN	Multi-scale Convolutional Neural Network
BI	Business Intelligence
PA	Precision Agriculture
GPS	Global Positioning System
ICU	Intensive Care Unit
CDW	Clinical Data Warehouse
GE	General Electric
IIoT	Industrial Internet of Things
ERP	Enterprise Resource Planning
PPC	Predictive Powertrain Control
RFID	Radio-frequency Identification
IT	Information Technology
PC	Personal Computer
kNN	k-nearest-neighbor

LOF	Local Outlier Factor
LRD	Local Reachability Density
COF	Connectivity-based Outlier Factor
INFLO	Influenced Outlierness
LOCI	Local Correlation Integral
HBOS	Histogram-based Outlier Score
CBLOF	Cluster-based Local Outlier Factor
AR	Autoregressive
MA	Moving Average
ARMA	Autoregressive Moving Average
ARIMA	Autoregressive Integrated Moving Average
ESD	Extreme Studentized Deviate
S-H-ESD	Seasonal Hybrid Extreme Studentized Deviate
MAD	Mean Absolute Deviation
EGADS	Extensible Generic Anomaly Detection System
TMM	Time-series Modeling Module
ADM	Anomaly Detection Module
SVM	Support Vector Machines
OC-SVM	One-Class Support Vector Machines
PCA	Principle Component Analysis
rPCA	Robust Principle Component Analysis
MCD	Minimum Covariance Determinant
HTM	Hierarchical Temporal Memory
XGBoost	Extreme Gradient Boosting
XGBOD	Extreme Gradient Boosting Outlier Detection
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
AE	Autoencoder
MLP	Multilayer Perceptron

ROC	Receiver Operating Characteristic
TP	True Positive
FP	False Positive
FN	False Negative
TN	True Negative
TPR	True Positive Rate
FPR	False Positive Rate
EER	Equal Error Rate
PR	Precision-Recall
P@n	Precision @ Rank n
NNAR	Neural Network Autoregression
mV	Millivolts
NAB	Numenta Anomaly Benchmark
AWS	Amazon Web Services
CPU	Central Processing Unit
CPC	cost-per-click
CPM	cost-per-thousand impressions
NYC	New York City
DTW	Dynamic Time Warping
CSV	comma-separated values
DeepAnT	Deep Anomaly Detection for Time-series
SGD	Stochastic Gradient Descent
ReLU	Rectified Linear Unit
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
AD	Anomaly Detection
RBF	Radial Basis Function
FuseAD	Fusion of Statistical and Deep Learning for Anomaly Detection

AIC	Akaike Information Criterion
TSViz	Time-series Visualization framework
TSXplain	Time-series Explanation framework
KS	Knowledge-based System
DeepEX	Deep Expert
seq2seq	sequence-to-sequence
KBANN	Knowledge-based Artificial Neural Networks
KL	Kullback–Leibler
KEHNN	Knowledge Enhanced Hybrid Neural Network
GRU	Gated Recurrent Units
KINN	Incorporating expert knowledge in neural networks
LDA	Latent Dirichlet Allocation
STL	Seasonal and Trend decomposition using Loess
CIF	Computational Intelligence in Forecasting
ATM	Automated Teller Machine
SMAPE	Symmetric Mean Absolute Percentage Error
MCMC	Markov Chain Monte Carlo
VI	Variational Inference
ELBO	Expected Lower BOund
SBN	Sigmoid Belief Network
OOD	out-of-distribution
MAF	Masked Autoregressive Flows
real NVP	real-valued Non-Volume Preserving
FC	Fully-connected
CDR	Cup-to-Disc Ratio
CV	Cross Validation
MLE	Maximum Likelihood Estimation

Part I

INTRODUCTION

1

INTRODUCTION

1.1 MOTIVATION

Data is the new gold

With every passing day, data is becoming more and more valuable. Companies from every sector like automotive, medicine, dairy, manufacturing, hospitality, and logistics are investing a lot of their resources [50, 147, 197, 200] to mine the data. They leverage the mined data to provide quality products and services at competitive prices. It would not be wrong to say that, data is the new gold [64, 217, 252].

Anomaly detection is one of the main tasks of time-series data mining

Time-series data mining process facilitates extracting meaningful insights and patterns from a given time-series using different classification, clustering, association, and anomaly detection methods [71, 76]. Anomaly detection has become the focus of many researchers and practitioners because of its applicability to a wide range of domains and use-cases [25]. It has been studied for fraud detection, network traffic monitoring, data leakage prevention, detecting issues in Electrocardiography (ECG), and machine failure detection [5, 35, 37, 46, 157]. Anomaly detection methods enable to 1) improve the product quality, 2) cut the product cost, 3) optimize the manufacturing process, 4) reduce production time, 5) take preemptive measures, 6) generate risk portfolios, 7) detect fraudulent behavior, and most importantly, 8) determine the root cause of a failure. There exists a lot of anomaly detection methods in literature to detect anomalies in a traditional setting; mostly for non-time-series data. Therefore, such methods have limited performance when applied to time-series data. To address this issue, anomaly detection methods are proposed in this thesis to detect different types of anomalies specifically in time-series data. Moreover, Deep Neural Networks (DNNs) have also not been explored extensively for anomaly detection in time-series data. To fill this gap, DNN based anomaly detection methods for time-series data are also proposed.

It has been well established that DNNs are superior to other methods available at hand and give near human-level accuracy for some tasks [33, 79]. The same implies to DNN-based anomaly detection methods, as they perform better than traditional anomaly detection methods for time-series data. However, DNN-based methods are not embraced in critical domains like health care, connected vehicles, counter-terrorism, and law enforcement because of the lack of confidence in their own decision and their black-box nature [48, 202, 210]. In critical domains, the minimum requirement is to know the confidence of a decision, so that it is clear either one can trust the decision

A system should be confident about its decision especially in critical domains

or not. Every aspect of a network's decision needs to be validated to avoid any possibility of a wrong decision that could affect human life. Deterministic deep learning models are unarguably one of the best choices for classification [119, 207, 270]. However, they don't provide any confidence measure about the decision. On the other hand, Bayesian neural networks place a probability distribution on network weights [27, 180] and provide a way to generate posterior distribution which can be used for an uncertainty estimation. The disadvantage of directly using Bayesian neural networks is the high computation time due to wider parameter space [180, 253]. To provide an uncertainty estimation of network decisions, a hybrid method is proposed in this thesis so that the decisions with low confidence can be validated again to void any unforeseen situation due to a bad decision.

Explanation is pivotal to gain trust on automatic decision-making systems.

Confidence together with the decision is a minimum requirement in critical domains. However, to better gain the trust of a user, interpretation and explanation of the decision are also a requisite. To open the DNN mysterious black-box, many visualization and interpretation tools have been provided in the literature [224, 234, 259, 261, 262]. Most of these proposed tools are for the image modality. To fill this vacuity, a DNN interpretability tool for time-series data is proposed in this thesis that helps end-user in understanding the decision-making process of a DNN. The visualizations in this tool highlight the anomalous data point that contributed towards a particular decision. Nevertheless, expert knowledge is required for the intelligibility of such visualizations which keep these approaches out of the range of an end-user. Also, they don't provide any justification or reason for the decision. When humans make a decision, they have certain experiences and rules in the back of their minds that can be presented as their version of a justification. Likewise, justification of a machine decision is mandatory in critical systems. To use Artificial Intelligence (AI) for pragmatic solutions, an explanation framework is presented in this thesis that provides natural language-based explanations of a network decision. Based on the most influential data points and other time-series features, simple and detailed explanations are provided for novice and expert users.

As a result of all these advancements, a hybrid framework is presented in this thesis. This framework consists of three major components or pillars: i) Anomaly Detection, ii) Uncertainty Estimation, and iii) Interpretability and Explainability.

1.2 RESEARCH QUESTIONS AND GOALS

Following are the research questions of this thesis:

1. **Is it plausible to use DNN for anomaly detection in time-series data?**
   Goal: Explore different DNN architectures and propose anomaly

detection method(s) for time-series data. Also, compare their strengths and weaknesses in different scenarios and for different data sets. Moreover, compare the DNN with traditional anomaly detection methods to validate the detection improvements if any.

2. **Is it possible to fuse two disjoint worlds of statistics-based and deep learning-based anomaly detection?**
   Goal: Combine statistics-based anomaly detection models with DNN-based anomaly detection models. Investigate how these two disjoint worlds can be combined on a model level instead of merely ensembling the results.

3. **Is it possible to get an uncertainty estimation of a network decision instead of merely relying on the decision?**
   Goal: Propose a solution to confidently give the network decision for the classification problem. Extend the current state of Bayesian neural networks as they are very computationally expensive by fusing the two different worlds i.e. deterministic and probabilistic to benefit from both. Furthermore, evaluate the method on time-series data and compare the results with traditional Bayesian neural network and deterministic neural network.

4. **Is it possible to go beyond the decision, and locate and explain the actual cause of the decision in natural language?**
   Goal: Use the interpretability tool(s) to get the insights of a neural network. Exploit those insights and other time-series features, to express the networks' decision in natural language that eventually facilitates novice and expert users. Furthermore, align statistical features of time-series with DNN to generate explanation and gain user trust.

1.3 CONTRIBUTIONS

Following are the contributions of this thesis:

Anomaly detection for HVAC systems

1. This thesis presents a novel approach to detect long-term anomalies in Heating, Ventilation and Air Conditioning (HVAC) time-series data. In contrast to the traditional anomaly detection methods, the proposed approach brings context into consideration and builds a knowledge base of long/short-term patterns based on normal data points that keep growing over time. The proposed method outperforms the state-of-the-art methods for anomaly detection in terms of different performance indicator measures i.e., Area Under the Curve (AUC) 99.4%, precision and recall of 0.91 and 0.80 respectively. Another contribution of this method is the generation of meaningful anomaly scores as

the anomaly scores produced by the existing methods for HVAC time-series is sometimes misleading and does not correlate with the significance of anomalies.

CNN-based anomaly detection

2. Deep Anomaly Detection for Time-series (DeepAnT) is proposed to detect time-series point and contextual anomalies as well as time-series discords in an unsupervised fashion. To the best of authors' knowledge, this is the first method to use Convolutional Neural Network (CNN) for time-series point anomalies. It is shown in the study that this method is applicable to univariant as well as to multi-variant time-series in anomaly detection and novelty detection scenarios. Generally, it is considered that Long short-term memory (LSTM) is the only and best choice for time-series analysis. To invalidate this hypothesis, it is highlighted in this study that the proposed CNN-based anomaly detection method performs better than LSTM-based anomaly detection. The evaluations show that in an anomaly detection setting, DeepAnT outperforms Yahoo EGADS and Twitter anomaly detection methods in three out of four Yahoo Webscope data sets by a significant improvement of 34% - 39% in the F-score. The detailed evaluations validate our hypothesis (mentioned in Section 1.2) that DNNs can be used for time-series anomaly detection.

Fusion of traditional statistics-based and DNN-based methods for anomaly detection

3. Since anomaly detection is an old research topic, there exist different statistics-, clustering-, machine learning-, and deep learning-based anomaly detection methods. In most of the industrial and practical use-cases, traditional statistics-based models are used because of their transparency and end-user trust. On the other hand, deep learning-based anomaly detection is mostly used in the research community because of its automatic feature engineering and high accuracy. To bring these two disjoint worlds together, this thesis presents a hybrid anomaly detection approach: Fusion of Statistical and Deep Learning for Anomaly Detection (FuseAD). The major contribution of this approach is the residual learning scheme that lets the network learn how to produce the best forecasting outcome based on two different kinds of models. In FuseAD, a statistics-based model – Autoregressive Integrated Moving Average (ARIMA) is merged with a deep learning-based model – CNN. This fusion mechanism enables the network to complement the strengths of the underlying two disjoint models by fusing the information encapsulated in them. This study validates our second hypothesis that the two different worlds of statistics and deep learning can be fused for time-series anomaly detection problem.

4. With all the rapid developments in the area of anomaly detection, there exists a small number of comparative studies for

anomaly detection methods. Even in the existing studies, the comparison of different anomaly detection methods only on traditional data set is presented, also there is no comparison with deep learning-based anomaly detection methods. In this thesis, a detailed comparative analysis of traditional and deep learning-based anomaly detection methods on time-series data sets is presented. In this study, 13 anomaly detection methods are compared on two commonly used streaming data sets. Four different evaluation metrics are used to evaluate the methods from different perspectives.

Comparative analysis of traditional and deep learning-based anomaly detection methods for time-series data

5. Deterministic models have unarguably achieved near-human performance in many image classification tasks. However, they are incapable of capturing all possible combinations of the network weights which results in a biased predictor towards their initialization. On the other hand, Bayesian neural networks provide uncertainty estimation; at the cost of high computation expense. In this thesis, a novel hybrid convolutional neural network is presented which combines the high accuracy of deterministic models with posterior distribution approximation of Bayesian neural networks. The evaluation results show that the proposed approach performs superior to both deterministic and Bayesian methods in terms of classification accuracy and also provides an estimate of uncertainty for every prediction. This study validates our third hypothesis mention in Section 1.2.

Uncertainty estimation

6. Interpretability plays a vital role in demystifying deep learning. There exist different interpretability methods and tools, however, only for the image domain. Generally, those methods are not directly applicable to the time-series domain. To fill this gap, an interpretability and visualization tool for time-series data is presented in this thesis. This tool identifies and highlights the parts of the input time-series that are influential to a network's decision.

Identification and visualization of most influential data points

7. Interpretability tools are not directly intelligible by an end-user. In most cases, expert knowledge is required to get the leads from the visualizations and reach to a certain conclusion. To bring research to the real world, an explainability tool is proposed in this thesis. The main contribution of this tool is the generation of natural language explanations of a network decision. In order to assess the reliability of a given explanation, a sanity check is also performed. Moreover, it is also important to evaluate the generated explanations and measure the correctness of those explanations. The survey conducted for the evaluation of the explanations showed that the generated explanations are relevant, correct, and satisfactory. This contribution validates our fourth hypothesis (Section 1.2) that explanations

Enable networks to explain their decisions in a natural language of DNN decisions can be provided which actually serves as the justification of a decision.

1.4 THESIS STRUCTURE

After the motivation (Section 1.1), research questions (Section 1.2), and contributions (Section 1.3) of this thesis, industrial perspective of data analytics (Chapter 2) is discussed in Part I. Chapter 2 provides a number of data analytics solutions and trends that are followed in major industries.

The rest of this thesis is divided into three pillars of the proposed hybrid framework for time-series analysis. The contributions of this thesis in the direction of time-series anomaly detection are mentioned in Part II. In this part, first a detailed background, basic concepts, characterization of anomaly detection methods, common data sets for anomaly detection, and commonly used methods are mentioned in Chapter 3. After that, an anomaly detection method for HVAC systems is presented in Chapter 4. Furthermore, a deep learning-based anomaly detection method – DeepAnT, and a fusion-based anomaly detection method – FuseAD are proposed in Chapter 5 and Chapter 6 respectively. As the last chapter of this part, a detailed comparison of different anomaly detection methods on different data sets is given in Chapter 7.

Other two pillars of this thesis; uncertainty estimation and interpretability and explainability are mentioned in Part III and Part IV respectively. In Part III, Chapter 8 explains in detail the confident classification of DNNs. The two important aspects of a network's decision; *HOW* and *WHY* are explained in Chapter 9, where the significance of interpretability and explainability, along with our solutions are explained.

Incorporation of expert knowledge in DNNs is presented in Chapter 10 as an associated research. Finally, the thesis is concluded in Chapter 11 along with the limitations and future work.

2

DATA ANALYTICS

Data analytics is a process of applying statistical and/or logical techniques to evaluate and describe data. The purpose of data analytics is to extract meaningful information from the data by reducing a large chunk of data into smaller fragments. This information can then be used to optimize processes to increase the overall efficiency of a system and to facilitate the business decision-making process. The whole data analytics process involves, but not limited to the following steps. The steps involved in this process varies according to the use-case and requirements. Each step in this workflow is key for good analytics.

Extracting information from the raw data

- Data Acquisition
  As a first step of the data analytics process, data engineers identify the data sets and their sources with the help of domain experts and field engineers. It is important to define the right set of data, as whole analytics will be based on it.

- Data Integration
  In most of the use-cases, data is coming from different sources. It needs to be combined via different integration routines and transformed into a common format that is required by the analytics system.

- Data Cleaning
  Once the data is acquired and transformed into a required format, the next step is to ensure the data quality. To do so, data profiling and data cleaning are performed that make sure that the information in the data is consistent and there are no duplicates and missing values in the data set.

- Data Modeling
  It is the core step of the data analytics process. This step aims to extract meaningful information out of the given data. To achieve it, data engineers may use different analytics software like *RapidMiner, IBM Analytics, KNIME, Google Analytics, and SAS Advance Analytics etc.*, or programming languages like *R, Python, Scale,* and *Matlab*. Moreover, several statistical, machine learning, and deep learning methods exist for data modeling.

- Data Visualization
  As the final step, the findings of the analytical models are communicated to the right audience. It is important to understand

---

This chapter is an adapted version of the work published in [181] the audience before visualization as business executives might only be interested in high-level numbers whereas, domain experts might be interested in getting full insights into the analytics. So, depending on the audience and the requirements, the visualizations are curated. There are different Business Intelligence (BI) dashboard applications like *R Shiny Dashboard* and *Tableau*, that give the possibility of displaying data on a single screen and can be updated in real-time as new information becomes available.

2.1 INDUSTRIAL PERSPECTIVE AND SOLUTIONS FOR STREAMING DATA

Smart devices

Over the past few years, a lot of devices and machines around us are becoming *'smart'*. Based on the idea of the Internet of Things (IoT), different devices and machines can connect to the internet and communicate with each other. IoT refers to the network of physical devices, vehicles, buildings, and other items embedded with electronics, software, sensors, actuators, and network connectivity that enable these objects to collect and exchange data [251]. Such internet-enabled devices are continuously observing their environment and logging a lot of data in the back-end database. By applying data analytics to the gathered *Big Data*, smart decisions can be taken to facilitate the end-user according to the current situation. This capability of adaptive decision making makes ordinary devices and machines *'smart'*. These devices and machines are becoming intelligent by learning about their surroundings from different sources and develop the ability to avoid unforeseen situations by analyzing the data.

Recent studies show that the global IoT market is growing and will continue to grow a lot over the next few years. It is expected that a market value of nearly 1,102 billion US dollars will be reached by the year 2026 [75]. Due to these internet-enabled devices, companies are now continuously getting live streams of huge amounts of data. This data is usually collected over a continuous interval of time, which results in time-series data.

Data analytics on streaming data

With the help of data analytics on streaming/time-series data, companies can keep an eye on different aspects, e.g., reducing maintenance costs, avoiding equipment failures, and empower them to upgrade existing processes by creating and tracking new business models. Moreover, retailers, restaurant chains, and makers of consumer goods can use the data from smartphones, wearable technologies, and in-home devices to do targeted marketing and promotions – the business side of the IoT's futuristic world of connected consumer gear.

Almost every industrial sector, be it health care, agriculture, manufacturing, dairy farming, logistics, or automotive is now redefining their products and enabling them to IoT for gaining maximum benefits. Due to these developments, different industrial solutions are also available focusing specifically on time-series based data analytics. This section provides a comprehensive overview of how different industrial players are using data analytics to provide better services to their customers and improve their internal processes and workflows. It is also discussed here how different industries use data analytics to gain vital insights for providing better healthcare to the public, making homes more secure, increasing crop yield, delivering goods more quickly, reducing the downtime of a machine, and avoiding disease. An overview of different analytics platforms and solutions used in different industries for time series and streaming data are also discussed in this section.

2.1.1 *Data Analytics in Agriculture*

With the increase of the world population and the amelioration of living standards, the demand for high-quality food is increasing. Agricultural mechanization is playing a vital role to fulfill this need with the help of large-scale production. However, the physical performance of mechanization and mass production are limited even with the advancements in the fields of IoT and cloud computing. In the agriculture sector, data analytics can be applied to the machinery and farming data to reduce loss, improve efficiency, and lower costs under the condition of unchanged physical properties. This enables a modern farming concept called Precision Agriculture (PA) or Satellite Farming. This kind of farming helps the farmer to recognize the variations in the farming land and to adjust input for different parts of the land to optimize the output. A Global Positioning System (GPS) is the backbone of PA. With the help of GPS, a farmer can identify the exact area where soil conditions vary. In conjunction with the precise location, different measures like air quality, moisture level, field terrain, crop yield, crop maturity, and gas levels are recorded and turned into meaningful information using data analytics.

Precision Agriculture

Data-driven decision-making has been extended from the business sector to the agricultural sector. Many large enterprises in the agribusiness are becoming involved in data analytics research and development. They are providing solutions for PA and for a variety of other issues in agriculture. Fierce competition between companies has already begun.

John Deere converted their equipment to the paradigm of IoT to help farmers manage their fleet, reduce downtime, and the cost of production. The local weather data, soil data, crop characteristics, and other data sets from different sources are combined to achieve the aforementioned goals. *MyJohnDeere.com* is a platform for data analytics, that provides the possibility to store, analyze, and visualize results on a web-portal (as well as on a mobile application called

John Deere

Mobile Farm Manager). With the help of such platforms, farmers can figure out when and where to plant which kind of crops, when to plough and when to harvest, and which optimized path should be followed during the work. The right decision can help farmers to improve their efficiency. The data collected during different phases of farming are massive. To take advantage of the collected data, John Deere already steps into big data analysis for the future of farming.

IBM in collaboration with SignalDemand

IBM and SignalDemand have developed a data analytics system which uses predictive analytics to predict the demand and optimize the margin to meet the needs of different agribusiness companies. While large agricultural enterprises have large data sets, advanced equipment, data scientists, and domain experts at their disposal; the majority of farmers neither have access to such information nor the resources to get benefits from advancements in technology. To help farmers who are working on a small-scale and lack the technology infrastructure, IBM built a back-office network. They supply corn-specific information on a regular basis, along with generalized information on fertilizer and weather conditions to registered farmers via their mobile phones. A farmer can get timely agronomic intelligence simply via automated voice mail or text messages on his mobile phone.

aWhere

aWhere (an American corporation) collects and analyzes over a billion points of data (which is a pivot element for analysis) from around the globe each day to create unprecedented visibility and insight which is known as Agricultural Intelligence. This intelligence is used for critical decision making from farm level through to national policy. High-quality weather data is combined and analyzed purely for agricultural use. Their major data analytics solutions are *In-Time Weather Data* , *Weather Insights*, and *Maps4ER*.

The Climate Corporation

The Climate Corporation (a San Francisco-based company) examines weather data to provide insurance to farmers who can lock in profits even in the case of drought, heavy rains, or other adverse weather conditions. *FieldView* is their data analytics solution, which combines farmers' field data with real-time and past – soil, crop, and weather data to help them efficiently manage their operations and gain insights into their fields [70]. In addition to the FieldView, they also provide a hardware solution *SeedSense* for Planter Monitoring. Perfect planter performance can be achieved by maximizing planter speed and adjusting vacuum pressure by using SeedSense. It also enables the farmer to sow precisely, maintain depth, avoid compaction, and troubleshoot mechanical problems.

Benson Hill Biosystems

The *CropOS* is a data analytics platform, which uses machine learning and cloud biology to improve crop performance and help scientists and breeders with some of the biggest challenges in the agriculture sector. It is developed and maintained by Benson Hill Biosystems, which is an agricultural solutions company. They unlock the global genetic potential of plants to enhance the sustainability of food, feed, fiber, and fuel production [176]. CropOS represents a uniquely powerful platform at the intersection of big data, machine learning, and plant biology. CropOS empowers researchers to significantly increase the yield of major food crops and identify the most promising plant genetics in weeks instead of studying long growing seasons.

CLAAS

CLAAS focuses very much on self-propelled machines, developing and producing combine harvesters, self-propelled forage harvesters, and tractors [108]. Self-propelled machines are very important especially for crops like wheat, rye, barley, and corn; that have to be harvested at just the right point of maturity. Once this harvest maturity has been reached, the combine harvesters work in the fields day and night. In this process, up to 50 parameters from the reel to the chopper influence the harvest yield. The operator has to continuously monitor and evaluate around a dozen of these parameters. Hardly any operator is capable of keeping an eye on everything and tapping the machine's full potential. To solve this problem, CLAAS also moved toward IoT enabled combine harvesters. In addition to this, an assistance and analytics system is used, that permanently monitors the harvesting process and automatically adjusts the machine setting to the current conditions — faster and more precise when compared to a human operator. Furthermore, together with the German Research Center for Artificial Intelligence, and the Fraunhofer Institute, CLASS is working on extending the data analytics to improve the performance of mobile work machines with unsupervised anomaly detection algorithms, which can detect unexpected events without any previous domain knowledge.

2.1.2 *Data Analytics in Healthcare*

Similar to agriculture, data analytics is playing a vital role in the advancement of the healthcare sector. With the easy availability of smart devices (including smart watches, smart phones, and smart wristbands), a new dimension of healthcare has emerged – Smart Healthcare. End-user smart devices are continuously collecting users' data regarding different activities performed over a day, month, or year using different sensors.

Smart healthcare

Data analytics on smart sensors' data have opened new dimensions of research and applications in Connected or Smart Healthcare. Smart healthcare is supporting, and slowly replacing traditional healthcare. By analyzing the streaming data generated by smart wearables, it is possible to see if a user is healthy, or if some preventive measures are required, in order to avoid a potential health problem. Now doctors can remotely examine their patients and suggest treatments on the go. Smart healthcare offers many new possibilities for patients too. Patients can stay updated with their health and fitness data all the time, find other patients suffering with the same disease to discuss various treatments, and easily track the post-surgical needs. The digitization of patients' health data encourages the communication and collaboration of all the stakeholders involved in the patient's health in the following ways:

- Government institutes can use the data to extract different statistics and to make policies as needed.
- Pharmaceutical companies can use the data to track the positive or negative effects of different medicines.
- Doctors can use this data to choose a treatment when a patient has high cardiovascular risk, etc.

With *smart healthcare*, healthcare is shifting from being episodic/reactive to preventive/proactive. Different companies (mentioned below) are providing solutions for connected-, smart-, and preventive-healthcare.

IBM

IBM Healthcare is a data analytics solution, which focuses on health monitoring and intervention, analyzing streaming data (such as data generated in Intensive Care Unit (ICU)), and helping in detecting signs of various changes occurring in a patient's health. The detected early signs are used to generate medical alerts for proactive intervention. It also enables healthcare providers to improve operational performance, reduce cost of care, and counter fraud in healthcare by using integrated data management and analytics. Furthermore, it provides consumer level analytics to understand consumer preferences and behaviors by capturing data from different sources such as claims, clinical history, and social platforms; and then merges all the data into one unified view. It also helps building a predictive model that evaluates the risk of readmission for patients with chronic obstructive pulmonary disease [127]. Researchers at National Institutes of Health (NIH) are using *IBM PureData* System for analytics to unlock new insights from data gathered over decades. With the help of this system, researchers can run analysis on large, complex data sets (both clinical and genomic research data) and generate reports faster than ever before [126].

SAP

SAP Real-Time Analytics is a complete solution for patient care, human resources, finance, care collaboration, and healthcare analytics. The big health data collected from electronic health records, research, physician notes, insurance claims, and social media data are used by SAP Real-Time Analytics to reduce cost and improve quality of care. This solution enables data scientists to separate noise from signals and derive meaningful insights from the data. The unified analytics model transform data from a wide range of sources into actionable information. Seoul National University Bundang Hospital (South Korea) has developed its Clinical Data Warehouse (CDW) using *SAP Data*

Services and *SAP HANA*. Their CDW is used to automate the clinical indicators system, gather critical data in real-time, provide instantaneous feedback to clinicians, and provide multidimensional analyses based on patient characteristics, diseases, and location [211].

General Electric

General Electric (GE) provides many healthcare solutions in general. Some of those solutions are based on data analytics in the areas of diagnosis, clinical decision-making, and asset monitoring. The GE *Marquette 12SL* ECG analysis program provides diagnostic confidence to care providers by giving fast and reliable cardiac care decisions. In the area of patient monitoring, GE provides *CARESCAPE Central Station* that allows the integration of different medical devices and systems to access patient's historical data. When a patient moves to a care area, this solution enables care providers to perform in-depth analyses and offers clinical decision support. They also provides *Centricity Imaging Analytics*, a real-time dashboard that provides visibility into the workflows of the radiology department for increasing department throughput and patient care [81].

Siemens

Combined Applications to Reduce Exposure (CARE) (by Siemens) combines a variety of advanced technologies and applications. *CARE for Patients* is an analytics solution, which is designed to improve dose monitoring in different interventional radiology systems. The dose of an individual patient is recorded in addition to other data, such as CT-dose index, dose length product, and total recording time. This data is also used to enhance dose reporting and assessment, transparency regarding dose per case, reporting on patient dose history, and cross-institutional reporting [222].

Apple Inc.

Apple CareKit is an open source platform for creating health related applications to regularly track care plans, monitor users' progress, and share their insights. *One Drop* (by Informed Data Systems, Inc.) is an example of such a mobile application created using CareKit. *Apple ResearchKit* equip developers to create applications, that enable researchers and doctors to gather robust and meaningful data for their health related studies, and obtain a complete history of their patients. The real life data collected is used to find physical patterns, correlation between physical history and medication, predict a particular problem, and recommend diet and fitness plans. With the help of *ResearchKit* and *CareKit*, researchers use Apple Watch to predict seizures before they actually happen. For instance, *EpiWatch* (an Apple Watch application by Johns Hopkins University) enables people to accurately track the onset and duration of seizures in real time. A patient sensing an impending seizure launches the application on Apple Watch and an alert is automatically sent to a designated family member or caregiver. Similarly, *Asthma Health* (by Weill Medical College), *Concussion Tracker* (by New York University Langone Medical Center), *GlucoSuccess* (by Massachusetts General Hospital), and *C*

Tracker (by Boston Children's Hospital) are examples of such applications that are built on top of *ResearchKit* and *CareKit* [11].

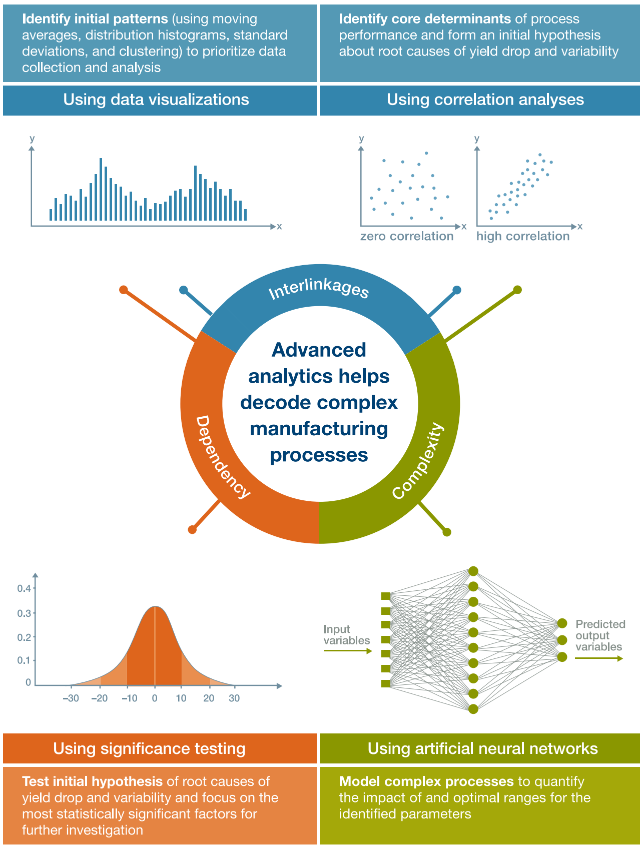
2.1.3 *Data Analytics in Manufacturing*

Industrial Internet of Things

Data analytics provides a granular approach to diagnose and improve whole manufacturing limitations. It is always in the manufacturers' interest to improve their production processes, product quality, production cycle, and the amount of output per unit of input. Due to the involvement of a number of players and processes in the manufacturing life cycle, it is challenging to find the cause of failure or inefficiency. With the growth of Industrial Internet of Things (IIoT) in recent years, everything is going digital and connected. By virtue of this digitization and connectivity, a lot of streaming data related to equipment, automation, production lines, systems, and products are generated and stored. Figure 2.1 shows how advance data analytics can help decode and improve complex manufacturing processes. Manufacturers can use data analytics to leverage the data collected from on-the-floor factory machinery alongside other traditional (factory logs) and social data. Some of the advantages of using data analytics in manufacturing are following:

- Get unexpected insights from different processes.
- Increase accuracy, quality, and yield (amount of output per unit of input).
- Improve the forecast of product supply and demand.
- Enhance the understanding of plant performance across multiple metrics.
- Boost the product quality.
- Track all products with defected components.
- Predict machine failure.
- Quantify how daily production impacts financial performance.
- Provide preemptive maintenance and service by continuously monitoring a product instead of fixed term maintenance.
- Identify the root cause of a failure.

The main challenges in manufacturing are a lack of collaboration across different departments, disparate systems and data sources, and difficulty in coordinating supply and demand chains. Such challenges, among others are tackled in the solutions discussed in this section by different companies using advanced data analytics.



Identify initial patterns (using moving averages, distribution histograms, standard deviations, and clustering) to prioritize data collection and analysis

Identify core determinants of process performance and form an initial hypothesis about root causes of yield drop and variability

Test initial hypothesis of root causes of yield drop and variability and focus on the most statistically significant factors for further investigation

Model complex processes to quantify the impact of and optimal ranges for the identified parameters

Using data visualizations
Using correlation analyses
y
x
zero correlation
high correlation
Interlinkages
Advanced analytics helps decode complex manufacturing processes
Dependency
Complexity
0.4
0.3
0.2
0.1
0
−30
−20
−10
0
10
20
30
Input variables
Predicted output variables
Using significance testing
Using artificial neural networks

Figure 2.1: Advanced analytics can be used in streamlining manufacturing value chains by finding the core determinants of process performance [13].

IBM Analytics provides a complete analytics solution to be used in automotive, defense, chemical, petroleum, energy, aerospace, electronics, and other industries to uncover deeper insights into operations, inventory, market demands, supply chain, and performance [130]. By applying advanced data analytics on aggregated data from different sources (such as different sensors, maintenance logs, and production systems), manufacturers can efficiently achieve their demand, production, and supply requirements; while properly managing all the resources at minimal cost. Their analytics solution can integrate structured as well as unstructured data from different sources. It can unveil a number of critical manufacturers questions, such as how operating costs can be reduced while having better project financial performance, how greater visibility into supply chains can be achieved, how the supply chain's needs can be predicted, and how the maintenance cost can be cut down. Moreover, it can also uncover insights into customers' behavior, their needs, and market trends to make better business decisions. Nowadays, production assets and consumer products are transmitting vital operational data to backend data warehouses. *IBM Predictive Maintenance and Quality* software solution leverages the data collected from different sources and predicts when a particular asset or machine needs maintenance. In contrast to the traditional scheduled maintenance, predictive maintenance recommends when maintenance is required and when it is not. This type of maintenance helps to keep critical production lines and consumer products running, while saving money and minimizing customer inconvenience. Muller Inc., USA, is a retailer and manufacturer of metal products. They used *IBM Cognos Business Intelligence, IBM Cognos TM1, IBM SPSS Modeler,* and IBM Business Analytics to pull data from all points of sale, inventory, and Enterprise Resource Planning (ERP) systems; so that the employees can view and analyze company data, measure individual performance, and access how their work affects the bottom line [128]. The Vaasan group (a leading bakery operator in Northern Europe) used IBM Analytics to enhance forecasting and inventory management. The solution based on the *IBM Cognos Controller, IBM Cognos Intelligence,* and IBM Cognos 8 Planning enabled the bakery to predict production requirements and helped them prepare for fluctuating orders [124].

IBM

SAP provides multiple solutions in the domain of manufacturing. SAP *Manufacturing Execution System* connects, monitors, and controls different manufacturing operations. With the help of automated data collection, it provides visibility into the manufacturing processes which helps process managers to find and resolve quality issues. Its asset utilization functionality improves overall equipment effectiveness, facilitates predictive maintenance, and minimizes downtime. SAP *ERP* is an enterprise level system for streamlining the manufacturing, services, sales, finances, and human resource processes. It is composed of different modules, which accelerate the entire manufacturing process, boost sales and customer satisfaction, provide support for administration tasks, streamline and automate financial operations, and provides real-time analytics based on ERP data. SAP *Manufacturing Integration and Intelligence* is the solution for smart manufacturing which exploit the data collected from IIoT. It automates the IIoT and facilitates in manufacturing data transformation and integration. This software is equipped with the *Manufacturing Analytics Platform*, which provides statistical process control and predictive analytics. It can also identify the root cause of machine downtime and efficiency loss; which makes the maintenance task easy for technicians and helps the operation team to improve efficiency. The SAP *Predictive Maintenance and Service* solution leverages the IoT data to transform reactive maintenance to predictive maintenance. It provides the visibility into manufacturing asset and consumer product health by remotely observing their behavior and patterns. By analyzing the Big Data collected, future needs are predicted [212].

SAP

Microsoft Azure IoT is a complete suite for connecting IoT devices, collecting IoT data, analyzing the collected data, and mining disparate data [177]. Existing data and systems can also be integrated with new data sources to create new insights and business models. A *Predictive Analytics* module in Azure provides insight into how a certain product behaves in normal conditions and in other special conditions by finding patterns and correlations in historical and new sensor data. Based on such analytics, this suite is able to provide warning signs, identify where a problem exists, and notify when equipment needs maintenance. With such preemptive warnings, small repairs can be made before big failures occur. It also helps in prioritizing the maintenance task by providing information about which equipment is at high risk. Once an actual root cause of the failure is detected, it can facilitate a technician by recommending the error code (with possible fixes) for that condition. The technician's time of finding the root cause of a failure is saved, now he just has to fix the defective component (with the help of some recommendations about possible fixes). This suite enables manufacturers to remotely monitor their assets, which are deployed outside the factory. Automatic notifications can be triggered on this live data to get real-time asset feedback and maintenance requests.

Microsoft

GE Brilliant Manufacturing is a software suite, which connects people, machines, materials, and processes in IoT. This suite maximizes manufacturing production performance and optimizes operations through advanced real-time analytics. It allows the integration and aggregation of whole manufacturing life cycle data from the beginning till the end. Data driven analytics from disparate manufacturing sources allow manufacturers to take optimal decisions to drive im-

General Electric

provements in end-to-end production [80]. This suite includes different products including the following:

- *Efficiency Analyzer* provides an up-to-date view of the entire production process and transforms real-time machine data into action efficiency metrics. Such unified metrics help plant managers to reduce unplanned downtime, maximize yield, improve production quality, increase flexibility, and maximize team productivity.
- *Production Quality Analyzer* analyzes data to catch non-conforming events before they occur to help quality engineers to easily identify the problem.
- *Production Execution Supervisor* digitizes documentation, instructions, orders, and process steps, enabling manufacturers to get the right information at the right time.
- *Product Genealogy Manager* builds a record of all equipment, raw materials, tools, and personnel which are required to build the finished goods. It helps service personnel to manage services in an efficient way.

BOSCH

Manufacturing Analytics by BOSCH is a solution for analyzing production data. Different types of data such as test, process, and machine data from different sources can be combined to improve the production process and product quality while reducing the cost with the help of this suite. This suite can integrate the existing production data with the new data. The predictive models can be applied to real-time data for predictive maintenance and root cause analysis. Data analytics unveils the previously unknown correlations in data and helps manufacturers in gaining new insights. The newly discovered data insights and prediction models can be applied using this suite to automate the analytics process.

SAS

SAS provides different solutions to get the best out of the manufacturing life cycle. SAS *Demand-Driven Planning and Optimization* suite improves the supply and demand planning processes. This suite uses analytical insights of demand patterns to help manufacturers in making supply plans, that are aligned with the demand forecast. Production and logistics can also be managed to match the ever-changing customer needs and market dynamics. SAS *Quality Analytics* suite includes data mining and predictive analytic technologies for predictive maintenance and identification of potential problems. It also helps in reducing the total cost of quality by reducing the scrap and rework, and identifying design and production defects. SAS *Field Quality Analytics* helps in making aftermarket service efficient by integrating and analyzing internal and external data sources. It helps in detecting and prioritizing warranty and service issues. SAS *Customer Intelligence 360* collects, analyzes, and reports on customer experiences to improve sales and marketing performance. It provides insight into customer segmentation: which customer groups are more likely to buy which kind of product and why. With the help of such forecasts, advertising and promotion campaigns can be planned and targeted at customer groups [213].

2.1.4 *Data Analytics in Connected Vehicles*

A connected vehicle is a vehicle designed with the capability of connecting to the internet and other connected devices including smart phones, traffic lights, other vehicles on the road, smart home appliances, etc. A study reveals that more than 286 million connected passenger cars will be added globally during the $2019-2025$ period and the revenues from connected cars globally are projected to grow five-fold, reaching over 24 billion US dollars by the year 2025 [170]. The accumulated data based on driver's behavior, car machinery, sensors installed in the car and in the surroundings can leverage data analytics in the following functional areas: autonomous driving, safety, infotainment, well-being of driver's health, vehicle management, mobility management, and smart home integration [244]. Different automotive manufactures including BMW and Volkswagen are making these connected vehicles smart by introducing functionalities like autonomous car parking and emergency assist respectively. Data analytics provides car manufacturers with crucial insights into the vehicle system, behavior of the vehicles in certain conditions, and drivers' patterns. Thousands of components inside the vehicle are continuously logging data. Even if the test driver observes an unexpected shifting characteristic, it is hard for a manufacturer to exactly find the defective component or the contributing components. But, with the help of data analytics, the defective component and the contributing components can be figured out precisely.

Ford in collaboration with IBM

Ford and IBM are working together to develop a platform which analyzes data collected from a vehicle. Based on the small chunks of vehicular data, this platform can spot patterns, correlations, and trends to help the driver make efficient transportation decisions. Data collected from Ford *Smart Mobility Experimentation Platform* helps their scientists to spot tendencies and behaviors, and their customers to have a better travel experience. They are working on using real-time analytics to learn about a problem on a particular route by taking data feed from different systems [129]. In the domain of predictive maintenance, Ford is working on sending personalized oil change and brake maintenance notifications to drivers. The collected data is statistically analyzed in order to evaluate the maintenance needs for each vehicle separately [187].

Daimler is making their cars and trucks intelligent by enabling them with anticipatory planning. Based on data from different

Daimler

sources, their vehicles are able to operate on an anticipatory basis in which they can foresee different things that the human eye cannot see. Their trucks and buses are equipped with *Predictive Powertrain Control (PPC)*, that can anticipate the terrain and adjust the vehicle accordingly. Based on the 3D map data, PPC adjusts the vehicle speed and gear selection optimally to the topography of the transport route [58]. This control reduces fuel consumption by up to 5%.

BMW in collaboration with IBM

BMW group is also using *IBM Big Data and Analytics* technology to optimize their products, repairs, and maintenance processes. *IBM SPSS* predictive analytics software is used to combine and analyze data from different sources like pre-production sensor data, workshop notes, and numerous test drives of prototypes [125]. In this way, different vulnerabilities can be identified quickly, and eliminated before the model goes into series production. Before this automated process, this evaluation took months to complete. IBM Big Data and Analytics are used to analyze data from all available sources to discover anomalous patterns and predict maintenance needs.

Volkswagen in collaboration with Microsoft

Volkswagen is introducing *Volkswagen Automotive Cloud* that is powered by Microsoft's Azure cloud and IoT Edge Platform. This cloud service allows the automotive group to leverage consistent mobility services and provide services and solutions for in-car consumer experiences, telematics, and the ability to securely connect data between the car and the cloud. Based on the Automotive cloud, the connections between vehicles, cloud-based platform, and customer-centric services will be significantly optimised [63]. By combining the customer data with vehicle data, and notes written by technicians at the service centers, upcoming maintenance can also be predicted.

Tesla

Tesla car manufacturer is collecting data from their connected cars and using telematics to batch stream key data points to back-end big data pool. The collected data enable engineers and manufacturing lines to resolve the issues and send back fixes with their over-the-air software updates. They are providing continuously improving customer experience based on the data and analytic views [85].

Audi

Audi is also making its vehicles intelligent with a vision to reduce fuel consumption. The predictive efficiency assistant enables the vehicle to slow down or automatically adjust the speed to the conditions in an anticipatory manner. The system analyzes the route topography, speed limits, road users ahead, and navigation data.

Caterpillar, Inc.

Caterpillar Inc. is the world's leading manufacturer of construction and mining equipment. They have created a new organizational division called Analytics and Innovation to form a broad and connected analytics ecosystem. The data collected from gigantic machines are used to develop predictive and prescriptive information. This predictive diagnostics is shifting their customers from reactive (repair after failure) to proactive (repair before failure) mode [93]. By using data analytics, they are able to point out inefficiencies in the operation of a particular machine by comparing its operational data with that machine's benchmark data.

2.1.5 *Data Analytics in Logistics*

Logistics service providers move masses of goods from one location to another. A lot of data related to shipments, origin, destination, size, weight, and content are stored per shipment. Some of the advantages of using data analytics in the logistics sector are:

- Optimization of delivery time, resource utilization, and geographical coverage.
- Goods storage capacity and required resources forecast.
- Valuable insight into customer sentiment and product quality.
- Insight into the global flow of goods.

DHL uses big data analytics to make their operations more efficient. Rapid processing of real-time information enables their *SmartTruck* to optimize the delivery route in real-time. Delivery routes are also automatically updated according to traffic conditions. Unsuccessful delivery attempts are avoided in intelligent routing, based on the availability and location information provided by the recipient. *SmartTrucks* are re-routed on the go, based on the combined analytics of geographical factors, environmental factors, and recipient data [131]. It is important for a logistics company to plan operational capacity in time. The optimal planning cannot be done by neglecting external factors, such as unexpected bankruptcy, a regional outbreak disease, or natural disasters etc. DHL Solutions and Innovation is working on an analytics tool to measure external factors on the expected volume of shipment to make efficient shipment volume prediction. Based on the shipment records, DHL provides an online geo marketing tool *Geovista*, to analyze business potential. This tool provides a sales forecast and local competitor analysis. DHL also offers *Supply Chain Risk Management Solution* that leverages emerging technologies to help businesses assess, predict, and mitigate evolving risks in their growing supply chain networks. This software improves the resilience of logistic providers entire supply chain with the help of predictive analytics on a global scale (by aggregating data from different local sources such as politics, economy, nature, health, etc.).

DHL

Amazon was the first company to give recommendations about items in which a user might be interested. Today, it uses different parameters (such as, which items are bought by a particular user before, what he has in his wish list and virtual cart, which items he has rated or viewed, and which items a similar user has bought) to customize the browsing and buying experience. Predictive analytics is

Amazon

used to ensure that the right item must be in stock when a customer orders it. Amazon is taking data analytics to a different level with its patent on *Anticipatory Shipping*. The patent is officially called 'Method and system for anticipatory package shipping'. The idea of anticipatory shipping is to predict who will order what and when, and then ship that item even before it is ordered. Another scenario is also discussed in patent for 'speculative shipping'. In this type of shipping, a package is sent to a geographical area, without completely specifying the delivery address at the time of shipment - the package might remain in near continuous transit on trucks until a customer makes a purchase [167]. In this way, the package is shipped to the customer instantaneously.

2.1.6 *Data Analytics in Milk Production and Cattle Monitoring*

IoT in dairy

The current trend of automation and data exchange in modern manufacturing is inextricably linked with the production industry as it helps making cars autonomous or factories more productive. Nowadays, not only these industries can benefit from IoT, but one of the oldest sector of mankind, i.e. milk production, is also taking advantage of smart technologies. For a long time, the dairy market has been suffering from low prices, which means that modern technologies and data analytics can neither influence market prices, nor the bargaining power of the dairy, nor the retail industry. However, these new technological trends can help farmers to reduce their production costs and enable them to produce more milk by keeping a keen eye on their livestocks' health.

Effects of the globalized milk market are already noticeable. Farmers are suffering mostly from the extremely sharp fall in prices. The low milk prices make it nearly impossible for farmers to obtain profits, as they are not covering costs. They are forced to optimize their production. Legal requirements and a change in social perception restricted many alternatives, like the prophylactic use of antibiotics in Europe, for optimization [84]. The only chance to raise their economic performance is to reduce costs and increase the efficiency of their production.

Tracking systems for health monitoring

The welfare of cows is of enormous importance for farmers because only healthy and happy cows give the maximum amount of milk. The farmers are able to determine the health of their cows themselves, but this is only true for small herd sizes. Farmers lack the time to monitor each cow individually in herds of dozens or hundreds of cows as can be found nowadays [249]. This is why farmers are making more frequent use of tracking systems and data analytics for the automatic health monitoring of their herd and cattle.

These tracking systems take advantage of the architecture of modern barns in Central Europe and North America, in which cows can move around freely. Everyday movement and activity behavior of cows is an important indicator of their health and whether they are in heat. In general, sick cows move less than cows without any diseases as shown in Figure 2.2. When cows are in heat, they move much more. The movement behavior is commonly measured with either accelerometers or pedometers embedded into the collar of each cow. These sensors are the central component in these systems as they are measuring the activity and vital parameters of the equipped cows continuously and autonomously.

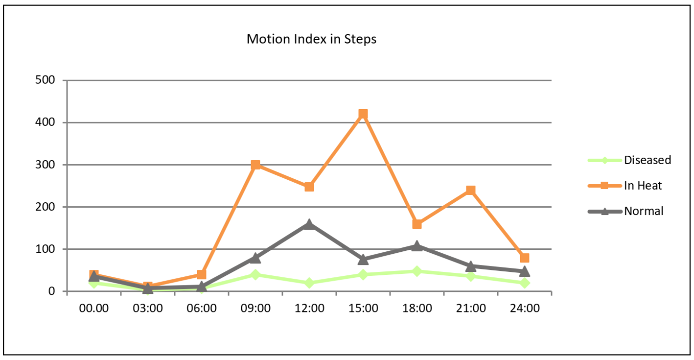


Figure 2.2: Simplified movement behavior of cows. Different conditions of cows can be classified based on their movement patterns.

In more recent times, acceleration sensors are used instead of pedometers. They are superior since they cannot only recognize the amount of activity, but also the precise type of movement: walking, running, or lying. From a data perspective, the sensors are just counting steps, which do not tell the farmer anything directly about the health of a cow. However, the number of steps per day is a strong indicator, and it is directly linked to diseases and in heat detection of individual cows in the herd. The smart dairy products are sold by *SCR Europe*, *Lely*, *DairyMaster*, and *DeLaval*. They have all placed sensors in the collars of cows and the data is transferred wirelessly to the server station (in most cases, by using proprietary radio standards). By analyzing that data, data analytics provide meaningful information about the cow's health and notify when it is in heat. SCR Europe product named *Heatime* and Lely product named *Qwes-H* also integrate rumination detection. It tells the farmer how much time each cow spends on ruminating which is an essential indicator for their health and whether they are in heat if the average time per day differs significantly [22].

SCR Europe and Lely

In the year 2014, the first tracking system based on locating cows within the barns entered the market. *Smartbow* and *CowView* draw the diagnosis from positioning data of cows instead of using pedometers or accelerometers. Both systems utilize an ultra-wide band Radio-frequency Identification (RFID) techniques in combination with an approach based on Time Difference of Arrival for locating the cows [23].

Indoor location techniques directly measure the distance traveled by cows instead of indirectly "guessing" them based on step counts or accelerometer values. Data analytics in this case works the same way as for the step count: under certain thresholds, which already had been figured out in studies a priori, cows are marked as in heat while they are classified as diseased above this threshold.

Automated in heat detection of cows

Beyond health monitoring, the determination of being in heat is a very sensitive process as the determination of the correct time is essential for a successful insemination. A failed insemination not only leads to repeated insemination costs, but also results in lower milk production. Nowadays, the insemination of cows in the dairy industry is done synthetically. In contrast to bulls (which can smell the hormones of cows and interpret their behavior), humans can only draw their conclusions based on the interpretation of their behavior. Studies show that the in heat observation plays a time-consuming role – three times a day, 15 minutes of observation are needed for complete heat detection (in addition to the normal working hours in the cowshed) [249]. It is understandable that the farmers need automated heat detection as an alternative to the time-consuming manual observation. The same kind of sensors as used for health monitoring can also be used for in heat detection. Cows in heat, feature a special characteristic in their movement behavior which significantly differs from healthy as well as diseased cows (see Figure 2.2). This movement behavior can be used to draw conclusions not only about health, but also about being in heat. The tracking systems help farmers to reduce their costs for insemination and again, increase their milk yield.

Now farmers are able to access data about the health and movement behaviors of their herd from their Personal Computer (PC), notebook, or smartphone anywhere and at anytime. More importantly, they are notified if a cow shows an abnormal pattern like a reduced feeding behavior. These alarms enable the farmer to look after their cows and call a veterinarian if required before it is too late. Not only the welfare of cows, but also the economic performance of farmers is also improved. Sick cows cause high veterinarian and drug costs for the farmers. Tackling these issues in time also leads to a better yield due to increased milk production. As a result, modern IoT-based products as well as data analytics improve the quality of dairy products and enable farmers to spend less time in the barn.

2.1.7 *Data Analytics in Smart Homes*

The Information Technology (IT) market research company, Gartner predicts that in 2022, there will be more than 500 smart objects in an average family household [78]. The smart home market is now flooded with IoT based devices. Many of the manufacturers are embedding wireless data exchange and interoperability into their devices.

Heating control is one of the areas in smart homes where people can actually save money. Products like *Thermostat+* (by ELV) and *Comet Blue* (by EUROtronic Technology) can easily be installed without even drilling a single hole [134, 135]. Heating control devices are easily plugged onto radiators, and are commonly shipped together with sensor windows (to get the knowledge if the window is close or open) and a gateway. The gateway bridges the heating control devices wirelessly so that a PC or smartphone can control the whole system. These smart devices enable customers to define the rules for temperature by the room, and to control and monitor their heating remotely from anywhere. Customers can specify the required temperature and define different time slots when they are not at home. With the help of data analytics applied on the collected data, people can analyze their habits and behaviors to save energy and more importantly for them; money.

Smart Heating

Radio-controlled sockets are cheap and small devices which can be plugged between normal sockets and the device to be powered, such as *Parce One* [136]. They are commonly equipped with Bluetooth 4.0 alias Low Energy and are easily connectible with modern Android or iPhone based smart phones. With the help of these smart sockets, i) customers can (gain the possibility to) monitor the exact power consumption of their electronic devices and, ii) they can define rules when the device gets switched on or off. With these smart sockets, all of the electrical devices can be turned into smart devices by switching them on and off autonomously. The data of the consumed energy can be analyzed per device, that gives customers the possibility to limit the use of a particular device which helps in minimizing the overall energy consumption.

Smart Sockets

The scope of smart homes is not confined only to the inside area of a home. Gardena is regularly offering new products in order to make gardens and gardening smart [103]. *Gardena's Sensor Control Set* contains a smart gateway (which has to be installed indoors and connected via Wi-Fi or cable to the network), magnetic valves for taps, and plant sensors. The plant sensor measures temperature, soil humidity, and light intensity. These values can be used to define irrigation profiles. The goal of this application is to automatically identify if the plant needs some water or fertilizer. There is a link between the level of photosynthesis within a plant and its energy supply. Once the soil is dry, the magnetic valve is automatically opened. Customers can fine-tune the irrigation rules, for instance, based on the type of plant. Aquatic plants need more water than a cactus that will survive even if the soil is dry. Environmental factors complicate the data analytics part in this application field. However, a smart irrigation system

Smart Gardens

saves a labor force and more importantly, helps plants to survive even if their owners are not present.

Now, most smart home devices are capable of measuring their surroundings, such as temperature, power consumption, or soil humidity. Additionally, they have the possibility to interact with their environment like switching off devices, activating the heating, or watering plants. What they currently lack is autonomous learning to interact with their environment based on the measured values. Nowadays, the customers still have to manually define some rules for each device. But, there are some systems which are becoming intelligent with the help of analyzing data from different sensors.

Apple HomeKit

Apple wanted to change this situation with the development of *HomeKit*: a powerful, interoperable smart home control system which is easy and fast to set up and usable on iOS devices out of the box [24]. Certified vendors and products (which are currently limited in number) can be connected to iOS over Wi-Fi or Bluetooth 4.0. Afterwards, the connected devices can be verbally configured, controlled, and monitored via Apple Siri.

Vivint SmartHome

Vivint is one of the largest home automation companies in North America. Different smart home devices including small appliances, HVAC, security systems, video devices, thermostats, smart doors and locks, smart bulbs, and smoke alarms are connected via Vivint touchscreen panel and construct a network of smart devices. That network produces a lot of streaming data, which is stored in Hadoop: an open source framework, for processing and storage of extremely large data sets. They use *Datameer* (a big data analytics platform) to shorten the time of using raw data for different analytics and actionable intelligence purposes [60]. The collected data is analyzed to better understand the usage patterns of different smart devices, which can be further used to improve the service and reduce energy consumption.

Google Nest

Google Nest offers smart devices including security cameras, thermostats, and smoke detectors. These are devices of daily use that have been in use for ages. But, data analytics and big data have changed the way these devices work. Before becoming ’smart’, these devices were used to just record videos, maintain heating to a certain level, and sound the alarm when smoke is detected, respectively. Now, by learning user behavior, Nest’s smart thermostat adapts to the user’s usage and seasonal changes. It automatically controls the temperature by learning the user schedule. By detecting unwanted events inside and outside a home, and making smart alerts, *Nest Aware* software makes security cameras intelligent. In contrast to the old security cameras that only record the video, Nest’s smart security cameras can make custom alerts for the activities a user is interested in. By making the smart notifications, Nest’s smoke detectors can tell the user (by speaking or by making mobile notification) in which room there is smoke and gives early warnings to avoid any emergency situation. It can distinguish between steam, food burn, carbon monoxide, and smoke. These smart devices can also be connected to each other to make a home safer and more secure. For example, security cameras, light bulbs, and window shades can work together to give an impression that you are at home when you are away. Or, when a thermostat is set to 'away', it can automatically turn on the security camera. By using data analytics, such smart devices can build up a profile that allows them to intelligently adjust themselves to the environment, minimize human effort, maximize human safety, improve service quality, and save energy [92].

Interoperability of smart devices

The smart home vision affords many business opportunities, but also faces many challenges. Currently, smart devices are hindered by a lack of interoperability and the communication standard between products designed by different manufacturers. There are different products which are trying to integrate and bridge as many different products, protocols, and wireless standards as possible. *Mediola Gateway V4+* produced by Mediola supports both 433 and 868 MHz [102]. The advantage is that various sensors and products of different manufacturers can interoperate which enable customers to mix them in rules and profiles. This works quite well; at least as long as Mediola supports them.

Challenges and concerns

The smart home market is a mix of many different networking technologies and protocols, that are mostly proprietary and not designed for interoperation. All producers in the domain of smart home want a big piece of the cake to consolidate their market position. Thus, they are intending to raise barriers for new producers to enter this market by using proprietary protocols and prevent interoperability between different products. From a consumer point of view, their biggest concern is data privacy. There is a need to develop a trust between the service provider and the consumer. It is very important for a consumer that the important information collected about their private life is only used to facilitate them, and not for earning money by selling that information to a third party without the consent of the consumer.

Part II

ANOMALY DETECTION

3

ANOMALY DETECTION

Anomaly detection has been one of the core research areas for a long time due to its ubiquitous nature. In everyday life, we observe the abnormalities that instantly become the focus of our attention. When something deviates largely from rest of the distribution, it is labeled as an anomaly. In computer science, anomaly detection refers to the techniques of finding specific data points, that do not conform to the normal distribution of the data set.

Versatile problem

The term 'Anomaly', is widely used and it refers to different problems in different domains. For example, an anomaly in network security system could be an activity related to a malicious software or a hacking attempt [88]. Whereas, in the manufacturing domain, a faulty product is considered as an anomaly.

Most important task in data analytics

Detecting and mitigating anomalies is vital in the manufacturing and industrial sector and also crucial in the healthcare and surveillance sector. A timely detected anomaly can improve machine performance, avoid a machinery downtime, reduce a disease outbreak, and even save a human life [183, 196]. In this era of digitization, companies from different sectors including manufacturing, automotive, healthcare, lodging, traveling, fashion, food, and logistics are investing a lot in collecting big data [50, 147]. They leverage data analytics to increase their profits and to provide their customers better user experience. In the whole data analytics process, anomaly detection is one of the most important tasks [20, 30]. In most of the cases, the collected data is streaming time-series data and due to its intrinsic characteristics, it is a challenging problem to detect anomalies in continuously changing environment.

Anomalies vs. Outliers

There is no defined distinction between anomalies and outliers. In some studies, outliers are considers as corruption in data (like measurement errors), whereas anomalies are considered as irregular data points with a specific pattern [96]. On the other hand, the following citation is referred to prove that both terms actually point to a same concept:

> *"Outliers are also referred to as abnormalities, discordants, deviants, or anomalies in the data mining and statistics literature."* – Aggarwal [3]

In the context of this thesis, both terms, anomalies and outliers are used interchangeably.

3.1 TERMINOLOGY

For the general understanding of the reader, most common terminologies used in this thesis are briefly explained in this section.

3.1.1 *Time-series*

Components/ characteristics

A series of data points indexed in time order or correlated in time compose a time-series. It is also referred to as a sequence of numeric data points recorded in successive order. A daily record of minimum and maximum temperature of New York city and daily closing value of Apple stock price are a couple of time-series examples. A time-series consist of one or more of the following components: 1) Trend, 2) Seasonality, 3) Cyclicity, 4) Structural break, and 5) White noise.

3.1.2 *Time-series Forecasting*

Forecasting Horizon

Time-series forecasting is the process of predicting future activity based on historical values and associated patterns. The number of timestamps that are intended to be predicted in the future is called time-series *Horizon*. A time-series horizon could start from 1 and go to any number. Forecast horizon of 1 means that only next timestamp is forecasted, whereas the forecast horizon of 5 means that the value of the next five timestamps will be forecasted. In some studies, all of the next five timestamps are forecasted, whereas, in others, only the fifth one is forecasted.

3.1.3 *Time-series Data Mining*

Steps involved in Data Mining

It is a process that generally consists of multiple steps including data cleaning, data integration, data selection, modeling, and evaluation. In this pipeline, the modeling step is the core step where a number of complex classification, prediction, clustering, association, and anomaly detection methods are deployed. These methods are used to extract meaningful information and examine how the changes in associated data can facilitate the decision. Time-series data mining term is also sometimes interchangeably used as 'Time-series Analysis'.

3.1.4 *Anomaly Detection*

Definition

Anomaly detection refers to the techniques of finding specific data points, that do not conform to the normal distribution of the data set. The most relevant definition of an anomaly with respect to computer science is given by Grubbs [94]: "An outlying observation, or 'outlier', is one that appears to deviate markedly from other members of the sample in which it occurs". In the context of this thesis, anomalies and outliers are used interchangeably as used by Aggarwal [3].

3.1.5 *Interpretability*

By definition, interpretability is *to explain or tell the meaning of something in an understandable way*. In the context of AI, "Interpretability is the degree to which a human can understand the cause of a decision in a given context" [178]. It is an attempt to open the machine learning black-box so that it becomes easier for an end-user to comprehend why certain decisions or predictions have been made by a machine learning model.

Opens up the machine learning black-box

3.1.6 *Explainability*

Explainability is an emerging term in the context of Explainable Artificial Intelligence (XAI) where it deals with the explanations of internal mechanics of a machine or deep learning system which eventually lead to a specific decision. In some studies, interpretability and explainability are used interchangeably. However, it is not the case in the context of this thesis. Explainability is the answer to a **WHY** question and gives reason to justify a decision. On the other hand, interpretability is about being able to discern the mechanics without explaining the reason.

Interpretability vs. explainability

3.2 FOUNDATION

In this section, the fundamental concepts and different aspects of anomalies categorizations are explained.

3.2.1 *Local vs. Global Anomalies*

Local anomalies are deviated data points with respect to their direct neighbors or inside some seasonal pattern. Whereas, global anomalies are data points that are deviated vastly from rest of the data points and it is far outside the entirety of the data set in which it is found. For the purpose of better understanding, an illustration of different clusters and anomalies is shown in Figure 3.1 [90]. Two data points, x_1 and x_2 can easily be identified as anomalies as they are far from all of the data clusters (c_1, c_2, and c_3). Based on their distinguishing attributes, these data points are called global anomalies. By considering the data set globally, x3 can be considered as a normal data point since it is not too far from the cluster c_2. However, this data point is considered as a local anomaly with regard to the cluster c_2. Data point x3 is clearly at more distance as compared to the average

Deviated data points w.r.t. the direct neighbors vs. deviated data points w.r.t. rest of the data points.

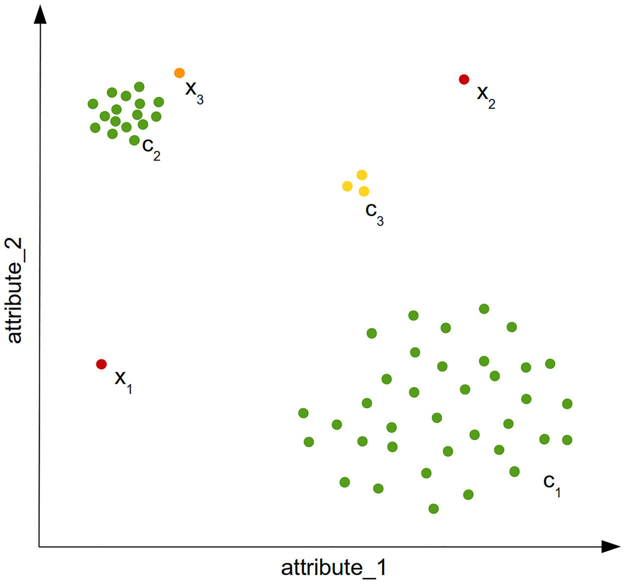


Figure 3.1: An illustration of global anomalies (x_1 and x_2), local anomaly (x_3), and micro cluster (c_3) [90].

distance between neighbors of cluster c_2. Regarding the cluster c_3, it can be considered as a normal cluster or three anomalous data points. Such clusters can be called micro cluster.

3.2.2 *Categorization of Anomaly Detection Approaches*

Due to the large variety of scenarios and use-cases, anomaly detection problem is categorized in different ways. The most common categorization is based on the level of supervision required by the algorithm:

- Supervised Anomaly Detection

  Fully labeled

  In supervised anomaly detection methods, all the data points present in training and testing data sets are fully labeled as normal or anomalous data points. This scenario is similar to most of the traditional pattern recognition task except the classes are likely to be very unbalanced. However, this scenario is not pragmatic because of the absence of anomalous data points labels in real scenarios. In most of the scenarios, anomalies are not known in advance as they might occur spontaneously.

- Semi-supervised Anomaly Detection

  Only normal data points are labeled

  Semi-supervised anomaly detection methods refer to the cases where only normal data points are labeled during the training process. It is also possible that training data set only consists of normal data points. The idea of providing only normal data points is to make the model learns only the normal behavior and whatever deviates from the learned normal behavior is considered an anomaly. In some studies, this scenario is also known as novelty detection [86]. One-class SVMs [215] and autoencoders [105] are algorithms that commonly used in this scenario.

- Unsupervised Anomaly Detection

  No labels

  It is the most flexible and practical scenario, where no labels are required at all. In this scenario, an anomaly detection algorithm is applied on unlabeled data and the algorithm identifies anomalous data points solely based on intrinsic properties of the data set. Most of the well known anomaly detection methods like LOF [31], COF [231], HBOS [89], and mostly DNN-based anomaly detection methods [173, 184, 185] lie in this category.

Anomaly detection methods can also be categorized based on the given data:

- Sequential vs. Non-sequential Data

  Input data categorization

  The choice of the anomaly detection method also depends on the input data. The input data can be divided in two major categories – Sequential and Non-sequential data. Data coming from video, speech, time-series, and text sources lie under sequential data category. Whereas, images are categorized as non-sequential data.

- Univariate vs. Multivariate data

  Data set variables

  Univariate refers to one- variable/dimension data. For example, a list of weights of all pupils in a class is a univariate data. On the other hand, multivariate data consists of data from multiple variables/dimensions. Data set comprising weight, height, age, and number of courses of all pupils in a class is an example of multivariate data.

Furthermore, anomaly detection methods can also be categorized based on the underlying methods and models [3]. For example, probabilistic models, statistical models, linear models, proximity-based, distance-based, and deep learning-based – anomaly detection. In the scope of this thesis, all the anomaly detection methods are divided into the following categories:

- Statistical Approaches
- Traditional Machine Learning Approaches
- Deep Learning Approaches

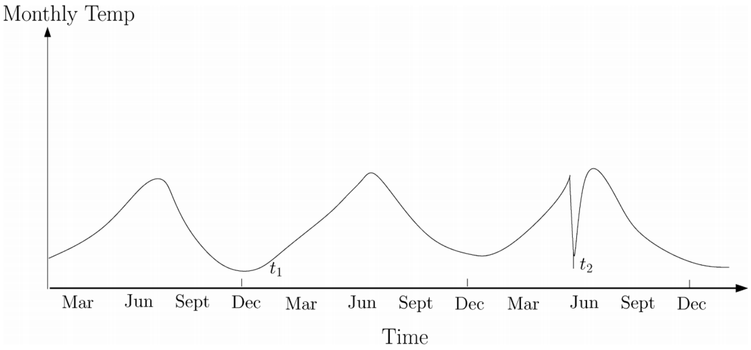


Figure 3.2: Monthly temperature over time is plotted here over months. Temperature at t_1 and t_2 are same, but t_2 occurs in a different context that makes it a contextual anomaly [42].

3.2.3 *Types of Anomalies*

Following are the three types of anomalies:

- Point Anomalies

  Deviates significantly from the rest of the data

  It is the simplest type of the anomaly where a data point is considered as anomalous if it deviates from the rest of the data. The point anomaly refers to an individual data point that stands out. In a real-life scenario, consider a fraud detection case for a credit card. Let the data set represents the daily spending. A transaction for which the spending is very high as compared to the normal range of daily spending, is considered as a point anomaly.

- Contextual Anomalies

  A normal point becomes anomalous in a specific context

  If a data point is anomalous in a certain context, but not otherwise, then that data point is known as contextual anomaly. It is important to note that, as name suggests, context is very important to define a contextual anomaly. Depending on the context and the surrounding behavior, a data point is considered anomalous, whereas it is considered normal in another context. In a real-life scenario, temperature of 35°F might be considered normal temperature during December (t_1 in Figure 3.2). Now consider having the same temperature during summers shown as t_2 in Figure 3.2. In the context of summers, such temperature is considered as an anomaly.

- Collective Anomalies

  The data points occur in a sequence

  In some cases, it is possible that a data point or a set of data points is not anomalous by itself. However, when they occur together in a collection, they are termed as collective anomalies. Figure 3.3 refers to an output of human ECG. Timestamps are shown across x-axis, whereas ECG values are shown across y-axis. ECG values shown in blue represent normal behavior, whereas the data points in red show anomaly. An individual data point from the red region is not considered anomalous if it occurs once. When these normal data points keep occurring collectively for a specific period of time, they highlight a problem in the ECG (shown in red).

3.2.4 *Time-series Characteristics*

Following are the time-series characteristics that make their analysis different from other types of data.

3.2.4.1 *Trend*

Time-series data may have a deterministic component that is proportionate to the time period. When it is the case, the time-series is said to have a trend. A trend can be positive or negative based on the increasing or decreasing values in the time-series. The stock price is a good example of a time-series that most of the time has a trend.

Refers to the deterministic component

3.2.4.2 *Seasonality*

Seasonality refers to a repeating pattern(s) in a time-series. These seasonal patterns may occur over a specific period e.g. week, month, or year. When a time-series is influenced by a seasonal factor (e.g., the quarter of the year, the month, day of the week, specific holidays like Christmas and New Year), then it is said to have regular seasonal pattern(s) in it. An example of a time-series with seasonality is retail sales of a clothing brand, that often increase between November to December due to Christmas and New Year eve.

Refers to the repeating pattern

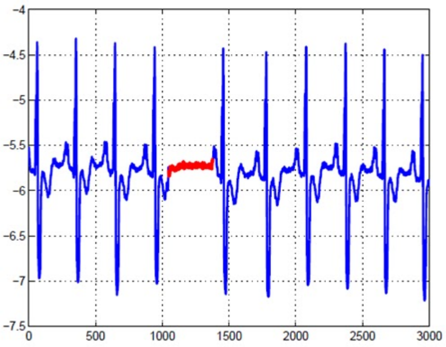


Figure 3.3: Human ECG output: Red data points show collective anomaly as the data points collectively occur for an undesired time period [42].

3.2.4.3 *Cyclicity*

Refers of the cyclic patterns

A time-series is said to have a cyclicity when a cyclic pattern exists in it. In cyclic patterns, rises and falls of data points can be observed that are not of fixed period. If the period is unchanging and associated with some aspect of the calendar and event, it is considered as seasonality. Whereas, in cyclicity, the fluctuations and cyclic patterns are not of fixed period.

3.2.4.4 *Structural Breaks*

Refers to the sudden shift

Sudden shifts in the level of the data at certain points show structural breaks in time-series. These sudden changes are also known as breakpoints. The structural break can occur due to any unexpected change.

3.2.4.5 *White Noise*

Refers to the irregular component

White noise in a time-series is due to the presence of an irregular component in it. This random variation is not explained by any other factor.

3.3 APPROACHES FOR TRADITIONAL ANOMALY DETECTION

In this section, commonly used traditional anomaly detection methods are described. These methods can be applied to time-series data, however, mostly in literature, these methods are applied on data sets that lack one or more time-series characteristics.

3.3.1 *k-nearest-neighbor (kNN) Anomaly Detection*

Targets global anomalies

The k-nearest-neighbor (kNN) Anomaly Detection [10] is one of the most commonly used distance-based anomaly detection methods. It is a simple technique that works out-of-the-box in most of the cases and detects global anomalies precisely. For each data point in a streaming data set, the k-nearest-neighbors have to be found. Based on these neighbors, the anomaly score is calculated. The anomaly score depends on the average distance to all the k neighbors. This technique is highly dependent on the value of k. If the value of this parameter is too low, the density estimation might not be reliable.

3.3.2 *Local Outlier Factor (LOF)*

Targets local anomalies

The LOF [31] is also a distance-based anomaly detection method. It is used for detecting local anomalies based on the local densities. In this method, the k-nearest-neighbors have to be found for each data point in a given streaming data set. By using k-nearest-neighbors, the local density of each data point is estimated by computing the Local Reachability Density (LRD). Finally, the anomaly score is computed by comparing the LRD of a data point with all the LRDs of its k neighbors.

3.3.3 *Connectivity-based Outlier Factor (COF)*

An extension of LOF

The connectivity-based outlier factor [231] is an improved version of LOF. In LOF, it is assumed that a given data is distributed in a spherical way around a given instance. For the cases in which this indirect condition is not fulfilled, the density estimation is incorrect and leads to poor anomaly detection. This limitation is addressed in COF by estimating the local density of the neighborhood using chaining distance. Chaining distance is a shortest-path approach which is the minimum of the sum of all distances connecting all k neighbors and the instance. Jin et al. [132] proposed Influenced Outlierness (INFLO) that is also an improved variant of LOF.

3.3.4 *Local Correlation Integral (LOCI)*

Targets local anomalies

In all of the distance-based anomaly detection approaches, the selection of parameter k plays a vital role in the overall performance. There is no fix rule on the basis of which the value of k can be estimated. So, it requires a lot of effort to come up with the best k for a particular data set. This limitation is addressed in LOCI [194] with the help of a maximization approach. It defines the r-neighborhood by using a radius r. The radius is expanded over time that makes this method very computational expensive. LOCI automatically flags outliers based on probabilistic reasoning. In addition to that, it also provides information regarding the data in the vicinity of the instance, determining clusters, micro-clusters, inter-cluster distances, and their diameters.

3.3.5 *Cluster-based Local Outlier Factor (CBLOF)*

Use clusters to determine the dense areas in the data

All the aforementioned anomaly detection methods are based on density estimation using nearest-neighbors. On the other hand, clustering based anomaly detection methods use clusters to determine the dense areas in the data. In CBLOF [107], data points are clustered using k-means (or any other) clustering algorithm. Then the set of clusters is sorted according to the amount of instances in each cluster. The anomaly score in this method is computed by the distance of each instance to its cluster center multiplied by the instances belonging to its cluster. For small clusters, the distance to the closest large cluster is used [90]. As this approach is based on clustering algorithm, the problem of choosing the right number of clusters arises, and reproduction of the same anomaly score also becomes impossible due to non-deterministic nature of clustering algorithms.

3.3.6 *Histogram-based Outlier Score (HBOS)*

Less computational expensive

It is a statistical unsupervised anomaly detection method. As the name of the method indicates, this method is based on histograms for detecting anomalies in a given streaming data. First, a histogram for each feature of the data is generated. Then the inverse height of the bins it resides of all features is multiplied for each instance of the data set. HBOS provides two histogram creation modes: i) static bin sizes with a fixed bin size and ii) dynamic bin width with a fixed amount of items in each bin [89]. This method is far less computational expensive as compared to distance-based and clustering-based anomaly detection methods.

3.4 APPROACHES FOR TIME-SERIES ANOMALY DETECTION

As the demand for time-series anomaly detection methods boosted a lot in recent years, there exists a variety of anomaly detection methods for time-series data in literature. These methods are categorized into following three categorizes as mentioned in Section 3.2.2.

3.4.1 *Statistical Approaches*

Statistical anomaly detection methods are widely used in practical use-cases and scenarios. Although some statistical approaches mentioned here are forecasting methods, the anomaly detection methods are closely linked to the forecasting methods.

3.4.1.1 *Autoregressive Model*

Autoregressive (AR) Model is a basic stochastic process used commonly for time-series. It specifies that the output variable depends linearly on its own previous (independent) values and an error value.

$$X_t = \sum_{i=1}^{p} a_i \cdot X_{t-i} + c + \varepsilon_t \tag{3.1}$$

Commonly used for time-series

The $AR(p)$ model is shown in Equation 3.1, where p shows length of preceding window. This equation can also be called AR process of order p and can be presented as $AR(p)$. X_t represents current data point and ε represents an error value. By using the training data and solving corresponding linear equation, the values of the coefficients $a_1 ... a_p$ and c can be approximated. After the approximation, ε for each X_t can be computed, that represents anomaly score [30].

3.4.1.2 *Moving Average Model*

Moving Average (MA) Model is also used commonly for time-series modeling. MA Model considers the current data point X_t as a linear combination of the last q prediction errors $\{\varepsilon_t, \varepsilon_{t-1}, ..., \varepsilon_{t-q}\}$ instead of linear transformation of the last p observations of the time-series as done in AR model.

$$X_t = \sum_{i=1}^{q} a_i \cdot \varepsilon_{t-i} + \mu + \varepsilon_t \tag{3.2}$$

The $MA(q)$ model is shown in Equation 3.2, where q shows length of preceding window. This equation can also be called MA process of order q and can be presented as $MA(q)$. Mean of time-series is represented by μ and the coefficients are learned. In MA process, it is complicated to learn the coefficients as compared to AR. In AR model, the preceding values are already known, whereas in MA model, the prediction errors $\{\varepsilon_t, \varepsilon_{t-1}, ..., \varepsilon_{t-q}\}$ are not known at beginning of the process. These error values are known after the model is fitted, that makes the optimization process sequential. Once the model is learned, the deviations from the actual data points represent anomalies.

3.4.1.3 *ARMA Model*

Autoregressive Moving Average (ARMA) is a stationary stochastic process in terms of two polynomials, one for the AR and other for the MA. It is a combination of AR and MA. An $ARMA(p, q)$ model is dependent on last p observations and q errors as shown in Equation 3.3.

$$X_t = \sum_{i=1}^{p} a_i \cdot X_{t-i} + \sum_{i=1}^{q} a_i \cdot \varepsilon_{t-i} + \varepsilon_t \tag{3.3}$$

Stationary stochastic process

The main challenge in ARMA modeling is the selection of appropriate values of p and q. If these values are too big, the model is likely to over fit that might results in too many false negatives. On the other hand, if these values are too small, the model is likely to underfit resulting false positives.

3.4.1.4 *ARIMA Model*

Addresses the issue of non-stationary data

In all of the aforementioned statistical approaches, it is mandatory to have a stationary time-series. However, in real-life scenarios, mostly non-stationary data is only available for analysis. To address this issue, Autoregressive Integrated Moving Average (ARIMA) was introduced which is a generalization of ARMA model. As name suggests, ARIMA consists of the following three parts: AR, *'Integrated'*, and MA. The difference between ARIMA and ARMA is the presence of the *'Integrated'* part in ARIMA. This part, also referred by d, is responsible for making a non-stationary time-series a stationary time-series. The stationarity is achieved by replacing the data values with the difference between their values and the previous values, this process is called differencing process.The parameter d defines the number of times the time-series is differenced. Non-seasonal ARIMA models are generally denoted by $ARIMA(p, d, q)$, where parameters p, d, and q are non-negative integers. After fitting the ARIMA model, the deviation of the predicted point to the observed point is considered as an anomaly.

3.4.1.5 *Seasonal Hybrid Extreme Studentized Deviate Model*

Based on ESD and STL

Seasonal Hybrid Extreme Studentized Deviate (S-H-ESD) [113] is a statistical technique for automatically detecting anomalies in time-series data. This technique is based on generalized Extreme Studentized Deviate (ESD) [204] to handle more than one outliers, and Seasonal and Trend Decomposition using Loess (STL) [49] to deal with the decomposition of time-series data and seasonality trends. ESD computes k test statistics iteratively to detect k point outliers. At each iteration, the most outlying observation is removed. Mean and standard deviation are used in ESD, that are sensitive to anomalous data. Whereas in S-H-ESD, statistics median and Mean Absolute Deviation (MAD) is used to detect anomalies. By using these metrics, the number of false positives detected by the model could be minimized, especially in the data sets with larger number of anomalies. S-H-ESD is computationally expensive as compared to ESD, but it is more robust to higher percentage of anomalies.

3.4.1.6 *Twitter Anomaly Detection*

For both, local and global anomalies

Twitter Inc. open-sourced its anomaly detection package[1] [138] in 2015, that is based on S-H-ESD algorithm. This method can detect both local and global anomalies. For long time-series such as six months of minutely data, the algorithm employs piecewise approximation. The following two anomaly detection functions for detecting anomalies in seasonal univariate time-series are provided:

- ***AnomalyDetectionTS*** function is used when input is a series of $< timestamp, value >$ pairs.
- ***AnomalyDetectionVec*** function is used when input is a series of observations.

3.4.1.7 *Yahoo EGADS*

Extensible Generic Anomaly Detection System (EGADS) [155] detects anomalies in large scale time-series data. It was released[2] by Yahoo Labs. EGADS consists of two main components: Time-series Modeling Module (TMM) and Anomaly Detection Module (ADM). For a given time-series, TMM models the time-series and produces an expected

---

[1] Source code of Twitter Anomaly Detection:
`https://github.com/twitter/AnomalyDetection/releases`

[2] EGADS Java Library:
`https://github.com/yahoo/egads`

value u_t at a timestamp t for a data point x_t. ADM compares the expected value with the actual value and computes number of errors E. The prediction error mentioned in Equation 3.4 is used as a notion of deviation and the relative error mention in Equation 3.5 is used for detecting errors or anomalies. Anomalous data points are marked by thresholding the relative error, RE_t.

Consists of two modules, TMM and ADM

$$PE_t = x_t - u_t \tag{3.4}$$

$$RE_t = \frac{x_t - u_t}{u_t} \tag{3.5}$$

An end user can pick one of the time-series model in TMM component: Olympic Model, MA Model, Exponential Smoothing Model, Regression Models, ARIMA, (T)BATS Family, and Spectral Kalman Filter. TMM component can be extended by adding more models. Most of the models used in TMM component are statistic-based models, but other models can also be added to this component. Furthermore, following three models are used in ADM component to detect anomalies: Extreme Low Density Model Outlier, Kernel-based Change Point Detection, and K-Sigma Model Outlier.

3.4.2 *Machine Learning Approaches*

In this section, classical machine learning approaches commonly used for anomaly detection are discussed. In contrast to the statistical anomaly detection approaches, machine learning approaches try to detect anomalies without assuming a specific generative model. Generally, it is assumed in statistical approaches that the data is generated by a specific statistical model. On the other hand, machine learning approaches try to learn only from the given data and consider the data generation process as a black-box [30].

3.4.2.1 *iForest*

iForest [166] anomaly detection method is based on the concept of 'isolation'; in contrast to the widely-used distance and density measures. In this approach, the anomalies are 'isolated' from normal instances. The data instances that are few in numbers and their attribute-values are very different from the rest of the data instances are the instances that are more susceptible to be put in isolation. This method uses a binary tree structure called isolation tree (`iTree`) to isolate such instances. The anomalous instances are more likely to be isolated closer to the root of an `iTree`.

Based on the concept of 'isolation'

3.4.2.2 *One-Class Support Vector Machines*

One-Class Support Vector Machines (OC-SVM) is based on the idea of Support Vector Machines (SVM). SVM was proposed by Vapnik and Chervonenkis [242] as a linear supervised approach. This algorithm was further extended by Boser, Guyon, and Vapnik [28]. They made SVM capable of making non-linear classification by introducing the kernel trick. Furthermore, in the context of novelties detection, Schölkopf et al. [215] introduced OC-SVM. OC-SVM is an unsupervised approach, where only one class, i.e. normal data points are required for the training step. Once the model is fitted on the given normal data, test data is classified as similar to the normal data or not. The data points that are not close to the normal data are considered as novelties. The basic idea of this machine learning-based approach is to learn a decision boundary that achieves the maximum separation between the points and the origin. Generally, OC-SVM is sensitive to the outliers when no labels are given. To tackle this shortcoming, Amer, Goldstein, and Abdennadher [8] enhanced OC-SVM for unsupervised anomalies by proposing two modifications that make the outliers contribute less to the decision boundary as compared to the normal instances. Hu et al. [116] proposed an anomaly detection method for detecting abnormal sub-sequences in a given time-series. First, the meta-features of a given univariate or multivariate time-series are generated and then the outliers are detected based on the OC-SVM optimized on the transformed samples.

Only normal class is required for training step

3.4.2.3 *Principle Component Analysis*

Principle Component Analysis (PCA) is a linear dimensionality reduction method that projects data to a lower dimensional space by using singular value decomposition. The possible correlated variables are converted into a set of linearly uncorrelated variables called major and minor principal components, known as principle components. In the context of anomaly detection, major components can show global deviations from the majority of data, whereas minor components can show local deviations. Shyu et al. [220] proposed an anomaly detection method based on PCA. The predictive model is generated based on the major and minor principal components of the normal data. Kwitt and Hofmann [152] proposed a Robust Principle Component Analysis (rPCA) method based on PCA. In this method, Minimum Covariance Determinant (MCD) is employed for the computation of covariance and correlation matrix.

Major components can show global deviations and minor components can show local deviations

3.4.2.4 *Extreme Gradient Boosting*

Extreme Gradient Boosting (XGBoost) [47] is a machine learning approach based on the Tree boosting algorithm used to solve many data science problems in a fast and accurate way. The main advantage of using XGBoost is its scalability. Extreme Gradient Boosting Outlier Detection (XGBOD) [266] is an ensembling method based on XGBoost. It is a relatively new semi-supervised method for detecting anomalies. XGBOD combines the strengths of both supervised and unsupervised machine learning methods that exploit each of their individual performance capabilities in anomaly detection. It ensembles multiple unsupervised outlier mining methods to extract useful representations of the provided data. XGBOD is a three-phase framework. In first phase, various outlier detection methods are applied to the original data to get transformed outlier scores. In the second phase, only the useful outlier scores are kept and combined with the original features, that creates a new feature space. In the last phase, XGBoost classifier is trained on the new feature space and its output is regarded as the prediction result. The predictive capabilities of this method are improved as compared to the other ensembling methods by using stacking-based outlier ensembling.

Three-phase framework

3.4.2.5 *Hierarchical Temporal Memory*

Hierarchical Temporal Memory (HTM) is a theory of intelligence based on neuroscience research. The neocortex is the seat of intelligence in the brain, and it is structurally homogeneous throughout. This means that a common algorithm is processing all the sensory input regardless which sense is sending the data. Following the same principles, in computer science domain, there are learning algorithms at the core of HTM that can store, learn, infer, and recall high-order sequences. HTM networks continuously learn time-based patters and model the spatio-temporal characteristics of their inputs. The real-time implementations of HTM are well suited for predictions task [56, 193]. However, they do not model anomalies and do not provide a mechanism for a usable anomaly score. To adapt it for the purpose of anomaly detection, two anomaly detection methods Numenta and NumentaTM [4, 157] are proposed by Numenta[3]. These techniques model the temporal sequences in a given data stream. At a given time t, HTM makes multiple predictions for next timestamp. These predictions are further compared with actual value to determine if a value is normal or anomalous. For each timestamp, anomaly likelihood score is calculated that is thresholded to finally reach a conclusion regarding the presence or absence of anomaly.

Models the spatio-temporal characteristics

3.4.2.6 *Density-Based Spatial Clustering of Applications with Noise*

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [72] is a clustering based outlier detection method. It is based on the idea of clusters and noise as it can be assumed that clusters can be of any arbitrary shape and data may contain noise.

---

[3] `https://numenta.org/`

Based on the idea of clusters and noise

For each point of the cluster, the neighborhood of a given radius has to contain at lest a minimum number of points. This algorithm has two parameters: ϵ and μ, where ϵ defines the neighborhood around a data point and μ is the minimum number of points each cluster must have. This method classifies a data point into three categories:

- Core Point:
  A point is classified as a core point if it has more than μ within ϵ.
- Border Point:
  A point is classified as a border point when it has fewer than μ within ϵ but it is in the neighborhood of a core point.
- Anomaly or Noise:
  A point that is not a core point nor a border point is classified as an anomaly.

Çelik, Dadaşer-Çelik, and Dokuz [39] used DBSCAN for anomaly detection in time-series data. They have used deseasoned temperature data for evaluation and comparison with statistical anomaly detection approaches. This study shows that DBSCAN is capable of detecting anomalies even when they are not extreme values.

3.4.3 *Deep Learning Approaches*

DNNs have been widely used for computer vision tasks including classification, object detection, and segmentation because of their exceptional performance. In recent years, DNNs are also getting famous for time-series anomaly detection task. They are similar to machine learning techniques as they also do not rely on the underlying data generation process.

3.4.3.1 *Multilayer Perceptron*

Feed-forward neural network

Multilayer Perceptron (MLP) belongs to the family of feed-forward neural networks. It is a basic Artificial Neural Network (ANN) that consists of at least three layers of nodes: input layer, hidden layer, and output layer. Each layer requires input from the previous layer. It utilizes supervised learning scheme called backpropagation for training and the parameters are learned from the data. This ANN can be used for time-series data. Like AR model, lagged values of a time-series can be used as input to a neural network. Hyndman and Athanasopoulos [120] called such network a Neural Network Autoregression (NNAR). This feed-forward network with one hidden layer is represented by $\mathrm{NNAR}(p, k)$, where p is number of lagged inputs and k is number of nodes in hidden layer. p can also be considered as a window size of sliding window used in time-series and window size is equal to number of neuron in the input layer. The network is applied iteratively for the purpose of forecasting. For one step ahead forecasting, the available historical inputs are used. By finding the error between forecasted and the actual value, anomalous data points can be detected using this approach. Haselsteiner and Pfurtscheller [104] used MLP for time-series classification.

3.4.3.2 *Convolutional Neural Networks*

Convolutional Neural Network (CNN) also belongs to the family of feed-forward neural networks. Convolutional networks were inspired by biological processes of visual cortex in which the visual information is passed from one cortical area to another and each cortical area is more specialized. The neurons in the specific visual field (also known as receptive field) only respond to the specific actions. The entire visual field is eventually covered as the receptive fields of different neurons partially overlap. In the context of AI, the heart of the CNN is a mathematical operation called *convolution*. A convolution is a linear operation that involves the multiplication of a set of weights with the input. The objective of the convolution is to extract the high-level features from the given input. Three main types of layers: *convolutional layers*, *pooling layers* and *fully-connected layers* are used to build CNN architectures in addition to input and output layers. One of the main advantage of CNN is that it reduces the input in a form that is easier to process, yet keeping all the important (feature) information intact. This makes the overall learning process efficient.

Inspired by the biological processes of visual cortex

CNNs have a lot of applications in image and video recognition, recommender systems, image classification, medical image analysis, natural language processing, and many more. Due to their success in a wide range of domains, there has been an increasing interest in deploying CNNs for time-series use-cases. Zheng et al. [268] used CNN for multivariate time-series classification. They proposed Multi Channel Deep Convolutional Neural Network (MC-DCNN) where each channel takes a single dimension of multivariate time-series data as input, and learns the features individually. This is followed by a layer of MLP to perform classification. Experimental results show that MC-DCNN outperforms competing baseline method that is K-nearest neighbor (based on Euclidean Distance and Dynamic Time Warping (DTW)). Instead of separating multivariate time-series into univariate for individually learning features, Zhao et al. [264] proposed a CNN framework for time-series classification in which the multivariate time-series is jointly trained for feature extraction. They alternatively used two convolution and two pooling layers to generate raw features of the given data set. Furthermore, the features are connected to a MLP for classification. Cui, Chen, and Chen [57] proposed a Multi-scale Convolutional Neural Network (MCNN) for time-series classification. The input is fed into multiple branches that are a set of consecutive convolutional and pooling layers. These branches perform different transformations of the time-series that helps in extracting features of different frequency and time scale.

3.4.3.3 *Long Short Term Memory*

Long short-term memory (LSTM) belongs to the family of Recurrent Neural Network (RNN). In contrast to the feed-forward neural networks, RNNs have a feedback connection that enable them to use the current output for the next input. The vanilla RNNs suffer from short-term memory due to the vanishing gradient problem. If a sequence is long enough, it will be hard for a RNN to carry information from the earlier timestamps to later ones. This limitation is addressed in LSTM [114] that is generally composed of a cell and three gates i.e. input gate, output gate, and forget gate. The purpose of these gates is to learn which data in a given sequence is important to keep and which information is unnecessary to keep. By doing so, it can pass the relevant and useful information down the long chain of sequences.

Recurrent neural network

LSTMs are applied to a wide range of problems including speech recognition, language modeling, translation, and image captioning. Due to the recurrent manner, LSTMs are used for univariate and multivariate time-series analysis. The anomaly detection technique proposed by Malhotra et al. [174] is based on stacked LSTMs model that is composed of two hidden LSTM layers. Their predictive model is trained on normal timestamps, that is further used to compute error vectors for given sequences. Based on the error threshold, a time-series sequence is marked as normal or anomalous. Chauhan and Vig [46] used similar approach to detect anomalies in ECG data. They used RNN, augmented with LSTM, to detect 4 different types of anomalies. The network is trained on non-anomalous data and used as a predictor. The prediction errors are used to fit a multivariate Gaussian distribution and a probability is assigned to each observation. Based on the probability and the threshold, anomalous behaviors are detected. Lipton et al. [164] employed LSTM to classify a time-series as normal or abnormal. They demonstrated that LSTM trained on only raw time-series with target replication outperforms MLP trained on hand engineered features. The evaluation was provided on an anonymized clinical data set.

3.4.3.4 *Autoencoder*

Autoencoder (AE) is a type of neural network which is used to learn efficient data codings in an unsupervised way. They belong to the family of feed-forward neural networks. AE tries to learn an approximation to the identity function, so that the output is similar to the input. It consists of two parts: encoder and decoder. The network learns how to efficiently compress the data (encoder) and how to reconstruct the data back to a representation close to the input data (decoder). The trick lies in limiting the number of hidden units; also referred to as *bottleneck*, that helps in discovering the interesting structure about the data. A simple AE might end up learning a low-dimensional representation similar to a PCA. In AE-based anomaly detection, AEs are used to detect anomalous instances by calculating the reconstruction error. Schreyer et al. [216] used deep AEs to detect anomalies in large-scale accounting data in the area of fraud detection. Amarbayasgalan, Jargalsaikhan, and Ryu [7] also proposed a novelty detection technique based on deep AEs. Their approach computes the error threshold from deep AE model and passes to a density-based cluster. Then, density-based clustering is applied to the compressed data to get novelty groups with low density. Sakurada and Yairi [208] proposed an approach to detect anomalies in time-series using AE and compared the results with linear PCA and kernel PCA.

Learns efficient data codings in an unsupervised way

3.5 DATA SETS

3.5.1 *Types of Data*

In this section, different characteristics of a data set are described that are generally considered for an anomaly detection method's evaluation.

3.5.1.1 *Labeled vs Unlabeled Data*

A data set is considered labeled when an annotation exists for each data point present in the data set. A data point can be normal or anomalous. When this information is attached to each data point, a supervised algorithm can learn this information which enables it to detect anomalies. A data set can consists of only normal data points, in this case, semi-supervised methods can be used to detect anomalies. On the other hand, unlabeled data is used by unsupervised anomaly detection methods.

Availability of data annotations

3.5.1.2 *Real vs Synthetic Data*

The data which comes directly from a scenario where system is up and running. For instance, data captured from a machine in an industrial setting is a real data. Real data captures the actual working setting which might be keep on changing over time. On the other hand, data captured by running a simulation of a particular scenario is considered as synthetic data. It is an alternate to recording real data, as it provides full control over the scenario and configurations. Also, sometime it is expensive to record real data, whereas synthetic data is inexpensive to generate and number of anomalies can be controlled in the generation process.

For example, data coming directly from an industry vs simulated data

3.5.2 *Data Sets Included in Thesis*

In this section, real and synthetic anomaly detection data sets which are used in this thesis are explained. Some of these data sets are also most commonly used data sets in anomaly detection studies.

3.5.2.1 *HVAC*

Real operational data

This data set contains real operational data gathered from 77 HVAC systems, that were operational at different locations in Germany. Data[4] collected from the household boilers over the span of 2.5 years are used to find operational anomalies. To analyze and control the behavior of a HVAC system, each system is equipped with a lot of sensors. These 'smart', IoT based HVAC systems transmit all of the operational data to a back-end database. A specific sensor used in boilers is analyzed in the scope of this thesis, which generates a univariate time-series. Each time-series generated by a HVAC system is treated separately. However, here the goal is to have same anomaly detection setting/threshold along all the systems. This data set contains some long-term anomalies; it means that the observed sensor is continuously and slowly deviating from its normal behavior. This kind of long-term or slow moving data anomalies are not easy to detect.

3.5.2.2 *ECG*

Real data

An ECG time-series from MIT-BIH ECG Database, available at PhysioNet [87] is also used in this thesis. According to PhysioNet, the ECG readings were digitized at 360 samples per second per channel with 11-bit resolution over a 10 Millivolts (mV) range. Two or more cardiologists independently annotated each record. Record numbered 108 is used in this thesis from this database[5]. It is a univariate time-series with 2160 data points. This time-series contains different ventricular abnormalities, which are represented as contextual anomalies.

3.5.2.3 *Yahoo Webscope A1*

Real network-traffic data

Yahoo Webscope A1 data set is a part of Yahoo Webscope Benchmark [257], open-sourced by Yahoo Labs. Yahoo Webscope Benchmark is a popular anomaly detection benchmark for time-series. The univariant, Yahoo Webscope A1 data set contains real network-traffic data to Yahoo log-in services. This data set is comprised of 67 different time-series that are annotated by humans. The hourly data is recorded

---

[4] Due to the data protection and privacy policy of the affiliated company, we are unable to open-source the data or provide details of the observed sensor.

[5] Direct link to database:
`https://www.physionet.org/physiobank/database/mitdb/`

in each time-series and each time-series consists of about 1400 data instances. There are almost 1.9% anomalies in this data set.

3.5.2.4 *Yahoo Webscope A2*

Yahoo Webscope A2 data set is also a part of Yahoo Webscope Benchmark [257]. This synthetic data set is comprised of 100 time-series. Each time-series consists of about 1421 time instances. Anomalies are inserted randomly by the publishers in this data set and there are almost 0.3% anomalies in this data set.

Synthetic data

3.5.2.5 *Yahoo Webscope A3*

Another part of Yahoo Webscope Benchmark [257] is Yahoo Webscope A3 data set. This synthetic data set is comprised of 100 time-series where each time-series consists of about 1680 data instances. In contrast to the Yahoo Webscope A1 and Yahoo Webscope A2 data sets, the time-series present in this data set have seasonality in addition to anomalies. The presence of this time-series characteristic makes the anomaly detection process hard in this data set. There are almost 0.3% anomalies in this data set.

Synthetic data

3.5.2.6 *Yahoo Webscope A4*

This data set is also part of Yahoo Webscope Benchmark [257]. It is also a synthetic data set having 100 time-series. Each time-series consists of 1680 data instances. There are almost 0.5% anomalies in this data set. The time-series present in this data set contain change-point anomalies. Other time-series characteristics are also present in this data like trend and noise.

Synthetic data

3.5.2.7 *NAB AWS Cloud Watch*

Amazon Web Services (AWS) Cloud Watch data set [157] is part of Numenta Anomaly Benchmark (NAB). NAB is an anomaly detection benchmark open-sourced by Numenta in 2015. Different AWS server metrics like *Central Processing Unit (CPU) Utilization*, *Network Bytes In*, and *Disk Read Bytes* are recorded in AWS Cloud Watch data set. These metrics are recorded with the help of *AmazonCloudwatch* service. This data set consists of 17 real time-series.

Real data

3.5.2.8 *NAB Ad Exchange*

Ad Exchange data set [157] is also part of the NAB. This data set records the online advertisement clicking rates. Two metrics, cost-per-click (CPC) and cost-per-thousand impressions (CPM) are included in this data set. This data set consists of 6 real time-series.

Real data

3.5.2.9 *NAB Known Cause*

Real data

As a part of NAB, Known Cause data set [157] consists of time-series for which the cause of anomaly is known. This data set is not hand labeled. For example, one time-series records the temperature sensor data of an internal component of a large industrial machine. First anomaly in this time-series is a planned shutdown of the machine. Another time-series in this data set records the number of taxi passengers in New York City (NYC). Here, five anomalies occur during NYC marathon, thanksgiving, Christmas, new years day, and a snow storm. There are 7 real time-series in this data set.

3.5.2.10 *NAB Traffic*

Real data

Traffic data set [157] is a real time-series data set which is also part of NAB. Real-time traffic data from Twin Cities Metro area in Minnesota, USA are recorded in this data set. The recorded metrics include occupancy, speed, and travel time. There are 7 real time-series in this data set.

3.5.2.11 *NAB Tweets*

Real data

Tweets data set [157] is another real data set provided in NAB. This data set is a collection of Tweets that mention large publicly-traded companies like Google, Facebook, and IBM. There are 10 time-series in this data set.

3.5.2.12 *NAB no Anomaly*

Synthetic data

No Anomaly data set [157] in NAB is a synthetic time-series data that contains no anomaly in it. There are 5 time-series in this data set.

3.5.2.13 *NAB with Anomaly*

Synthetic data

With Anomaly data set [157] is also part of NAB. This data set consists of artificially-generated time-series with different types of anomalies. There are 6 time-series in this data set.

3.5.2.14 *Shuttle*

Real data

Shuttle is NASA's shuttle data set that is already divided into train and test set by the publisher. This real data set is available at UCI Machine Learning Repository [65] and OpenML [241]. All the data instances that belong to class 4 are removed as it is done by Liu, Ting, and Zhou [165]. Rest of the classes except class 1, are treated as anomalies. This multivariate time-series consists of 9 features and it has 49097 data instances. There are 7% anomalies in this time-series.

3.5.2.15 *Pima*

Pima is a diabetes data set collected at the National Institute of Diabetes and Digestive and Kidney Diseases, USA. This real data set is available at UCI Machine Learning Repository [65] and OpenML [241]. Pima is a diagnostic data set that shows if a patient has signs of diabetes or not. The target value 'pos' indicates that a patient is suffering from diabetes and the corresponding data point is treated as anomalous. This multivariate time-series consists of 8 features and it has 768 data instances. There are 34.9% anomalies in this time-series.

Real data

3.5.2.16 *Forest Cover*

Forest Cover data set (also known as Covertype in UCI repository) has target values in integers, which are different tree species. This real data set is available at UCI Machine Learning Repository [65] and OpenML [241]. The data set comprised of 54 features in total, where 44 features are categorical. Therefore, we only use 10 non-categorical features for training a model. Out of the 7 target classes, we use 2 classes as done in [165]. All the instances from class 4 are considered anomalous, while instances from class 2 are considered normal. This multivariate time-series has 286048 data instances and there are 0.96% anomalies in this time-series.

Real data

3.5.2.17 *Ionosphere*

Ionosphere is a radar data set. The target attribute is ionosphere, which is considered as 'good' if radar shows evidence of some type of structure in the ionosphere, otherwise it is considered as 'bad'. 'Bad' ionospheres are considered as anomalous. This real data set is available at UCI Machine Learning Repository [65] and OpenML [241]. Ionosphere is a multivariate time-series which consists of 32 features and 351 data instances. There are 36% anomalies in this time-series.

Real radar data

3.5.2.18 *HTTP*

HTTP is a subset of KDD CUP '99 network intrusion data. A wide variety of anomalies (i.e. network attack) were hand-injected in the normal network data. Th data set is used in a lot of studies. We have used this data set in a standard way described in [258]. It can be downloaded from UCI Machine Learning Repository [65] and OpenML [241]. This multivariate time-series consists of 3 features and it has 567497 data instances. There are 0.39% anomalies in this time-series.

A subset of KDD CUP '99 network intrusion data

3.5.2.19 *SMTP*

SMTP is also a subset of KDD CUP '99 network intrusion data. This data set is also used as described in [258]. It can be downloaded

A subset of KDD CUP '99 network intrusion data from UCI Machine Learning Repository [65] and OpenML [241]. Attack class is considered as an anomalous class. This multivariate time-series consists of 3 features and it has 95156 data instances. There are 0.03% anomalies in this time-series.

3.5.2.20 *Mulcross*

Synthetic data

Mulcross data set is obtained from a synthetic data generator known as Mulcross. Mulcross [203] generates a multi-variate normal distribution with a selectable number of anomaly clusters. We have used same settings (contamination ratio, distance factor, and anomaly clusters) for this data set as mentioned in [165]. It can be downloaded from UCI Machine Learning Repository [65] and OpenML [241]. This multivariate time-series consists of 4 features and it has 262144 data instances. There are 10% anomalies in this time-series.

3.5.2.21 *Mammography*

Real data

Mammography data set is publicly available at OpenML [241]. All the data instances with class value 1 are considered anomalous. This multivariate time-series consists of 6 features and it has 11183 data instances. There are 2% anomalies in this time-series.

3.5.2.22 *NASA Shuttle*

Current measurements on a Marotta MPV-41 series valve

NASA space shuttle valve data set [74] consists of multiple time-series. The time-series in this data set are current measurements on a *Marotta MPV-41* series valve. These valves are used to control flow of fuel on the space shuttle. This data set is different from aforementioned data sets as whole time-series is labeled as normal or abnormal sequences.

3.6 EVALUATION METRICS

The evaluation metrics used in this thesis are explained in this section.

3.6.1 *Precision*

Fraction of relevant instances among the retrieved instances

As name of this metric implies, it is the fraction of relevant instances among the retrieved instances. In other words, it highlights the proportion of positive identifications that were actually correct. In the form of a formula, precision is a ratio of True Positive (TP) to the total of the TPs and False Positives (FPs) as shown in Equation 3.6. If there are no false positives, then the precision of the employed model will have the maximum value of 1. Based on the false positives, precision will decrease.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3.6}$$

3.6.2 *Recall*

Recall is the fraction of retrieved instances among all relevant instances. In literal terms, recall highlights how many of the TPs are actually found or recalled. The recall rate is penalized whenever a False Negative (FN) is predicted. Recall is defined in Equation 3.7. The maximum recall value of 1 shows that a model has no false negative. Recall is also known as True Positive Rate (TPR).

Fraction of retrieved instances among all relevant instances

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3.7}$$

3.6.3 *F-score*

F-score is a one of the best singleton metrics which serves as a good indicator of the model's performance. Precision and recall highlights different aspects of a system. To combine these different aspects of a system, F-score is used. This score is also defined as harmonic mean of a model's precision and recall as shown in Equation 3.8. The highest possible value of F-score is 1, which indicates best precision and recall. If either of the precision or recall is 0, the F-score is 0, that is the lowest value of this metric.

Combines different aspects of a system

$$\text{F-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3.8}$$

In some cases, it is required to weight precision or recall more highly than the other. The weighted F-score is shown in Equation 3.9. By setting β to 2 in Equation 3.9, recall is considered twice as important as precision. If β is set to 1, this equation becomes equivalent to Equation 3.8.

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \tag{3.9}$$

3.6.4 *False Positive Rate*

The False Positive Rate (FPR) measures the ratio of false positives within the negative samples. It is the proportion of negative cases incorrectly identified as positive cases in the data, that's why, this measure is also known as *false alarm ratio*. The FPR is calculated as the ratio between FPs and the total number of ground truth negatives (i.e. FP and True Negative (TN)). FPR is defined in Equation 3.10.

Ratio of false positives within the negative samples

$$\mathsf{FPR} = \frac{\mathsf{FP}}{\mathsf{FP} + \mathsf{TN}} \tag{3.10}$$

3.6.5 *ROC*

Receiver Operating Characteristic (ROC) curve is a graphical representation of a classification model which shows the diagnostic ability of that model. The ROC curve is created by plotting TPR against FPR for different threshold settings, providing a broader overview of an algorithm's classification capability. This metric helps in uncovering the maximum potential of an algorithm whose performance is dependent on selecting the best threshold. The best threshold depends upon the need and other criteria for a use-case, such as the maximum number of true positives and the minimum number of false alarms. By lowering the threshold, more items are classified as positive (anomalies in the context of this thesis) which increase both false positives and true positives. In the context of this thesis, ROC is referred to ROC-AUC where ROC is a probability curve and AUC represents degree or measure of separability. The ROC value near to 1 represents a good measure of separability.

Shows the diagnostic ability of a model

3.6.6 *Precision-Recall Curve*

To have a better performance overview of a system with an unbalanced class distribution, Precision-Recall (PR) curve is preferred. Precision-Recall (PR) curve is a plot of the precision against the recall for different thresholds. In the context of this thesis, the area under the Precision-Recall (PR) curve is reported which ranges from 0 to 1. A value close to 1 represents an accurate classifier.

Plot of the precision against the recall

3.6.7 *Inference Time*

The time required to 'infer' the results on a given test set is referred here as an inference time. The machine used to compute inference time was equipped with Intel Xeon(R) processor with 8 cores and one NVIDIA GeForce GTX 1070 GPU.

4

CONTEXTUAL ANOMALY DETECTION IN HVAC SYSTEMS

Anomalies have always been of great interest to humans. In everyday life, we observe that abnormal things and happenings attract our attention. In some cases, anomaly detection is also used as leading indicator of unwanted events, also known as early warning.

IoT-based household devices

Each year, the usage trend, as well as the number of IoT-based household devices (e.g., smart heaters, smart air conditioners, smart fire alarms, smart TVs, and smart security cameras), are enormously increasing. According to a study released by Statistica [229], HVAC market size worldwide is expected to reach 367.5 billion US dollars by the year 2030. It is also projected that HVAC market volume worldwide will reach to 151 million units by the year 2024.

Shift of trend in IoT-based household devices

Data analytics on IoT-based HVAC devices

Figure 4.1 shows the current share of internet-connectable household devices. A clear shift from traditional computer network equipment to other smart and IoT-based household devices can be seen in this figure. In 2010, the share of personal computers and wireless routers (which lie in computer network equipment category) was about 75% of all household devices, which was decreased to 45% in 2016 and further decreased to about 25% by 2020. On the other hand, the share of internet enabled HVAC systems is increasing day by day. It is clear from this figure that the dominating segment in IoT-based household devices in the year 2020 is HVAC systems with a total share of almost 35% [192]. HVAC manufacturers have already enabled their devices with internet connectivity; so that they can store and analyze the operational and usage log of their devices. This connectivity with the surrounding environment and the back-end servers has opened new horizons for HVAC manufacturers and their users. The analysis of the collected data enables HVAC manufacturers and third-party companies to offer new services specifically tailored according to the customers' need. For service providers and manufacturers, the operational data can be used to monitor the internal state of a system, and to directly identify the root cause of a problem in case of system breakdown. By applying predictive data analytics on the devices' internal data, service provider can offer predictive maintenance even before the actual issue arises in a system.

An important use case of analyzing the internal state of HVAC systems is to find the anomalies in the running system and finally reach the root cause of the problem. The early correct detection of such anomalies is the basis of predictive maintenance.

This chapter is an adapted version of the work published in [183].

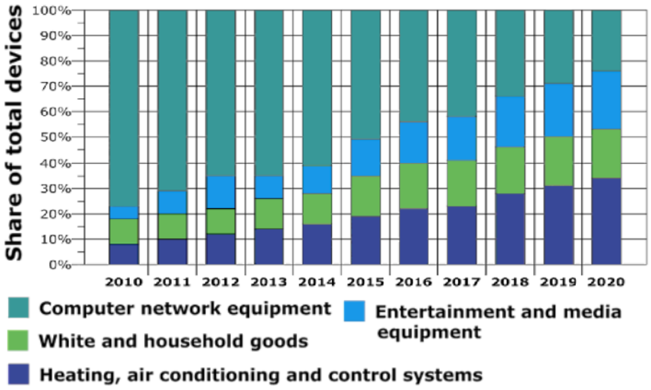


Figure 4.1: Share of Internet-connectable consumer household devices [192].

Long-term anomalies

In the context of HVAC systems, the long-term anomalies are of great interest, where the sensor value is deviating from its normal behavior continuously and slowly. Such anomalies can be detected if data is analyzed in the context of the previous data. At this point, most of the existing distance-based and statistical-based anomaly detection techniques fail. The main reason of this failure is the co-existence of a lot of data points that have deviated from the rest of the normal data points.

Issues in existing anomaly detection methods for HVAC systems

There are many methods available for anomaly detection [8, 29, 89, 107, 138, 152, 162, 201]. However, in most of the cases, the performance is unsatisfactory when real HVAC time-series data are presented to them. Most of the methods are unable to detect important anomalies, and/or if detected, have meaningless or misleading anomaly score. Sometimes, significant anomalies have low anomaly score as compared to less significant anomalies. Furthermore, as the existing methods generate anomaly score based on the distribution of each HVAC system, the range of anomaly score also varies from device to device. Due to this factor, it becomes difficult to select a suitable threshold which can efficiently detect anomalies in a number of same HVAC systems. The evaluation of different distance-based, statistical-based, and time-series anomaly detection algorithms shows that they are less precise to detect long-term anomalies in HVAC data set. To address these issues, an unsupervised pattern-based contextual anomaly detection technique is presented in this chapter. Furthermore, existing anomaly detection techniques are also evaluated on real HVAC data set. The presented method uses a knowledge base of only normal data points extracted from the given data. The knowledge base is updated with the passage of time where new normal points are included automatically in it and all of the anomalous points are neglected. The presented method is capable of detecting long/short -term contextual anomalies as well as point anomalies in time-series data.

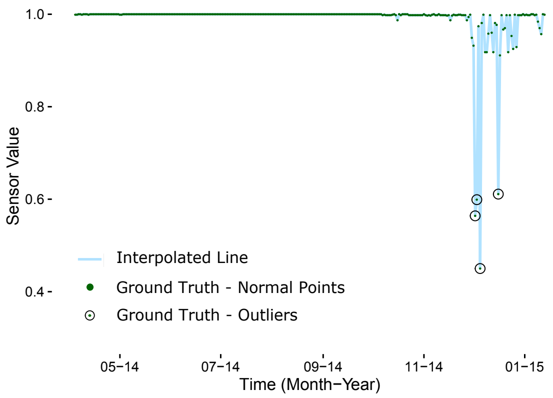
Main contributions of this chapter are:

- A pattern-based contextual anomaly detection approach for IoT-based HVAC systems.
- Generation of meaningful anomaly score (an added advantage of the proposed method) for detected anomalies, i.e. high score to most significant anomalies and less score to less significant anomalies.
- A detailed evaluation of existing and proposed approaches on real HVAC operational data set.

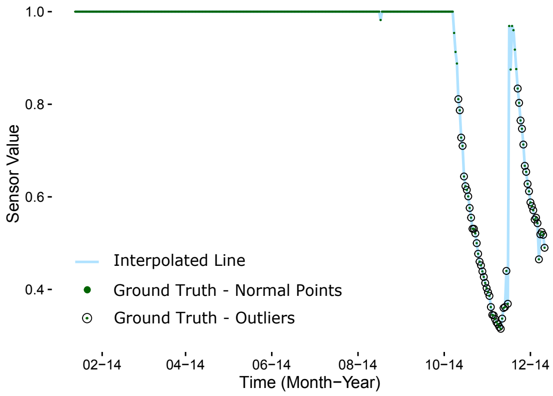
4.1 LITERATURE REVIEW

DTW- and CNN-based methods

Leng, Chen, and Li [162] proposed a method to detect anomalous patterns in time-series data. Their method has two stages. First, a time-series is segmented into different patterns and then anomalous patterns are detected. Each pattern is compared to other patterns of different sizes using Dynamic Time Warping (DTW) to calculate the anomaly factor. This method highly depends on the correct time-series segmentation and other thresholds. It is also not clear if this method is capable of detecting point anomalies or not. A fast variant of this anomaly detection method was proposed by Vy and Anh [246]. Another DTW-based anomaly detection method for ECG data was proposed by Boulnemour, Boucheham, and Benloucif [29]. In this improved version of DTW, called I-DTW, time-series of different lengths and periods are aligned better to each other as compared to standard DTW. For anomaly detection, a normal ECG segment and an ECG segment with the abnormality (a query segment) are given to the system. The I-DTW reconstructs normal segment onto query segment and the morphological difference between two segments less than a threshold shows anomaly. They only reported results on ECG data, so the performance of this method on HVAC data set is not known. Also, only visual anomaly detection is shown with no information about anomaly score. Miyata et al. [179] introduced a CNN-based fault detection and diagnosis method for HVAC systems. First they generate a fault database by detailed simulation, then a CNN is trained on that database. Finally, they apply the trained model on real data set to detect faults. The detected faults are further used for diagnosis purposes. They have used a six layered CNN including two convolution layers, two max pooling layers, and two fully connected layers. Chakraborty and Elzarka [40] have introduced an XGBoost-based method for early detection of faults in HVAC systems. They have also proposed dynamic threshold method to determine occurrences of faults in real time. Their method adjusts the threshold value dynamically according to the real-time moving average and moving standard deviation of the predictions.



(a)



(b)

Figure 4.2: Examples of HVAC time-series data set. This data set contains both point and contextual anomalies.

4.2 DATA SETS

Real HVAC and ECG data

Two real data sets, HVAC data set and ECG data set are used for the evaluation of the proposed technique. These data sets are already explained in Section 3.5.2.1 and Section 3.5.2.2 of Chapter 3. Figure 4.2 shows examples of normalized time-series of observed sensor from two HVAC systems. In the early stages, when a boiler is installed, it is observed that the sensor works perfectly fine (1.0 is the normal/expected running value of the observed sensor). However, with the passage of time, the observed sensor value starts deteriorating or shows anomalous behavior on some days. These abnormal behaviors indicate that the sensor needs to be repaired before it completely breaks. Normal data points are shown in green color, whereas black circles having a dot inside them show anomalous data points. These data points serve as anomaly ground truth, that were marked by a domain expert. This data set contains both point (Figure 4.2a) and long-term (Figure 4.2b) anomalies.

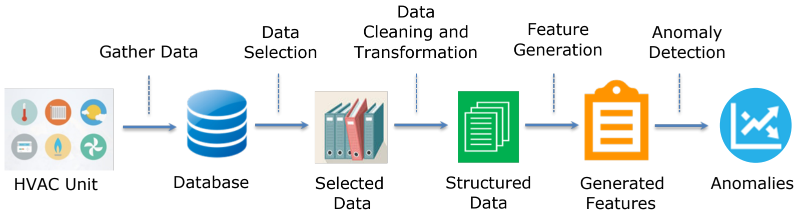


Figure 4.3: Proposed pipeline for anomaly detection on HVAC data.

4.3 PROPOSED PIPELINE FOR ANOMALY DETECTION

This section provides a detailed insight of the presented pipeline for anomaly detection. Figure 4.3 shows the complete workflow of the presented anomaly detection method that is divided into three main steps i.e. pre-processing, feature generation, and anomaly detection. The pre-processing and feature generation steps are explained according to the HVAC data set. However, these steps can be followed for other HVAC data sets. The other data set used in this study ECG, does not require any pre-processing as it is already in a format that is generally required by machine-learning algorithms.

4.3.1 *Pre-processing*

The data recorded from a HVAC system usually have missing data and some data that is of no use to anomaly detection process. Therefore, it is required to transform the HVAC data into the form that is generally required by anomaly detection methods. Following are the challenges that are faced in the used HVAC data set pre-processing. This step is further divided into two steps:

- **Data Selection**
  The purpose of this step is to select relevant data that is suitable for anomaly detection. It is possible that not all of the HVAC systems are ready for analysis; there might be some systems, that have logged very small amount of data since their installation due to various reasons. This may happen because the router, that is registered to transmit the data over the internet is permanently changed or switched off. In addition, each HVAC system logs a lot of information from multiple sensors. However, only few of the sensors might be relevant and suitable for further analysis. Therefore, in our case, those HVAC systems are shortlisted for further analysis that stayed online at least for one year. The selected systems transmit huge amount of the data, consisting hundreds of internal and external sensors. In practice, not all of the sensor data can be passed to anomaly detection algorithms. Therefore, the sensors, which are most critical to be ob-

Select relevant data

served are selected, based on the expert's knowledge. However, in the scope of this chapter, our focus is only on one sensor.

- **Data Cleaning and Transformation**
  The purpose of this step is to tackle the issue of missing data. From a HVAC system to the database, the data passes through different connectivity bridges. Connectivity loss in any one of these bridges causes missing data in the database. A missing gap can be of few minutes, an entire day, or even of months. The systems which have missing gaps of more than consecutive 3 days are called unhealthy systems. 24 healthy systems are shortlisted for further analysis which have the maximum number of consecutive data and minimum number of missing gaps. The sensor data are stored in the back-end database in an unstructured format and could not be used directly for analysis purposes. So, it is transformed into a structured comma-separated values (CSV) format in which the sensor values are extracted against a unique timestamp for each HVAC system.

Convert data into a structured format

4.3.2 *Feature Generation*

The operational data logged in the database is unevenly spaced time-series data. In an unevenly spaced or an irregular time-series data, observation time is not constant. In other words, the observations for each system are logged at the different time. It is, therefore, hard to find a common pattern among different systems as there is no common logging time-line among different systems. To compare different systems and the behavior of the recorded system, it is important to define the data on a common timeline. In feature generation step, excessive and unevenly spaced time-series data are removed and meaningful features out of the raw data are generated.

In general, a feature is defined as an aggregated quantity (of some parameter) per unit. In our case, a feature is *the mean value of a selected parameter per day*. So, we take mean of the observed sensor at a particular time of day that shows the actual behavior of that sensor. This results into the data that is evenly spaced time-series and can be compared to all other systems for further analysis.

Hand-engineered feature

4.3.3 *Anomaly Detection*

The presented pattern-based anomaly detection is an unsupervised technique, that builds a knowledge base of long-term patterns. The knowledge base, which is based on normal data points, keeps growing over the time. The presented approach works based on the assumption that first n instances of an operational log are normal data points and does not contain contaminated/anomalous data. This as-

Unsupervised method

sumption is generally true, because a new device when installed for the first time, is expected to have a normal behavior. These initial n points (in our case, $n = 50$) serves as a basis of knowledge base containing normal patterns.

The size of the knowledge base (K_B) keeps growing over the time. A new point is added to the knowledge base if it is detected as a normal data point. To mark a data point as normal or anomaly, current data point (x_t), as well as its contextual information, is used. The sequence containing the context and the original value can be defined as follows:

$$S_t = \bigcup_{i=t-N}^{t} x_i \tag{4.1}$$

Using equation 4.1, knowledge base (K_B) in initial state can be defined as follows:

$$K_B = \{S_N\} \tag{4.2}$$

Knowledge base

This sequence/window S_t is compared with the knowledge base (K_B) by using an overlapping window of the same size as input sequence. This means that knowledge base is divided into overlapping windows of size $N+1$. All of these overlapping windows from knowledge base are compared with the window/sequence of the current data point. The comparison between the context of the current window and all the windows in the knowledge base is done using DTW. The minimum DTW distance (equation 4.3) finds the most similar sequence from the knowledge base.

$$\mathrm{dist}(S_t) = \min_i(DTW(S_t, K_i)) \; \forall \, K_i \in K_B \tag{4.3}$$

A threshold τ is used to distinguish between a normal data point and an anomalous data point. Equation 4.4 defines the use of equation 4.3 for detection of anomalous point as well as accumulation of knowledge base.

$$f(\mathrm{dist}(S_i)) = \begin{cases} Anomaly, & \text{if } \mathrm{dist}(S_i) > \tau \\ S_i \bigcup K_B, & \text{otherwise} \end{cases} \tag{4.4}$$

Where dist is a DTW distance between the current window and all the sequences in the knowledge base. All of the remaining data points that do not satisfy the first criteria in Equation 4.4 are marked as normal and added to the knowledge base. In this way, the knowledge base keeps on growing with time (t) incorporating long-term patterns.

4.4 EVALUATION

This section provides a detailed analysis of the following existing state-of-the-art unsupervised anomaly detection techniques and

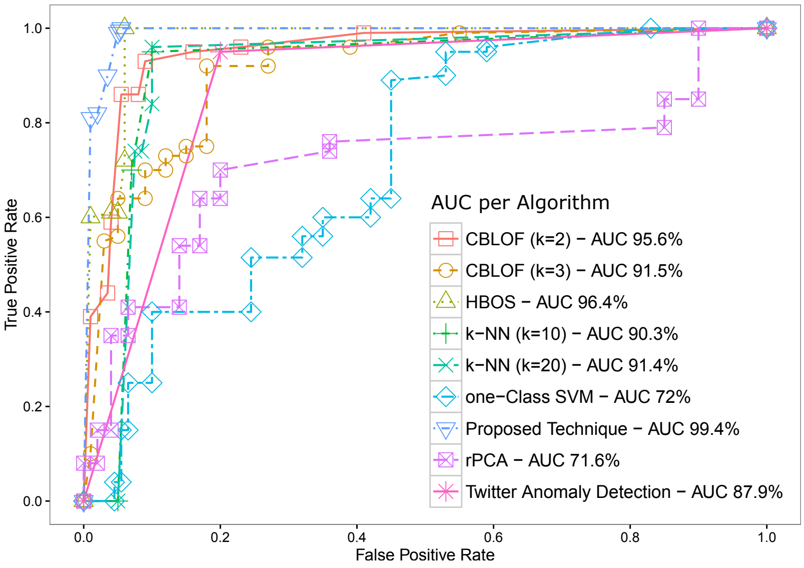


Figure 4.4: ROC plot comparing the proposed technique with the state-of-the-art anomaly detection techniques.

the presented anomaly detection technique on real HVAC and ECG data sets: kNN Global Anomaly Score [201], CBLOF[107], HBOS [89], OC-SVM [8], Twitter Anomaly Detection [138], rPCA [152].

Anomaly Detection Extension for RapidMiner

For comparison with the existing techniques, the standard implementation of the above-mentioned algorithms in RapidMiner is used. These anomaly detection algorithms are freely available as an extension to RapidMiner. This Anomaly Detection Extension[1] contains many other unsupervised anomaly detection algorithms too. There are different hyperparameters of the selected algorithms that need to be tuned. A number of experiments on different hyperparameters was performed and the best parameters were used for the final evaluation. Same parameters are used for both data sets. A detailed evaluation is only provided for HVAC data set (on the basis of ROC, ROC-AUC, Precision, and Recall). Whereas, for ECG data set, visual evaluation is provided to show the issues in other anomaly detection methods and to highlight the generic nature of the proposed method.

Evaluation metrics

Performance measures like ROC and ROC-AUC are used in this analysis section to show an overall performance of different algorithms and the proposed method. In the ROC plot shown in Figure 4.4, the performance of a method is considered best whose curve is most close to the upper left corner. In addition to that, precision and recall are also calculated to gain insights of each algorithm.

Generally, all anomaly detection algorithms provide an anomaly score for a given data point. To classify a data point as normal or anomalous, a suitable threshold is required (that might vary for each

---

[1] RapidMiner Anomaly Detection Extension is available at: `https://marketplace.rapidminer.com/UpdateServer/faces/product_details.xhtml?productId=rmx_anomalydetection`

Table 4.1: Precision and recall of proposed and other observed techniques on HVAC data set. The best threshold is selected by equal error rate.

Algorithm	Precision	Recall
Proposed Technique	**0.91**	**0.80**
HBOS	0.47	0.90
CBLOF (K=2)	0.50	0.74
CBLOF (K=3)	0.56	0.54
kNN (k=10)	0.34	0.54
kNN (k=20)	0.36	0.61
OC-SVM	0.19	0.38
rPCA	0.42	0
Twitter Anomaly Detection	0.23	0.95

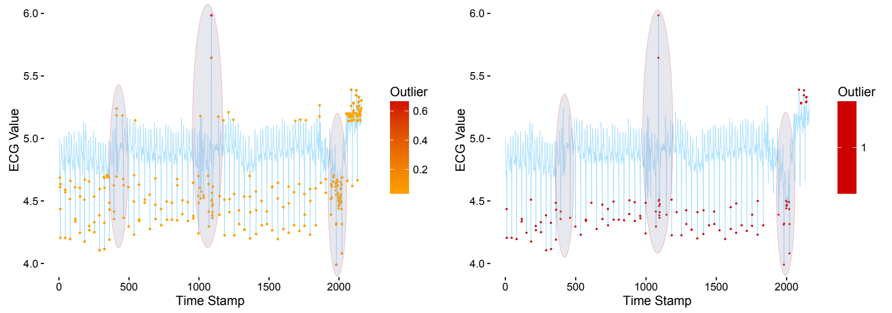
algorithm). It is important to find the threshold that is a true representative of the classifier. In this study, the best threshold is selected on the basis of Equal Error Rate (EER). FPR and TPR from ROC curve are used to find EER. Usually, the threshold where both FPR and TPR become equal is considered as the best threshold. However, in some cases, it is also possible that these two measures do not match exactly. For such cases, we calculate the difference between the two measures. The best threshold is considered at the point where the difference is minimum. Table 4.1 shows the precision and recall of all the evaluated methods on the HVAC data set. The overall precision and recall of the proposed technique are better than other anomaly detection methods. The recall of HBOS and Twitter Anomaly Detection is better than proposed technique, but a balance between precision and recall is more important than only relying on precision or recall.

4.4.1 *Analysis*

Here, it is important to note the notation we have used for normal and anomalous data points – anomalies are considered as positive records, whereas normal data points are considered as negative records. As we are interested in anomalous data points, so they are marked as a positive hit.

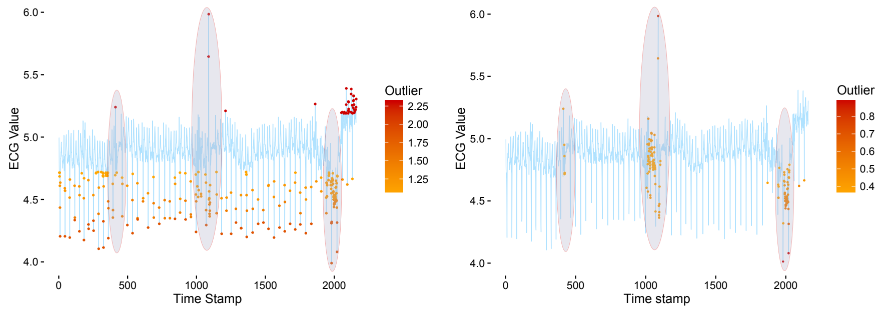
Labeling notation

The analysis of Table 4.1 shows that HBOS performs best on HVAC data set from the group of the state-of-the-art techniques with maximum AUC of 96.4%. HBOS clearly performs better than distance-based and clustering-based techniques. It is mainly of two reasons: i) The data points are spread in a way that very less number of data points lie in the same bin, that generates high anomaly score, and ii) The height of histograms is normalized, so the anomaly score of different systems is more representative. It can be observed in Table 4.1 that



(a) k-NN Global Anomaly (k=20), Threshold = 0.1

(b) Twitter Anomaly Detection.A point is marked as normal or abnormal. Red points are detected anomalies.



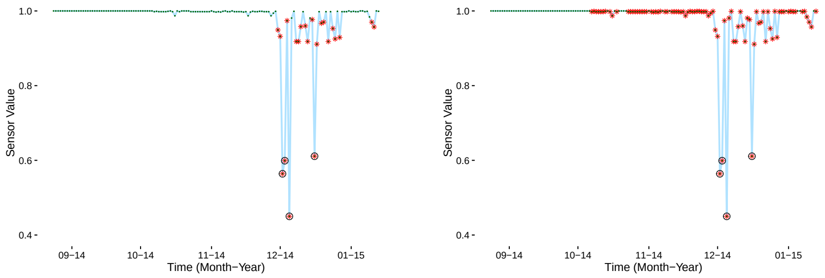
(c) HBOS, Threshold = 1.0

(d) Pattern-based Anomaly, Threshold = 0.3

Figure 4.5: Color coded anomalies show anomalies detected by respective methods. Abnormal patterns exit in the highlighted area. Only the proposed method is able to detect these contextual and point anomalies.

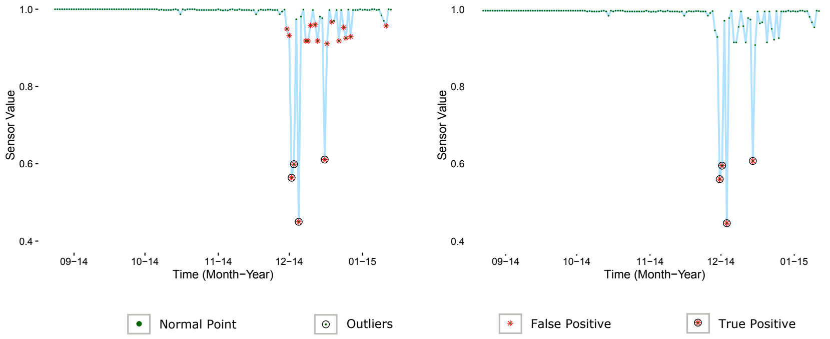
the overall precision and recall of HBOS is better than other state-of-the-art techniques. Whereas, on ECG data set, HBOS is only able to detect point anomalies as shown in Figure 4.5c. In Figure 4.5c, the data points that lie far from the most of the data points space, are incorrectly marked as anomalies (in red) towards the end of the time-series. This behavior shows the limitation of HBOS. In Figure 4.5, the three highlighted areas in all sub-figures are where actual anomalous points exist.

After HBOS, clustering-based technique, CBLOF performs well on HVAC data set with the AUC of 95.6%. We did experiments with different number of clusters and report the results on two clusters size, i.e. 2 and 3 clusters. It can be observed in Figure 4.4, that AUC of CBLOF method is decreased with the increase in the number of clusters for the HVAC data set. Depending on the nature of data and number of clusters, this technique can cause a problem when a number of anomalous data points form a separate cluster because of their high co-existence. This can happen in the case of long-term anomalies, where a sensor starts to continuously give slightly deviated value. In



(a) k-NN Global Anomaly (k=20) P: 18.18% R:100%

(b) Twitter Anomaly Det. P: 4.3% R:100%



(c) HBOS P: 22.2% R:100%

(d) Pattern-based Anomaly P:100% R:100%

Figure 4.6: Comparison of precision (P) and recall (R) of different anomaly detection techniques and the presented technique on a specific HVAC system. Here, the precision and recall are calculated for a single HVAC system.

this case, the whole cluster can be wrongly detected as normal data points. However, in the case of two clusters and existence of a relatively low number of anomalous data points, as compared to normal data points, 2 clusters give relatively good results as compared to larger cluster number.

kNN-based anomaly detection

Widely used, kNN-based anomaly detection technique shows AUC of 91.4% for HVAC data set, that is less than CBLOF and HBOS. This method is also tested for different set of parameters and best are reported, i.e. K is set to 10 and 20. This distance-based technique is good for the detection of point anomalies as they are easily distinguishable from the rest of the data. However, in the case of time-series data sets, where anomalies occur due to the change in pattern or due to the small deviations, this technique is unable to detect important anomalies. Also, in the case of long-term anomalies, this technique is unable to detect anomalies because the distance between normal and abnormal points does not change drastically. Same is observed in ECG data set, all of the anomalies are not detected in this case too (Figure 4.5a).

Experiments with different parameter combinations are performed using Twitter Anomaly Detection technique. In most of the cases for HVAC data set, this technique is able to detect all of the anomalies, *OC-SVM and rPCA based anomaly detection*

but with high false alarm rate. But for ECG data set, this method is unable to detect contextual anomalies and the marked anomalies are incorrect as shown in Figure 4.5b. Overall results of OC-SVM and rPCA are poor on HVAC data set.

The above-mentioned issues in different anomaly detection techniques are addressed in the proposed pattern-based contextual anomaly detection technique. As shown in Figure 4.4, the proposed approach performs better than other anomaly detection techniques with the AUC of 99.4%.

The proposed technique out-performed others

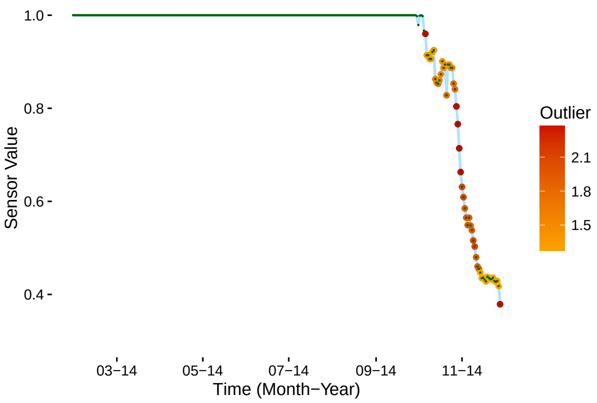
Figure 4.6 shows the results of some of the observed anomaly detection techniques on a specific HVAC system. The anomalous data points (marked with red asterisk ($*$)) detected by the mentioned technique are shown along the ground truth information. When ($*$) is inside a black circle, it represents true positive; i.e. an anomalous data point is correctly detected by the respective algorithm. Whereas, only asterisk mark shows false alarm. The purpose of this figure is to visually understand the issues of current anomaly detection techniques on HVAC data set. Precision and recall shown in this figure are calculated for this HVAC system based on the commonly selected threshold.

Figure 4.6a shows the result of kNN Global Anomaly Score technique, where all of the anomalies are detected correctly (recall is 100%) with a lot of false alarms, that degraded the overall performance and precision decreases to 18.18%. For the kNN technique, the accuracy is slightly increased with greater k on the whole data set as shown by the AUC in Figure 4.4. The parameters, like the number of clusters and k in nearest neighbor techniques, cannot be generalized as they depend on the distribution of the data. Twitter Anomaly Detection technique performs worst on the observed HVAC system (Figure 4.6b). Even the small variations in data are falsely marked as anomalies by this technique. Figure 4.6c shows that HBOS is able to detect all anomalies correctly. However, similar to kNN-based method, its precision is low. In comparison to existing techniques, Figure 4.6d shows the result of the proposed pattern-based technique, that outperforms other methods and achieves a precision and recall of 100% on the observed HVAC system. Likewise on ECG data set, the proposed technique is clearly able to detect point and contextual anomalies (Figure 4.5d) that are missed by most of the other mentioned methods.

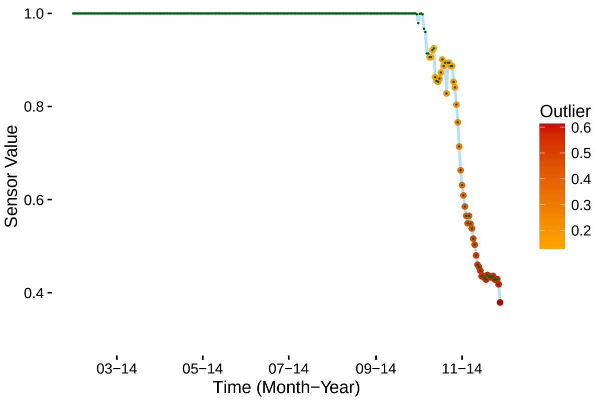
4.4.2 *Problem of Misleading Anomaly Score*

Issues in the existing techniques

In addition to the low precision and recall, another problem with most of the existing anomaly detection techniques is the misleading anomaly score, across multiple systems of the same type. Ideally, an anomalous data point that is far off should have a high anomaly score, whereas a low anomalous score should be assigned to a data point that is not so far from the distribution of normal data points. The



(a) HBOS



(b) Presented technique.

Figure 4.7: Color-coded anomaly scores given by HBOS and the presented pattern-based technique. Anomalies toward the bottom of the curve are given relatively low anomaly score by HBOS. Whereas, correct anomaly score behavior is given by the proposed technique.

best threshold must provide best precision and recall on all systems. In most of the cases, the recall score is very good, but the precision is not so good. The data distribution of all the systems is different due to which it is hard to standardize the anomaly score threshold for all the systems. Without a defined threshold, the anomaly detection algorithm cannot be used practically. In HVAC data set, there are some systems in which small deviations of sensor value are observed. On such small deviations, existing algorithms give high anomaly score because of the inability of incorporating context, which makes the threshold selection process (for all systems of the same type) very difficult.

The second best technique mentioned in Figure 4.4 for the HVAC data set is HBOS. Figure 4.7a shows color-coded anomalies marked by HBOS for a HVAC system. This is an example of a long-term anomaly scenario. All the anomalies are detected by HBOS in this case. However, the anomaly score is not the true representative of the anomalies when the data of the whole system is observed. The highest anomaly score is given to anomalous data point in the middle of the curve, and then anomaly score decreases towards the end of the curve (which should be increasing). It is due to the fact that when more data points occur in the anomalous region, then anomaly score decreases. In the context of HBOS algorithm, if more data points exist for a defined bin, their possibility of being anomalous decreases. Whereas in real data, there are cases when a relatively high number of data points exist in the anomalous region and it is important to detect all of those anomalous points with correct anomaly score. Same anomaly score behavior is also observed in kNN Global Anomaly Score technique for HVAC data set. In comparison to the existing methods, the presented technique provides correct anomaly score behavior as shown in Figure 4.7b, in which, anomaly score increases for more distant anomalous data points. The anomaly score generation technique makes the anomaly score true representative of the anomalies. Figure 4.5d also shows correct anomaly score behavior on ECG data set. For an anomalous data point, anomaly score is calculated from the normal data points saved in the knowledge base. This gives the correct representation how far an anomalous data point is.

Long-term anomaly scenario

5

DEEP LEARNING FOR ANOMALY DETECTION

Detecting anomalies in time-series is a challenging task

The term 'Anomaly', is widely used and it refers to different problems in different domains. For example, an anomaly in network security system could be an activity related to a malicious software or a hacking attempt [88]. Whereas, in the manufacturing domain, a faulty product is considered as an anomaly. Companies from different sectors including manufacturing, automotive, healthcare, lodging, traveling, fashion, food, and logistics are investing a lot [50, 147] in collecting big data and exploring the hidden anomalous patterns in it to facilitate their customers. In most of the cases, the collected data are streaming time-series data and due to its intrinsic characteristics (cyclicity, trend, seasonality, and noise), it is a challenging problem to detect a slightly deviating data point in a period. Also, it is very important to detect anomalies in good time so that the big issues like financial system hack, total machine failure, or a tumor in human body can be avoided.

DNN in general

Recent years have witnessed an enormous increase in the use of DNN. They are used as a function approximation or an estimation tool. Nowadays, DNNs are widely used in domains like image/video processing, data mining, and finance for classification and prediction. They can learn different features based on the training data and classify the test data into different target classes on which a network is learned. DNNs achieve good performance and flexibility by learning to represent the data as a nested hierarchy of concepts within layers of the neural network [41]. Simple features are learned in lower layers, whereas, complex features are learned in higher layers. In most of the cases, DNNs work better than traditional machine learning and shallow neural networks as shown in Figure 5.1.

Unsupervised anomaly detection

Furthermore, in most of the real life scenarios, it is practically impossible to label enormous amount of data, therefore, the demand of unsupervised anomaly detection methods is very high. Although many unsupervised methods are available, but most of them don't handle the intrinsic characteristics of time-series data very well. In most of the cases, they are evaluated for a particular scenario or on a domain specific data. Flexibility and adaptability of anomaly detection systems are open questions that need to be addressed. The proposed unsupervised approach incorporates context, seasonality, and trend into account for detecting anomalies. This approach can be adapted for different scenarios and use cases, and works on different types of data sets.

This chapter is an adapted version of the work published in [185].

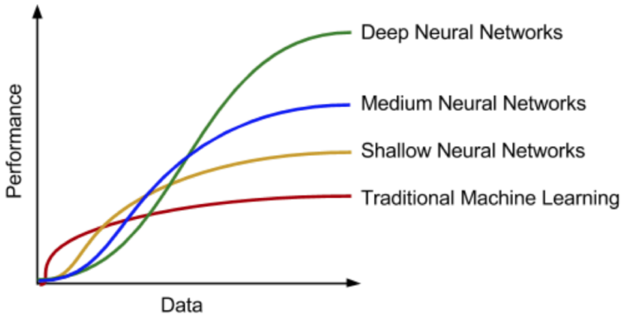


Figure 5.1: Performance comparison of traditional machine learning and Deep Neural Network (DNN) [6].

This chapter presents Deep Anomaly Detection for Time-series (DeepAnT): a novel unsupervised deep learning-based anomaly detection approach for streaming data. This approach doesn't rely on labeling of anomalies, it rather leverages the original time-series data even without removing anomalies (given that the number of anomalies in the data set is less than 5% [88]). DeepAnT leverages the CNN as its forecasting module. This module predicts the next timestamp of a given time-series window. Furthermore, the forecasted value is passed to a detector module which compares that value with the actual data point to detect anomalies in real-time. The approach is realistic and suitable even for domains where time-series data are collected from heterogeneous sources and sensors. DeepAnT achieves good generalization capabilities in data scarce scenarios where less training data are available. Only a few number of training samples (e.g. 568 data points) are sufficient to build a prediction model due to its effective parameter sharing during feature extraction. DeepAnT when tested on publicly available anomaly detection benchmarks, outperformed the state-of-the-art anomaly detection methods in most of the cases. In addition to classifying whole time-series as normal or abnormal (as done in [43, 46, 164, 174]), DeepAnT is also capable of detecting point anomalies. In particular, following are the main contributions of this chapter:

DeepAnT

1. To the best of our knowledge, DeepAnT is the first deep learning-based approach which is capable of detecting point anomalies, contextual anomalies, and discords in time-series data in an unsupervised setting.
2. The proposed pipeline is flexible and can be easily adapted for different use-cases and domains.
3. In contrast to the LSTM-based approach, CNN-based DeepAnT is not data hungry. It is equally applicable to big data as well as

small data. We are only using 40% of a given time-series to train a model.

4. We gathered different anomaly detection benchmarks at one place and provided extensive evaluation of 15 state-of-the-art methods in different settings on 10 data sets (covering both steaming and non-streaming cases) which contain 433 time-series in total. DeepAnT has gained the state-of-the-art performance on most of the data sets.

5.1 PROPOSED APPROACH FOR TIME-SERIES ANOMALY DETECTION

The proposed DeepAnT consists of two modules. The first module, **Time-series Predictor** predicts timestamps for a given horizon and the second module, **Anomaly Detector** is responsible for tagging the given data points as normal or abnormal. Deep learning has been employed in a wide range of applications primarily because of its capability to automatically discover complex features without having any forehand knowledge. This automatic feature learning capability makes the neural networks a good candidate for time-series anomaly detection problem. Therefore, DeepAnT also uses only raw data and uses CNN as a time-series predictor. Also, it is robust to variations as compared to other neural networks and statistical models. It is shown in literature [82, 114] and in practice that LSTM performs well on temporal data, due to its capability to extract long-term trends in the encountered time-series. However, it is shown in this chapter that CNNs can be a good alternate for uni-variate as well as multi-variate time-series data due to its parameter efficiency. Deep CNN has been employed for forecasting in *time-series predictor*, whereas CNNs and LSTMs are usually used for time-series classification in literature (as done in [164, 268]).

Consists of two modules

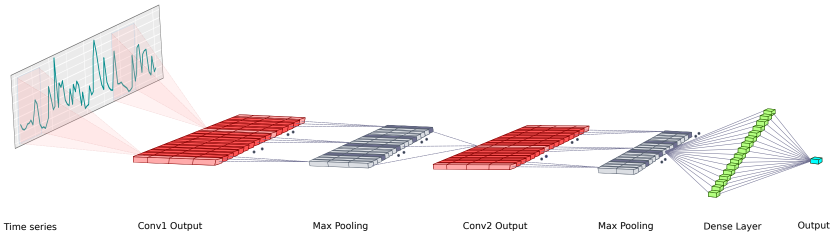


Figure 5.2: DeepAnT architecture for time-series prediction: A convolutional neural network with two convolutional layers, two max pooling, and a fully connected layer.

5.1.1 *Time-series Predictor*

Based on CNN

The predictor module of DeepAnT is based on CNN. CNN is a type of artificial neural network that has been widely used in different domains like computer vision and natural language processing in a range of different capacities due to its parameter efficiency. As the name indicates, this network employs a mathematical operation called *convolution*. Normally, CNN consists of sequence of layers which includes convolutional layers, pooling layers, and fully connected layers. Each convolutional layer typically has two stages. In the first stage, the layer performs the convolution operation which results in linear activations. In the next stage, a non-linear activation function is applied on each linear activation. In simplest form, convolution is a mathematical operation on two functions of real valued arguments to produce a third function. The convolution operation is normally denoted as asterisk:

$$s(t) = (x * w)(t) \tag{5.1}$$

This new function s can be described as a smoothed estimate or a weighted average of the function $x(\tau)$ at the timestamp t, where weighting is given by $w(-\tau)$ shifted by amount t. In Equation 5.1, function x is referred to as the input and function w is referred to as the kernel. The output is referred to as the feature map. One dimensional discrete convolution is defined as:

$$s(t) = \sum_{\tau=-\infty}^{\infty} x(\tau)w(t-\tau) \tag{5.2}$$

Max-pooling

In DeepAnT, similar to other well known methods [123, 149], the output of a convolutional layer is further modified by a pooling function in a pooling layer. A pooling function statistically summarizes the output of the convolutional layer at a certain location based on its neighbors. Most commonly used, *max-pooling* operation is used in DeepAnT which outputs the maximum activation in a defined neighborhood. Since there exists more than one feature maps, therefore the pooling function is applied on all of these feature maps.

After pair of convolutional and max-pooling layers, the final layer of connections in DeepAnT is a fully connected layer. In this layer, each neuron from a previous layer is connected to all output neurons. The activation for convolutional and fully connected layers are given in Equation 5.4 and Equation 5.6 respectively, where k is defined as $\lfloor FliterSize/2 \rfloor$.

$$z^l_{ji} = \sum_{-k}^{k} W^l_{jk} a^{l-1}_{i-k} + b^l_j \tag{5.3}$$

$$a^l_{ji} = \max\left(z^l_{ji}, 0\right) \tag{5.4}$$

$$z_j^l = \sum_{k=1}^{e} W_{jk}^l a_k^{l-1} + b_j^l \tag{5.5}$$

$$a_j^l = \max\left(z_j^l, 0\right) \tag{5.6}$$

In Equation 5.4, a_{ji}^l refers to the activation of the j^{th} neuron in the l^{th} layer at the i^{th} input location of a convolutional layer. Whereas, a_j^l refers to the activation of the j^{th} neuron in the l^{th} fully connected layer in Equation 5.6.

Like other artificial neural networks, a CNN uses training data to adapt its parameters (weights and biases) to perform the desired task. In DeepAnT, parameters of the network are optimized using Stochastic Gradient Descent (SGD). The idea of training or learning of a neural network is to reduce the cost function C. In this predictor module, a cost function computes the difference between the network's predictions and the desired prediction. In a learning process, that difference is minimized by adapting the weights and biases of the network. The process of calculating the gradient that is required to adjust the weights and biases is called backpropagation. It is obtained by calculating the partial derivatives of the cost function with respect to any weight w or bias b as $\partial C/\partial w$ and $\partial C/\partial b$ respectively.

Backpropagation

There are four steps of backpropagation algorithm:

1. Feed-forwad pass
2. Backpropagation to the output layer
3. Backpropagation to the hidden layer(s)
4. Update weights and biases

Initially, the weights of a network are randomly initialized. In the feed-forward pass, the output of all the hidden neurons along with the final output of a network is computed. The error between the network output and the cost function is backpropagated to the initial layers and the gradient with respect to network parameters is computed. The error (δ) of j^{th} neuron at the output layer L is calculated as shown in Equation 5.8.

$$\delta_j^L = \frac{\partial C}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L} \tag{5.7}$$

$$\delta_j^L = \frac{\partial C}{\partial a_j^L} \max{}'\left(z_j^L, 0\right) \tag{5.8}$$

The gradient of error with respect to the output layer is backpropagated to all the neurons in the hidden layer as shown in Equation 5.9. If the value of input x exceeds 0, then the gradient of Rectified Linear Unit (ReLU) is 1 and remains 0 otherwise. As max-pooling layer is applied on ReLU activation of convolutional layer, so the gradient of max-pooling layer is 1 when the maximum quantity occurred and remains 0 otherwise as shown in Equation 5.10.

$$\delta_j^l = \left(\left(w_{ij}^l \right)^T \delta_j^{l+1} \right) \odot \max' \left(z_j^l, 0 \right) \tag{5.9}$$

$$\max'(x, 0) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.10}$$

The rate of change of cost with respect to the bias and weights is given in Equation 5.11 and Equation 5.12 respectively.

$$\frac{\partial C}{\partial b_j^l} = \delta_j^l \tag{5.11}$$

$$\frac{\partial C}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l \tag{5.12}$$

Forecasting horizon

In order to leverage CNN for forecasting, time-series data needs to be changed in a compatible form for the system to operate on. For each element x_t at timestamp t in a time-series, next element x_{t+1} at timestamp $t+1$ is used as its label. Input data is transformed into several sequences of overlapping windows of size w. This window size defines the number of timestamps in history, that are taken into account (referred as a *history window*). It also serves as the context of x_t. The number of timestamps required to be predicted is referred as *prediction window* (p_w). In some studies, prediction window is also called as (Forecasting) Horizon [68, 233].
Consider a time-series:

$$\{x_0, x_1, ..., x_{t-1}, x_t, x_{t+1}, ...\}$$

For $w = 5$ and $p\_w = 1$, the sequence at index t will be as follow:

$$x_{t-4}, x_{t-3}, x_{t-2}, x_{t-1}, x_t \rightarrow x_{t+1}$$

In a regression problem (as ours), the left hand side is treated as input data and right hand side is treated as label. In this case, it can be called as *many_to_one prediction*. When $p\_w > 1$, it can be called as *many_to_many prediction*.

Layers and nodes

Architecture Summary: We did extensive experiments to finalize the architecture and its hyperparameters. Two convolutional layers, each followed by a max-pooling layer are used in this architecture (as shown in Figure. 5.2). The input layer has w input nodes, as we have converted the data into w vectors. Each convolution layer is composed of 32 filters (kernels) followed by an activation function. Element-wise activation function, ReLU as shown in Equation 5.13 is used as an activation function. Last layer of the network is a fully connected layer in which each neuron is connected to all the neurons in the previous layer. This layer represents the network prediction for the next timestamp. The number of nodes used in the output layer are equal to p_w. In this case, the model is predicting only the next timestamp, so the number of output node is 1. However, in Section 5.5 of this chapter, when the model is predicting a sequence instead of a single data point, the the number of nodes in output layer is changed accordingly.

$$f(x) = max(0, x) \tag{5.13}$$

Loss Function: Mean Absolute Error (MAE), shown in Equation 5.14 has been employed as an indicator of the discrepancy between the desired and the predicted output. By reducing the error between the predicted and the desired value, the network can learn to predict the normal behavior of the time-series. We normalized each time-series based on the training data.

$$MAE = \frac{1}{n}\sum_{j=1}^{n} |y_j - \hat{y}_j| \tag{5.14}$$

5.1.2 *Anomaly Detector*

Once the prediction of next timestamp x_{t+1} is made by the *Time-series Predictor*, this module detects anomalies in a given time-series. The value predicted by the predictor module is passed to this module and the difference between actual and predicted value is calculated. Euclidean distance shown in Equation 5.15 is used as a measure of the discrepancy.

$$(y_t, y_t') = \sqrt{(y_t - y_t')^2} \tag{5.15}$$

where y_t is actual value and y_t' is predicted value.

Euclidean distance

The Euclidean distance is used as anomaly score. A large anomaly score indicates a significant anomaly at the given timestamp. A threshold, based on the time-series type needs to be defined for this module; that is generally required by most of the anomaly detection algorithms.

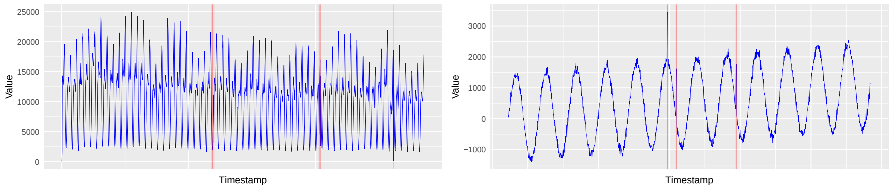
5.2 EXPERIMENTAL SETTING-I

For the better understanding, the experimental setup is divided into several parts, because different anomaly detection methods in literature are evaluated on different benchmarks based on different metrics.

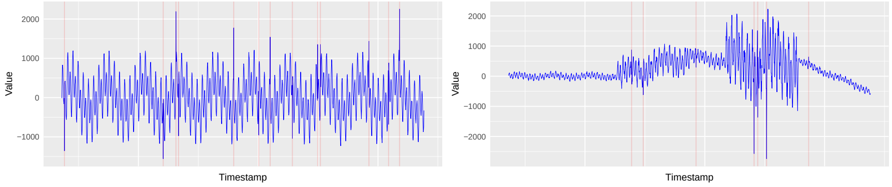
5.2.1 *Data set*

Yahoo Webscope data sets

One real and three synthetic data sets from Yahoo Webscope data set (A1, A2, A3, and A4) are used in this experimental setting. Details of these data sets are given in Section 3.5.2. One time-series from each data set is shown in Figure 5.3. Actual time-series is shown in blue, whereas red vertical lines show anomaly ground truth.



(a) A1 Benchmark (b) A2 Benchmark



(c) A3 Benchmark (d) A4 Benchmark

Figure 5.3: One time-series from all of the four Yahoo Webscope data set is shown here. Actual streaming data are shown in blue, whereas red vertical lines highlight anomalous data points based on the provided labels.

5.2.2 *Evaluation Metric and Experimental Setup*

F-score is employed as the evaluation metric for our models. All the anomaly detection methods in this experimental setting are applied on each time-series of all the data sets separately. Average F-scores per data set is reported for each method.

DeepAnT Parameters: We are using only 40% of each time-series as a training set and rest of the 60% data as a test set. We further split the training set and use 10% of the training set for validation. Since it is an unsupervised approach, we don't use any label information in training process. For each time-series, only next timestamp is predicted and marked as either normal or anomalous data point. To compare the performance of CNN with LSTM in the context of anomaly detection, we have also used LSTM in the *Time-series Predictor* module of DeepAnT. We used the same 40% training data scheme for training LSTM as we did for CNN. For LSTM-based model, we used two LSTM layers (as done in [174]) of 32 memory cells each. For both techniques, we used same w for whole data set.

Table 5.1: This table shows the selected history window and thresholds which are used to evaluate DeepAnT on Yahoo data set.

Sub-benchmark	Threshold	History Window
A1	0.50	45
A2	0.75	45
A3	0.65	35
A4	0.55	35

Thresholding

Finding the best threshold is very important for evaluation. Normally, each time-series has its own characteristics, so finding a generic threshold which works for all of the time-series, is not a straightforward task. Since each Yahoo Webscope data set shares common properties, therefore, we searched for the best threshold for each data set based on the validation data. Based on the data set threshold, we are calculating F-score of DeepAnT.

History window

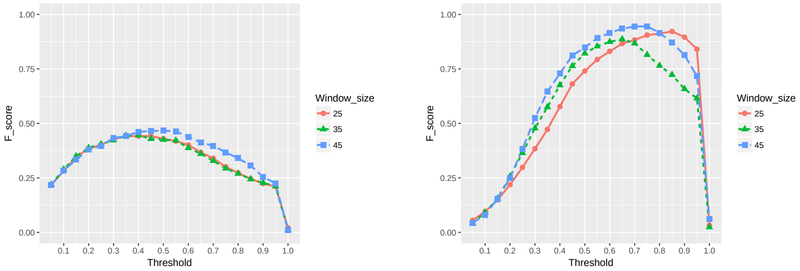
Another parameter, *History Window* (w), also plays a vital role in improving the prediction model. Again, there is no fixed window size, that can be used for all of the time-series. For reproducibility, we list the combination of thresholds and window size yielding the best F-score in Table 5.1. We shortlisted the window sizes of 25, 35, and 45 after hyperparameter optimization. Figure 5.4 shows the effect of w on average F-score, in each data set. These plots also show the importance of selecting the right number of w.

Twitter Anomaly Detection Parameters: We used *AnomalyDetectionTS* function provided in Twitter anomaly detection for A2, A3, and A4 data sets. For A1 data set, we used *AnomalyDetectionVec* function, because timestamps are replaced by integers with an increment of 1 in this data set by the publisher. We used all default parameters of this method except the following two:

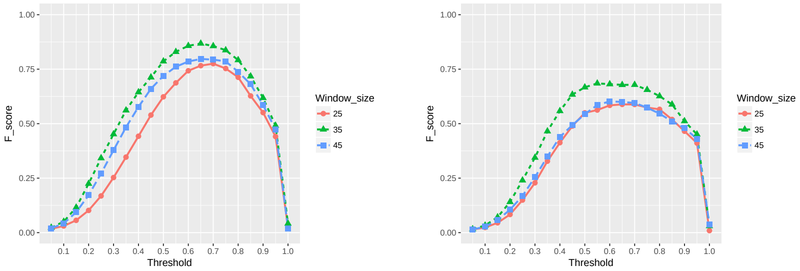
1. ***Alpha:*** This parameter defines the level of statistical significance with which to accept or reject anomalies. We used three values for this parameter i.e. $0.05, 0.1$, and 0.2.
2. ***Direction:*** This parameter defines the directionality (positive or negative) of anomalies to be detected. We used 'both', as anomalies can be in any direction in this data set.

Used models

Yahoo EGADS Parameters: We are using *Olympic Model* in TMM and *EGADS ExtremeLowDensityModel Outlier* in ADM. Default values of all the other parameters are used. Both Twitter anomaly detection and EGADS calculate threshold themselves for each time-series and give timestamps or indexes (if input data does not contain timestamp) of the anomalous data points as output. To evaluate these two methods, we used the same 60% test data of each time-series that is used to evaluate DeepAnT.



(a) $A1$ (b) $A2$



(c) $A3$ (d) $A4$

Figure 5.4: Average F-score of each Yahoo Webscope data set is plotted per history window used in DeepAnT. Plots of three shortlisted windows per data set are shown. For $A1$ and $A2$ data sets, $w = 45$ provides better average F-score, but for $A3$ and $A4$ data sets, $w = 35$ performs good.

5.2.3 *Results*

DeepAnT anomaly detection results on a single time-series are shown in Figure 5.5. In this figure, actual time-series is depicted in blue color, the predictions on training data are depicted in yellow color (not used in reported results), while the predictions on test data are depicted in red color. Vertical blue doted lines are anomalies ground truth in training and testing data. Small dotes are placed on the ground truth when same data points are detected as anomalous points (true positive).

Even small deviations are detected

It can be seen in this example that there are anomalies in training data, but the network correctly captured the data generating distribution disregarding the anomalies. Predicted data points (red) are super imposed on the actual time-series in order to highlight the generalization capabilities of the model. The observed time-series is a combination of periodicity/cyclicity and a trend. In such cases, anomaly is not just a spike that is clearly distinguishable, but, it can be a data point that is locally deviated from the actual cycle. These local deviations are hard to detect robustly. Examples of such anomalies are magnified in Figure 5.5. It can also be seen in this figure that w data points are missing from the beginning of training and testing data set. In

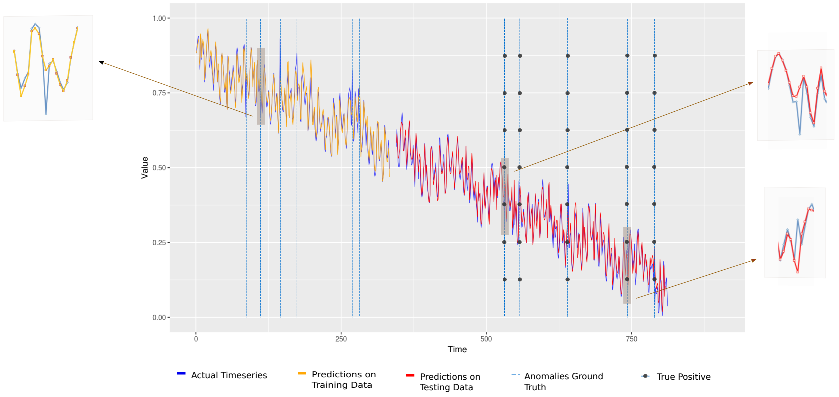


Figure 5.5: An example of time-series prediction and anomaly detection using DeepAnT is shown in this figure. Actual time-series is shown in blue color. 40% data of actual time-series are used as train set, while the rest of the 60% as test set. Time-series shown in yellow color are prediction results on train data, and time-series shown in red color are prediction results on test data. Vertical blue dotted lines are anomalies ground truth, and vertical blues line with dotes on them show point anomalies detected by DeepAnT (true positive). DeepAnT F-score is 1 for this time-series, whereas, Yahoo EGADS and Twitter anomaly detection F-score is 0.

both of the cases, this is the starting sequence (history window), after which the predictions are made.

On a detailed level, Table 5.2 shows a comparison of DeepAnT with EGADS, Twitter Anomaly Detection (AD), and LSTM (DeepAnT using LSTM as a predictor) on whole Yahoo Webscope data set. DeepAnT outperforms other methods in two data sets and for the rest, it is a runner up. A1 data set consists of anomalies where there is no trend and seasonality effect. Mostly, the anomalies are just the spikes in A1 data set. Since we are computing F-score based on the data set-level threshold, DeepAnT is not on top. Whereas, other methods are computing threshold separately for each time-series. For A3 and A4 data sets, F-scores of DeepAnT are significantly better than other methods. Twitter anomaly detection didn't work at all on A2 data set.

Quantitative evaluation

Table 5.2 also shows that the parameter 'Alpha' does not have a significant impact in this case. It is also important to note in this table that CNN-based DeepAnT performs better than LSTM on three data sets and performs slightly poor for one data set. It shows that CNN could be used in the cases when only limited amount of training data are available.

As DeepAnT's *Anomaly Detector* module is dependent on the *Time-series Predictor* module, good forecasting performance will result in better anomalous points detection. Figure 5.6 shows a plot of ground truth versus prediction evaluated on the test data of a time-series. Ideally, this should be a smooth line, but in practice, this line is uneven

Table 5.2: Average F-scores of Twitter AD, Yahoo EGADS, DeepAnT, and LSTM (DeepAnT using LSTM as time-series predictor) on Yahoo data set are given in this table. Bold F-scores are the best scores for corresponding Yahoo Webscope data set.

Data set	Yahoo EGADS	Twitter AD Alpha = 0.05	Twitter AD Alpha = 0.1	Twitter AD Alpha = 0.2	DeepAnT	LSTM
A1	0.47	**0.48**	**0.48**	0.47	0.46	0.44
A2	0.58	0	0	0	0.94	**0.97**
A3	0.48	0.26	0.27	0.30	**0.87**	0.72
A4	0.29	0.31	0.33	0.34	**0.68**	0.59

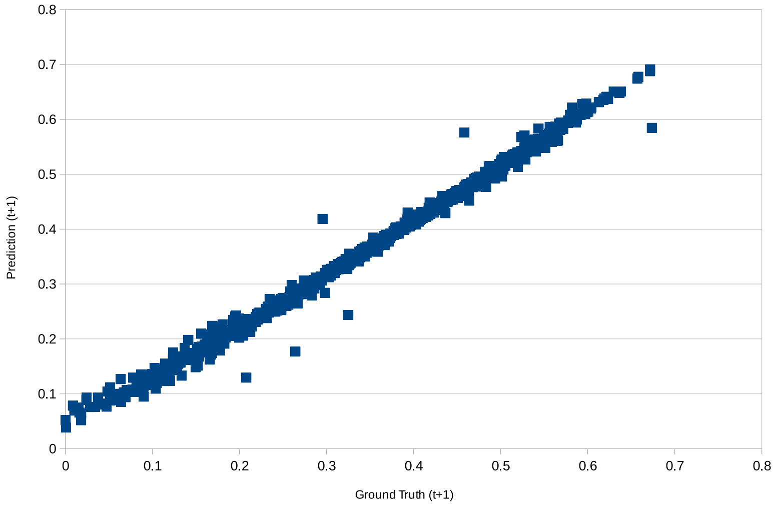


Figure 5.6: Time-series ground truth values at $(t+1)$ are plotted against predictions at $(x+1)$ to show the accuracy of the prediction model. The data points away from the diagonal line show anomalies.

due to minor errors in the prediction model. In this figure, points away from the diagonal line show the anomalous data points in the test data.

5.3 EXPERIMENTAL SETTING-II

5.3.1 *Data set*

NAB data sets

In this experimental setting, five real and two synthetic NAB data sets, namely Artificial no Anomaly, Artificial with Anomaly, Real Tweets, Real Traffic, Real Known Cause, Real Ad Exchange, and Real AWS Cloud Watch are used. Details of these publicly available NAB data sets are provided in Section 3.5.2. These data sets contain 58 data streams, each with $1,000$ - $22,000$ instances. These data sets are comprised of streaming data from different domains including road-traffic, network utilization, on-line advertisement, and internet-traffic. The data set is labeled either based on the known root cause of an anomaly or as a result of following the defined labeling procedure (described in [157]). Each data file consists of timestamps and actual data values. Anomaly labels of each data file are given in a separate set of files.

Challenges

Although NAB provides a diverse labeled streaming anomaly detection data set, but there are a few challenges [225] that makes it hard to be used as a practical anomaly detection benchmark. Each data point with ground truth anomaly label is centered by a defined anomaly window (10% the length of a data file), that makes the ground truth label of normal data points as an anomaly. Each data point with ground truth anomaly label is centered by a defined anomaly window, and the data points in the whole anomaly window are also labeled as anomalous. For example, for an anomaly window of size 350, all of the 350 data points in a data stream are labeled anomalous, whereas, there could be just 2-3 actual point anomalies in the center of this anomaly window. This kind of labeling helps in calculating good NAB score and leaves the recall very low. NAB score is introduced in [157] as an anomaly detection score that is designed to reward early anomaly detection and penalize later detection based on the true and false detection within an anomaly window.

5.3.2 *Experimental Setup and Evaluation Metric*

Two levels of experiment

A high NAB score shows that a particular algorithm has a higher tendency to detect early anomalies. However, it does not show how good that algorithm is in terms of maximum true detection of anomalies and minimum false alarms. In real life scenarios, in addition to early anomaly detection, it is equally important to detect correct number of anomalies. It is shown in [225] that in some cases, the NAB score is high, but the precision and recall is low, which means that the algorithm was not able to detect maximum number of anomalies. Two levels of same experiment are shown in this section. On the first level, we applied five time-series anomaly detection algorithms in addition to DeepAnT, on 20 NAB time-series from different domains. We picked same time-series as mentioned in [225]. The algorithms are evaluated on the basis of precision and recall. On the second level of this experiment, we have done the detailed analysis of 11 algorithms and compared them with DeepAnT on all the NAB data sets. The evaluated algorithms include Twitter AD (Twitter ADVec), context OSE, Skyline, Numenta, Multinomial Relative Entropy [247], Bayes changepoint detection [1], EXPoSE [214], and simple sliding threshold. All of these algorithms are used in same settings and with same parameters as mentioned in [4]. Ahmad et al. [4] have done extensive parameter tuning for each algorithm and used the optimal parameters. We have used

F-score for this detailed evaluation so that an overall performance of an algorithm can be reported. We are not reporting NAB score here because we want to evaluate an algorithm on the basis of the number of detected and rejected anomalies and the other arguments made in [225]. Since NAB benchmark consists of multiple time-series from different domains, we have reported the mean F-score per domain.

5.3.3 *Results*

Table 5.3 shows results of the first level of our NAB experiment. In most of the cases, high precision is followed by low recall. The main reason of such low recall is the labeling mechanism used in the NAB data set. It can be observed in this table that each algorithm is capable of achieving the precision close to 1, but recall stays in between $0.001 - 0.36$. In such cases, algorithms detect $1 - 4$ anomalies out of $346 - 401$ anomalies. Whereas, DeepAnT gives relatively better recall with equivalent precision as other algorithms (e.g., ec2-request-latency-system-failure, speed-t4013). Table 5.4 shows mean F-scores of a wide range of algorithms on the whole NAB data set that represents the results of second level of this experimental setting. It can be noted here that DeepAnT outperforms other algorithms with significant margin. DeepAnT is $2 - 13$ times better than the best performing algorithm for different domains in the NAB data set.

Low recall

5.4 EXPERIMENTAL SETTING-III

5.4.1 *Data set*

In this experimental setting, we have used 7 real and 1 synthetic data sets (mentioned in Table 5.5) that are most commonly used in classic anomaly detection settings. These multi-variant data sets are explained in Section 3.5.2. Each data set consists of different number of features and varying percentage of anomalies. Known anomaly cases are marked as ground truth in these data sets. We have removed all non-continuous attributes as done in [86, 165]. Data properties of these data sets are shown in Table 5.5. The number of features and anomaly percentage varies significantly between these data sets. The target class (anomaly) of each data set is also given in this table.

Classic anomaly detection data sets

5.4.2 *Evaluation Metric and Experimental Setup*

For the evaluation of different anomaly detection algorithms and the DeepAnT, ROC-AUC measure has been utilized in this experimental setting. ROC-AUC is used most commonly for reporting results of anomaly detection techniques for mentioned data sets. The evaluation is done in a semi-supervised fashion. In a semi-supervised set-

ROC-AUC

Table 5.3: Comparative evaluation of different state-of-the-art algorithms and the proposed algorithm on 20 NAB time-series from different domains. Precision and recall are reported in this table.

Time-series		ContextOSE		Numenta		NumentaTM		Skyline		ADVec		DeepAnT	
		Pre.	Rec.	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.
Real Ad Exchange	exchange-2-cpc-results	0.5	0.006	0	0	0	0	0	0	0	0	0.03	0.33
	exchange-3-cpc-results	0.75	0.02	1	0.013	1	0.007	0	0	1	0.02	0.71	0.03
Real AWS Cloud Watch	ec2-cpu-utilization-5f5533	1	0.005	1	0.007	1	0.01	1	0.002	1	0.002	1	0.01
	rds-cpu-utilization-cc0c53	1	0.005	1	0.002	1	0.002	1	0.1	0.62	0.012	1	0.03
Real Known Cause	ambient-temperature-system-failure	0.33	0.001	0.42	0.004	0.5	0.006	0	0	0	0	0.26	0.06
	cpu-utilization-asg-misconfiguratio	0.12	0.001	0.64	0.018	0.52	0.01	0	0	0.74	0.01	0.63	0.36
	ec2-request-latency-system-failure	1	0.009	1	0.009	1	0.009	1	0.014	1	0.02	1	0.04
	machine-temperature-system-failure	1	0.001	0.58	0.004	0.27	0.004	0.97	0.01	1	0.02	0.8	0.001
	nyc-taxi	1	0.002	0.77	0.007	0.85	0.006	0	0	0	0	1	0.002
	rogue-agent-key-hold	0.33	0.005	1	0.005	0.5	0.005	0	0	0	0	0.34	0.05
	rogue-agent-key-updown	0	0	0.14	0.002	0	0	0	0	0.11	0.002	0.11	0.001
Real Traffic	occupancy-6005	0.5	0.004	0.33	0.004	0.2	0.004	0.5	0.004	0.5	0.004	0.5	0.004
	occupancy-t4013	1	0.008	0.4	0.008	0.66	0.008	1	0.04	1	0.02	1	0.036
	speed-6005	0.5	0.004	0.66	0.008	0.25	0.008	1	0.01	1	0.01	1	0.008
	speed-7578	0.57	0.03	0.66	0.03	0.6	0.02	0.86	0.16	1	0.01	1	0.07
	speed-t4013	1	0.008	1	0.01	0.8	0.01	1	0.06	1	0.01	1	0.08
	TravelTime-387	0.6	0.01	0.25	0.004	0.33	0.004	0.62	0.07	0.2	0.004	1	0.004
	TravelTime-451	1	0.005	1	0.005	0	0	0	0	0	0	1	0.009
Real Tweets	Twitter-volume-GOOG	0.75	0.002	0.36	0.003	0.38	0.005	0.59	0.02	0.81	0.01	0.75	0.01
	Twitter-volume-IBM	0.37	0.002	0.15	0.002	0.22	0.005	0.22	0.01	0.5	0.009	0.5	0.005

Table 5.4: Comparative evaluation of the state-of-the-art anomaly detection methods on the NAB data set. Mean F-score is reported for each domain as each domain contains different number of time-series. DeepAnT out-performs all the other methods on whole data set (best mean F-score in bold).

Data set	Bayes Changepoint	Context OSE	EXPoSE	HTM JAVA	KNN CAD	Numenta	NumentaTM	Relative Entropy	Skyline	Twitter ADVec	Windowed Gaussian	DeepANT
Artificial no Anomaly	0	0	0	0	0	0	0	0	0	0	0	0
Artificial with Anomaly	0.009	0.004	0.004	0.017	0.003	0.012	0.017	0.021	0.043	0.017	0.013	**0.156**
Real Ad Exchange	0.018	0.022	0.005	0.034	0.024	0.040	0.035	0.024	0.005	0.018	0.026	**0.132**
Real AWS Cloud Watch	0.006	0.007	0.015	0.018	0.006	0.017	0.018	0.018	0.053	0.013	0.060	**0.146**
Real Known Cause	0.007	0.005	0.005	0.013	0.008	0.015	0.012	0.013	0.008	0.017	0.006	**0.200**
Real Traffic	0.012	0.02	0.011	0.032	0.013	0.033	0.036	0.033	0.091	0.020	0.045	**0.223**
Real Tweets	0.003	0.003	0.003	0.010	0.004	0.009	0.010	0.006	0.035	0.018	0.026	**0.075**

Table 5.5: Data properties of the used anomaly detection benchmarks.

Data set	No. of Instances	No. of Features	Anomaly Class	Anomaly Percentage
Shuttle	49097	9	classes $\neq$ 1 (class 4 removed)	7%
Pima	768	8	pos	34.9%
ForestCover	286048	10	class 4 (vs. class 2)	0.96%
Ionosphere	351	32	bad	36%
Http	567497	3	attack	0.39%
Smtp	95156	3	attack	0.03%
Mulcross	262144	4	2 clusters	10%
Mammography	11183	6	class = 1	2%

ting (also known as novelty detection [86]), training data consist of only normal data. In this setting, all the anomalies from the training set are removed in a pre-processing step.

We compare the results of three state-of-the-art anomaly detection methods, iForest, OC-SVM, and LOF with DeepAnT on the aforementioned data sets. For the model-based methods (iForest, OC-SVM, and DeepAnT), 40% of the actual data are used for training and rest for testing. To train iForest model, we have used the default parameters i.e. $\psi = 128$ and $t = 100$, as suggested in [165]. For OC-SVM, we have used Radial Basis Function (RBF) kernel. Commonly used setting of $k = 10$ is applied for LOF. For DeepAnT, history window of 2 is used, with the rest of the parameters being intact.

5.4.3 *Results*

Evaluation results of semi-supervised or novelty detection setting are shown in Table 5.6. DeepAnT shows best AUCs for most of the used data sets. In novelty detection setting, OC-SVM is considered the best method, but DeepAnT outperforms it in most of the data sets. These results show that DeepAnT is capable of finding anomalies in multivariant data set too.

5.5 EXPERIMENTAL SETTING-IV

5.5.1 *Discord Detection*

In the previous sections, we have shown that DeepAnT has the capability of detecting point anomalies as well as contextual anomalies in streaming and non-streaming data. In this section, we show that DeepAnT is also capable of detecting time-series discord. Time-series discords are sub-sequences of a longer time-series, that are different from rest of the sub-sequences [140]. Discords are considered as the

Table 5.6: Comparison of the state-of-the-art anomaly detection methods in semi-supervised (novelty detection) setting. DeepAnT performs best in most of the cases (best AUC in bold).

	iForest	OC-SVM	LOF	DeepAnT
Shuttle	0.98	**0.99**	0.56	**0.99**
Pima	0.40	0.26	**0.48**	0.31
ForestCover	0.78	0.70	0.57	**0.85**
Ionosphere	0.82	0.84	0.83	**0.85**
Http	0.98	**0.99**	0.42	**0.99**
Smtp	**0.80**	0.67	0.41	0.75
Mulcross	**0.99**	**0.99**	0.59	**0.99**
Mammography	0.85	0.89	0.72	**0.99**

NASA space shuttle data set

anomalous sequences in a time-series. For this experiment, we have picked NASA space shuttle valve data set [74] that is explained in Section 3.5.2. In this data set, some sub-sequences are normal whereas few are abnormal. We have down sampled this data set by 70% to show that time-series discord can be detected on far less data using CNN. Normal sub-sequences are shown in blue highlighted area in upper plot of Figure 5.7, whereas, the abnormal sub-sequence is shown in red highlighted area (to the right of the plot). Each sub-sequence is separated by a blue vertical line in this figure. Instead of extracting all the sub-sequences and converting them to some symbolic representation (as done in [140]), we simply train our predictor model on normal time-series. Same DeepAnT architecture and parameters are used here except the history window and the horizon. On a given test time-series, the DeepAnT predictor tries to predict the whole sub-sequence. By aggregating the anomaly score calculated at each timestamp (shown in bottom plot of Figure 5.7) of a sub-sequence, an anomaly score of whole sub-sequence is calculated. Based on the threshold applied to a sub-sequence anomaly score, discord is detected. In addition to the discord detection, the behavior of actual time-series that actually caused the discord can be detected using DeepAnT. It can be seen in the red highlighted area of the bottom plot that the anomaly score of the corresponding abnormal behavior is much higher that actually caused the discord.

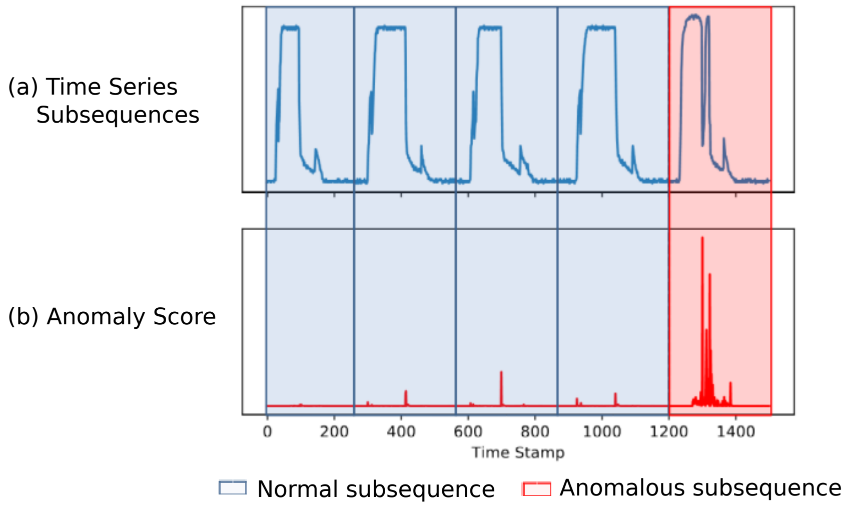


Figure 5.7: DeepAnT can also be used to detect time-series discords. Normal sub-sequences of a time-series are highlighted in blue color in plot (a), whereas, sub-sequence highlighted in red color is a discord. Lower plot (b) shows corresponding point-wise anomaly score of a sub-sequence, which is accumulated (per sub-sequence) to detect a discord.

6

FUSION OF STATISTICAL AND DEEP LEARNING FOR ANOMALY DETECTION

In the current era of smart and connected devices, the sensors in IoT devices are continuously generating streaming data that can be analyzed to a) monitor the device health, b) foresee the problems that could arise in the device, and c) make the device intelligent by adapting to varying behaviors. Nowadays, common use of the streaming data is to detect the anomalies in a system for fault diagnosis and predictive analytics [19, 36, 155, 181]. The connected devices are generating a large amount of data per second, so it is nearly impossible to analyze them manually. Therefore, it is vital to have a robust anomaly detection technique for streaming data.

The streaming data can be analyzed for multiple purposes

Considering the importance of anomaly detection and its wide area of applicability, there exists a lot of methods for anomaly detection in general [31, 89, 165, 206] and for streaming data in specific [138, 155, 157, 185]. In the context of streaming data, different methods have shown their supremacy over other methods for a particular set of use-case. However, no such method exists that can be deployed in every use-case [41]. Statistical models have proved to be quite effective in some areas for anomaly detection, while deep learning-based anomaly detection techniques have shown promising results in other domains. Each technique has its own advantages and limitations. Nowadays, much of the research is focused on deep learning-based approaches, whereas statistical models are widely accepted in a practical environment i.e. in industry, specifically due to their transparency. Both techniques are well suited for anomaly detection, but the choice of determining a technique depends on the use-case and the type of data. To fill this gap of picking 'one' model for a specific use-case and to increase the accuracy of the detected anomalies, we propose a fusion technique – Fusion of Statistical and Deep Learning for Anomaly Detection (FuseAD). This technique is based on the idea of fusing statistical and deep learning models for anomaly detection. By combining these two disjoint worlds as of now, we can profit from both. The main advantage of such a fusion is that where one model is weak, the strength of other model plays its role and improves the overall process of anomaly detection. In particular, the contributions of this chapter are as follows:

Combining two disjoint worlds

- A novel fusion method for combining deep learning-based and statistical model-based anomaly detection techniques is proposed. In contrast to the ensembling based anomaly detection

---

This chapter is an adapted version of the work published in [184].

methods in which one out of different forecasting results is picked based on the lowest error, the proposed residual scheme lets the network learn itself how to produce the best forecasting outcome based on two different kinds of models. In addition, the fusion mechanism enables the network to complement the strengths of the underneath two disjoint models by fusing the information encapsulated in them. As a result, the fused network performs better in such cases where a single model is unable to produce good results.

- Extensive evaluation of different distance-based, machine learning-based, and deep learning-based anomaly detection methods including iForest [165], OC-SVM [8, 169], LOF [31], PCA [220], Twitter AD [138], DeepAnT [185], Bayes ChangePT [1], Context OSE [4], EXPoSE [214], HTM Java [4], Numenta [157], Relative Entropy [247], Skyline [4], and Windowed Gaussian [4] on 11 anomaly detection data sets is provided. These data sets contain in total 425 time-series.
- An ablation study is provided in order to identify the contribution of the different components in FuseAD. In this study, we highlight the significance of using the fused model by comparing the results with each individual model.

6.1 LITERATURE REVIEW

Generally, for streaming data, the anomaly detection methods consist of two modules; a value at the next timestamp is forecasted first and then it is compared with the actual value to mark the data point as normal or anomalous [32, 185]. In most of traditional anomaly detection cases [256, 260], the forecasting module is based on ARIMA, that is a generalization model of ARMA [250]. It consists of three components, i) the *Auto-Regression* part uses the dependent relation between an observation and prior (lagged) values, ii) the *Moving Average* part incorporates the dependency between an observation and a residual error from a model applied to lagged observations, and iii) the *Integrated* part represents the difference between the observed values and the previous values.

Traditional methods

With the rapid increase in the applicability of ANN in different domains like automotive [137], government [238], health [55], security & surveillance [151]; more deep learning-based anomaly detection techniques are being introduced. Malhotra et al. [173] introduced LSTM-based anomaly detection technique for time-series data. They train stacked LSTM on non-anomalous data and use it as a predictor over different timestamps. The prediction errors are further modeled to access the likelihood of anomalous sequences. Chauhan and Vig [46] also proposed a similar approach based on deep LSTMs. The anomalous pattern detection technique for multivariate clinical time-series proposed by Lipton et al. [164] is also based on LSTMs. They have shown in their study that a LSTM trained on raw data is superior to a MLP trained on hand engineered features. Zheng et al. [268] proposed a CNN-based approach for multivariate time-series classification problem. Each channel of the proposed multi-channel deep CNN learns features individually when multivariate data is presented and classifies it as a normal or anomalous sequence. We have also proposed a CNN-based anomaly detection technique for time-series data in Chapter 5. In the area of network monitoring, Lopez-Martin et al. [168] introduced a method for network traffic classification. They combined LSTM and CNN models to better classify the network sequences without providing any hand engineered features. In order to apply CNN to time-series data, they proposed an approach to render the data as an associated pseudo-image. In contrast to all the aforementioned DNN-based anomaly detection techniques, DeepAnT detects point and contextual anomalies. It is relatively difficult to precisely detect point anomalies in streaming data as compared to the traditional classification of a sequence into normal or abnormal class because of the presence of seasonality and trend.

Deep learning-based methods

Du et al. (2017) [69] introduced a method for network fusion, in which they fuse together the soft metrics from different networks to generate the final confidence score. However, their approach is not directly applicable in our case where we aim to fuse a statistical model and a deep learning model to get benefit from the two different approaches. The anomaly detection technique proposed by Buda, Caglayan, and Assem [32] is merging the predictions from different LSTMs models and statistical models in the forecasting module. They proposed two approaches for merging the results of time-series forecasting. In the *Single-step merge*, each model forecasts the next value and the forecasted value with the lowest Root Mean Square Error (RMSE) is selected. In *Vote merge*, the best forecasting model is voted based on the training data. In their approach, each model works independently, and the best forecast is selected from a number of models. However, this is not the case in the technique that we propose in this chapter. In FuseAD, a network learns itself how and when to fuse the statistical and deep learning forecasting to generate the best forecasting results.

Fusion-based methods

6.2 METHODOLOGY

In this section, we explain the two forecasting models from the statistical and deep learning domains that we have selected for fusion in the presented technique. ARIMA and CNN forecasting models are building blocks of the proposed FuseAD and combined in such a way that both complement each other. These two models are used to fore-

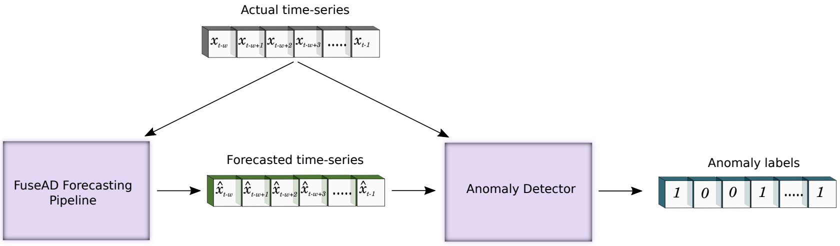

Actual time-series
x_{t-w} x_{t-w+1} x_{t-w+2} x_{t-w+3} ..... x_{t-1}
FuseAD Forecasting Pipeline
Forecasted time-series
$\hat{x}_{t-w}$ $\hat{x}_{t-w+1}$ $\hat{x}_{t-w+2}$ $\hat{x}_{t-w+3}$ ..... $\hat{x}_{t-1}$
Anomaly Detector
Anomaly labels
1 0 0 1 1

Figure 6.1: FuseAD overview: The system consists of two modules, Forecasting Pipeline and an Anomaly Detector.

cast the next timestamp in a given time-series. The forecasted value is further passed to an anomaly detector module that marks a data point as normal or anomalous.

6.2.1 *Statistical Model: ARIMA*

Used widely in industry for time-series regression task

ARIMA is a well-known and widely used statistical technique for time-series forecasting [122]. We used ARIMA as our statistical model since it has been employed successfully for a wide range of use-cases in industry to handle time-series regression tasks [52, 54, 95]. To get best out of the ARIMA model, it is important to find the right set of parameters for a given time-series. Non-seasonal ARIMA models are denoted as *ARIMA(p, d, q)*, where p, d, and q are non-negative integer parameters. Lag order *(p)* is the number of lag observations included in the model, degree of differencing *(d)* is the number of nonseasonal differences needed to make the series stationary, and moving average window size *(q)* is the number of lagged forecast errors in the prediction. Other details related to ARIMA can be found in Section 3.4.1.

6.2.2 *Deep Learning-based Model: CNN*

Proved superiority in many research and practical use-cases

CNN has proved its superiority over other ANN variants in many computer vision applications [146] and also in time-series anomaly detection applications [185]. The CNN model used in this study is composed of 2 convolutional layers, where each convolutional layer is followed by a max-pooling layer. Finally, the output is generated through a fully-connected layer producing continuous valued outputs. The network is trained through MAE as the loss function since the output is real-valued. We keep the architecture simple with a minimal number of parameters in order to make sure that the network can be successfully constrained to a reasonable solution with a very limited amount of data, which is a common case in publicly available time-series data sets. CNN is commonly used as a directly forecasting model. This formulation can be represented as:

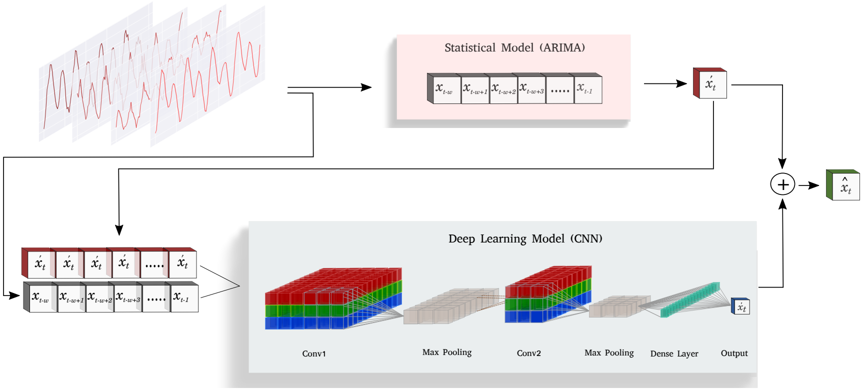


Figure 6.2: FuseAD forecasting pipeline.

$$\hat{x}_t = \Phi([x_{t-w}, ..., x_{t-1}]) \tag{6.1}$$

where $\Phi([x_{t-w}, ..., x_{t-1}])$ indicates the output of the network and $\hat{x}_t$ indicates the output of the system that are same in this case. In Equation 6.1, w is the size of history window. Therefore, the network learns a mapping from the input space $\mathcal{X}$ to the output space $\mathcal{Y}$.

6.2.3 *FuseAD: The Proposed Method*

Consists of two modules

The proposed technique consists of two modules as shown in Figure 6.1. First module is called *Forecasting Pipeline*: actual time-series is fed into this module and it generates a forecasted time-series. This forecasted time-series is further passed to an *Anomaly Detector* module, that is responsible for detecting anomalies. Based on the forecasted time-series and the actual time-series, *Anomaly Detector* marks each timestamp as normal or abnormal. Both modules are discussed in detail in this section.

6.2.3.1 *Forecasting Pipeline*

Instead of using both statistical and deep learning-based models in isolation, we combine these models in a novel residual learning scheme as shown in Figure 6.2. This enables the system to complement each other's strengths by using the information encapsulated in the other system. In this formulation, instead of treating the CNN as a mapping from the input space $\mathcal{X}$ to the output space $\mathcal{Y}$, we consider it to be a mapping from input space $\mathcal{X}$ to an intermediate space $\mathcal{Z} \in \mathbb{R}$. We then add an offset (the output of ARIMA) to transform it back to the output space $\mathcal{Y}$.

Residual learning scheme

The new formulation, therefore, is a mixture of the two models in a residual scheme. We simply augment the output of the CNN by introducing a summation layer in the end. In this way, the output of CNN can be considered as a correction term for the output of the

ARIMA model. In a case where the output of ARIMA is accurate, CNN can suppress its output in order to retain the prediction made by ARIMA. On the other hand, when the prediction is significantly off, the network can generate large offsets in order to compensate for the error made by the ARIMA model. In this way, the network itself can decide its reliance on the output of ARIMA during training to adapt its behavior so as to overcome its limitations. The new formulation can be written as:

$$\hat{x}_t = \Phi([x_{t-w}, ..., x_{t-1}; x_t^{'}]) + x_t^{'} \tag{6.2}$$

Correction term

where $x_t^{'}$ indicates the output from ARIMA, $\Phi([x_{t-w}, ..., x_{t-1}; x_t^{'}])$ indicates the output of the CNN network and $\hat{x}_t$ indicates the output of the whole forecasting pipeline. It is important to note that we condition the output of CNN on the output of ARIMA. This step is essential as we want to generate an offset or a correction term to the prediction made by ARIMA, therefore, the network should have access to the prediction made by ARIMA.

Single-step ahead forecasting

There can be many possible strategies to achieve this conditioning. We resorted to the easiest possible formulation by directly stacking the prediction of ARIMA in a different channel to the actual signal. This enables the network to keep in consideration the conditioning term at every point in the sequence. This conditioning strategy might be problematic for cases where multi-step-ahead prediction is desired, however, we restrict ourselves to single-step-ahead forecasting that is required for anomaly detection scenarios. This is an unsupervised learning technique that can benefit from a large amount of unlabeled data. Instead of ARIMA and CNN, other statistical and deep learning-based forecasting models can also be used to make predictions.

6.2.3.2 *Anomaly Detector*

Euclidean distance

When the forecasting model generates a prediction, it is passed to the anomaly detection module. Based on the anomaly score produced by this module, a timestamp is marked as a normal (0) or an abnormal (1) instance. The anomaly score (shown in the lower plot of Figure 6.3) is computed based on the distance between the predicted value and the actual value. We use Euclidean distance (as mentioned in [185]) given in Equation 6.3 as an anomaly score.

$$(x_t, \hat{x}_t) = \sqrt{(x_t - \hat{x}_t)^2} \tag{6.3}$$

where x_t is the actual value and $\hat{x}_t$ is the predicted value by the system computed using Equation 6.2.

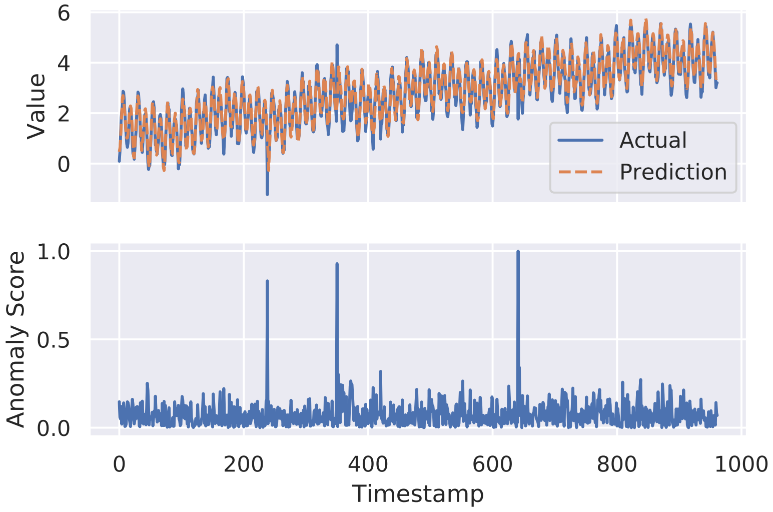


Figure 6.3: Forecasting and anomaly detection results of FuseAD on *TS11* time-series from Yahoo *A3* data set. The upper plot shows actual time-series and forecasting results on test data, whereas the lower plot shows anomaly score at each timestamp. Anomaly label (i.e. 1) is assigned to data points that have a high anomaly score.

6.3 EXPERIMENTAL SETTING-I

6.3.1 *Data set and Experimental Setup*

Yahoo Webscope data sets

In this experimental setup, 4 Yahoo Webscope data sets namely, Yahoo Webscope A1, Yahoo Webscope A2, Yahoo Webscope A3, and Yahoo Webscope A4 are used. Details of these data sets are given in Section 3.5.2.

We have used *Auto ARIMA* [122] to get the best ARIMA model for forecasting, as ARIMA requires data specific tuning to obtain the best results. Since each time-series has a different trend, change point, and seasonality, we tune the ARIMA model separately for each time-series. ARIMA model with a different set of parameters is tuned on 40% of a time-series, and the best model is selected based on the lowest Akaike Information Criterion (AIC) value. The best model is used to make a single-step-ahead (horizon of 1) forecast on the rest of the 60% data. For CNN-based forecasting, we have used the same hyperparameters as mentioned in Chapter 5. Same 40/60 data split is used here as used for the ARIMA model.

ROC-AUC

We have used area under the ROC curve (ROC-AUC) to provide an aggregated measure of the used models' performance. AUC value near to 1 represents a good measure of separability. Average AUC per Yahoo Webscope data set is reported as each data set contains multiple time-series.

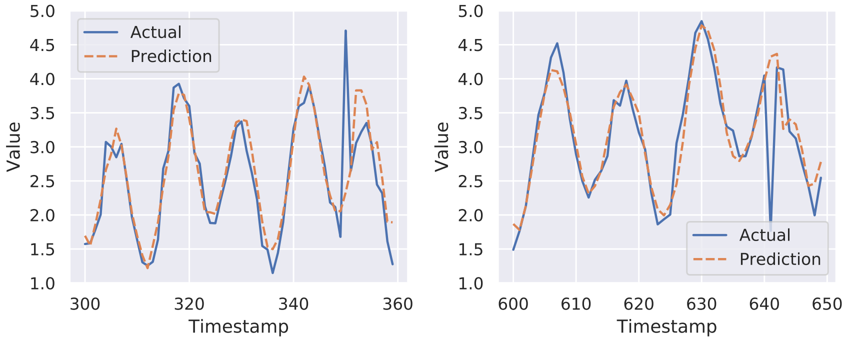


Figure 6.4: Zoomed-in plots of two out of three anomalies detected in Figure 6.3. It shows that FuseAD is capable of correctly detecting point anomalies in streaming data where traditional anomaly detection methods fail normally.

Table 6.1: Comparative evaluation of state-of-the-art anomaly detection methods on the Yahoo Webscope data set. Average AUC per sub-benchmark is shown in this table.

Data set	iForest [165]	OCSVM [169]	LOF [31]	PCA [220]	TwitterAD [138]	DeepAnT [185]	FuseAD
A1	0.8888	0.8159	0.9037	0.8363	0.8239	0.8976	**0.9471**
A2	0.6620	0.6172	0.9011	0.9234	0.5000	0.9614	**0.9993**
A3	0.6279	0.5972	0.6405	0.6278	0.6176	0.9283	**0.9987**
A4	0.6327	0.6036	0.6403	0.6100	0.6534	0.8597	**0.9657**

6.3.2 *Results*

The proposed method precisely detects point anomalies

Figure 6.3 shows the forecasting and anomaly detection results of FuseAD on a sample time-series from the Yahoo Webscope A3 data set. In the upper plot of this figure, the actual time-series is shown in blue, whereas the predictions are superimposed in orange. It can be seen in this plot that the network is able to learn the time-series trend and cycles. In the lower plot of this figure, anomaly score per timestamp is given. FuseAD detected three instances with a high anomaly score (peaks in lower plot) in the mentioned time-series. The zoomed-in plots of two of the detected anomalies (of Figure 6.3) are shown in Figure 6.4. In these plots, the normal behavior learned by the model is predicted per timestamp, that deviates from the observed behavior at index 350 and 641 respectively. It is clear from these zoomed-in plots that FuseAD is able to precisely detect point anomalies that are otherwise easily overlooked by traditional distance-based and density-based anomaly detection methods in time-series data. We have compared the ROC-AUC of FuseAD with other state-of-the-art anomaly detection methods including LOF [31], iForest [165], OC-SVM [169], PCA [220], Twitter AD [138], and DeepAnT [185] on Yahoo Webscope data sets. Table 6.1 compares the average AUCs of FuseAD with the mentioned methods. It can be seen in this table that FuseAD has outperformed the other methods.

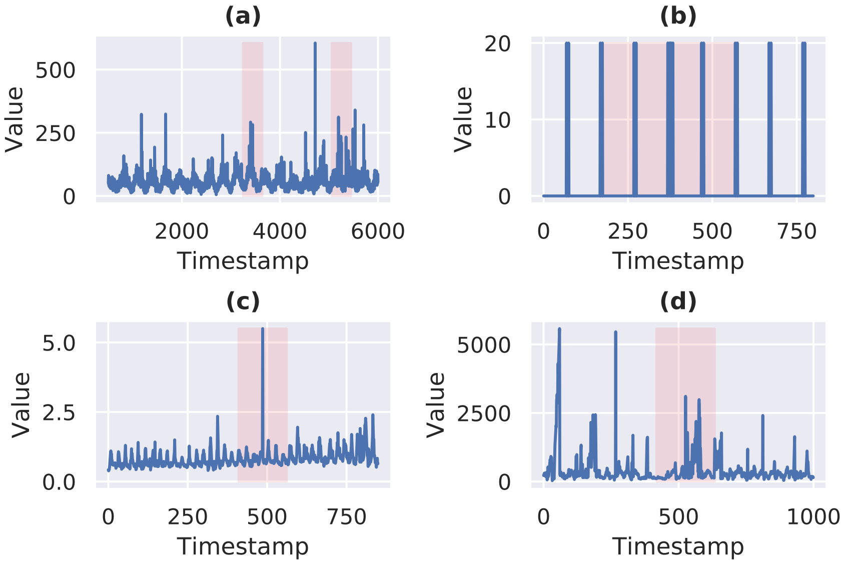


Figure 6.5: Snippets of NAB data sets from different time-series. Actual time-series are shown in blue, whereas the highlighted area shows an anomaly window. (a) Real Tweets (Twitter_volume_AMZN), (b) Artificial With Anomaly (art _increase_spike_density), (c) Real Ad Exchange (exchange-3_cpm_results), (d) Real Traffic (TravelTime_451_whole).

6.4 EXPERIMENTAL SETTING-II

6.4.1 *Data set and Experimental Setup*

NAB data sets are used in this experimental setup. There are 2 synthetic and 5 real NAB data sets, namely Artificial no Anomaly (Artificial nA), Artificial with Anomaly (Artificial wA), Real Tweets (Real-Tw), Real Traffic (Real-Tr), Real Known Cause (Real-KC), Real Ad Exchange (Real-AdE), and Real AWS Cloud Watch (Real-AWS). Details of these data sets are given in Section 3.5.2. There are total 58 time-series in these data sets, where each sequence is comprised of 1000 - 22000 instances. The snippets of few time-series are shown in Figure 6.5. Actual time-series is shown in blue, whereas the highlighted region shows anomaly labels.

NAB data sets

We have used 40% of each time-series of these data sets to train FuseAD in the same fashion as it is done for Yahoo Webscope data sets. We compared FuseAD with 9 other anomaly detection methods for the evaluation. The results of all the algorithms are reported on 60% of the actual time-series (test data).

For this experimental setup, we have used ROC-AUC to provide an aggregated measure of the used models' performance. Average AUC per NAB data set is reported as each data set contains multiple time-series.

Table 6.2: Comparative evaluation of anomaly detection methods on the NAB data set. Average AUC per domain is reported here. Bold numbers show highest AUC in a particular domain.

	Bayes ChangePT [1]	Context OSE [4]	EXPoSE [214]	HTM Java [4]	NUMENTA [157]	Relative Entropy [247]	Skyline [4]	Twitter ADVec [138]	Windowed Gaussian [4]	DeepAnt [185]	FuseAD
Artificial-nA	0	0	0	0	0	0	0	0	0	0	0
Artificial-wA	0.502	0.316	0.5144	**0.653**	0.531	0.505	0.558	0.503	0.406	0.555	0.544
Real-AdE	0.509	0.307	0.581	0.568	0.576	0.505	0.534	0.504	0.538	0.562	**0.588**
Real-AWS	0.499	0.311	0.594	0.587	0.542	0.506	0.602	0.503	**0.614**	0.583	0.572
Real-KC	0.501	0.486	0.533	0.584	0.590	0.503	**0.610**	0.504	0.572	0.601	0.587
Real-Tr	0.507	0.310	0.613	**0.691**	0.679	0.508	0.556	0.505	0.553	0.637	0.619
Real-Tw	0.498	0.304	**0.594**	0.549	0.586	0.500	0.559	0.505	0.560	0.554	0.546

6.4.2 *Results*

No clear winner

Table 6.2 shows comparative results of FuseAD and other streaming and kernel-based anomaly detection methods on NAB data sets. Average AUC per domain is reported for each anomaly detection method. We have used DeepAnT [185], Bayes ChangePT [1], Context OSE [4], EXPoSE [214], HTM Java [4], Numenta [157], Relative Entropy [247], Skyline [4], Twitter AD [138], and Windowed Gaussian [4] anomaly detection methods for the comparison. It can be observed in this table that there is no single anomaly detection method that outperforms others. There is high variance in the performance of every method on different data sets, where the average performance of every method is close to random guess. It is not because any of these methods are not capable of detecting anomalies in streaming data, but due to the poor labeling mechanism used in the NAB data sets. It can be seen in Figure 6.5(b) and Figure 6.5(c) that there is a small number of data points that are actually anomalous, but NAB labeling mechanism has labeled all the data points in an anomaly window as anomalous data points. Most of the data points in these windows are apparently normal. On the other hand, there are no anomalous data points in the anomaly windows shown in Figure 6.5(a) and Figure 6.5(d), whereas the actual anomalous data points are not labeled as anomalous. Due to these issues and other issues mentioned by Singh and Olinsky [225], it is hard for an anomaly detection method to have high AUC under these conditions. Most of the methods end up in random detection of anomalies for the NAB data sets.

Table 6.3: Ablation study on the Yahoo Webscope data set.

	A1			A2			A3			A4		
ARIMA	✓		✓	✓		✓	✓		✓	✓		✓
CNN		✓	✓		✓	✓		✓	✓		✓	✓
AUC	0.920	0.936	**0.947**	**0.999**	**0.999**	**0.999**	0.992	0.986	**0.998**	0.949	0.928	**0.965**

6.5 ABLATION STUDY

We performed an ablation study on FuseAD framework in order to identify the contribution of different components in the overall pipeline. FuseAD combines the statistical model – ARIMA, and the deep learning model – CNN, in a novel residual scheme. This combined prediction is fed to the anomaly detection module that decides if an instance is normal or abnormal. In the ablation study, we remove one of the forecasting modules i.e. either the CNN or the ARIMA model to check the influence of that model on the overall anomaly detection process.

Verify the contribution of each components

We first remove the CNN from the formulation presented in Equation 6.2. This leaves the final prediction to only rely on the prediction made by the forecasting model. Therefore, the new formulation becomes:

$$\hat{x}_t = x_t^{'} \tag{6.4}$$

where $x_t^{'}$ represents the prediction made by the statistical model and $\hat{x}_t$ represents the overall output of the system. Similarly, we remove the ARIMA model from the original formulation (Equation 6.2). In this case, CNN learns the complete input to output space projection. This formulation can be written as:

$$\hat{x}_t = \Phi([x_{t-w}, ..., x_{t-1}]) \tag{6.5}$$

where $\Phi([x_{t-w}, ..., x_{t-1}])$ represents the prediction made by the network and $\hat{x}_t$ represents the overall output of the system. We also remove the conditioning term since there is no statistical model available in this case. This formulation is exactly the same as the one from Equation 6.1 where we trained a CNN for forecasting except the presence of the FuseAD anomaly detection module in the end.

We use the same data splits as used by FuseAD for training the respective model, in order to have a fair comparison. The results from the ablation study on the Yahoo Webscope data sets are presented in Table 6.3. It is apparent from the table that a novel combination of the components leveraged by FuseAD significantly improves performance in most of the cases. The results from the ablation for NAB data sets are presented in Table 6.4. As evident from the table, the results are again chaotic due to the poor quality of the data set itself.

The fusion of components shows improvement in the performance

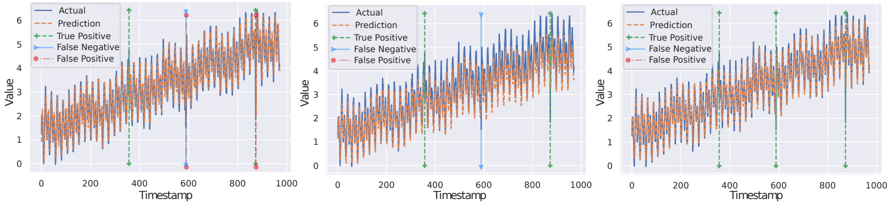
To highlight the importance of fusing statistical and deep learning models, comparative analysis of FuseAD based on the ablation

Table 6.4: Ablation study on the NAB data set.

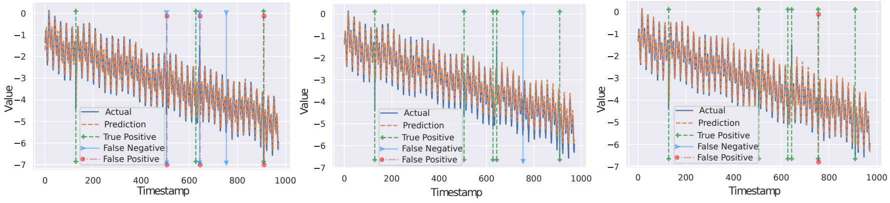
	Artifical -nA			Artifical-wA			Read-AdE			Real-AWS			Real-KC			Real-Tr			Real-Tw		
ARIMA	✓		✓	✓		✓	✓		✓	✓		✓	✓		✓	✓		✓	✓		✓
CNN		✓	✓		✓	✓		✓	✓		✓	✓		✓	✓		✓	✓		✓	✓
AUC	0	0	0	0.49	0.53	**0.54**	0.56	**0.58**	**0.58**	0.55	**0.58**	0.57	0.50	**0.60**	0.58	0.58	**0.61**	**0.61**	**0.55**	**0.55**	0.54

study are shown in Figure 6.6 on two different time-series from Yahoo Webscope data set. First, we remove the CNN forecasting model from FuseAD and show anomaly detection results generated by the ARIMA model (first column). Then, we remove the ARIMA forecasting model, which means only the CNN model is used for forecasting in FuseAD (second column). Finally, we use both forecasting models in actual FuseAD setting (third column). In this figure, actual time-series are shown in blue, whereas predictions are superimposed in orange. To highlight the detected anomalies and the performance of different models, true positives are shown in green, false negatives in blue, and false positives in red vertical lines. We use the same parameters as mentioned in [185] to detect anomalies and calculate F-score (mentioned in Figure 6.6). It can be observed in these plots that in both ARIMA plots, all of the anomalies are not detected (false negative) and there are many false alarms (false positive) too. In ARIMA cases, the false alarms are given just after the actual anomalous data point (normally 1 - 2 indexes), that shows that the forecasting is not very robust when there exist cycles and trends. In CNN plots, there are no false alarms, but there exist false negatives. FuseAD results presented in this figure show that an anomaly which is not detected by using a model in isolation is detected by fusing both models. By learning from both models, the number of false alarms is also reduced in FuseAD.

Reduced number of false alarms in FuseAD



ARIMA (0.57) CNN (0.80) FuseAD (1.00)



ARIMA (0.50) CNN (0.90) FuseAD (0.92)

Figure 6.6: Comparative analysis of FuseAD and anomalies detected by ARIMA and CNN models on two Yahoo Webscope time-series. The first row shows results on TS29 and the second row shows results on TS18 from $A3$ Yahoo Webscope data set. Respective F-scores are shown in brackets.

7

COMPARATIVE STUDY OF TRADITIONAL AND DEEP LEARNING-BASED ANOMALY DETECTION

Modalities of anomaly detection

Anomaly detection is an old research topic and a lot of advancements have been made in this area. There are two main modalities in which this wide research area can be divided: image-based and non-image-based. The notable modality is the image-based, in which it is easy to highlight a detected anomaly and most of the research studies are available for this modality. However, in the current era of the IoT and manufacturing advancement, where each device is connected to the internet and generating a bulk of data, non-image-based modality or time-series modality is also getting prominent for quite some time. Recent amplification in the usage of internet-enabled devices has created a high demand for robust anomaly detection methods for streaming sensor data. The presence of the intrinsic characteristics of a sensor-based streaming data makes the process of robustly detecting and representing anomalies hard.

There exist no silver bullet that works for every use-case

Vital developments in the area of DNN have proved them to be a very good option for most of the classification and regression problems [149]. There exist different deep learning-based and traditional approaches for anomaly detection, but their direct comparison on non-image streaming data is missing in the literature. Generally, when a new algorithm is proposed, it is reported in a particular setting and for a specific data set. Each method has its strengths and weaknesses. However, no such method exists that can be deployed in every use-case [41].

Focus of study

The focus of this study is to draw a comparative analysis for most commonly used traditional and deep learning-based anomaly detection methods for non-image streaming data. Given the superiority of DNNs in many recent studies, we hypothesize that DNN-based anomaly detection algorithms may outperform commonly used traditional methods on streaming data. We analyze these methods to find out if there is any benefit in using deep learning for anomaly detection or the same results can be achieved using traditional methods? Also, this study will help in deciding for which type of data set it is beneficial to use deep learning-based methods.

7.1 LITERATURE REVIEW

Recently, DNN-based approaches have enjoyed significant attention, owing to their amazing performance in various domains. Hence, it

---

This chapter is an adapted version of the work published in [182].

is no surprise that there has been an increase in DNN-based methods for anomaly detection as well. Chalapathy and Chawla [41] provides a review of DNN-based methods for anomaly detection employed in different application scenarios. Techniques reviewed in their study include both image as well as time-series domains with applications ranging from fraud detection, intrusion detection to medical anomaly detection and video surveillance. Although, their survey provides a comprehensive review of DNN techniques, there is no comparative analysis of DNN-based techniques with traditional anomaly detection techniques. Similarly, Chandola, Banerjee, and Kumar [42] provides a comprehensive survey of anomaly detection techniques comprising not only of methods based on simple machine learning, like clustering and nearest neighbours, but also based on statistical approaches and information theory with a diverse range of applications. Again, there is no comparative study among the techniques discussed in the article. Blázquez-García et al. [25] provided a review study on anomaly detection specifically for time-series data. They have provided anomaly detection methods review on both uni-variant and multi-variant time-series, albeit without any experimental results

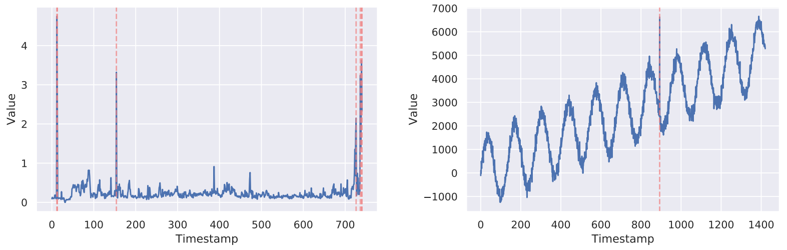
Deep learning-based methods

Goldstein and Uchida [90] provides a comparative study of different anomaly detection techniques for a range of different data sets. Although this study does compare different techniques on the same data sets to give a better comparison, it primarily focuses on multivariate tabular data and does not incorporate recent DNN-based anomaly detection approaches. Similarly, Gupta et al. [99] have performed an extensive survey on outlier detection techniques for temporal data. Their study provides an extensive overview of different techniques employed in multiple temporal data sets, but it lacks DNN-based techniques and their comparative analysis. Similarly, Kiran, Thomas, and Parakkal [144] provides a review of DNN-based approaches for anomaly detection in videos. Adewumi and Akinyelu [2] provides a comprehensive survey of DNN-based approaches in the domain of fraud detection. Similarly, Hodge and Austin [115] provides an extensive survey on statistical and some of the earlier machine learning-based outlier detection methodologies. A similar work is recently presented by Braei and Wagner [30] in which they have provided a quantitative evaluation on five data sets based on three evaluations metrics. Whereas, in our study, we have evaluated anomaly detection methods on 10 data sets based on 4 evaluation metrics.

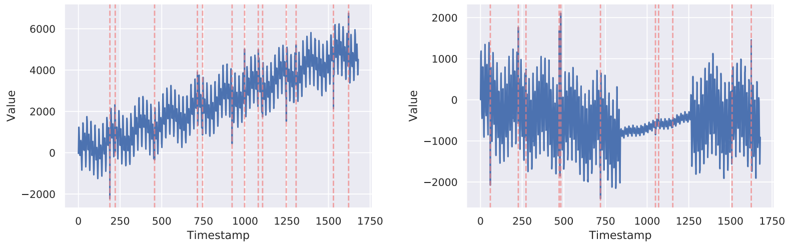
Comparative studies

Most of the surveys in the literature are general in nature that span over techniques that are proposed for different problems. Moreover, most of the review articles lack quantitative comparison that can determine the performance of a method. In this study:

- A wide range of **distance**-based, **density**-based, **kernel**-based, **cluster**-based, and **DNN**-based anomaly detection methods have been evaluated. The total number of methods is 13.



(a) A1 Data set (b) A2 Data set



(c) A3 Data set (d) A4 Data set

Figure 7.1: Sample time-series from Yahoo Webscope data sets. Actual streaming data are shown in blue, whereas red vertical lines are anomalous data points based on the provided labels.

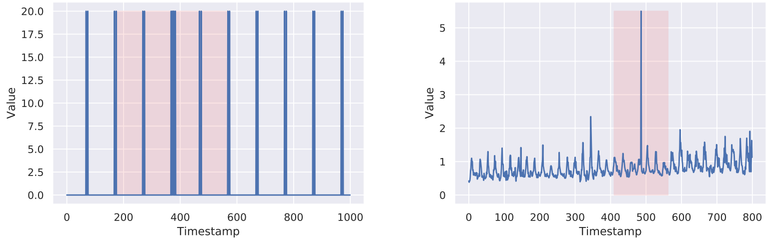
- To analyze the anomaly detection methods from different perspectives, we used the following evaluation metrics: *Precision @ Rank n (P@n)*, *ROC-AUC*, *Area Under Precision-Recall (PR) curve*, and *Inference Time*.

7.2 EXPERIMENTAL SETTING

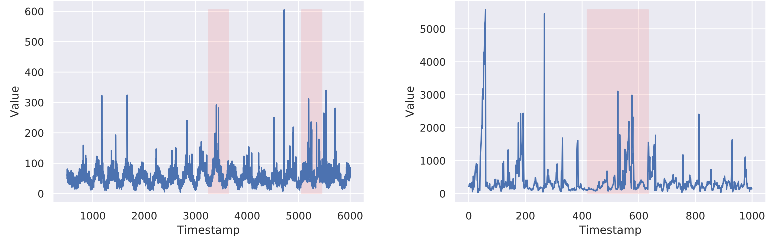
7.2.1 *Data sets*

We have selected time-series data sets from Yahoo Webscope and NAB benchmarks, that are already labeled by the publishers. There are total 10 real and synthetic data sets, namely Yahoo Webscope A1, Yahoo Webscope A2, Yahoo Webscope A3, Yahoo Webscope A4, Artificial with Anomaly, Real Tweets, Real Traffic, Real Known Cause, Real Ad Exchange, and Real AWS Cloud Watch that consist of 425 time-series. Details of these data sets are given in Section 3.5.2. Examples of time-series from Yahoo Webscope data sets are shown in Figure 7.1. In these figures, red vertical lines represent the labels of anomalies. Figure 7.1a and Figure 7.1b have random point anomalies, whereas, Figure 7.1c and Figure 7.1d have changing trends with pre-specified seasonalities. Figure 7.2 shows sample time-series from NAB data sets. In these figures, red highlighted areas represent the anomaly window. Details of these data sets are given in Section 3.5.2.

Yahoo Webscope and NAB data sets



(a) Artificial With Anomaly (b) Real Ad Exchange



(c) Real Tweets (d) Real Traffic

Figure 7.2: Sample time series from different NAB data sets. Actual streaming data are shown in blue, whereas the highlighted area shows anomaly window.

It is important to mention that only the data sets that have time-series characteristics and contain point and contextual anomalies are used in this study. There exists a lot of other data sets [38] that are converted to time-series from image and signal domains (e.g. *Breast Cancer Wisconsin* [38]). Such data sets are generally used for time-series classification, that is not the scope of this study.

7.2.2 *Experimental Setup*

The initial 40% of a time-series is used to build a DNN model and the rest of 60% for testing. For the distance-based and density-based anomaly detection methods that do not require a training process, we have only used 60% of the data for consistency and fair comparison with deep learning-based methods. Since all data sets consist of multiple time-series for a particular domain, the presented results are averaged per data set. We have selected the following algorithms for the evaluation: i) kNN Anomaly Detection [10], ii) LOF [31], iii) COF [231], iv) LOCI [194], v) iForest [166], vi) OC-SVM [215], vii) PCA [220], viii) HBOS [89], ix) XGBOD [266], x) AE [267], xi) XGBoost [47], xii) DeepAnT [185], xiii) FuseAD [184]. All these anomaly detection methods are explained in Chapter 3, 5 and Chapter 6. For all the anomaly detection methods used for evaluation, the default settings mentioned in [267] are used, except for DeepAnT and FuseAD. For these two methods, the settings mentioned in Chapter 5 and Chapter 6 are used.

Train-test split is 40%-60%

7.2.3 *Evaluation Metrics*

To analyse the anomaly detection methods from different perspectives, we have used following evaluation metrics:

- P@n
- Area Under ROC Curve (ROC-AUC)
- Area Under PR Curve
- Inference Time

7.3 RESULTS AND ANALYSIS

The comparative results of distance-based, density-based, kernel-based, and deep learning-based anomaly detection methods on 10 streaming data sets are shown in Table 7.1 and Table 7.2. Important evaluation metrics, the area under the ROC and PR curves are provided in Table 7.1, while the supporting metrics, P@n and inference time are provided in Table 7.2. For the Yahoo Webscope data sets, DNN-based anomaly detection methods are in lead by a clear margin in the terms of ROC and PR curves. For Yahoo Webscope A1 and Yahoo Webscope A2 data sets, kNN and LOF also show high ROC and perform on par in terms of the PR curve as compared to deep learning-based methods. It is mainly because of the relatively less complicated time-series in these data sets, also the spikes are very sharp in these time-series that are easily detected by distance-based methods. For Yahoo Webscope A3 and Yahoo Webscope A4 data sets, DNN-based methods are way ahead of the traditional methods in terms of both ROC and PR curves. The presence of trends, seasonality, and change-points make these data sets hard for traditional anomaly detection methods. These time-series characteristics also cause low PR curve value in traditional anomaly detection settings. DNN-based anomaly detection methods showed around 29% improvement in the ROC as compared to the best traditional anomaly detection method COF for Yahoo Webscope A3 data set and 25% improvement for Yahoo Webscope A4 data set. There are also noticeable improvements in terms of the PR curve for Yahoo Webscope A3 and Yahoo Webscope A4 data sets that actually shows the robustness of DNN-based methods in the streaming data. The PR curve increased 34% in DNN-based methods as compared to XGBoost for Yahoo Webscope A3 data set and 16% for Yahoo Webscope A4 data set. In terms of P@n metric, the same improvement trend is observed for DNN-based methods. For Yahoo Webscope A4 data set, even an improvement of 50% in P@n is observed in DNN-based methods. One downside of using DNN-based anomaly detection methods for this data set is a relatively high inference time. Although the inference time of deep learning-based methods is on par or even better

Deep learning-based methods are generally in lead for Yahoo Webscope data sets than some statistical and distance-based methods, it is not the minimum inference time. For the Yahoo Webscope data sets, PCA detects anomalies in minimum time. In comparison with other traditional anomaly detection methods, PCA performance is quite good. Its ROC, PR curve, and P@n is on par with other traditional methods, but it is far more superior than others in terms of producing results. LOCI turns out to be a bad choice for anomaly detection because of its moderate results and very high time complexity. Due to its high time complexity, we are unable to report the results for all of the data sets.

No clear winner for NAB data set

The overall performance of all anomaly detection methods is not very convincing on the NAB data sets. It is not due to the incompetence of these methods, but the labeling mechanism used in these data sets. For these data sets, a mix performance of anomaly detection methods is observed. There is no clear winner for these data sets as all methods perform on par. For some NAB data sets, DNN-based methods perform better in terms of ROC, whereas the traditional methods perform better in terms of other evaluation metrics. For these data sets, HBOS provides anomaly results in minimum time and kNN has maximum P@n in most of the cases.

Table 7.1: Comparative evaluation of different anomaly detection methods on Yahoo Webscope and NAB data sets. The area under the ROC and Precision-Recall (PR) curves are provided in this table. The highest score per data set is shown in bold.

			kNN [10]	LOF [31]	COF [231]	LOCI [194]	iForest [166]	OC-SVM [8]	PCA [220]	HBOS [89]	XGBoost [47]	XGBOD [266]	AE [267]	DeepAnT [185]	FuseAD [184]
Yahoo Webscope	A1	ROC	0.911	0.904	0.826	0.879	0.898	0.895	0.836	0.869	0.522	0.535	0.903	0.912 ± 0.010	**0.917** ± 0.018
		PR	**0.755**	0.665	0.466	0.260	0.710	0.705	0.672	0.571	0.516	0.517	0.740	0.648 ± 0.018	0.602 ± 0.011
	A2	ROC	0.920	0.901	0.858	0.851	0.662	0.913	0.923	0.652	0.500	0.500	0.878	0.962 ± 0.006	**0.982** ± 0.010
		PR	0.742	0.742	0.671	0.150	0.435	0.733	0.698	0.434	0.503	0.503	0.719	**0.881** ± 0.006	0.833 ± 0.008
	A3	ROC	0.654	0.641	0.698	-	0.628	0.657	0.628	0.630	0.511	0.551	0.653	0.922 ± 0.007	**0.976** ± 0.014
		PR	0.264	0.251	0.309	-	0.380	0.265	0.165	0.382	0.486	0.472	0.227	**0.829** ± 0.017	0.799 ± 0.009
	A4	ROC	0.648	0.640	0.682	-	0.629	0.651	0.610	0.636	0.504	0.532	0.661	0.870 ± 0.007	**0.935** ± 0.018
		PR	0.203	0.201	0.260	-	0.235	0.200	0.175	0.270	0.485	0.419	0.196	**0.642** ± 0.010	0.632 ± 0.008
Numenta Anomaly Benchmark (NAB)	Artificial	ROC	0.560	**0.605**	0.527	0.540	0.565	0.582	0.589	0.550	0.500	0.500	0.515	0.548 ± 0.010	0.545 ± 0.006
	With Ano.	PR	0.449	0.419	0.356	0.340	0.353	0.415	0.350	0.405	**0.609**	**0.609**	0.290	0.209 ± 0.005	0.196 ± 0.003
	Real Ad	ROC	0.530	0.504	0.503	0.499	0.520	0.519	0.415	0.510	0.511	0.532	0.477	0.562 ± 0.004	**0.590** ± 0.006
	Exchange	PR	0.146	0.003	0.171	0.143	0.139	0.143	0.120	0.149	**0.396**	0.215	0.134	0.156 ± 0.004	0.154 ± 0.001
	Real AWS	ROC	0.539	0.525	0.512	-	0.539	0.561	0.520	0.554	0.507	0.486	0.540	**0.583** ± 0.006	0.572 ± 0.002
	Cloud Watch	PR	0.211	0.189	0.209	-	0.240	0.241	0.193	0.267	**0.466**	0.345	0.333	0.197 ± 0.003	0.190 ± 0.007
	Real	ROC	0.632	0.578	0.493	-	**0.647**	0.637	0.596	0.641	0.519	0.544	0.613	0.608 ± 0.009	0.595 ± 0.012
	Know. Cause	PR	0.376	0.291	0.206	-	0.340	0.325	0.291	0.373	**0.515**	0.453	0.411	0.259 ± 0.011	0.236 ± 0.013
	Real	ROC	0.578	0.561	0.522	-	0.572	0.577	0.517	0.585	0.503	0.529	0.569	**0.630** ± 0.009	0.620 ± 0.004
	Traffic	PR	0.363	0.296	0.291	-	0.324	0.333	0.290	0.338	**0.570**	0.544	0.377	0.269 ± 0.004	0.262 ± 0.004
	Real	ROC	0.524	0.521	0.526	-	**0.553**	0.550	0.478	0.520	0.511	0.524	0.539	0.551 ± 0.002	0.547 ± 0.001
	Tweets	PR	0.207	0.169	0.197	-	0.138	0.140	0.114	0.183	**0.342**	0.179	0.193	0.123 ± 0.123	0.119 ± 0.119

Table 7.2: Comparative evaluation of different anomaly detection methods on the basis of Precision @ Rank n (P) and the inference time (T). The time is reported in seconds. For each data set, the maximum P@n and the minimum inference time are shown in bold.

			kNN [10]	LOF [31]	COF [231]	LOCI [194]	iForest [166]	OCSVM [8]	PCA [220]	HBOS [89]	XGBoost [47]	XGBOD [266]	AE [267]	DeepAnT [185]	FuseAD [184]
Yahoo Webscope	A1	P	0.7275	0.6517	0.4645	0.2491	0.5278	0.6166	0.6678	0.2327	0.0000	0.0345	**0.7387**	0.5656 ± 0.0087	0.5340 ± 0.0015
		T	0.0782	0.0025	1.8057	2155.1	0.0604	0.0074	**0.0002**	0.0245	0.0012	3.4775	0.0362	0.0457	0.0486
	A2	P	0.7200	0.7200	0.6472	0.0835	0.0100	0.6700	0.6817	0.0000	0.0000	0.0000	0.7000	**0.8738** ± 0.0045	0.8147 ± 0.0074
		T	0.0757	0.0022	1.7474	2196.7	0.0560	0.0078	**0.0003**	0.0011	0.0011	3.4474	0.1563	0.0466	0.0491
	A3	P	0.2570	0.2498	0.3136	-	0.1753	0.2579	0.1628	0.2030	0.0084	0.1052	0.2210	**0.8136** ± 0.0190	0.7723 ± 0.0069
		T	0.0907	0.0025	2.6382	-	0.0604	0.0108	**0.0002**	0.0012	0.0017	3.8734	0.3721	0.0608	0.0639
	A4	P	0.2022	0.2012	0.2670	-	0.1207	0.2011	0.1731	0.0889	0.0306	0.0486	0.2029	**0.6178** ± 0.0102	0.6122 ± 0.0003
		T	0.0911	0.0026	2.5313	-	0.0624	0.0108	**0.0003**	0.0011	0.0017	3.8460	0.5825	0.0584	0.0607
Numenta Anomaly Benchmark (NAB)	Artificial	P	0.3302	0.3722	0.2637	0.3076	0.3654	0.3518	0.3469	**0.3926**	0.0000	0.0000	0.2544	0.1972 ± 0.0142	0.1882 ± 0.0033
	With Ano.	T	0.2107	0.0105	9.4820	44322.9	0.0906	0.0341	0.0003	**0.0002**	0.0017	6.7487	0.3467	0.1336	0.1420
	Real Ad	P	0.1231	0.1213	0.0719	0.0877	0.1177	0.1228	0.0842	0.1274	0.0725	0.0724	0.0567	**0.1641** ± 0.0044	0.1603 ± 0.0044
	Exchange	T	0.0913	0.0057	1.6342	1533.4	0.0671	0.0063	0.0003	**0.0001**	0.0022	2.7408	0.1607	0.0603	0.0664
	Real AWS	P	**0.2171**	0.1718	0.1296	-	0.1792	0.1745	0.1998	0.1880	0.0574	0.0279	0.1391	0.1974 ± 0.0031	0.1901 ± 0.0080
	Cloud Watch	T	0.2176	0.0088	10.549	-	0.0883	0.0417	0.0003	**0.0002**	0.0033	7.1031	0.4798	0.1407	0.1537
	Real	P	**0.3375**	0.2674	0.1131	-	0.2881	0.3056	0.2848	0.3368	0.0716	0.0876	0.1115	0.2705 ± 0.0144	0.2544 ± 0.0118
	Know. Cause	T	0.4998	0.0168	101.82	-	0.1612	0.3609	0.0004	**0.0003**	0.0060	20.2553	1.3271	0.2830	0.2997
	Real	P	0.3746	0.3159	0.2740	-	0.3126	0.3217	0.2643	**0.3843**	0.0255	0.0700	0.1646	0.2493 ± 0.0041	0.2495 ± 0.0019
	Traffic	T	0.1253	0.0038	3.3476	-	0.0718	0.0124	0.0003	**0.0002**	0.0014	3.7763	0.3248	0.0888	0.0946
	Real	P	**0.3004**	0.2331	0.2204	-	0.1633	0.1644	0.1480	0.2322	0.1937	0.1616	0.0960	0.1485 ± 0.0024	0.1416 ± 0.0015
	Tweets	T	0.9121	0.0734	155.90	-	0.2242	0.6048	0.0004	**0.0003**	0.0160	34.7278	2.4480	0.4569	0.5019

Part III

UNCERTAINTY ESTIMATION

8

UNCERTAINTY ESTIMATION OF DNN DECISION

Over the last decade, CNNs have made phenomenal strides in various classification tasks using a wide array of input modalities. These powerful algorithms have achieved impressive performance, often at par with human experts, in many challenging natural scene image recognition tasks [119, 207, 270] and even in sensitive and critical application areas like medical image analysis for disease prediction [16, 73, 100, 198, 199]. These CNNs gained significant attention due to their parameter efficiency, in contrast to other deep learning models like densely connected MLPs, resulting in comparatively better generalisation performance. They are particularly powerful in analysing visual modalities like images and videos [159] but have also proved their worth in time-series analysis where they have been used for classification [268] and anomaly detection [185] (as shown in Chapter 5).

Performance of CNN

The fundamental principle behind conventional CNNs is to learn the optimal combination of network parameters (weights and biases) that can capture encoded representation of input training data. These conventional CNNs use point-estimates to represent network parameters and although they work astonishingly well in most image recognition tasks, where they have large insatiable appetite for data [112]. Additionally, the `softmax` function tips the odds in favor of one class by squashing classification probabilities for others. Therefore, it might results in overly confident predictions even when the network is completely wrong. This compulsive behavior of traditional point-based neural networks to always be relentlessly assertive in their prediction raises serious concerns in many crucial application areas like medical image analysis, security, autonomous driving, financial transactions and IoT-based human health monitoring. Also, the very nature of these point-based classifiers prohibits them to associate uncertainty with their predictions, which is a highly desired characteristic of any AI-based classifier.

Compulsive behavior of point-based classifiers

Bayesian estimation introduces a probabilistic perspective to the neural networks and addresses many shortcomings of traditional point-based neural networks. It represents each parameter with a probability distribution instead of a single point-estimate. As a result, Bayesian neural networks are able to learn effectively from relatively small amount of data and thus are fairly robust to over-fitting [218]. They can provide an inherent regularisation effect [227] by constraining the network parameters within a distribution instead of letting

Probability distribution

This chapter is an adapted version of the work published in [15].

them increase out of bound. Most importantly, Bayesian inference can allow to estimate network's uncertainty about any prediction. However, a full Bayesian estimation over all network parameters is computationally expensive and finding true posterior probability is intractable. These limitations are normally addressed by employing various tricks like Markov Chain Monte Carlo (MCMC) sampling [188] and Variational Inference (VI) [133], or a combination of the two [209], to approximate the true posterior with a manageable distribution. A CNN trained using Bayesian estimates for network parameters is shown to lag its counterpart, trained using point-estimates, in terms of classification accuracy [218, 219].

Hybrid training paradigm

In this chapter, we recognize specific merits of each approach discussed above and combine them into a hybrid training paradigm. This hybrid approach integrates deterministic CNNs, where each parameter assumes only one value. With probability driven Bayesian CNNs, where each parameter may take any value drawn from a probability distribution characterized by a mean and a standard deviation. This probability distribution is learned for each parameter during training. The proposed hybrid training method provides an estimate of uncertainty, using Bayesian classifier, without compromising on classification accuracy owing to deterministic feature extractor. It also captures maximum weight configurations from small data sets while still remaining computationally manageable. The proposed approach is tested on 13 different classification data sets including benchmark image data sets, fine-grained medical image data sets and time-series data sets. The proposed hybrid method is proved to be superior to both fully deterministic and fully Bayesian CNN approaches in terms of classification accuracy.

8.1 LITERATURE REVIEW

Conventional CNNs have demonstrated their worth in various image recognition tasks since long [160] and have resurged into popularity in 2012 with Alexnet [149]. They have lately evolved into awfully complicated networks spanning thousands of layers [106].

Bayes by backpropagation

Although applications of Bayesian method into neural networks have also been investigated for many decades [26], it was only after Blundell et al. [27] proposed Bayes by backpropagation that training of deep neural networks was made possible using Bayesian estimation. This method of VI allowed backpropagation of so called Expected Lower BOund (ELBO) loss and regularising weight distributions. A CNN trained using Bayesian method was recently proposed by Shridhar et al. [219] as a fundamental construct for other network architectures. They used Bayes by backpropagation for training convolutional network and reported comparable results on many benchmark data sets.

Acknowledging the excessive computational cost of Bayesian models, Gal and Ghahramani [77] proposed a Monte Carlo dropout method to approximate Bayesian inference in deep Gaussian processes. The method is equivalent to performing multiple forward passes through the network and taking the average of results to model the uncertainty of the network. Kwon et al. [153] recognized the importance of uncertainty quantification especially in medical domain and proposed to calculate it by splitting the uncertainty into aleatoric, that corresponds to model's uncertainty; and epistemic uncertainty, that represents inherent noise in the data. Kendall and Gal [139] studied the advantages of modelling epistemic uncertainty as compared to aleatoric uncertainty in deep Bayesian models.

Monte Carlo dropout

Combining deterministic and probabilistic models in various fashions has also been studied for long. Tang and Salakhutdinov [232] pointed out that the conditional distribution $p(Y|X)$ does not need to be unimodel, as normally assumed by MLPs, but can also be represented as a multimodel output distribution for many structured prediction problems. They proposed a hybrid Sigmoid Belief Networks (SBNs) with some stochastic hidden variables and some deterministic hidden variables and achieved superior performance on synthetic and facial expression data sets. Similarly, other neural networks with partially Bayesian parameters have been proposed for regression tasks as alternative to Gaussian Processes [158, 227], that do not scale well with the number of training samples.

Combining deterministic and probabilistic models

The problem of estimating uncertainty has been addressed in variety of ways, for example out-of-distribution (OOD) samples detection [110, 161] and density estimation using flow based models. Normalizing flows and autoregressive models have been successfully combined to produce state-of-the-art results in density estimation, via Masked Autoregressive Flows (MAF) [195]; and to accelerate state-of-the-art WaveNet-based speech synthesis to 20x faster than real-time [190], via Inverse Autoregressive Flows [143]. Huang et al. [118] presented Neural Autoregressive Flows and demonstrated that these models are universal approximators for continuous probability distributions, and their greater expressivity allows them to better capture multimodal target distributions. Adding on to their work, De Cao, Titov, and Aziz [62] proposed Block Neural Autoregressive Flow that is a much more compact universal approximator of density functions, where a bijection is directly modeled using a single feedforward network. Dinh, Sohl-Dickstein, and Bengio [67] introduced a set of transformations called real-valued Non-Volume Preserving (real NVP) as a tractable and expressive way to modelling high-dimensional data. Ardizzone et al. [12] extended real NVP architecture and argued that their proposed Invertible Neural Networks are well suited for determining full posterior parameter distribution conditioned on training data. They noted that alternating backward and forward training

Estimating uncertainty

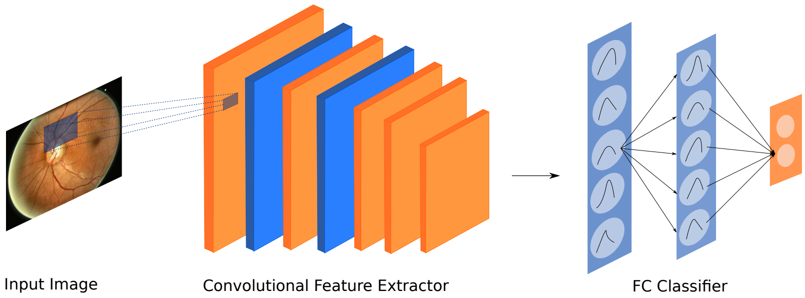


Figure 8.1: Proposed Hybrid Model. Convolutional Layers are trained separately using point estimates. The parameters of the convolutional layers are then frozen and Bayesian classifier is trained.

passes and accumulating gradients from both sides before updating parameters allows efficient training. Kingma and Dhariwal [142] furthered flow-based generative models [66] that are useful for calculating exact log-likelihood, performing exact latent-variable inference, and parallelising training and synthesis pipelines. Their Generative flow model uses an invertible 1×1 convolution and is shown to be capable of efficient and accurate synthesis of large images.

8.2 METHODOLOGY

Convolutional layers

A CNN primarily consists of two main modules: a feature extractor and a classifier. The proposed network consists of a set of convolutional layers trained with point estimates followed by fully-connected layers trained using Bayesian estimate. It provides a trade-off between high accuracy of deterministic models and uncertainty estimation of Bayesian models. It also restricts the parameter space of the network as compared to fully Bayesian models because only the classifier part of the network treats its parameters as random variables. Figure 8.1 shows schematic diagram of the hybrid model proposed in this work. The network initially trains to optimize parameters for both convolutional feature extractor and dense classifier as given below.

$$\mathcal{W}_C^*, \mathcal{W}_D^* = \underset{\mathcal{W}_C, \mathcal{W}_D}{\arg\min} \frac{1}{|\mathcal{X}|} \sum_{(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}} \mathcal{L}\Big(\psi\big(\Phi(\mathbf{x}; \mathcal{W}_C); \mathcal{W}_D\big), y\Big) \tag{8.1}$$

where $\mathcal{L}$ denotes the loss function, Φ represents the convolutional part of the network parameterised by $\mathcal{W}_C$ and ψ represents the dense layers (forming the classifier) parameterised by $\mathcal{W}_D$.

Once the network is trained using point-estimates, we reinitialize fully connected layers with random variables following normal distribution and retrain them using Bayesian estimation. The parameters of convolutional feature extractor are frozen throughout this retraining. This whole training paradigm allows us to capitalize on economically learned features by deterministic convolutional block and use expensive Bayesian inference only to approximate posterior distribution, that might then be used for uncertainty estimation. Mathematically, the learning of Fully-connected (FC) classifier of hybrid model is given by;

$$\theta_D^* = \arg\min_{\theta_D} \frac{1}{|\mathcal{X}|} \sum_{(\mathbf{x},y)\in\mathcal{X}\times\mathcal{Y}} \mathcal{L}\Big(\Psi\big(\Phi(\mathbf{x}; \mathcal{W}_C^*); \theta_D\big), y\Big) \tag{8.2}$$

where Ψ represents the Bayesian layers learned through Bayes by backpropagation and θ_D denotes the distribution of weights. Since the weights are described by a distribution instead of point-wise estimates, $\mathcal{L}$ in this case denotes the ELBO loss. Convolutional feature extractor trained with point-estimates learns crisp features of the input data while probabilistic classifier allows to sample from posterior distribution and offers an insight into network's confidence.

Posterior distribution

After this retraining is finished, we perform inference by passing test samples a number of times from our network. Since the parameters of the last FC layers of the network are sampled from a probability distribution, each pass of the same test sample gives a different prediction. These output predictions are used to draw a posterior distribution and estimate network's uncertainty. Complete algorithms used for this task is given in Algorithm 1.

Uncertainty analysis

For uncertainty analysis in Bayesian and hybrid architectures during inference, the algorithm works by sampling 10 classifier models from Bayesian weights distribution for every test sample and take their output predictions. This way, instead of a single prediction, we get a set of predictions representing a probability distribution on network's output. This set of predictions are normalized in $[0-1]$ range using min-max normalization for direct comparison. Predictions for top two classes are taken and difference in their values is recorded. After having the normalized differences, we build a distribution of all these differences and use a percentile value (40% in this case) to automatically select a threshold for the measure of uncertainty. The percentile value of 40% is determined heuristically. This parameter can be considered as a knob to control how confident predictions are desired in any given application area. In circumstances where no prediction is deemed better than a wrong prediction (medical diagnosis, for example), this value can be raised to ensure that only the most confident predictions are given by the network. For other, relatively less critical scenarios, this knob can be adjusted accordingly. The underlying assumption for our uncertainty estimation is that if the output for two classes is fairly distinctive then the difference in top two classes should be greater than the threshold and the model is regarded as certain about prediction otherwise it is considered uncertain. If a test sample is regarded as certain by more than half models

Algorithm 1 Uncertainty Estimation

Inputs modelOutput: Array containing softmax probabilities of all images for all models
allPredictions: Array containing class predictions for all images and for all models
allTargets: Array containing actual targets for all images and for all models
percentile: A scalar parameter to ascertain uncertain images to ignore
consensus: A scalar parameter representing minimum number of confident models to reach certain prediction
Outputs certainAccuracy: Accuracy when model is certain
uncertainImages: A percentage of uncertain images filtered out

```
procedure estimateUncertainty
    for each model $i$ in $allModels$ do
        for each image $j$ in $allImages$ do
            $differences$ = differences of top two classes' probabilities in $modelOutput[i][j]$
        end for
    end for
    $threshold$ = calculate for each model by filtering $percentile$ number of images from $differences$ of each model and average them.
    for each image $j$ in $allImages$ do
        Let $confPred = 0$, $uncertain = 0$, $confModels = 0$ be new variables
        for each model $i$ in $allModels$ do
            if $differences[i][j] > threshold$ then
                if $allPredictions[i][j] == allTargets[i][j]$ then
                    increment $confModels$
                end if
            end if
        end for
        if $confModels >= consensus$ then
            increment $confPred$
        else
            increment $uncertain$
        end if
    end for
    return $confPred/(len(allImages) - uncertain)$, $uncertain/len(allImages)$
end procedure
```

Table 8.1: Time and space requirement of deterministic, Bayesian and hybrid models for some data sets.

Data set	**Network Parameters (Millions)**			**Execution Time per epoch (s)**		
	Deterministic	**Bayesian [219]**	**Hybrid [Ours]**	**Deterministic**	**Bayesian [219]**	**Hybrid [Ours]**
MNIST	2.457	4.914	2.459	15	70	27
CIFAR-10	5.851	11.703	9.528	25	129	49
ISIC-Subset	58.294	116.587	112.840	338	832	602
ORIGA	58.29	116.579	112.831	5	16	6
Electric Devices	0.655	3.277	0.577	2	16	3
Mallat	3.801	33.423	3.486	2	10	3
Thorax-1	2.726	24.589	2.569	2	10	5

(represented by *consensus* parameter), using simple majority voting, then it is output as a fairly certain prediction.

8.2.1 *Time and space complexity analysis*

Fewer parameters

The proposed hybrid model uses fewer parameters than its Bayesian counterpart as is evident from Table 8.1. This table shows the number of trainable parameters in each method and training time per epoch for some of the data sets. The hybrid model does not incur any additional cost for combining the benefits of both deterministic and Bayesian methods.

The time complexity of the proposed algorithm is $O(2M \times I)$, where M represents number of Models sampled and I denotes the number of test samples. Also, the algorithm computes in constant space since, regardless of number of total models and test samples, only one model and one test sample are loaded at any given time.

8.3 EXPERIMENTAL SETUP AND DATA SET

13 *data sets*

We used 13 data sets of disparate modalities and from diverse areas of application to ascertain the viability of the proposed hybrid CNN architecture. A brief description of all the data sets used and overall experimental setup is given in this section.

8.3.1 *Data sets*

Table 8.2 gives an overview of all the data sets used in this work. We picked standard benchmark image data sets, as well as challenging fine-grained medical image classification data sets and many time-series data sets so that the validity of our approach on a broad range of data sets may be extensively investigated.

Table 8.2: Distribution of data sets used to evaluate proposed architecture

Data sets	Modality	No. of Classes	No. of Samples		
			Train	Test	Total
Image Data sets					
MNIST	Grey Images	10	70k	10k	80k
CIFAR-10	Color Images	10	50k	10k	60k
Medical Image Data sets					
ORIGA	Color Retinal Fundus Images	2	520	130	650
ISIC-Subset	Color Clinical Skin Images	3	5201	600	5801
Time Series Data sets					
Fish	Image-derived data	7	175	175	350
ShapesAll	Image-derived data	60	600	600	1200
Plane	Sensor data	7	105	105	210
TwoPattern	Simulation data	4	1000	4000	5000
ECG5000	ECG data	5	500	4500	5000
MedicalImages	Image-derived data	10	381	760	1141
ElectricalDevices	Device data	7	8926	7711	16637
Mallat	Simulation data	8	55	2345	2400
ECG Thorax1	ECG data	42	1800	1965	3765

8.3.1.1 *Image Data sets*

MNIST, CIFAR-10, ORIGA, ISIC

We used two of the most common benchmark data sets i.e. MNIST [160] and CIFAR-10 [235] and two publicly available medical image data sets i.e. ORIGA [263] and a subset of ISIC Archive to evaluate the performance of our proposed approach. For MNIST and CIFAR-10, standard pre-defined train and test splits are used. ORIGA data set provides clinical ground truth to benchmark segmentation of optic disc and classification of healthy and glaucomatous images. It provides optic Cup-to-Disc Ratio (CDR) and labels for each image as glaucomatous or healthy. This data set has been used as a standard data set in many recent state-of-the-art researches for glaucoma classification. Since this data set is very small and no predefined train and test splits are given, we used 5-fold Cross Validation (CV) for this data set such that in each iteration of CV there are 130 images in validation fold and 520 images in training fold. The second data set of medical images was taken from ISIC Archive 2018 version. It consists of around 24,000 clinical and dermoscopic images of skin lesions categorized into 7 classes. Some of the classes in this data set have as fewer as 122 images per class, therefore, we took a subset of the whole data with three largest classes namely Benign Keratosis-like Lesions

(BKL), Melanoma (MEL), and melanocytic Nevi (NV) and randomly divided them into training and test sets.

8.3.1.2 *Time series Data sets*

We selected 9 publicly available data sets, namely Fish, ShapesAll, Plane, TwoPattern, ECG5000, MedicalImages, ElectricalDevices, Mallat, and ECG Thorax1 from UCR archive [61]. The time series data sets were generated based on different sources including device usage, sensors data, ECG, motion sensor, or simulation etc. Each time series contains different number of classes; and the number of observations also vary in each data set. All data sets are already divided into train and test sets by the publisher.

9 publicly available data sets

8.3.2 *Data Pre-processing*

To pre-process benchmark image data sets (MNIST and CIFAR-10), we used random crop and normalization by mean subtraction. On medical image data sets (ORIGA and ISIC Subset), histogram equalization is applied to enhance contrast and normalize brightness. We also made use of different data augmentation techniques like rotations, flipping, and random crops to increase the data set size. Note that in addition to pre-processed images, original images are also kept in the data set. Data augmentation was done keeping in mind the class ratio, such that the minor class can have more augmentations and more copies generated. Time-series data sets are used without any pre-processing.

Normalization and histogram equalization

8.3.3 *Hyperparameter Selection*

All of the image data sets were trained and compared with similar experimental setup. We used a 5-layer convolutional block as baseline CNN, however, our experiments with varying depths and breadths of CNN shows that the approach is fairly scalable to more advance CNN architectures. We trained this CNN using Maximum Likelihood Estimation (MLE) for 60 epochs with a learning rate of 0.001, weight decay of 5×10^{-4}, and batch size of 32. For probabilistic models, we used the same setup as described above but instead of using point estimates we trained convolutional and fully connected layers with distribution-based weights using Bayes by backpropagation for 60 epochs. In our proposed hybrid approach, we employed a FC classifier with frozen convolutional feature extractor, pre-trained using MLE, and fine-tuned it using Bayesian estimation for 60 epochs with similar parameters. Two hyperparameters used in the proposed algorithm, $percentile$ and $consensus$ can be selected as per use case requirements. In critical application areas, for example medical image diagnosis or stock

Image modality

Table 8.3: Comparison of deterministic, Bayesian, and proposed hybrid models on different data sets without using uncertainty estimation

Data sets	**Deterministic Accuracy (%)**	**Bayesian [219] Accuracy (%)**	**Hybrid [Proposed] Accuracy (%)**
Benchmark Data sets			
MNIST	99.0	99.01	**99.3**
CIFAR-10	88	72.0	**88.7**
Medical Image Data sets			
ORIGA	76	74.4	**80.3**
ISIC-Subset	74	65.5	**75.7**
Time Series Data sets			
Fish	**85.1**	80.7	84.7
ShapesAll	67.0	70.9	**72.3**
Plane	**97.0**	96.7	95.1
TwoPattern	89.0	81.0	**89.4**
ECG5000	92.0	**93.2**	91.9
MedicalImages	**69.0**	62.4	64.7
ElectricalDevices	55.0	54.0	**56.6**
Mallat	88.0	82.5	**89.3**
ECG Thorax1	90.0	89.1	**91.3**

market prediction, where there is little room for incorrect classification, higher values of these parameters can be selected to ensure only the most certain predictions are given by the network. In other applications, a relaxed criterion for uncertainty estimation might be acceptable. In our experiments, we used $percentile = 40\%$ and $consensus$ of more than half models (i.e. 6 models). These values were selected empirically and they worked well in all 13 data sets of different kind. It should be emphasized here that, for a given data set, we used the same underlying architecture (number, width, and depth of convolutional layers and size of dense layers) in all three training paradigms, i.e. fully deterministic, fully Bayesian and Hybrid, to ensure fair comparison among three approaches.

Time-series modality

For time-series modality, we used CNN with two convolutional layers, each followed by a max pooling layer for deterministic model analysis. On top of that, two FC layers were added as classifier. For probabilistic and hybrid approach, we used the same setting as explained before.

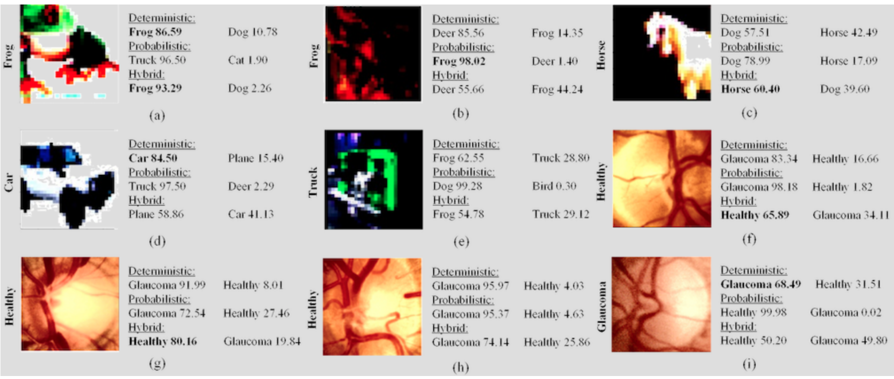


Figure 8.2: An analysis of confidence comparison for all three approaches on various samples of CIFAR10 and ORIGA data sets. The actual class is mentioned on left side of each image in bold vertical text.

8.4 EVALUATION AND DISCUSSION

Comparison

Table 8.3 summarizes classification accuracies obtained by traditional fully deterministic CNN, Bayesian CNN [219], and our proposed hybrid approach. The table shows that the proposed hybrid approach outperforms not only purely Bayesian CNNs but also their deterministic counterparts in 9 out of 13 data sets while giving comparable results on rest of them. Even when the hybrid approach lagged other methods in classification accuracies, the difference was very small and came at no additional cost in terms of time or number of parameters as shown in Table 8.1. The results in *Bayesian Accuracy* field in Table 8.3 are generated by our own experiments using the implementation of Shridhar et al. [219] for Bayesian CNN.

Figure 8.2 shows output probabilities of deterministic, Bayesian and hybrid models for various correctly classified and misclassified images from CIFAR-10 and ORIGA data sets. It can be observed in Figure 8.2 that when hybrid model was unable to make a correct prediction (sub-figures (b), (d), (e), and (h)), it associated relatively smaller probability scores with its misclassification than its competing models who also misclassified but did so with overconfidence. For example, consider Figure 8.2 (b) where the original label of the image is Frog. Although hybrid model failed to correctly classify this image but it predicted Deer with only 55.66 probability score and had Frog at second place with 44.24 probability score. In contrast, deterministic model also misclassified this image predicting it was a Deer with 85.56 probability score and assigned only 14.35 score to second prediction. Additionally, in cases where both deterministic and Bayesian models failed to correctly classify an image and hybrid network succeeded (sub-figures (c), (f), and (g)), it predicted very cautiously with reasonable probability scores. The probability scores of hybrid model were at par with other two methods for relatively easy examples as shown in sub-figure (a).

Table 8.4: Comparison of Bayesian and proposed hybrid models on different data sets with uncertainty estimation

Data sets	**Bayesian Model** [219]			**Hybrid Model**		
	Overall Accuracy (%)	**Certain Accuracy (%)**	**Uncertain Samples (%)**	**Overall Accuracy (%)**	**Certain Accuracy (%)**	**Uncertain Samples (%)**
Image Data sets						
MNIST	99.01	99.17	20.5	**99.26**	**99.28**	9.6
CIFAR-10	65.41	72	66.9	**88.70**	**91.11**	46.2
Medical Image Data sets						
ORIGA	74.42	77.10	35.65	**80.31**	**77.21**	38.7
ISIC-Subset	58.15	65.48	34.3	**75.67**	**81.5**	53.8
Time Series Data sets						
Fish	80.7	92.4	9.1	**84.7**	**100.0**	6.8
ShapesAll	70.9	71.8	1.0	**72.3**	**72.9**	1.3
Plane	**96.7**	**98.9**	0.95	95.1	97.1	0.0
TwoPattern	81.0	84.4	25.0	**89.4**	**91.3**	24.9
ECG5000	**93.2**	93.8	36.2	91.9	**93.9**	36.8
MedicalImages	62.4	62.9	0.13	**64.7**	**66.5**	0.13
ElectricalDevices	54.0	55.8	14.6	**56.6**	**57.9**	14.8
Mallat	82.5	84.2	35.6	**89.3**	**92.1**	37.7
ECG Thorax1	89.1	90.9	14.9	**91.3**	**91.6**	14.8

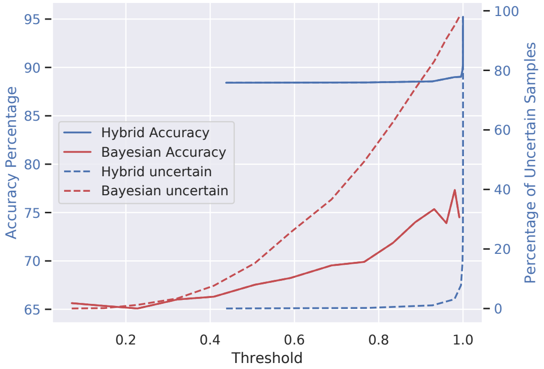
8.4.1 *Uncertainty Estimation*

Comparison of Bayesian and hybrid models

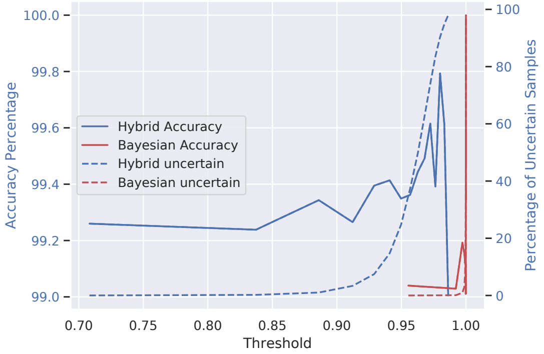
Since deterministic model does not have intrinsic ability to estimate uncertainty (although some works like [77, 97] have used deterministic models and applied some post-processing to get confidence estimates), in this section we focus on Bayesian and Hybrid models only and compare their performance. Since the classifier part of both Bayesian and Hybrid methods are trained using Bayesian estimates, both networks provide posterior distribution that is used to estimate uncertainty using Algorithm 1. Table 8.4 compares the accuracies of both training methods before and after using Algorithm 1. In this table, *Overall Accuracy* refers to the accuracy of the model before applying Algorithm 1, whereas *Certain Accuracy* refers to the accuracy on the predictions for which the network was certain according to Algorithm 1. When the algorithm is not sure about the prediction it tags the test sample as uncertain. We can observe that accuracies for both fully Bayesian and hybrid approaches improved after uncertainty estimation algorithm was applied. The accuracy of our hybrid approach is higher than fully Bayesian model especially when it was fairly certain about the predictions.

Figure 8.3 depicts the trade-off between number of uncertain samples and classification accuracy for both Bayesian and Hybrid models. We can see from this figure that the accuracy of the networks increases with the increase in percentage of uncertain samples. It can be argued from these curves that since, *difficult* samples have been passed over by the classifier and prediction is given for *easy* samples only, that is why we see a positive trend in accuracy with growing number of uncertain samples. However, in many crucial application areas, it is better to abstain from giving any half-cooked prediction than making a potentially costly mistake. In medical image analysis, for instance, such non-compulsive classifiers can reduce the workload of human experts by screening relatively easy disease patterns and allowing the physicians to focus their time and energy only on the most challenging of the cases.

Trade-off between number of uncertain samples and classification accuracy



(a) CIFAR-10



(b) MNIST

Figure 8.3: Trade-off between number of uncertain samples and the accuracy on remaining predictions. The threshold on x-axis is calculated using *percentile* parameter as shown in Algorithm 1.

Part IV

INTERPRETABILITY AND EXPLAINABILITY

9

INTERPRETABILITY AND EXPLAINABILITY OF DNN

DNNs have become ubiquitous these days. They have been successfully applied in a wide range of sectors including automotive [137], government [238], wearable [191], dairy [156], home appliances [226], security and surveillance [151], health [55], and many more, mainly for regression, classification, and anomaly detection problems [34, 88, 184, 185, 268]. Neural network's capability of automatically discovering features to solve any task at hand makes them particularly easy to adapt to new problems and scenarios. However, this capability to automatically extract features comes at the cost of a lack of transparency/intelligibility of their decisions. The applicability of DNNs has also been compromised due to their deficiency of explaining a decision [175]. This is specifically true for domains like business, finance, natural disaster management, health-care, self-driving cars, industry 4.0, and counter-terrorism where reasons for reaching a particular decision are equally important as the prediction itself [145].

DNN's deficiency of explaining its decision

In the domains where human lives are directly or indirectly linked to a machine's decision or the high-stakes decisions are based on them, the trustworthiness of the decision-making system is more important than accuracy. This trustworthiness can be achieved by enabling a system to answer *HOW* and *WHY* of a decision. The *HOW* part is answered when a system is capable of showing how it has taken a particular decision. In this process, the system must highlight the major observables to show how they are behaving and changing. The *WHY* part is answered when a system provides an explanation of a decision. It is important to provide reasons for a particular decision taken by a system. The attached facts to an explanation makes an explanation more transparent which eventually makes the whole system trustworthy.

Trustworthiness of decision-making system

In this chapter, we enable a system to answer *HOW* part with the help of visualizations. There have been significant attempts to uncover the black-box nature of deep learning-based models [150, 224, 234, 259, 261, 262], where visualization of the model has been the most common strategy. Almost all of the proposed visualization systems are image-centric where visualizing the images is directly interpretable for humans (natural association to similar looking objects like eyes, faces, dogs, cars etc.). These visualizations help humans understand the thinking process of an ANN. Most of these visualization and interpretability ideas are equally applicable to time-series,

The HOW part

This chapter is an adapted version of the work published in [221] and [186].

but the unintuitive nature of the time-series data makes it difficult to directly transfer these ideas to aid human understanding. To demystify a deep model for time-series analysis, a portion of Time-series Visualization framework (TSViz) is introduced in this chapter.

The importance of explanation (WHY part)

Though a picture is worth thousand words, still it provides an overview, not a detailed explanation. To understand the details, it is necessary to have a logical description of the picture. It has been well established in the prior literature that an explanation of the decision made by a DNN is essential to fully exploit the potential of these networks [9, 205], and explanations help in making a system trustworthy [59, 240, 245]. With the rise in demand for these deep models, there is an increasing need to have the ability to explain their decisions. For instance, big industrial machines cannot be powered down just because a DNN predicted a high anomaly score, until and unless the system is trustworthy. It is important to understand the reason for reaching a particular decision, i.e. why the DNN computed such an anomaly score. Adequate reasoning of the decision increases a user's confidence in the system.

The WHY part

To address this *WHY* part, Time-series Explanation framework (TSXplain) is introduced in this chapter. This framework is inspired by the human psychology of logical reasoning for a particular decision. It contributes to the WHY part by generating natural language explanations of the decisions made by a DNN. Powerful statistical features are aligned with the most influential data points for deep networks (discovered by TSViz) to generate textual explanations. Main contributions of this chapter are as follow:

- An influence tracing algorithm to compute the input saliency map in introduced as part of TSViz. It enables an understanding of the regions of the input that were responsible for a particular prediction. To the best of our knowledge, it is the first framework for time-series interpretability that helps in understanding the decision making process of a DNN.
- To fulfill the need of the hour, an explanation framework, TSXplain is proposed that provides natural language explanations of a DNN decision. To the best of our knowledge, TSXplain is a pioneer in providing explanations for time-series data.
- The mechanism of providing two levels of explanations provides ample description of the decision made by the network to aid an expert as well as a novice user alike.
- Our survey and reliability assessment test confirm that the generated explanations are meaningful and correct. We believe that generating natural language based descriptions of a network's decisions is a big step towards opening up the ANN black-box.

9.1 LITERATURE REVIEW

Over the past few years, there have been numerous advancements in the field of network interpretation and visualization. For understanding a DNN on an image classification task, Zeiler and Fergus [261] proposed a technique for the visualization of deep convolutional neural networks. Their visualization reveals features in a fully trained network. Highlighting the input stimuli that excites individual feature maps at different layers, helps in understanding what the network has learned. For each feature map, they also visualize the corresponding image patches to enhance the visual understanding for humans. Yosinski et al. [259] introduced *DeepVizToolbox* that helped in understanding how neural network models work and which computations they perform at the intermediate layers of the network at two different levels. The first level visualizes the activations produced on each layer while processing an image on a trained convolutional neural network. Based on regularized optimization in image space, the second level visualizes features that the different filters are responding to at each layer of the network. This toolbox visualizes the top images for each unit, forward activation values, deconvolutional highlighting, and preferred stimuli. Simonyan, Vedaldi, and Zisserman [224] also presented an approach to visualize the convolutional neural networks developed for image classification. Their visualization provides image-specific class saliency maps for the top predicted class, that are extracted using a single backward pass through the network.

Visualization studies

Bau et al. [18] introduced a framework to interpret the deep visual representations and quantify the interpretability of the learned CNN model. First, they gather a broad set of human-labeled visual concepts and then gather the response of hidden variables to known the concepts. To understand a neural network, Mahendran and Vedaldi [171] highlight the encoding learned by the network through inversion of the image representations. They also study the locality of the information stored in the representations. Melis and Jaakkola [175] designed self-explaining models where the explanations are intrinsic to the model for the robust interpretability of a network. They also argued regarding the necessity of explicitness, faithfulness, and stability for interpretability. Bach et al. [14] introduced an approach to achieve pixel-level decomposition of an image classification decision. They generate heatmap for a well-classified image that segments the pixels belonging to the predicted class. Their visualization helps to highlight the contribution of single pixels to the prediction of kernel-based classifiers as well as DNNs.

Interpretability studies

Theoretical contributions have also been made in order to understand the amazing generalization capabilities of these deep models. Zhang et al. [262] presented an empirical analysis to divert attention to the philosophical topic of what is actually perceived as generalization. Koh and Liang [148] presented influence functions as a methodology to trace back model predictions in terms of its training data.

Visualizations for time-series

Despite these advancements, the area of network visualizations for time-series analysis has remained unexplored until now. A recent attempt has been made by Kumar, Taylor, and Wong [150] to visualize the input points that were most influential for a particular prediction through gradients (saliency).

Explanation studies

Current DNN visualizations and interpretations only help an expert to understand and improve the overall process. However, they still lack the actual reasoning why a particular decision has been taken by the learned model. There has been some work in the domain of image captioning where visual attributes are leveraged to support the DNN decision. Guo et al. [98] proposed a textual summarization technique of image classification models. They train a model with the image attributes that are used to support the classification decision. A filter-level attribute probability density function is learned as a posterior probability with the given images. In the same domain, Hendricks et al. [109] proposed a model that predicts a class label and explains the reason for the classification based on the discriminating properties of the visual objects. Kim et al. [141] introduced a textual explanation system for self-driving vehicles. They generate introspective explanations to represent the causal relationships between the system's input and its behavior which is also the target of our study. On the basis of the vehicle's sensor measurements and the attention-based video-to-text model, the textual descriptions and explanations are generated. They also train another neural network, (LSTM), based on the human-annotated explanations, to generate the final explanations. In most of the aforementioned techniques, a neural network black-box is further used to generate descriptions and explanations of another neural network. Despite being a promising direction, this introduces another level of opaqueness into the system.

In the domain of relation extraction, Hancock et al. [101] proposed a supervised rule-based method to train classifiers with natural language explanations. In their framework, an annotator provides a natural language explanation for each labeling decision. Furthermore, the provided explanations are parsed into labeling functions that are logical forms of explanations. These logical forms convert unlabeled data into a large labeled data set that is used to train the classifier. Similar work has been presented by Srivastava, Labutov, and Mitchell [228], but they jointly train a task-specific semantic parser and classifier instead of a rule-based parser. These systems, however, rely on a labeled set of training examples that are not available in most of the real-world applications.

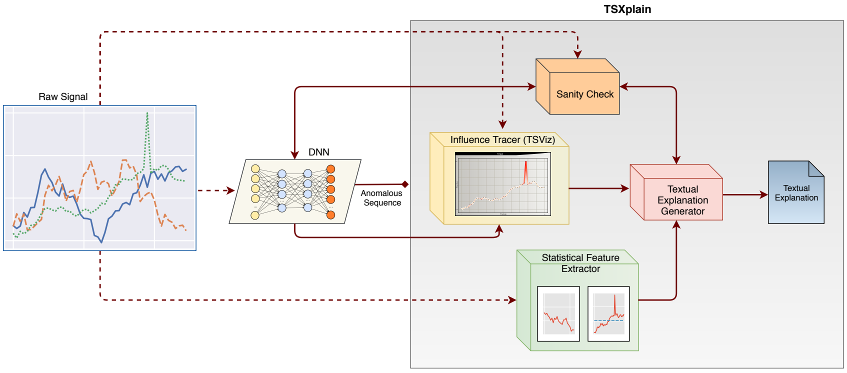


Figure 9.1: TSXplain system diagram. Based on a time-series classifier, the time-series are passed on to the *influence tracer* module which highlights the most influential data points. Furthermore, these influential data points are used along with the point-wise and sequence-wise statistical features to generate textual explanations of the time-series.

9.2 METHODOLOGY

Different visualization and interpretation techniques developed specifically to understand deep models aid an expert in understanding the learning and decision-making processes of the network. However, the provided interpretation/visualization cannot be readily understood by a novice user. It is up to the user to draw conclusions about the network's decision with the help of the available information. Many of the existing techniques use a separate deep network that is trained for the generation of explanations using the primary model [98, 109, 141]. These explanations still suffer from a lack of transparency, as they are also generated by a deep model. Therefore, we approach the problem of generating explanations in a way that significantly improves the intelligibility of the overall process. We leverage the statistical time-series features to provide a concrete natural language-based explanation of a sequence. These features also help in gaining a user's trust because of their lucid nature. The proposed system is composed of different modules as highlighted in Figure 9.1. The raw input is first passed on to DNN for classification. If the sequence is classified anomalous, the whole TSXplain system is activated, which is composed of four modules: namely *influence tracer TSViz*, *statistical feature extractor, sanity check*, and *textual explanation generator*.

A user has to draw conclusions himself from the available visualizations

The *influence tracer* is employed to discover the most salient regions of the input. The *statistical feature extractor* module extracts different statistical features from the sequence. The results from previous two modules are passed onto the *textual explanation generator* module in order to come up with a natural language description of the encountered anomaly. Furthermore, we introduce a *sanity check* module to get a coarse estimate of the system's confidence regarding the gener-

Major components

ated explanation. All of these modules are explained in detail in this section.

9.2.1 *Influence Tracer (TSViz)*

Targets the HOW part

The influence tracer module is based on the TSViz framework and it targets the *HOW* part. The proposed influence tracing algorithm can be used to trace the influence at several different levels. However, we only consider the main influence for our method i.e. the influence of the input on the output. This information provides important insights regarding the data points in the input that the network is actually responding to for computation of its output. The information regarding the parts of the input that were responsible for a particular prediction is considered a viable explanation in many scenarios including domains like self-driving cars [141], finance [150] and medical imaging [269]. It is important to note that we also compute the input's influence for every filter along with the final output.

This value can be obtained by computing the gradient of the current layer l w.r.t. the input layer. We use the absolute value of the gradient as the magnitude is of relevance, irrespective of direction. We denote the input as a^0, therefore, this influence of the input can be computed using Equation 9.3.

$$\delta_j^l = \frac{\partial a^l}{\partial z_j^l} \frac{\partial z_j^l}{\partial a_j^0} \tag{9.1}$$

$$\delta_j^0 = \frac{\partial a^l}{\partial a_j^0} \tag{9.2}$$

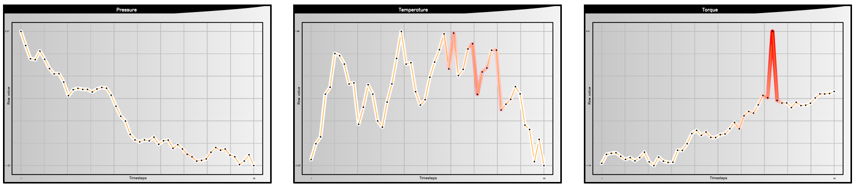
$$I_{input}^d = \left|\delta^0\right| \tag{9.3}$$

In order to be able to visualize and compare the saliency of the different filters, the absolute values of the influences are scaled using the min-max scaling presented in Equation 9.4.

$$I_{\text{input}} = \frac{I_{\text{input}} - \min\limits_j I_{\text{input}}^j}{\max\limits_j I_{\text{input}}^j - \min\limits_j I_{\text{input}}^j} \tag{9.4}$$

Visualization

The computed influence values are visualized on top of the original signals to provide a hint regarding the encapsulated information. This visualization is presented in Figure 9.2, where color-coded data points in each input channel are shown with respect to their influence on the final decision of the DNN. The dark shade represents the high influence of a data point. It can be observed in Figure 9.2 that the network decision is mostly based on the small and big spikes in the observed time-series.



(a) Pressure signal (b) Temperature signal (c) Torque signal

Figure 9.2: Visualization of the saliency information along with the raw signal values. Influential data points are highlighted in the shades of red, dark being more influential.

Siddiqui et al. [221] discussed the problem of extremely confident predictions for the influence tracing algorithm and suggested a remedy to overcome this issue by imposing regularization on top of the activations itself when training the network. The new objective can be written as:

$$\mathcal{W}^* = \arg\min_{\mathcal{W}} \frac{1}{|\mathcal{X}|} \sum_{(\mathbf{x},y)\in\mathcal{X}\times\mathcal{Y}} \mathcal{L}(\Phi(\mathbf{x};\mathcal{W}),y) + \lambda\|\mathcal{W}\|_2^2 + \beta\|z^L\|_2^2 \tag{9.5}$$

where Φ defines the mapping from the input to the output space, $\mathcal{W}$ encapsulates all the parameters of the network, z^L denotes the activation values of the last layer before application of the sigmoid layer, and λ and β denotes hyperparameters controlling the contribution of the regularization terms and the empirical risk. We use the same modified objective to train our network in order to avoid extremely confident predictions.

The *influence tracer* module consumes both the DNN model as well as the raw input. Then, it performs the backward pass through the network from output to input in order to obtain these influence values. The output from this module is consumed by the *textual explanation generator* module (Section 9.2.3).

9.2.2 *Statistical Feature Extractor*

Sequence-wise features

This module extracts different statistical features from the input sequence. Since we are dealing with sequential data comprising of time-series, different point-wise as well as sequence-wise features are calculated. These features include, but not limited to, lumpiness, level shift, Kullback–Leibler (KL) score, number of peaks, and ratio beyond r-sigma (explained below). These features have been previously used by Bandara, Bergmeir, and Smyl [17] and have been proposed in [121, 248].

1. *Lumpiness:* Initially, daily seasonality from the sequence is removed by dividing it into blocks of n observations. Variance of

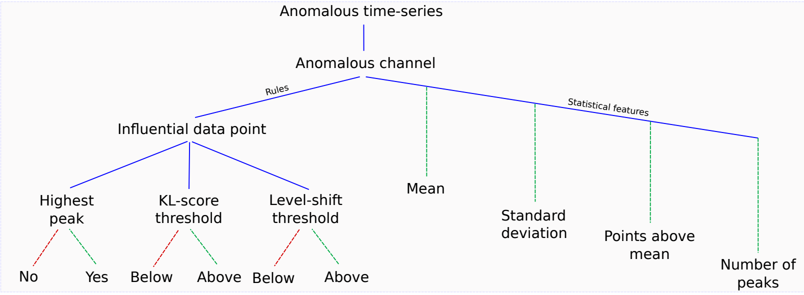


Figure 9.3: An example of rules and statistical features used to generate textual explanations for a given anomalous time-series.

the variances across all blocks is computed, which represent the lumpiness of the sequence.

2. *Level shift:* The sequence is divided into n observations and the maximum difference in mean between consecutive blocks is considered as the level shift. It highlights the block that is different from the rest of the sequence.

3. *KL score:* To calculate this score, the sequence is divided into consecutive blocks of n observations. This score represents the maximum difference in Kullback-Leibler divergence among consecutive blocks. A high score represents high divergence.

4. *Number of peaks:* This feature identifies the number of peaks in a sequence. The sequence is smoothed by a Ricker wavelet for widths ranging from 1 to n. It detect peaks with sufficiently high signal-to-noise ratio.

5. *Ratio beyond r-sigma:* It gives the ratio of data points that are r standard deviations away from the mean of a sequence.

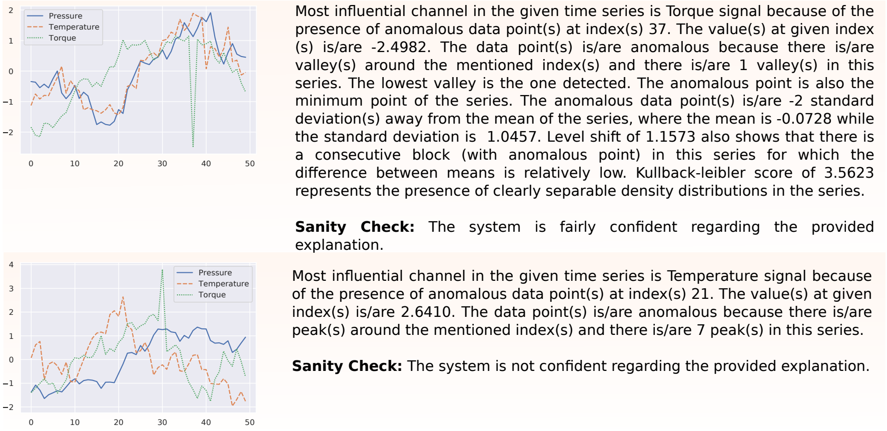
6. *Standard deviation:* This feature represents the standard deviation of a sequence.

Point-wise features

In addition to the above mentioned sequence-wise features, we also use point-wise features including peak, valley, maximum point, minimum point, highest spike, and lowest valley, etc. The fusion of sequence-wise and point-wise features provides vivid characteristics of the highly influential data points.

9.2.3 *Textual Explanation Generator*

The influential points determined by the *influence tracer* module along with the features computed by the *statistical feature extractor* module are passed onto this module for the generation of the textual explanations of a given anomalous sequence. This module also receives input



Most influential channel in the given time series is Torque signal because of the presence of anomalous data point(s) at index(s) 37. The value(s) at given index (s) is/are -2.4982. The data point(s) is/are anomalous because there is/are valley(s) around the mentioned index(s) and there is/are 1 valley(s) in this series. The lowest valley is the one detected. The anomalous point is also the minimum point of the series. The anomalous data point(s) is/are -2 standard deviation(s) away from the mean of the series, where the mean is -0.0728 while the standard deviation is 1.0457. Level shift of 1.1573 also shows that there is a consecutive block (with anomalous point) in this series for which the difference between means is relatively low. Kullback-leibler score of 3.5623 represents the presence of clearly separable density distributions in the series.

Sanity Check: The system is fairly confident regarding the provided explanation.

Most influential channel in the given time series is Temperature signal because of the presence of anomalous data point(s) at index(s) 21. The value(s) at given index(s) is/are 2.6410. The data point(s) is/are anomalous because there is/are peak(s) around the mentioned index(s) and there is/are 7 peak(s) in this series.

Sanity Check: The system is not confident regarding the provided explanation.

Figure 9.4: Channel-wise natural language explanations generated for the expert users. The second time-series in this figure represents a failure-case where the explanations are not accurate, which is also detected in the sanity check.

from the *sanity check* module (explained in Section 9.2.4) that allows the system to specify its confidence over the generated explanation.

A set of rules

We designed a set of rules to incorporate a range of features. These rules are defined based on the statistical time-series features in a way that explains different characteristics of a given sequence. Based on the classification decision given by the network, along with the time-series features, this module provides explanations of the data points that influence the network decision. The explanations are generated channel-wise so that the anomalous data points in each channel can be highlighted. An example of defining rules and an overall hierarchy of this process is shown in Figure 9.3. The influential data points are provided by the *influence tracer* module and the rules defined under the 'Influential data point' parent node in Figure 9.3 are applied on those data points. The statistical features (mean, standard deviation, points above mean, and number of peaks calculated on whole sequence mentioned on right branch of 'Anomalous channel' parent node in Figure 9.3) provide supporting arguments to the defined rules. The values of child nodes (highest peak, KL-score, level-shift, mean, standard deviation, points above mean, and number of peaks) in green are incorporated in the textual explanations. For multi-channel input, the salient data points from an individual channel are passed onto this module.

Two levels of abstraction

We have defined two levels of abstraction for the textual explanations. The first level of explanation is defined for the users who are not interested in detailed explanations or don't have enough knowledge of time-series data (novice user), but would like to get information regarding the most salient regions and channels of the input. The second level of explanation, on the other hand, is defined for the expert users. Such users are generally interested in knowing the details, such as, why a network took a particular decision? Sample explanations for the expert users are shown in Figure 9.4. The explanations shown in this figure clearly point out the anomalous channel in a given anomalous time-series sequence. Moreover, characteristics of anomalous data points and anomalous channel are also explained in the form of feature values. Figure 9.5a shows an example of a simple explanation that is generated for the novice users.

9.2.4 *Sanity Check*

Reliability of the generated explanations

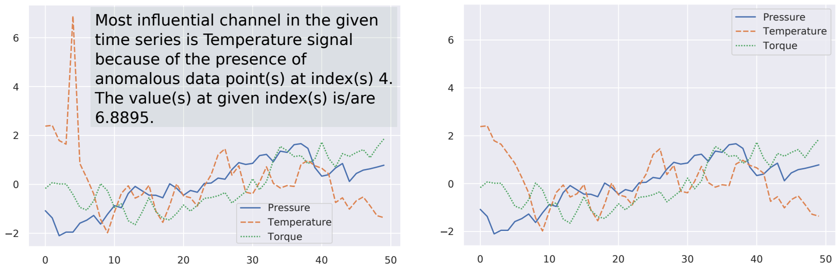
In order to assess the reliability of the explanation generated by the system, a simple sanity check is performed. The output of the *textual explanation generator* module along with the original input are passed onto this module. The data points corresponding to the explanation are suppressed and this masked sequence is fed again to the network for inference. The masking is performed by linear interpolation between the last and the first retained points.

Helps in computing confidence over the provided explanations

If the removed data points were indeed causal for the network prediction, we expect the prediction to flip. We use this sanity check to compute confidence over the provided explanation. If we observe a flip in the prediction, we assign high confidence to the provided explanation. On the other hand, if the prediction is retained, we assign low confidence to the provided explanation. This check confirms that the generated explanations are correctly referring to the data points that are actually contributing towards the classification decision taken by the network. Finally, the sanity check output is passed back to the *textual explanation generator* module with the confidence information which is mentioned in Figure 9.4. In Figure 9.4, the second example didn't observe any flip in the prediction after the suppression of the deemed causal point (a failure case), resulting in low system confidence for the provided explanation. Figure 9.5 visualizes the process of sanity check on an anomalous sequence. In Figure 9.5a, the textual explanation highlights an anomalous data point in the temperature signal (in orange color). The network prediction is flipped after suppressing the mentioned anomalous data point in the temperature signal as shown in Figure 9.5b.

9.3 EXPERIMENTAL SETUP AND DATA SET

To classify a time-series as normal or anomalous, we trained a CNN model with three convolutional layers comprising of 16, 32 and 64 filters respectively, with Leaky ReLU as the activation function, followed by a single dense layer. Since the focus of this chapter is on a generation of textual explanations, we selected the hyperparameters (e.g. number of layers, number of filters) based on our experience and did not invest any significant effort into hyperparameter optimization



Most influential channel in the given time series is Temperature signal because of the presence of anomalous data point(s) at index(s) 4. The value(s) at given index(s) is/are 6.8895.

(a) Anomalous sequence.

(b) Effect of sanity check.

Figure 9.5: As a results of sanity check, the anomalous peak is suppressed and sequence in (a) is classified as normal in (b).

or model selection. It is important to mention that we have chosen a convolutional neural network as our DNN, because CNNs are generally easier to optimize, achieved state-of-the-art results in anomaly detection [185, 268], and the base module of TSXplain (*influence tracer* module) is also currently based on CNNs. To generate natural language explanations on time-series data, we used the data sets mentioned in this section.

9.3.1 *Machine Anomaly Detection*

It is a synthetic time-series classification data set[1] curated by Siddiqui et al. [221]. This data set comprises of $60,000$ time-series with 50 timestamps each. Each sequence consists of three channels which represent values from pressure, torque, and temperature sensors. Random point anomalies were introduced in the data set and the sequences containing such point anomalies are marked as anomalous. Point anomalies only exist in the torque and temperature signals, while the pressure signal is kept intact. The data set is split into $45,000$ training sequences with $7,505$ anomalous sequences, $5,000$ validation sequences with 853 anomalous sequences, and $10,000$ test sequences with $1,696$ anomalous sequences.

Contains random point anomalies

9.3.2 *Mammography Data set*

This breast cancer screening data set contains $11,183$ time-series and it is commonly used for classification purposes. Anomalies at certain points/features make the whole time-series anomalous. The data set is split into $8,000$ training time-series with 186 anomalous time-series, $1,000$ validation time-series with 25 anomalous time-series, and $2,183$ test time-series with 49 anomalous time series. This data set is available at OpenML[2] [241].

Commonly used for time-series classification

---

[1] Machine Anomaly Detection data set: `https://bit.ly/2UNk0Lo`

[2] Mammography Data set: `https://www.openml.org/d/40907`

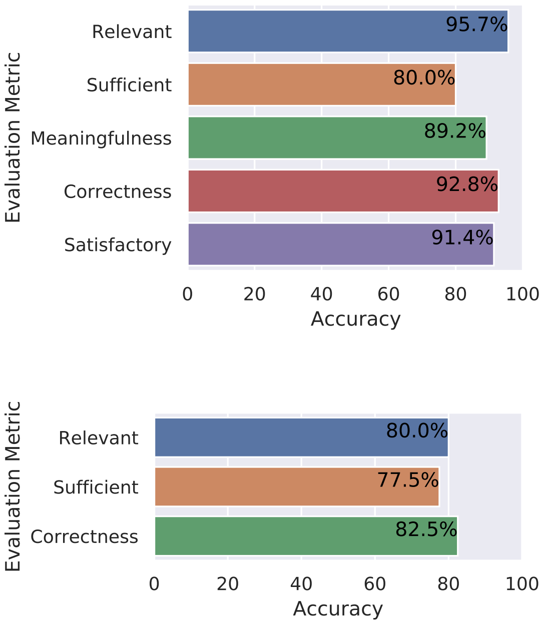


Figure 9.6: Summary of evaluation done by expert (left) and novice (right) users on Machine Anomaly Detection data set [221] explanations.

9.3.3 *Shuttle Data set*

Describes radiator positions in a NASA shuttle

This data set describes radiator positions in a NASA shuttle with 9 attributes. There are 46,464 time-series in this data set. We split the data set into 25,000 training time-series with 483 anomalous time-series, 5,000 validation time-series with 102 anomalous time-series, and 16,464 test time-series with 293 anomalous time series. This data set is available at OpenML[3] [241].

9.4 EVALUATION AND DISCUSSION

A survey to evaluate the explanations

In order to completely assess the relevance and correctness of a given explanation, we conducted a survey in which novice and expert users were asked to evaluate the generated explanations. We provided 20 time-series from the *Machine Anomaly Detection* data set along with the generated explanations to the participants and asked questions related to whether the generated explanation was, i) relevant, ii) sufficient, iii) meaningful, iv) correct, and v) satisfactory from the experts? Whereas, the novice users were only questioned about i), ii), and iv). There were 7 expert and 6 novice participants, who provided their binary (agree/disagree) feedback to the aforementioned question category. The results of this survey are summarized in Figure 9.6. The

---

[3] Shuttle Data set with 7% anomalies: `https://www.openml.org/d/40901`

Table 9.1: Effect of masking the data points in *Machine Anomaly Detection* data set which are relevant for the explanation.

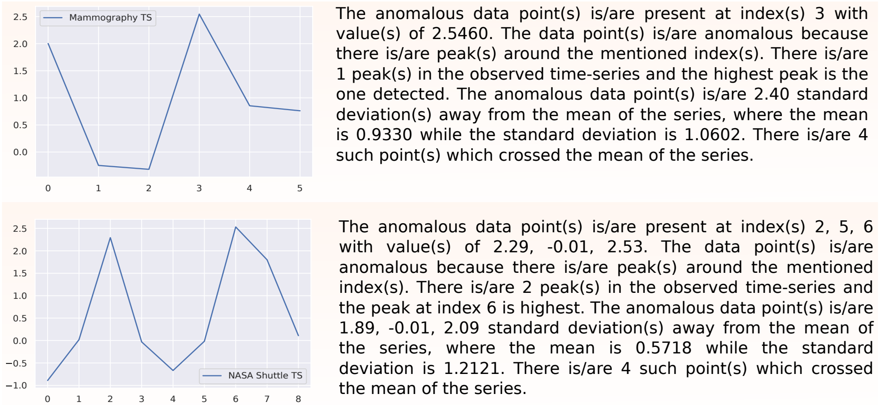
Window size	Anomalous sequence	Flipped prediction after masking	Percentage flipped
1	1511	1104	73.0%
3	1511	1319	87.3%

bar charts show the percentage of the participants who agreed to the asked questions related to a specific aforementioned metric. By analyzing the accumulated feedback of the participants shown in Figure 9.6, it is clear that most of the participants considered the provided explanations relevant, meaningful, and correct. The majority of the experts were satisfied with the reasoning of the ANN decision provided in the explanation. Although, 20% experts and 22.5% novice participants thought that the provided explanation is not sufficient.

Reliability of the generated explanations

In this study, we are trying to infer the causality through the provided explanations, so it is also important to assess the reliability of the generated explanations. Therefore, we introduced the sanity check module in the system pipeline to obtain a measure of confidence over the provided explanations (as explained in Section 9.2.4). We also computed this confidence estimate over the entire test set of *Machine Anomaly Detection* data set in order to get an impression regarding the overall reliability of the generated explanations. The cumulative results are presented in Table 9.1. Since it is important to compute these statistics only over examples where the classification from the network was correct, we were left with $1,511$ out of $1,696$ total anomalous sequences in the test set. In the first setup, a masked sequence is generated by suppressing the exact data points for which the explanations have been generated by the *textual explanation generator* module. Since we are suppressing the exact point, we represent this setup with a window size of one. In the second setup, a sequence is masked with a window size of three, covering one preceding and one following value in order to cover up any minor misalignment of the most salient region highlighted by the *textual explanation generator* module. In this case, a total of three data points were suppressed. We represent this setup with a window size of three. The results shown in Table 9.1 indicate that for 73.0% of the anomalous sequences, the predictions were flipped by masking out the exact data points highlighted by the explanation module. When we relaxed the sanity check criteria to a window size of three, the percentage of flipped sequences rose up to 87.3%. This high success rate makes it evident that in most of the cases, plausible explanations for the predictions made by the network could be provided. However, it is important to note that this experiment does not strongly imply causality.

In the traditional interpretation settings where only visual explanations are available, it is difficult for a user to understand why a par-



The anomalous data point(s) is/are present at index(s) 3 with value(s) of 2.5460. The data point(s) is/are anomalous because there is/are peak(s) around the mentioned index(s). There is/are 1 peak(s) in the observed time-series and the highest peak is the one detected. The anomalous data point(s) is/are 2.40 standard deviation(s) away from the mean of the series, where the mean is 0.9330 while the standard deviation is 1.0602. There is/are 4 such point(s) which crossed the mean of the series.

The anomalous data point(s) is/are present at index(s) 2, 5, 6 with value(s) of 2.29, -0.01, 2.53. The data point(s) is/are anomalous because there is/are peak(s) around the mentioned index(s). There is/are 2 peak(s) in the observed time-series and the peak at index 6 is highest. The anomalous data point(s) is/are 1.89, -0.01, 2.09 standard deviation(s) away from the mean of the series, where the mean is 0.5718 while the standard deviation is 1.2121. There is/are 4 such point(s) which crossed the mean of the series.

Figure 9.7: Sample explanations generated for Mammography (top) and NASA Shuttle (bottom) time-series.

Easy to understand the reason behind the decision by following the provided explanations

ticular decision is taken by the network just by looking at the plots. However, it is relatively easy to understand the classification reason by reading the explanations provided by our system for the corresponding plots in Figure 9.4 and Figure 9.7. We specifically opted for statistical features due to their strong theoretical foundations and transparency. The point-wise features of an anomalous data point help a user in understanding how that data point is different from the rest of the sequence. Whereas, sequence-wise features help in highlighting the overall behavior of an anomalous sequence. An end-user can confirm part of the explanation by looking at the plot, which elevates his trust in the system.

Part V

ASSOCIATED RESEARCH

10

INCORPORATING EXPERT KNOWLEDGE IN DNN

Recent advances in computational hardware have made it possible to achieve state-of-the-art performance in various domains, by utilizing DNNs, ranging from image classification [271], playing board games [223], natural language processing [53] to speech recognition [111]. As a result, there is heightened interest, both academically and industrially in DNNs, with deep learning being listed at the top of Gartner hype cycle for emerging technologies [51]. This increased interest coupled with advances in hardware has paved the way for the development of more sophisticated DNN algorithms, which may contain millions of parameters to train and optimize. Version of NASNet [271] model, for example, with highest accuracy on ImageNet data set contains around 88.9M parameters. Optimizing such a huge number of parameters is a challenge itself and requires equivalently bigger training data set that allows the model to extract enough features to train its parameters. As a result, these models perform exceptionally well in domains where ample data is available but in data scarce domains, these models suffer as they can easily overfit. This issue is even more significant in time-series domain, where scantiness of data is further compounded by the fact that time-series often do not have enough features for deep networks to work with.

Scantiness of data

In contrast to DNNs, humans tend to rely on their knowledge while solving problems. This knowledge is acquired not only from problem specific examples but also from other sources, like education and experiences [154]. However, the very notion of "knowledge" is tricky to explain, and equivalently difficult to collect and store in a form that is understandable or transferable to a computing program. Knowledge-based System (KS) aims to store such knowledge expressed in the form of logic rules or some other declarative language that can then be used to find solution to complex problems [239]. Similarly, there are statistical methods that are based on strong logical reasoning, like ARIMA, that do perform exceptionally well in their respective domains and are used by many experts to aid them in decision-making process.

'Knowledge', for solving problems

Complementing DNNs with expert knowledge or some form of extra knowledge has been actively researched upon [32, 117, 243]. Most of the work in the literature, although improves performance of the DNNs but adds extra dependency on the network on quality of expert information used [117, 236]. In this chapter, Deep Expert (DeepEX) is proposed that combines the knowledge driven and data driven

---

This chapter is an adapted version of the work published in [45].

Combining knowledge driven and data driven streams

streams for multi-step time-series forecasting problems. The focus of this work is to combine these seemingly parallel streams in a way that retains advantages of both, while suppressing their individual disadvantages. Specifically, we aim to reduce the dependency of DNNs on the data by leveraging information contained in the knowledge stream. Finding state-of-the-art KS or DNN model is not the focus here, but instead, the goal is to devise a knowledge incorporation scheme that bridges the gap between data and knowledge driven approaches and combines their strengths. We tested DeepEX on the CIF2016 and NN5 time-series forecasting benchmarks to signify its performance.

In particular, following are the contributions of this chapter:

- A novel approach to combine knowledge and data driven systems in an end-to-end learning framework is introduced.
- A new regularization on the activity of the network that helps the network in identifying strengths and weakness of both domains and decide optimal combination of both domains is also introduced.
- Introduction of a new network to capture the trend in order to decompose the problem into sub-problems which can be effectively solved.
- Scaling of the proposed method to multi-step ahead prediction which is significantly difficult for the current generation of expert knowledge incorporation techniques.

10.1 LITERATURE REVIEW

Knowledge-grounded conversation model

Integrating domain knowledge or any sort of extra information to boost DNN performance has been actively researched upon. Ghazvininejad et al. [83] presented a knowledge-grounded conversation model based on neural networks. In addition to utilizing data containing the conversation history, they also conditioned the output of their sequence-to-sequence (seq2seq) model on external details within the context of conversation. The resulting conversation model was able to produce more accurate responses that were labeled as more informative and appropriate by human judges. Such schemes encouraged knowledge incorporation to improve the performance of the system, however, it made the system more dependent on external contextual information data.

Knowledge-based Artificial Neural Networks (KBANN) was proposed by Towell and Shavlik [236]. They utilized propositional rules for knowledge representation which were structured in a hierarchical manner. The neural network was designed to have a one-to-one correspondence with the elements of the rule set. The rule set directly defined the number of neurons along with their corresponding weights.

Additional neurons were also introduced to learn features not specified in the rule set. Tran and Garcez [237] followed a similar approach where the network defined a logic rule set. Such techniques directly incorporate the information contained in the knowledge stream into the neural network. However, as a result of this direct incorporation, the network is confined to a structure that strictly complies to the hierarchical structure defined in the rule set. Additionally, this also abolishes the flexibility to be able to use different network architectures.

Leveraging knowledge for video descriptor

Venugopalan et al. [243] also proposed a neural network based video descriptor model that leveraged knowledge from both a neural language model as well as semantics obtained from a large text corpus in a LSTM-based architecture. The results demonstrated significant improvements in grammar while also improving the overall descriptive quality. They introduced two fusion techniques, namely Late fusion and Deep fusion where they concatenated the hidden states from both video to text network and language LSTM network, fusing the information contained in both of the domains. The system is strongly dependent on the quality of the expert which in turn is dependent of large amount of data.

Statistical forecasting models to aid neural network

Buda, Caglayan, and Assem [32] used statistical forecasting models to aid neural network in producing forecasting results for an anomaly detection problem. They utilized multiple statistical forecasting models in conjunction with deep learning model. Predictions made by all of these individual models were combined into one framework. The predicted values from all models were compared with the groundtruth and value giving the lowest Root Mean Square Error (RMSE) score was selected as the final prediction. They refer this approach as single-step merge. Another voting based approach was also proposed where RMSE score is used to select a single model for all of the predictions. The system treats the predictions from the individual models separately, hence, it is not able to leverage the advantages from both streams simultaneously. We have also used statistical methods to enhance performance of the neural network in anomaly detection problem (Chapter 6). We employed auto-ARIMA to forecast future values of the time-series. These predictions were then integrated into neural network by using a residual scheme. The resulting model, FuseAD, achieves better performance compared to individual networks when used separately.

First-order logic rules

Hu et al. [117] again leveraged expert knowledge in the form of first order logic rules. Iterative knowledge distillation technique is used to transfer knowledge to network parameters. The expert network acts as a teacher network to the student network. The student network tries to follow the teacher network by mimicking its predictions. Both the student and the teacher networks are updated at each iteration step. The goal is to find the best teacher network that fits the prediction of the rule set while also staying close to prediction made by the student network. Kullback–Leibler (KL)-divergence between the probability distribution of the predictions made by the teacher network and the output distribution of predictions made by the student network is used to minimize the difference between the two distributions. The proposed framework achieved state-of-the-art performance for the evaluated classification tasks. However, the framework strongly relied on the expert network as the student network is trying to emulate predictions made by the teacher network.

Expert knowledge incorporation

Expert knowledge is incorporated for key phrase extraction by [91] where they defined label-distribution rules that dictates the probability of a word being a key phrase. For example, the rule enunciates that a noun that appears in the document as well as in the title is 90% likely to be a key phrase and thus acts as posterior regularization providing weak supervision for the classification task. Similarly, KL-divergence between the distribution given by the rule set and the model estimates is used as the objective function to be used for the optimization. Again, as the model utilizes knowledge to strengthen the predictions of the network, it shifts the dependency of the network from the training data to accurate expert knowledge which might just be an educated guess in some cases. Similarly, [255] incorporated symbolic knowledge into the network by deriving a semantic loss function that acts as a bridge between the network outputs and the logical constraints. The semantic loss function is based on constraints in the form of propositional logic and the probabilities computed by the network. During training, the semantic loss is added to the normal loss of the network and thus acts as a regularization term. This ensures that symbolic knowledge plays a part in updating the parameters of the network.

Knowledge Enhanced Hybrid Neural Network

Wu et al. [254] proposed a Knowledge Enhanced Hybrid Neural Network (KEHNN). KEHNN utilizes knowledge in conjunction with the network to cater for text matching in long texts. Here, knowledge is considered to be the global context such as topics, tags, etc. obtained from other algorithms that extracts information from multiple sources and data sets. They employed the twitter Latent Dirichlet Allocation (LDA) model [265] as the prior knowledge which was considered useful in filtering out noise from long texts. A special gate, known as the knowledge gate is added to the traditional bidirectional Gated Recurrent Units (GRU) in the model which controls how much information from the expert knowledge flows into the network.

Residual learning scheme

Chattha et al. [44] proposed a residual learning scheme, called as Incorporating expert knowledge in neural networks (KINN), where they incorporated expert knowledge in the form of prediction in the network by adding it to the network's output. Although the approach is highly promising, it couldn't be scaled for multi-step predictions.

The first limitation is its inability to control the network's correction factor. The network makes useless corrections even in cases where it is not necessary. The second limitation is its inability to cope up with trend present in the sequence. This proves to be an impediment in the production of convincing results for complex time-series data. DeepEX addresses both these limitations.

10.2 METHODOLOGY

Components

DeepEX consists of different components, each targeted to cater for a specific task as visualized in Figure 10.1. The first component is the expert module which contains information about expert opinion. The second component is the Seasonal and Trend decomposition using Loess (STL) decomposition module which decomposes the input signal into its constituent parts. Finally, there are two CNN models, where one model is dedicated to handle the trend part while the other one handles the remaining signal.

The data and the expert model output is log normalized before feeding it to the STL decomposition layer. Log normalization has two major advantages: (i) re-scaling values and (ii) transformation of the multiplicative relation between the STL components to an additive

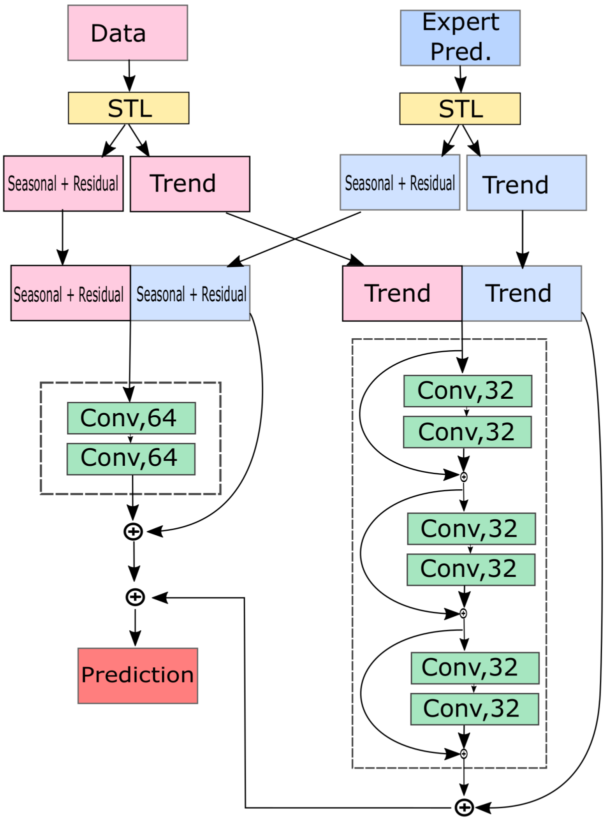


Figure 10.1: DeepEX pipeline.

one, which makes it easier to decouple the decomposed components. Each component further explained in detail in the following subsections.

10.2.1 *STL Decomposition*

Decomposition of a time-series

STL is a well-known time-series decomposition method which splits a time-series into three components, namely i) trend, ii) seasonality, and the iii) residual. In DeepEX, input data and output of the expert model are fed into the STL decomposition module. The trend from each of the signals is extracted from the rest of the signal. Residual and seasonal components are added together since only the trend is of relevance to us. Hence, output of STL decomposition contains two signals, trend and the rest of the signal that is a de-trended version of the input comprising of the seasonal and the residual components. These signals are then given to their respective CNN estimators as inputs. Although it is a common notion that neural networks are capable of modeling complex structures in data owing to their strong self adapting generalizing capabilities, more recent studies argue decomposing input signal or filtering out some component prior to modeling can produce better forecasting results [17, 189, 230]. Hence, we opt for a similar approach and decompose the original signal into two relatively less complex components.

10.2.2 *Expert Model*

4Theta

Knowledge driven techniques offer their own advantages. Knowledge, however, can take many shapes and forms. It can be in the form of a human expert, logic rules, or even some statistical method. One of the strengths of DeepEX is that it does not limit itself to any specific knowledge model as it is not dependent on architecture of the expert model but rather its predictions. Therefore, any knowledge model capable of producing predictions can be used. The expert model can be human feedback integrated into the system or a KS. For this particular study, we used the $4\mathrm{Theta}$ method[1] as our expert network. $4\mathrm{Theta}$ is based on theta model that decomposes the original signal into theta lines, where theta lines are derived by modifying the local curvature of the time-series through the coefficient θ. This θ is then applied to the second differences of the data. $4\mathrm{Theta}$ is an improvement to the Theta model, enabling it to handle complex time-series more efficiently which is evident from its performance on M4 benchmark data set [172].

---

[1] https://github.com/Mcompetitions/M4-methods/blob/master/4Theta%20method.R

10.2.3 *CNN Models*

Two different models

DeepEX has two different CNN models aimed to estimate trend and de-trended signal which are obtained after STL decomposition. Although the focus of this work is to develop a knowledge incorporation technique and not the individual DNN or knowledge models, but considerable effort has been invested in estimating hyperparameters of DNN. It was particularly observed that simple CNN model struggled the most in modeling the trend component of the signal, hence, model responsible for the estimation of trend is relatively complex compared to the seasonal and residual signal estimator. The trend estimation network comprises of three residual blocks, each containing two convolutional layers with 32 filters each. The other CNN model comprises of two convolution layers with each layer having 64 filters. It is important to mention that although we have chosen convolutional neural network as our DNN (because CNNs are generally easier to optimize), DeepEX is flexible enough to work with any other DNN architecture.

10.2.4 *Knowledge Incorporation Scheme*

Conditioning the DNN

Expert predictions are incorporated in the form of a residual scheme, where expert predictions are added to the output of the DNN model and are also used for conditioning the DNN. This conditioning is achieved by sequentially stacking the expert predictions ($x_t^{tr'}$) to the input from the data. The proposed residual scheme changes the underlying mapping learned by the network and instead of learning the complete input to output projection, the network only learns the modification factor needed in the input to give the desired output. In a way it can be said that the DeepEX framework estimates efficacy of expert model and makes corrections to it by using information from the data. As we have used a statistical method (Section 10.2.2) as our expert model, some portion of the data (25% of the training set) is used to estimate parameters of $4\mathrm{Theta}$ model. The resulting expert model is then used for making predictions on remaining portion of the data set, by employing a rolling window approach where forecast for the next horizon is obtained by using all of the previous data. DeepEX is trained on 75% of the training data, which is the only portion of the data set where expert predictions are available, since we do not have any expert predictions on 25% of the data on which $4\mathrm{Theta}$ is trained. It should be noted that the test set was never used in either estimating parameters of the $4\mathrm{Theta}$ model or in training DeepEX. Validation set was obtained by $\max(0.2 * |X|, H)$ where $|X|$ denotes the cardinality of the training set, while H denotes the horizon.

The input signal and expert predictions both are decomposed using STL (Section 10.2.1). Trend from both, the expert predictions and the data is fed to CNN, that is responsible for trend estimation. This optimization problem for the trend estimating CNN ($\Phi : \mathbb{R}^h \mapsto \mathbb{R}^h$) can be represented as:

$$[\hat{x}_t^{tr}, \hat{x}_{t+1}^{tr}, .., \hat{x}_{t+h-1}^{tr}] = \Phi([x_{t-1}^{tr}, x_{t-2}^{tr}, ..., x_{t-w}^{tr}; x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}]; \mathcal{W}) + [x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}] \quad (10.1)$$

Output of the system is the sum of the output of the two CNNs

where $x_t^{tr'}$ represents the decomposed trend values predicted by the expert model, x_t^{tr} represents the decomposed trend values of the original input and $\hat{x}_t^{tr}$ output trend values by the network. As the network is just producing an offset, i.e. the required change in the input signal to produce the desired output, instead of finding the complete input to output mapping, it makes the optimization problem significantly easier to tackle. Expert predictions are incorporated in a similar fashion as in case of other CNN model. Finally, the overall output of the system is the sum of the output of the two CNNs, which can be represented as:

$$[\hat{x}_t, \hat{x}_{t+1}, .., \hat{x}_{t+h-1}] = [\hat{x}_t^{tr}, \hat{x}_{t+1}^{tr}, .., \hat{x}_{t+h-1}^{tr}] + [\hat{x}_t^{sr}, \hat{x}_{t+1}^{rr}, .., \hat{x}_{t+h-1}^{rr}] \quad (10.2)$$

where $\hat{x}_t^{sr}$ represents the output from the trend network, $\hat{x}_t^{rr}$ represents the output of the seasonality and residual network, and $\hat{x}_t$ represents the output of the overall system. Both of the CNN models are optimized separately. A regularization term β is also added on top of the network activations in order to hinder the network from making unnecessary modifications. Therefore, the optimization problem for the CNN can be represented as:

$$\begin{aligned}\mathcal{W}^* = \arg\min_{\mathcal{W}} \|[x_t^{tr}, x_{t+1}^{tr}, .., x_{t+h-1}^{tr}] - \\ \Big(\Phi([x_{t-1}^{tr}, x_{t-2}^{tr}, ..., x_{t-w}^{tr}; x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}]; \mathcal{W}) + [x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}]\Big)\|_2 \\ + \beta\|\Phi([x_{t-1}^{tr}, x_{t-2}^{tr}, ..., x_{t-w}^{tr}; x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}]; \mathcal{W})\|_2 \quad (10.3)\end{aligned}$$

where β is a hyperparameter which controls the activity of the network. This formulation is used for both of the CNN models and the value of β is obtained via validation.

10.3 DATA SETS

Two famous forecasting benchmarks, CIF2016 and NN5 are used in this study.

10.3.1 CIF2016 *Data set*

The Computational Intelligence in Forecasting (CIF) data set was part of the International Time-series Forecasting Competition[2] held in 2016. This data set contains 72 monthly time-series, where 24 time-series are originated from the banking domain and rest are synthetic time-series. Two forecasting horizon, 6 and 12 were originally used in the competition. 57 time-series have forecasting horizon of 12, whereas the remaining 15 have horizon of 6.

Time-series from banking domain and synthetically generated data

10.3.2 NN5 *Data set*

NN5 is a neural forecasting competition[3] data set which was held in 2008. This data set contains time-series from roughly two years of daily cash money withdrawal amounts from one of various Automated Teller Machine (ATM) or cash machines [230]. There are total 111 daily time-series in this data set. The forecasting horizon is 56 for all the time-series and there are 735 data points in each time-series. The data may contain a number of time-series patterns including multiple overlying seasonality, local trends, and structural breaks.

Time-series of daily cash withdrawal amounts

10.4 RESULTS AND ANALYSIS

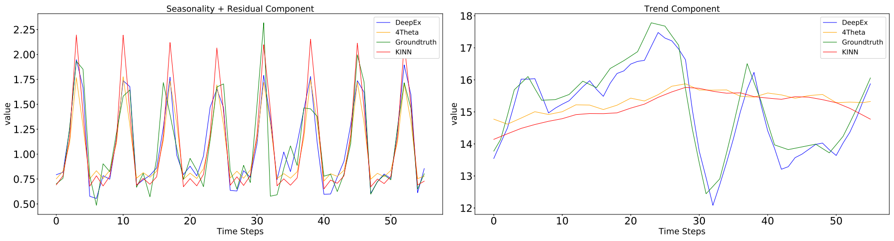
Figure 10.2 shows and compares the performance of DeepEX on a randomly selected time-series from the NN5 data set. It is evident from the Figure 10.2b that DeepEX does a better job at following the trend of the time-series compared to the expert network, which struggled in correctly modeling the magnitude of the peaks. KINN [44] also closely followed the expert model. Similar pattern can be observed from Figure 10.2a where DeepEX was able to capture minor variations in seasonal and residual components, especially in cases of minimas, as compared to the other models. DeepEX had a Symmetric Mean Absolute Percentage Error (SMAPE) score of 15.04 on this particular time-series whereas the SMAPE score of the expert model was 22.40, highlighting the efficacy of DeepEX in modeling the sequence. This dominance of DeepEX was found to be consistent on the entire data set.

Experimental details

We performed a series of experiments in order to validate the effectiveness of DeepEX's knowledge incorporation scheme in helping DNN to reduce its dependence on data, along with its ability to scale to multi-step ahead prediction. As mentioned previously, for the first set of experiments, only 75% of the training data was utilized to train parameters of the DNNs. For NN5 data set, the performance was eval-

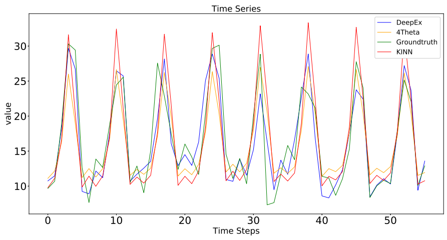
---

[2] Competition website: https://irafm.osu.cz/cif/main.php
[3] Competition website: http://www.neural-forecasting-competition.com/NN5/



(a) Prediction of seasonal and residual components

(b) Prediction of trend component



(c) Overall prediction of the time-series

Figure 10.2: Prediction of DeepEX and other networks on a randomly selected time-series from the NN5 data set with a horizon 1

uated for a horizon of 1, 3, 8 and 56. Whereas, for CIF2016, the performance was evaluated for a horizon of 1, 3 and 6/12. In the next set of experiments, training data was further reduced to 50% while the horizons were kept same. When the size of the data set was reduced to half, many time-series in CIF2016 became so small that even having a horizon of one in the validation set was not possible. Hence, in this particular experimental setting, CIF2016 data set was not evaluated for a horizon of 6/12.

DeepEX achieved lower SMAPE

Table 10.1 shows the results of the aforementioned experiments. In most of the cases, DeepEX achieved lower SMAPE score compared to the other techniques. Even in data scarce scenarios, DeepEX showed an improvement of almost 46% in terms of SMAPE score when trained on only 50% of the data from the NN5 data set with a horizon of 1.

Table 10.1: Results of CIF2016 and NN5 data sets when applied different techniques. Minimum SMAPE is highlighted.

Data set	CIF2016				NN5			
Percentage	Horizon	4Theta	KINN	DeepEX	Horizon	4Theta	KINN	DeepEX
75%	1	9.2	99.2	**7.5**	1	20.1	24.4	**17.2**
	3	10.1	117.0	**9.4**	3	20.6	27.2	**18.2**
	6/12	13	96.0	**12.8**	8	20.6	29.3	**19.8**
					56	21.5	65.0	**21.4**
50%	1	30.4	99.0	**18.9**	1	35.3	39.5	**19.0**
	3	32.3	105.0	**22.5**	3	35.1	44.8	**21.3**
	6/12	-	-	-	8	34.9	46.8	**28.1**
					56	34.0	93.0	**32.8**

Table 10.2: Results of CIF2016 data set of different techniques ordered by mean SMAPE. DeepEX result is highlighted.

Method	Mean SMAPE
LSTM.Cluster	10.53
LSTMs and ETS	10.83
ETS	11.87
MLP	12.13
REST	12.45
Seq-2-Seq(75%)	12.57
DeepLSF(75%)	12.70
ES	12.73
DeepEX (trained on 75% data)	**12.80**
FRBE	12.90
HEM	13.04
Avg	13.05
BaggedETS	13.13
LSTM	13.33
Fuzzy c-regression	13.73
PB-GRNN	14.50
PB-RF	14.50
ARIMA	14.56
Theta	14.76

Similarly, for the same experimental setting, it showed an improvement of 38% for CIF2016 data set. It was also observed that for bigger horizon, the percentage gain in terms of SMAPE was lower compared to experiments with smaller horizon since the complexity of the task was significantly enhanced. Nevertheless, even in these cases, DeepEX still outperformed other techniques. KINN [44] particularly struggled on these data sets as it could not handle time-series with trend component.

Comparison with the best performing methods

We compared DeepEX with the top performing techniques for both of these competitions i.e. NN5 and CIF2016. Table 10.2 shows the comparison of results on CIF2016 data set. DeepEX trained with 75% of the training data outperformed most of the other techniques, including BaggesETS [21], ARIMA and Theta methods, which were considered as benchmarks in the competition and achieved comparative performance with that of the top performing models.

Table 10.3 shows the results obtained on the NN5 data set including both DeepEX and other state-of-the-art models. Similar to the case of CIF2016, DeepEX trained on 75% of the data outperformed most of the techniques and is even slightly better then *LSTM.Cluster* [17] which was the best performing model for the CIF2016 data set. This demonstrates the robustness of DeepEX and its ability to work on different data sets.

Table 10.3: Results of NN5 data set of different techniques ordered by Mean SMAPE. DeepEX results are highlighted.

Name	Mean SMAPE
Wildi	19.9
Andrawis	20.4
DeepLSF(75%)	20.5
D'yakonov	20.6
Noncheva	21.1
DeepEX(trained on 75% data)	**21.4**
LSTM.Cluster	21.6
Rauch	21.7
Luna	21.8
Lagoo	21.9
Wichard	22.1
Gao	22.3
LSTM.All	23.4
Puma-Villanueva	23.7
Seq-2-Seq(75%)	22.8
Autobox(Reilly)	24..1
Lewicke	24.5
Brentnall	24.8
Dang	25.3
Pasero	25.3
Adeodato	25.3
undisclosed	26.8
undisclosed	27.3
Tung	28.1
Naive Seasonal	28.8
DeepEx (trained on 50% data)	**32.8**
undisclosed	33.1

Part VI

CONCLUSION

11

CONCLUSION AND OUTLOOK

This chapter summarizes the challenges in the domain of time-series anomaly detection and their solutions in the form of the framework proposed in this thesis. The three main pillars of this framework: Anomaly Detection, Uncertainty Estimation, and Interpretability and Explainability are also summarized in this chapter as the solutions to tackle the major problems in the area of time-series analysis. Furthermore, the limitations of the presented work along the possible research directions to overcome those limitations are also discussed.

11.1 CONCLUSION

This thesis is aimed at providing a comprehensive framework that helps researchers and analysts in robustly finding time-series anomalies and possible reasons of their occurrences. The back-bone, or the first pillar of this thesis is time-series anomaly detection which serves as a major contribution of this thesis. The proposed traditional, deep learning-based, and hybrid anomaly detection methods are capable of detecting point anomalies as well as time-series discords. In critical domains where human lives are directly or indirectly linked to a machine's decision, mere classification does not completely serve the purpose, but a classification with high certainty score is actually required. A hybrid confidence estimation method is proposed in this thesis as a second pillar of the framework which provides a confidence score on top of the classification. By keeping the importance of interpretability and explainability of DNN in view, the third pillar of the proposed framework contributes to highlighting the most influential data points of a time-series that help analysts in interpreting the underneath DNN model. In addition to that, the decision of a DNN is also explained in natural language, so that the reason for a particular decision can easily be studied and argued.

Anomaly detection is the backbone of data analysis processes

In the current era of IoT and digitization, everything is connected to the internet. With this connectivity, the internal state of machines and their surroundings are recorded by employing hundreds of sensors. The data is collected and stored in data warehouses that is further used for different kind of analysis, including i) Diagnostic Analysis, ii) Predictive Analysis, and iii) Prescriptive Analysis. As anomaly detection is the backbone of different data analysis, major contributions of this thesis are in the area of anomaly detection. First anomaly detection approach proposed in this thesis is characterized as a traditional anomaly detection approach. This approach is proposed specifically for HVAC systems, as it is evident from other studies [192] that HVAC systems have a dominating share in IoT-based household devices. HVAC systems are now part of everyday human life, so their well-being and avoidance of any breakdown are very important for the ease of customers. To achieve this goal, the proposed HVAC anomaly detection method generates alarms for service providers as soon as the method foresees a possible issue in the machine. This method utilizes contextual information in addition to the current data points to detect an anomaly as the environment of such systems keeps on changing. It is shown in the thesis that the proposed method outperforms different traditional anomaly detection methods including CBLOF, rPCA, and Twitter anomaly detection. This method is evaluated on two real data sets.

Utilization of contextual information for HVAC anomaly detection

Due to the presence of intrinsic time-series properties like trend, cycle, seasonality, and change-point in most of the real time-series data sets, the traditional anomaly detection methods show limited performance. This thesis contributes to tackling this issue by offering DeepAnT. To the best of the author's knowledge, DeepAnT is the first deep learning-based anomaly detection method that is capable of detecting point and contextual anomalies. In addition to that, this method can be used to detect time-series discords. This CNN-based unsupervised anomaly detection method leverages the original time series data even without removing the anomalies from the training set. Another highlight of this method is its less appetite for data, unlike other LSTM-based anomaly detection methods. It is equally applicable to big data as well as small data. Only 40% of a given time-series is used to train a model. A detailed evaluation of 15 state-of-the-art methods in different settings on 10 data sets, that contain 433 time-series in total are provided in Chapter 5. DeepAnT has gained the state-of-the-art performance on most of the data sets.

CNN-based unsupervised anomaly detection

Another contribution of this thesis is to combine the strengths of statistical-based anomaly detection and deep learning-based anomaly detection for time-series in FuseAD. These two domains are considered as two disjoint worlds in which each has its own advantages and disadvantages. Statistical-based anomaly detection methods are well established, trusted, and used in most industrial settings mainly due to their transparency. Conversely, deep learning-based anomaly detection methods are gaining a lot of hype in the research community mainly due to their automatic features selection and nearly human-level accuracy in some tasks. The aim of the FuseAD is to benefit from these two worlds by deploying residual mechanism that enables the network to complement the strengths of the underlying two disjoint models, by fusing the information encapsulated in them. The evaluation shows that the proposed method outperformed other 12 anomaly detection methods on 4 Yahoo Webscope data sets.

Fusion of statistical-based and deep learning-based anomaly detection

Anomaly detection, being an old and active research area has a lot of practical applications and use-cases, both in the research community and industry. Due to this long history, there exists a variety of distance-based, density-based, kernel-based, and cluster-based anomaly detection algorithms. As some of these methods were originally proposed for a specific use-case, their comparison with each other on a variety of data sets, especially on time-series data is missing in the literature. In this thesis, a comparative study is presented in which different traditional anomaly detection methods are evaluated on time-series data. In addition to that, these methods are also compared with deep learning-based anomaly detection methods. To analyze the anomaly detection methods from different perspectives, four different evaluation methods are used. In this study, a quantitative comparison of 13 commonly used anomaly detection methods on real and synthetic time-series data from a wide range of domains including internet traffic, cloud services, automotive traffic, and online advertisement is provided.

Comparison of 13 commonly used anomaly detection methods

In many domains including health-care, automotive, and law enforcement, only classification of events or a given time-series is not sufficient. In such domains, the impact of a machine's decision on human life can be very significant. Even with the human-level accuracy of deterministic methods, there is still a possibility that they generate a false alarm. Confident decisions are required in the aforementioned critical domains and use-cases. Bayesian neural networks provide a way to generate posterior distribution, which can be used for the model's uncertainty estimation. Due to a wide parameter space, Bayesian neural networks are computationally very expensive. Another contribution of this thesis is to combine deterministic models with the posterior distribution approximation of Bayesian neural networks. The proposed hybrid approach performs superior to both deterministic and Bayesian methods in terms of classification accuracy, and also provides an estimate of uncertainty.

Uncertainty estimation

According to Lipton [163], the objective of machine learning might be to reduce error, but their real-world purpose is to provide useful information. The deficiency of this useful information from the current state of DNNs makes their use in the aforementioned critical domains questionable. To be deployed for real-world and practical purposes, DNNs should be confident and trustworthy. Interpretability and explainability of DNNs are the processes of opening-up these black-boxes, so that their decision, and decision-making process can be interpreted and explained to humans. Another major contribution of this thesis is an effort of demystifying DNN and providing natural language explanations of their decisions. This is achieved by enabling a system to answer HOW and WHY a decision is made. TSViz – a visualization framework that demystifies convolutional deep learning models for time-series data, contributes to the HOW part. This frame-

Natural language explanations of DNN decisions

work highlights the parts of the input that might be most influential to a network's decision. To answer the WHY part, TSXplain is proposed. By aligning the powerful time-series statistical features with the most influential data points, TSXplain generates natural language explanations of DNN. Based on the defined rules, the explanations are generated for novice and expert users. A survey presented in Chapter 9 shows that the generated explanations are actually relevant and correct. To the best of the author's knowledge, TSXplain is a pioneer in explaining a DNNs decision for time-series data.

Knowledge incorporation into the data-driven systems

In addition to the above mentioned contributions, this thesis also looked into the area of knowledge incorporation into DNNs. In contrast to machines, humans reach to a decision, based on their knowledge collected over time from different sources. This knowledge gives added advantage to humans over machines in their decision-making process. To make machines more intelligent, DeepEX is presented in the associated research section that incorporates knowledge into the data-driven systems. This new knowledge incorporating residual framework combines best of both knowledge- as well as data-driven approaches. Expert predictions are incorporated in the form of a residual scheme, where expert predictions are added to the output of the DNN model and are also used for conditioning the DNN.DeepEX aims to reduce the dependency of DNNs on the data by leveraging information contained in the knowledge stream. The evaluation shows that DeepEX not only alleviates data dependence but also significantly boosts the performance of the network. DeepEX trained on only 75% of the data ranked at 6th place overall in the *NN5* competition and at 7th place in the *CIF2016* competition.

11.2 LIMITATIONS

Thresholding

The anomaly detection algorithms proposed in this thesis have two modules, as most of the other deep learning-based methods. The first module is generally responsible for predicting the next n timestamp(s), and the second module detects anomalies. The anomalies are detected by finding a difference between the actual and the predicted values. At this point, the defined threshold plays its role in marking a data point as normal or abnormal. Although, the thresholds used in the contextual anomaly detection (Chapter 4) and in DeepAnT (Chapter 5) are selected systematically, they might not give the best results on a different data set. This parameter needs to be tweaked for a different data set to get the best out of the proposed methods, as it is generally true for other methods too.

This thesis presents some initial efforts in the area of interpretability and explainability for time-series data. With the help of the presented work in this direction, we want to highlight the significance of DNN decision explanation and its practical applications. One of the limitations of the TSXplain is its specificity for the task at hand. The computed features and the rule-base are not easily transferable between different tasks. Currently, the explanations are generated based on the defined rules. As the target audience, their expectations, and data set change; the rules need to be amended according to the new requirements. Statistical features are used in addition to sequence-wise and point-wise features in the explanations. Another limitation is regarding the availability of a suitable set of statistical features for a particular task. In that case, existing features might need to be dropped and new features might add on.

Specificity for the task at hand

11.3 FUTURE WORK

In recent years, a lot of anomaly detection research papers for time-series have been published. It shows the interest and need for robust anomaly detection methods, specifically for time-series data. Most of these papers test the algorithms on one or more of these most common time-series benchmarks: NASA, Yahoo Webscope, and NAB. These benchmarks contain both real and synthetic time-series, but with one or more of the following issues:

Time-series anomaly detection benchmarking

- **Incorrect Labeling:** When a given time-series is analyzed visually, some anomalies are not labeled as anomalous, while some normal data points are labeled as anomalous.
- **Anomaly Density:** Some time-series contain a large ratio of anomalies as compared to the normal data points. By definition, anomalies should be a very small ratio of whole time-series, only then it is considered as an anomaly.
- **Insignificance:** There exist some time-series that might be very trivial for an algorithm.

The presence of these problems in some time-series makes them unsuitable for a good comparison. To avoid these problems in the future, and for the sake of better, fair, and useful comparisons, it is planned to offer a time-series anomaly detection benchmark.

XAI is another area in which a lot of papers have been proposed recently which again highlights the need of the hour. As the trend of practically using AI systems is increasing in many sectors, including finance, medical, and automotive, the demand for having their decision's justification is also increasing. Researchers and practitioners are using interpretability and explainability of DNNs in an effort to provide some kind of justification of the system's decision-making process and the actual decision. Due to the applicability of DNNs in a wide range of domains, the proposed solutions span to different niches in which the very own notions of interpretability and explainability are used. The current state of XAI is that there exists not even

Formally define interpretability and explainability

a single agreed-upon definition of interpretability and explainability. There is not only a lack of clear understanding of these terms, there exist no framework or a reference point according to which people can claim that their proposed system is actually interpretable and explainable. The traits of interpretable and explainable XAI methods have also not been defined yet. We have already started a study to formally define the interpretability and explainability terms and to come up with a framework that can define if a given solution is actually interpretable and explainable, or not. If the research community don't pay attention to these concerns, then it will also be a challenge to compare XAI methods as the consensus of basic terminologies will be missing from them, resulting in no common grounds for comparison. Evaluation of XAI methods is also an open question. In most of the studies, quantitative evaluation is missing, and expert evaluations are conducted. We aim to address this issue too in future research work.

BIBLIOGRAPHY

[1] R. P. Adams and D. J. MacKay. "Bayesian online changepoint detection." In: *arXiv preprint arXiv:0710.3742* (2007) (cit. on pp. 91, 98, 108).

[2] A. O. Adewumi and A. A. Akinyelu. "A survey of machine-learning and nature-inspired based credit card fraud detection techniques." In: *International Journal of System Assurance Engineering and Management* 8.2 (2017), pp. 937–953 (cit. on p. 114).

[3] C. C. Aggarwal. In: *Outlier Analysis*. Springer, 2016 (cit. on pp. 33, 35, 37).

[4] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha. "Unsupervised real-time anomaly detection for streaming data." In: *Neurocomputing* 262 (2017), pp. 134–147 (cit. on pp. 47, 91, 98, 108).

[5] A. Alazizi, A. Habrard, F. Jacquenet, L. He-Guelton, F. Oblé, and W. Siblini. "Anomaly Detection, Consider Your Dataset First An Illustration on Fraud Detection." In: *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE. 2019, pp. 1351–1355 (cit. on p. 3).

[6] Alejandro Correa Bahnsen. *Building AI Applications Using Deep Learning*. `https://blog.easysol.net/building-ai-applications/`. June 2017 (cit. on p. 78).

[7] T. Amarbayasgalan, B. Jargalsaikhan, and K. Ryu. "Unsupervised novelty detection using deep autoencoders with density based clustering." In: *Applied Sciences* 8.9 (2018), p. 1468 (cit. on p. 51).

[8] M. Amer, M. Goldstein, and S. Abdennadher. "Enhancing one-class support vector machines for unsupervised anomaly detection." In: *ACM SIGKDD Workshop on Outlier Detection and Description*. ACM. 2013 (cit. on pp. 46, 60, 65, 98, 119, 120).

[9] R. Andrews, J. Diederich, and A. B. Tickle. "Survey and critique of techniques for extracting rules from trained artificial neural networks." In: *Knowledge-based systems* 8.6 (1995), pp. 373–389 (cit. on p. 140).

[10] F. Angiulli and C. Pizzuti. "Fast outlier detection in high dimensional spaces." In: *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer. 2002, pp. 15–27 (cit. on pp. 40, 116, 119, 120).

[11] Apple Inc. *Empowering medical researchers, doctors, and you*. `https://www.apple.com/researchkit/` (cit. on p. 16).

[12] L. Ardizzone, J. Kruse, S. Wirkert, D. Rahner, E. W. Pellegrini, R. S. Klessen, L. Maier-Hein, C. Rother, and U. Köthe. "Analyzing Inverse Problems with Invertible Neural Networks." In: *arXiv preprint arXiv:1808.04730* (2018) (cit. on p. 125).

[13] E. Auschitzky, M. Hammer, and A. Rajagopaul. *How Big Data can Improve Manufacturing*. `http://www.mckinsey.com/business-functions/operations/our-insights/how-big-data-can-improve-manufacturing`. Online; Accessed: 2016-06-21. July 2014 (cit. on p. 17).

[14] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." In: *PloS one* 10.7 (2015), e0130140 (cit. on p. 141).

[15] M. N. Bajwa, S. Khurram, M. Munir, S. A. Siddiqui, M. I. Malik, A. Dengel, and S. Ahmed. "Confident classification using a hybrid between deterministic and probabilistic convolutional neural networks." In: *IEEE Access* 8 (2020), pp. 115476–115485. DOI: `10.1109/ACCESS.2020.3004409` (cit. on p. 123).

[16] M. N. Bajwa, M. I. Malik, S. A. Siddiqui, A. Dengel, F. Shafait, W. Neumeier, and S. Ahmed. "Two-stage framework for optic disc localization and glaucoma classification in retinal fundus images using deep learning." In: *BMC medical informatics and decision making* 19.1 (2019), p. 136 (cit. on p. 123).

[17] K. Bandara, C. Bergmeir, and S. Smyl. "Forecasting Across Time Series Databases using Recurrent Neural Networks on Groups of Similar Series: A Clustering Approach." In: *arXiv preprint* (2017) (cit. on pp. 146, 162, 167).

[18] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. "Network Dissection: Quantifying Interpretability of Deep Visual Representations." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2017, pp. 6541–6549 (cit. on p. 141).

[19] A. Beghi, R. Brignoli, L. Cecchinato, G. Menegazzo, and M. Rampazzo. "A data-driven approach for fault diagnosis in HVAC chiller systems." In: *2015 IEEE Conference on Control Applications (CCA)*. Sept. 2015, pp. 966–971. DOI: `10.1109/CCA.2015.7320737` (cit. on p. 97).

[20] A. Beghi, R. Brignoli, L. Cecchinato, G. Menegazzo, and M. Rampazzo. "A data-driven approach for fault diagnosis in HVAC chiller systems." In: *2015 IEEE Conference on Control Applications (CCA)*. IEEE. 2015, pp. 966–971 (cit. on p. 33).

[21] C. Bergmeir, R. J. Hyndman, and J. M. Benítez. "Bagging exponential smoothing methods using STL decomposition and Box–Cox transformation." In: *International journal of forecasting* 32.2 (2016), pp. 303–312 (cit. on p. 167).

[22] L. Beringhoff. "Kau-Sensoren melden die Brunst." In: *top agrar* (2014) (cit. on p. 26).

[23] Berkemeier, Ostermann-Palz, and Stöcker. "Kuh-Navis für den Stall." In: *Elite Magazin* (2015) (cit. on p. 26).

[24] E. Betters. *Apple HomeKit and Home app: What are they and how do they work?* `https://tinyurl.com/kcx7c84`. Online; Accessed: 2016-06-26. June 2016 (cit. on p. 28).

[25] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano. "A review on outlier/anomaly detection in time series data." In: *arXiv preprint arXiv:2002.04236* (2020) (cit. on pp. 3, 114).

[26] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. "Variational inference: A review for statisticians." In: *Journal of the American Statistical Association* 112.518 (2017), pp. 859–877 (cit. on p. 124).

[27] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. "Weight uncertainty in neural networks." In: *arXiv preprint arXiv:1505.05424* (2015) (cit. on pp. 4, 124).

[28] B. E. Boser, I. M. Guyon, and V. N. Vapnik. "A training algorithm for optimal margin classifiers." In: *Proceedings of the fifth annual workshop on Computational learning theory*. 1992, pp. 144–152 (cit. on p. 46).

[29] I. Boulnemour, B. Boucheham, and S. Benloucif. "Improved Dynamic Time Warping for Abnormality Detection in ECG Time Series." In: *International Conference on Bioinformatics and Biomedical Engineering*. Springer. 2016, pp. 242–253 (cit. on pp. 60, 61).

[30] M. Braei and S. Wagner. "Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art." In: *arXiv preprint arXiv:2004.00433* (2020) (cit. on pp. 33, 42, 45, 114).

[31] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. "LOF: identifying density-based local outliers." In: *ACM sigmod record*. Vol. 29. 2. ACM. 2000, pp. 93–104 (cit. on pp. 37, 40, 97, 98, 106, 116, 119, 120).

[32] T. S. Buda, B. Caglayan, and H. Assem. "DeepAD: A Generic Framework Based on Deep Learning for Time Series Anomaly Detection." In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 2018, pp. 577–588 (cit. on pp. 98, 99, 157, 159).

[33] A. Buetti-Dinh, V. Galli, S. Bellenberg, O. Ilie, M. Herold, S. Christel, M. Boretska, I. V. Pivkin, P. Wilmes, W. Sand, et al. "Deep neural networks outperform human expert's capacity in characterizing bioleaching bacterial biofilm composition." In: *Biotechnology Reports* 22 (2019), e00321 (cit. on p. 3).

[34] Y. Cai, K. Guan, J. Peng, S. Wang, C. Seifert, B. Wardlow, and Z. Li. "A high-performance and in-season classification system of field-level crop types using time-series Landsat data and a machine learning approach." In: *Remote Sensing of Environment* 210 (2018), pp. 35–47 (cit. on p. 139).

[35] M. Canizo, I. Triguero, A. Conde, and E. Onieva. "Multi-head CNN–RNN for multi-time series anomaly detection: An industrial case study." In: *Neurocomputing* 363 (2019), pp. 246–260 (cit. on p. 3).

[36] A. Capozzoli, M. S. Piscitelli, A. Gorrino, I. Ballarini, and V. Corrado. "Data analytics for occupancy pattern learning to reduce the energy consumption of HVAC systems in office buildings." In: *Sustainable cities and society* 35 (2017), pp. 191–208 (cit. on p. 97).

[37] D. Carrera, B. Rossi, P. Fragneto, and G. Boracchi. "Online anomaly detection for long-term ecg monitoring using wearable devices." In: *Pattern Recognition* 88 (2019), pp. 482–492 (cit. on p. 3).

[38] G. Casalicchio, J. Bossek, M. Lang, D. Kirchhoff, P. Kerschke, B. Hofner, H. Seibold, J. Vanschoren, and B. Bischl. "OpenML: An R package to connect to the machine learning platform OpenML." In: *Computational Statistics* 32.3 (2017) (cit. on p. 116).

[39] M. Çelik, F. Dadaşer-Çelik, and A. Ş. Dokuz. "Anomaly detection in temperature data using dbscan algorithm." In: *2011 international symposium on innovations in intelligent systems and applications*. IEEE. 2011, pp. 91–95 (cit. on p. 48).

[40] D. Chakraborty and H. Elzarka. "Early detection of faults in HVAC systems using an XGBoost model with a dynamic threshold." In: *Energy and Buildings* 185 (2019), pp. 326–344 (cit. on p. 62).

[41] R. Chalapathy and S. Chawla. "Deep learning for anomaly detection: A survey." In: *arXiv preprint arXiv:1901.03407* (2019) (cit. on pp. 77, 97, 113, 114).

[42] V. Chandola, A. Banerjee, and V. Kumar. "Anomaly detection: A survey." In: *ACM computing surveys (CSUR)* 41.3 (2009), pp. 1–58 (cit. on pp. 38, 39, 114).

[43] V. Chandola, D. Cheboli, and V. Kumar. “Detecting anomalies in a time series database.” In: *Computer Science Department, University of Minnesota, Tech. Rep* (2009) (cit. on p. 78).

[44] M. A. Chattha, S. A. Siddiqui, M. I. Malik, L. van Elst, A. Dengel, and S. Ahmed. “KINN.” In: *arXiv preprint arXiv:1902.05653* (2019) (cit. on pp. 160, 165, 167).

[45] M. A. Chattha, S. A. Siddiqui, M. Munir, M. I. Malik, L. van Elst, A. Dengel, and S. Ahmed. “DeepEX: Bridging the Gap Between Knowledge and Data Driven Techniques for Time Series Forecasting.” In: *International Conference on Artificial Neural Networks*. Springer. 2019, pp. 639–651. URL: `https://link.springer.com/chapter/10.1007/978-3-030-30484-3_51` (cit. on p. 157).

[46] S. Chauhan and L. Vig. “Anomaly detection in ECG time signals via deep long short-term memory networks.” In: *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*. IEEE. 2015, pp. 1–7 (cit. on pp. 3, 50, 78, 98).

[47] T. Chen and C. Guestrin. “XGBoost: A scalable tree boosting system.” In: *KDD*. ACM. 2016, pp. 785–794 (cit. on pp. 46, 116, 119, 120).

[48] J. Choo and S. Liu. “Visual analytics for explainable deep learning.” In: *IEEE computer graphics and applications* 38.4 (2018), pp. 84–92 (cit. on p. 3).

[49] R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning. “STL: A seasonal-trend decomposition procedure based on loess.” In: *Journal of Official Statistics* 6.1 (1990), pp. 3–73 (cit. on p. 44).

[50] L. Columbus. *53% Of Companies Are Adopting Big Data Analytics*. Dec. 2017. URL: `https://goo.gl/tN5eNC` (cit. on pp. 3, 33, 77).

[51] L. Columbus. *Gartner's hype cycle for emerging technologies, 2017 adds 5G and deep learning for first time*. `https://www.forbes.com/sites/louiscolumbus/2017/08/15/gartners-hype-cycle-for-emerging-technologies-2017-adds-5g-and-deep-learning-for-first-time/#15856b145043`. 2017 (cit. on p. 157).

[52] A. J. Conejo, M. A. Plazas, R. Espinola, and A. B. Molina. “Day-ahead electricity price forecasting using the wavelet transform and ARIMA models.” In: *IEEE transactions on power systems* 20.2 (2005), pp. 1035–1042 (cit. on p. 100).

[53] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. “Supervised learning of universal sentence representations from natural language inference data.” In: *arXiv preprint arXiv:1705.02364* (2017) (cit. on p. 157).

[54] J. Contreras, R. Espinola, F. J. Nogales, and A. J. Conejo. “ARIMA models to predict next-day electricity prices.” In: *IEEE transactions on power systems* 18.3 (2003), pp. 1014–1020 (cit. on p. 100).

[55] B. F. Crabtree, S. C. Ray, P. M. Schmidt, P. T. O’Connor, and D. D. Schmidt. “The individual over time: time series applications in health care research.” In: *Journal of clinical epidemiology* 43.3 (1990), pp. 241–260 (cit. on pp. 98, 139).

[56] Y. Cui, S. Ahmad, and J. Hawkins. “Continuous online sequence learning with an unsupervised neural network model.” In: *Neural computation* 28.11 (2016), pp. 2474–2504 (cit. on p. 47).

[57] Z. Cui, W. Chen, and Y. Chen. “Multi-scale convolutional neural networks for time series classification.” In: *arXiv preprint arXiv:1603.06995* (2016) (cit. on p. 49).

[58] Daimler AG. *Predictive Powertrain Control - Clever cruise control helps save fuel*. `https://media.daimler.com/marsMediaSite/en/instance/ko.xhtml?oid=9917205`. Online; Accessed: 2016-06-18 (cit. on p. 22).

[59] H. K. Dam, T. Tran, and A. Ghose. “Explainable software analytics.” In: *Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results*. 2018, pp. 53–56 (cit. on p. 140).

[60] Datameer, Inc. *Vivint Drives Smart Home Automation With Datameer*. `https://www.datameer.com/wp-content/uploads/2015/09/vivint_testimonial.pdf`. Online; Accessed: 2017-04-20 (cit. on p. 28).

[61] H. A. Dau et al. *The UCR Time Series Classification Archive*. `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`. Oct. 2018 (cit. on p. 131).

[62] N. De Cao, I. Titov, and W. Aziz. “Block Neural Autoregressive Flow.” In: *arXiv preprint arXiv:1904.04676* (2019) (cit. on p. 125).

[63] E. Dedezade. *Volkswagen and Microsoft partner to create new Automotive Cloud*. `https://news.microsoft.com/europe/features/volkswagen-and-microsoft-partner-to-create-new-automotive-cloud/`. Oct. 2018 (cit. on p. 22).

[64] Deloitte. *Data is the new gold.* 2020. URL: `t.ly/XH2G` (cit. on p. 3).

[65] D. Dheeru and E. Karra Taniskidou. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml (cit. on pp. 54–56).

[66] L. Dinh, D. Krueger, and Y. Bengio. "Nice: Non-linear independent components estimation." In: *arXiv preprint arXiv:1410.8516* (2014) (cit. on p. 126).

[67] L. Dinh, J. Sohl-Dickstein, and S. Bengio. "Density estimation using real nvp." In: *arXiv preprint arXiv:1605.08803* (2016) (cit. on p. 125).

[68] P. Du Jardin and E. Séverin. "Predicting corporate bankruptcy using a self-organizing map: An empirical study to improve the forecasting horizon of a financial failure model." In: *Decision Support Systems* 51.3 (2011), pp. 701–711 (cit. on p. 83).

[69] X. Du, M. El-Khamy, J. Lee, and L. Davis. "Fused DNN: A deep neural network fusion approach to fast and robust pedestrian detection." In: *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*. IEEE. 2017, pp. 953–961 (cit. on p. 99).

[70] S. Eathington. *Pulling Back the Curtain on Our Innovation Pipeline*. http://www.climateinsights.com/2017-innovation-pipeline/. Jan. 2017 (cit. on p. 12).

[71] P. Esling and C. Agon. "Time-series data mining." In: *ACM Computing Surveys (CSUR)* 45.1 (2012), pp. 1–34 (cit. on p. 3).

[72] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. "A density-based algorithm for discovering clusters in large spatial databases with noise." In: *Kdd*. Vol. 96. 34. 1996, pp. 226–231 (cit. on p. 47).

[73] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. "Dermatologist-level classification of skin cancer with deep neural networks." In: *Nature* 542.7639 (2017), p. 115 (cit. on p. 123).

[74] B. Ferrell and S. Santuro. *NASA Shuttle Valve Data*. 2005. URL: http://www.cs.fit.edu/~pkc/nasa/data/ (cit. on pp. 56, 95).

[75] Fortune Business Insights. *Internet of Things (IoT) Market*. https://www.fortunebusinessinsights.com/industry-reports/internet-of-things-iot-market-100307. 2019 (cit. on p. 10).

[76] T.-c. Fu. "A review on time series data mining." In: *Engineering Applications of Artificial Intelligence* 24.1 (2011), pp. 164–181 (cit. on p. 3).

[77] Y. Gal and Z. Ghahramani. "Dropout as a bayesian approximation: Representing model uncertainty in deep learning." In: *international conference on machine learning*. 2016, pp. 1050–1059 (cit. on pp. 125, 134).

[78] Gartner, Inc. *Gartner Special Report "Digital Business Technologies"*. `https://www.gartner.com/en/newsroom/press-releases/2014-09-08-gartner-says-a-typical-family-home-could-contain-more-than-500-smart-devices-by-2022`. Online; Accessed: 2020-04-25. Sept. 2016 (cit. on p. 27).

[79] R. Geirhos, D. H. Janssen, H. H. Schütt, J. Rauber, M. Bethge, and F. A. Wichmann. "Comparing deep neural networks against humans: object recognition when the signal gets weaker." In: *arXiv preprint arXiv:1706.06969* (2017) (cit. on p. 3).

[80] General Electric. *GE Launches Brilliant Manufacturing Suite to Help Manufacturers Increase Production Efficiency, Execution and Optimization through Advanced Analytics*. `t.ly/gWzW`. Sept. 2015 (cit. on p. 20).

[81] General Electric. *GE Healthcare*. `http://www3.gehealthcare.com/en`. Online; Accessed: 2016-06-11 (cit. on p. 15).

[82] F. A. Gers, D. Eck, and J. Schmidhuber. "Applying LSTM to time series predictable through time-window approaches." In: *Neural Nets WIRN Vietri-01*. Springer, 2002, pp. 193–200 (cit. on p. 79).

[83] M. Ghazvininejad, C. Brockett, M.-W. Chang, B. Dolan, J. Gao, W.-t. Yih, and M. Galley. "A knowledge-grounded neural conversation model." In: *AAAI*. 2018 (cit. on p. 158).

[84] N. Gilbert. *Rules tighten on use of antibiotics on farms*. `http://www.nature.com/news/rules-tighten-on-use-of-antibiotics-on-farms-1.9761`. Online; Accessed: 2016-06-25. Jan. 2012 (cit. on p. 24).

[85] P. Gittins. *Tesla, big data and industrial disruption*. `https://www.capgemini.com/blog/insights-data-blog/2015/09/tesla-big-data-and-industrial-disruption`. Online; Accessed: 2016-06-16. Sept. 2015 (cit. on p. 22).

[86] N. Goix. "How to evaluate the quality of unsupervised anomaly detection algorithms?" In: *arXiv preprint arXiv:1607.01152* (2016) (cit. on pp. 37, 91, 94).

[87] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. "PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals." In: *Circulation* 101.23 (2000 (June 13)). Circulation Electronic Pages: http://circ.ahajournals.org/content/101/23/e215.full PMID:1085218; doi: 10.1161/01.CIR.101.23.e215, e215–e220 (cit. on p. 52).

[88] M. Goldstein. "Anomaly Detection in Large Datasets." PhD-Thesis. München, Germany: University of Kaiserslautern, Feb. 2014, p. 248. URL: `http://www.goldiges.de/phd` (cit. on pp. 33, 77, 78, 139).

[89] M. Goldstein and A. Dengel. "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm." In: *KI-2012: Poster and Demo Track* (2012), pp. 59–63 (cit. on pp. 37, 42, 60, 65, 97, 116, 119, 120).

[90] M. Goldstein and S. Uchida. "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data." In: *PloS one* 11.4 (2016), e0152173 (cit. on pp. 35, 36, 41, 114).

[91] S. D. Gollapalli, X.-L. Li, and P. Yang. "Incorporating Expert Knowledge into Keyphrase Extraction." In: *AAAI*. 2017, pp. 3180–3187 (cit. on p. 160).

[92] Google Nest. `t.ly/Dojq`. Online; Accessed: 2017-04-24 (cit. on p. 29).

[93] W. Grayson. *Caterpillar's predictive diagnostics will enable more "repairs before failure" in future telematics updates*. `http://www.equipmentworld.com/caterpillar-predictive-diagnostics-telematics/`. Online; Accessed: 2016-06-20. Mar. 2015 (cit. on p. 22).

[94] F. E. Grubbs. "Procedures for detecting outlying observations in samples." In: *Technometrics* 11.1 (1969), pp. 1–21 (cit. on p. 34).

[95] R. D. Gunathilaka, G. A. Tularam, et al. "The tea industry and a review of its price modelling in major tea producing countries." In: *Journal of Management and Strategy* 7.1 (2016), pp. 21–36 (cit. on p. 100).

[96] N. Günnemann, S. Günnemann, and C. Faloutsos. "Robust multivariate autoregression for anomaly detection in dynamic product ratings." In: *Proceedings of the 23rd international conference on World wide web*. 2014, pp. 361–372 (cit. on p. 33).

[97] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. "On calibration of modern neural networks." In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 1321–1330 (cit. on p. 134).

[98] P. Guo, C. Anderson, K. Pearson, and R. Farrell. "Neural Network Interpretation via Fine Grained Textual Summarization." In: *arXiv preprint arXiv:1805.08969* (2018) (cit. on pp. 142, 143).

[99] M. Gupta, J. Gao, C. C. Aggarwal, and J. Han. "Outlier detection for temporal data: A survey." In: *IEEE Transactions on Knowledge and Data Engineering* 26.9 (2013), pp. 2250–2267 (cit. on p. 114).

[100] H. Haenssle, C. Fink, R. Schneiderbauer, F. Toberer, T. Buhl, A. Blum, A. Kalloo, A. B. H. Hassen, L. Thomas, A. Enk, et al. "Man against machine: diagnostic performance of a deep learning convolutional neural network for dermoscopic melanoma recognition in comparison to 58 dermatologists." In: *Annals of Oncology* 29.8 (2018), pp. 1836–1842 (cit. on p. 123).

[101] B. Hancock, M. Bringmann, P. Varma, P. Liang, S. Wang, and C. Ré. "Training classifiers with natural language explanations." In: *Proceedings of the conference. Association for Computational Linguistics. Meeting*. Vol. 2018. NIH Public Access. 2018, p. 1884 (cit. on p. 142).

[102] S. Hansen. "Das Gateway – Mediolas Smart-Home-Hub." In: *Magazin für Computertechnik c't* 9 (Apr. 2016), p. 67 (cit. on p. 29).

[103] S. Hansen. "Harke, Schlauch, Handy." In: *Magazin für Computertechnik c't* 13 (June 2016), p. 62 (cit. on p. 27).

[104] E. Haselsteiner and G. Pfurtscheller. "Using time-dependent neural networks for EEG classification." In: *IEEE transactions on rehabilitation engineering* 8.4 (2000), pp. 457–463 (cit. on p. 49).

[105] S. Hawkins, H. He, G. Williams, and R. Baxter. "Outlier detection using replicator neural networks." In: *International Conference on Data Warehousing and Knowledge Discovery*. Springer. 2002, pp. 170–180 (cit. on p. 37).

[106] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition." In: *CoRR* abs/1512.03385 (2015). arXiv: `1512.03385`. URL: `http://arxiv.org/abs/1512.03385` (cit. on p. 124).

[107] Z. He, X. Xu, and S. Deng. "Discovering cluster-based local outliers." In: *Pattern Recognition Letters* 24.9 (2003), pp. 1641–1650 (cit. on pp. 41, 60, 65).

[108] L. Henderson. *CLAAS: The Harvesting Specialists*. `http://www.agrimarketing.com/s/38408`. Jan. 2006 (cit. on p. 13).

[109] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell. "Generating visual explanations." In: *European Conference on Computer Vision*. Springer. 2016, pp. 3–19 (cit. on pp. 142, 143).

[110] D. Hendrycks and K. Gimpel. "A baseline for detecting misclassified and out-of-distribution examples in neural networks." In: *arXiv preprint arXiv:1610.02136* (2016) (cit. on p. 125).

[111] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al. "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups." In: *IEEE Signal processing magazine* 29.6 (2012), pp. 82–97 (cit. on p. 157).

[112] D. Ho, E. Liang, I. Stoica, P. Abbeel, and X. Chen. "Population Based Augmentation: Efficient Learning of Augmentation Policy Schedules." In: *arXiv preprint arXiv:1905.05393* (2019) (cit. on p. 123).

[113] J. Hochenbaum, O. S. Vallis, and A. Kejariwal. "Automatic anomaly detection in the cloud via statistical learning." In: *arXiv preprint arXiv:1704.07706* (2017) (cit. on p. 44).

[114] S. Hochreiter and J. Schmidhuber. "Long short-term memory." In: *Neural computation* 9.8 (1997), pp. 1735–1780 (cit. on pp. 50, 79).

[115] V. Hodge and J. Austin. "A survey of outlier detection methodologies." In: *Artificial intelligence review* 22.2 (2004), pp. 85–126 (cit. on p. 114).

[116] M. Hu, Z. Ji, K. Yan, Y. Guo, X. Feng, J. Gong, X. Zhao, and L. Dong. "Detecting anomalies in time series data via a meta-feature based approach." In: *IEEE Access* 6 (2018), pp. 27760–27776 (cit. on p. 46).

[117] Z. Hu, X. Ma, Z. Liu, E. Hovy, and E. Xing. "Harnessing deep neural networks with logic rules." In: *arXiv preprint arXiv:1603.06318* (2016) (cit. on pp. 157, 159).

[118] C.-W. Huang, D. Krueger, A. Lacoste, and A. Courville. "Neural autoregressive flows." In: *arXiv preprint arXiv:1804.00779* (2018) (cit. on p. 125).

[119] Y. Huang, Y. Cheng, D. Chen, H. Lee, J. Ngiam, Q. V. Le, and Z. Chen. "GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism." In: *arXiv preprint arXiv:1811.06965* (2018) (cit. on pp. 4, 123).

[120] R. J. Hyndman and G. Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018 (cit. on p. 48).

[121] R. J. Hyndman, E. Wang, and N. Laptev. "Large-scale unusual time series detection." In: *2015 IEEE international conference on data mining workshop (ICDMW)*. IEEE. 2015, pp. 1616–1619 (cit. on p. 146).

[122] R. Hyndman and Y. Khandakar. "Automatic Time Series Forecasting: The forecast Package for R." In: *Journal of Statistical Software, Articles* 27.3 (2008), pp. 1–22. DOI: `10.18637/jss.v027.i03`. URL: `https://www.jstatsoft.org/v027/i03` (cit. on pp. 100, 103).

[123] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. “Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size.” In: *arXiv preprint arXiv:1602.07360* (2016) (cit. on p. 81).

[124] IBM Corporation. *Big Data & Analytics in the Manufacturing Industry: The Vaasan Group*. https://www.slideshare.net/IBMBDA/ibm-bda-vassangroupslidesharefinal. Online; Accessed: 2017-04-10. 2014 (cit. on p. 18).

[125] IBM Corporation. *Leading German car manufacturer boosts customer satisfaction using IBM Big Data & Analytics*. https://www-03.ibm.com/press/us/en/pressrelease/43392.wss. Online; Accessed: 2016-06-18. Mar. 2014 (cit. on p. 22).

[126] IBM Corporation. *National Institutes of Health: Finding cures faster by transforming big data into valuable clinical insight*. https://www.ibm.com/industries/au-en/healthcare/case-studies/national-institutes-of-health.html. Online; Accessed: 2016-06-09. July 2015 (cit. on p. 14).

[127] IBM Corporation. *PinnacleHealth System: Transforming healthcare delivery with insight into the "when" and "why" of readmissions*. http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=AB&infotype=PM&appname=SWGE_YT_YT_USEN&htmlfid=YTC03945USEN&attachment=YTC03945USEN.PDF. Online; Accessed: 2016-06-09. 2015 (cit. on p. 14).

[128] IBM Corporation. *Using enhanced cognitive analytics to gain a competitive edge by finding valuable answers to questions not yet asked*. https://www.spssanalyticspartner.com/case-study-mueller-inc/. July 2015 (cit. on p. 18).

[129] IBM Corporation. *Ford, IBM Use Data Analytics to Find More Efficient Ways to Move*. http://www-03.ibm.com/press/us/en/pressrelease/48774.wss. Online; Accessed: 2016-06-21. Jan. 2016 (cit. on p. 21).

[130] IBM Corporation. *IBM Analytics*. http://www.ibm.com/analytics/us/en/industry/#. Online; Accessed: 2016-06-22. 2016 (cit. on p. 18).

[131] M. Jeske, M. Grüner, and F. Weiß. *BIG DATA IN LOGISTICS: A DHL perspective on how to move beyond the hype*. http://www.dhl.com/content/dam/downloads/g0/about_us/innovation/CSI_Studie_BIG_DATA.pdf. Online; Accessed: 2016-06-19. Dec. 2013 (cit. on p. 23).

[132] W. Jin, A. K. Tung, J. Han, and W. Wang. “Ranking outliers using symmetric neighborhood relationship.” In: *Advances in Knowledge Discovery and Data Mining*. Springer, 2006, pp. 577–593 (cit. on p. 41).

[133] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. "An introduction to variational methods for graphical models." In: *Machine learning* 37.2 (1999), pp. 183–233 (cit. on p. 124).

[134] N. Jurran. "Bluetooth Smart Home." In: *Magazin für Computertechnik c't* 15 (June 2015), p. 59 (cit. on p. 27).

[135] N. Jurran. "Heizkörperthermostat: Eve Thermo." In: *Magazin für Computertechnik c't* 11 (May 2016), p. 48 (cit. on p. 27).

[136] N. Jurran. "HomeKits Anknipser." In: *Magazin für Computertechnik c't* 14 (June 2016), p. 56 (cit. on p. 27).

[137] M.-J. Kang and J.-W. Kang. "Intrusion detection system using deep neural network for in-vehicle network security." In: *PloS one* 11.6 (2016), e0155781 (cit. on pp. 98, 139).

[138] A. Kejariwal. *Introducing practical and robust anomaly detection in a time series*. Jan. 2015. URL: `https://blog.twitter.com/2015/introducing-practical-and-robust-anomaly-detection-in-a-time-series` (cit. on pp. 44, 60, 65, 97, 98, 106, 108).

[139] A. Kendall and Y. Gal. "What uncertainties do we need in bayesian deep learning for computer vision?" In: *Advances in neural information processing systems*. 2017, pp. 5574–5584 (cit. on p. 125).

[140] E. Keogh, J. Lin, and A. Fu. "Hot sax: Efficiently finding the most unusual time series subsequence." In: *Data mining, fifth IEEE international conference on*. Ieee. 2005, 8–pp (cit. on p. 95).

[141] J. Kim, A. Rohrbach, T. Darrell, J. Canny, and Z. Akata. "Textual explanations for self-driving vehicles." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 563–578 (cit. on pp. 142, 143).

[142] D. P. Kingma and P. Dhariwal. "Glow: Generative flow with invertible 1x1 convolutions." In: *Advances in Neural Information Processing Systems*. 2018, pp. 10215–10224 (cit. on p. 126).

[143] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. "Improved variational inference with inverse autoregressive flow." In: *Advances in neural information processing systems*. 2016, pp. 4743–4751 (cit. on p. 125).

[144] B. Kiran, D. Thomas, and R. Parakkal. "An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos." In: *Journal of Imaging* 4.2 (2018), p. 36 (cit. on p. 114).

[145] W. Knight. *MIT Technology Review: The Financial World Wants to Open AI's Black Boxes*. 2017. URL: `https://www.technologyreview.com/s/604122/the-financial-world-wants-to-open-ais-black-boxes/` (cit. on p. 139).

[146] G. Koch, R. Zemel, and R. Salakhutdinov. “Siamese neural networks for one-shot image recognition.” In: *ICML Deep Learning Workshop*. Vol. 2. 2015 (cit. on p. 101).

[147] J. Koetsier. *IoT In The USA*. Dec. 2017. URL: `https://goo.gl/CPKYrc` (cit. on pp. 3, 33, 77).

[148] P. W. Koh and P. Liang. “Understanding Black-box Predictions via Influence Functions.” In: *International Conference on Machine Learning (ICML)*. 2017 (cit. on p. 142).

[149] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Imagenet classification with deep convolutional neural networks.” In: *Advances in neural information processing systems*. 2012, pp. 1097–1105 (cit. on pp. 81, 113, 124).

[150] D. Kumar, G. W. Taylor, and A. Wong. “Opening the Black Box of Financial AI with CLEAR-Trade: A CLass-Enhanced Attentive Response Approach for Explaining and Visualizing Deep Learning-Driven Stock Market Prediction.” In: *ArXiv e-prints* (Sept. 2017). arXiv: `1709.01574 [cs.AI]` (cit. on pp. 139, 142, 143).

[151] A. K. Kushwaha, J. K. Dhillon, et al. “Deep Learning Trends for Video Based Activity Recognition: A Survey.” In: *International Journal of Sensors Wireless Communications and Control* 8.3 (2018), pp. 165–171 (cit. on pp. 98, 139).

[152] R. Kwitt and U. Hofmann. “Robust methods for unsupervised PCA-based anomaly detection.” In: *Proc. of IEEE/IST Workshop on Monitoring, Attack Detection and Mitigation* (2006), pp. 1–3 (cit. on pp. 46, 60, 65).

[153] Y. Kwon, J.-H. Won, B. J. Kim, and M. C. Paik. “Uncertainty quantification using Bayesian neural networks in classification: Application to ischemic stroke lesion segmentation.” In: (2018) (cit. on p. 125).

[154] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. “Human-level concept learning through probabilistic program induction.” In: *Science* 350.6266 (2015), pp. 1332–1338 (cit. on p. 157).

[155] N. Laptev, S. Amizadeh, and I. Flint. “Generic and scalable framework for automated time-series anomaly detection.” In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2015, pp. 1939–1947 (cit. on pp. 44, 97).

[156] R. M. Lark, B. L. Nielsen, and T. T. Mottram. “A time series model of daily milk yields and its possible use for detection of a disease (ketosis).” In: *Animal Science* 69.3 (1999) (cit. on p. 139).

[157] A. Lavin and S. Ahmad. “Evaluating Real-Time Anomaly Detection Algorithms–The Numenta Anomaly Benchmark.” In: *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*. IEEE. 2015, pp. 38–44 (cit. on pp. 3, 47, 53, 54, 90, 97, 98, 108).

[158] M. Lázaro-Gredilla and A. R. Figueiras-Vidal. “Marginalized neural network mixtures for large-scale regression.” In: *IEEE transactions on neural networks* 21.8 (2010), pp. 1345–1351 (cit. on p. 125).

[159] Y. LeCun, Y. Bengio, and G. Hinton. “Deep learning.” In: *nature* 521.7553 (2015), p. 436 (cit. on p. 123).

[160] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al. “Gradient-based learning applied to document recognition.” In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324 (cit. on pp. 124, 130).

[161] K. Lee, H. Lee, K. Lee, and J. Shin. “Training confidence-calibrated classifiers for detecting out-of-distribution samples.” In: *arXiv preprint arXiv:1711.09325* (2017) (cit. on p. 125).

[162] M. Leng, X. Chen, and L. Li. “Variable length methods for detecting anomaly patterns in time series.” In: *Computational Intelligence and Design, 2008. ISCID’08. International Symposium on*. Vol. 2. IEEE. 2008, pp. 52–56 (cit. on pp. 60, 61).

[163] Z. C. Lipton. “The mythos of model interpretability.” In: *Queue* 16.3 (2018), pp. 31–57 (cit. on p. 175).

[164] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzell. “Learning to Diagnose with LSTM Recurrent Neural Networks.” In: *arXiv preprint arXiv:1511.03677* (2015) (cit. on pp. 50, 78, 79, 99).

[165] F. T. Liu, K. M. Ting, and Z.-H. Zhou. “Isolation forest.” In: *Data Mining, 2008. ICDM’08. Eighth IEEE International Conference on*. IEEE. 2008, pp. 413–422 (cit. on pp. 54–56, 91, 94, 97, 98, 106).

[166] F. T. Liu, K. M. Ting, and Z.-H. Zhou. “Isolation-based anomaly detection.” In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 6.1 (2012), p. 3 (cit. on pp. 45, 116, 119, 120).

[167] N. Lomas. *Amazon Patents "Anticipatory" Shipping — To Start Sending Stuff Before You’ve Bought It*. `https://techcrunch.com/2014/01/18/amazon-pre-ships/`. Online; Accessed: 2016-06-19. Jan. 2014 (cit. on p. 24).

[168] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret. “Network traffic classifier with convolutional and recurrent neural networks for Internet of Things.” In: *IEEE Access* 5 (2017), pp. 18042–18050 (cit. on p. 99).

[169] J. Ma and S. Perkins. “Time-series novelty detection using one-class support vector machines.” In: *Neural Networks, 2003. Proceedings of the International Joint Conference on*. Vol. 3. IEEE. 2003, pp. 1741–1745 (cit. on pp. 98, 106).

[170] A. Madhok. *Global Connected Car Revenues to Grow Five-Fold by 2025*. `https://www.counterpointresearch.com/connected-car-revenues-grow-five-fold-2025/`. 2019 (cit. on p. 21).

[171] A. Mahendran and A. Vedaldi. “Understanding deep image representations by inverting them.” In: *CVPR*. 2015, pp. 5188–5196 (cit. on p. 141).

[172] S. Makridakis, E. Spiliotis, and V. Assimakopoulos. “The M4 Competition: Results, findings, conclusion and way forward.” In: *International Journal of Forecasting* (2018) (cit. on p. 162).

[173] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal. “Long short term memory networks for anomaly detection in time series.” In: *Proceedings*. Vol. 89. Presses universitaires de Louvain. 2015 (cit. on pp. 37, 98).

[174] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal. “Long Short Term Memory Networks for Anomaly Detection in Time Series.” In: *European Symposium on Artificial Neural Networks*. Vol. 23 (cit. on pp. 50, 78, 85).

[175] D. A. Melis and T. Jaakkola. “Towards robust interpretability with self-explaining neural networks.” In: *Advances in Neural Information Processing Systems*. 2018, pp. 7775–7784 (cit. on pp. 139, 141).

[176] Michael Kellner. *Benson Hill Biosystems Launches CropOS*. `t.ly/ktau`. May 2016 (cit. on p. 13).

[177] Microsoft Corporation. *Azure IoT Suite*. `https://www.microsoft.com/en-us/cloud-platform/internet-of-things-azure-iot-suite`. Online; Accessed: 2016-06-22 (cit. on p. 19).

[178] T. Miller. “Explanation in artificial intelligence: Insights from the social sciences.” In: *Artificial Intelligence* 267 (2019), pp. 1–38 (cit. on p. 35).

[179] S. Miyata, J. Lim, Y. Akashi, Y. Kuwahara, and K. Tanaka. “Fault detection and diagnosis for heat source system using convolutional neural network with imaged faulty behavior data.” In: *Science and Technology for the Built Environment* 26.1 (2020), pp. 52–60. DOI: `10.1080/23744731.2019.1651619` (cit. on p. 61).

[180] V. Mullachery, A. Khera, and A. Husain. “Bayesian neural networks.” In: *arXiv preprint arXiv:1801.07710* (2018) (cit. on p. 4).

[181] M. Munir, S. Baumbach, Y. Gu, A. Dengel, and S. Ahmed. "Data Analytics: Industrial Perspective & Solutions for Streaming Data." In: *Data Mining in Time Series and Streaming Databases*. Vol. 83. Series in Machine Perception and Artificial Intelligence. World Scientific, Mar. 2018. Chap. 7, pp. 144–168. DOI: `10.1142/9789813228047_0007` (cit. on pp. 9, 97).

[182] M. Munir, M. A. Chattha, A. Dengel, and S. Ahmed. "A Comparative Analysis of Traditional and Deep Learning-based Anomaly Detection Methods for Streaming Data." In: *18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE. 2019, pp. 561–566. DOI: `10.1109/ICMLA.2019.00105` (cit. on p. 113).

[183] M. Munir, S. Erkel, A. Dengel, and S. Ahmed. "Pattern-based contextual anomaly detection in hvac systems." In: *IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE. 2017, pp. 1066–1073. DOI: `10.1109/ICDMW.2017.150` (cit. on pp. 33, 59).

[184] M. Munir, S. A. Siddiqui, M. A. Chattha, A. Dengel, and S. Ahmed. "FuseAD: unsupervised anomaly detection in streaming sensors data by fusing statistical and deep learning models." In: *Sensors* 19.11 (2019), p. 2451. DOI: `10.3390/s19112451` (cit. on pp. 37, 97, 116, 119, 120, 139).

[185] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed. "DeepAnT: A deep learning approach for unsupervised anomaly detection in time series." In: *IEEE Access* 7 (2018), pp. 1991–2005. DOI: `10.1109/ACCESS.2018.2886457` (cit. on pp. 37, 77, 97, 98, 101, 103, 106, 108, 110, 116, 119, 120, 123, 139, 150).

[186] M. Munir, S. A. Siddiqui, F. Küsters, D. Mercier, A. Dengel, and S. Ahmed. "TSXplain: Demystification of DNN Decisions for Time-Series using Natural Language and Statistical Features." In: *International Conference on Artificial Neural Networks*. Springer. 2019, pp. 426–439. URL: `https://link.springer.com/chapter/10.1007/978-3-030-30493-5_43` (cit. on p. 139).

[187] M. Murphy. *Ford Brakes the Mould with Car Maintenance Prediction Algorithm*. `https://tinyurl.com/m6ky3du`. Online; Accessed: 2016-06-21. Jan. 2015 (cit. on p. 21).

[188] R. M. Neal. "Probabilistic inference using Markov chain Monte Carlo methods." In: (1993) (cit. on p. 124).

[189] M. Nelson, T. Hill, W. Remus, and M. O'Connor. "Time series forecasting using neural networks: Should the data be deseasonalized first?" In: *Journal of forecasting* 18.5 (1999), pp. 359–367 (cit. on p. 162).

[190] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, et al. "Parallel wavenet: Fast high-fidelity speech synthesis." In: *arXiv preprint arXiv:1711.10433* (2017) (cit. on p. 125).

[191] F. Ordóñez and D. Roggen. "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition." In: *Sensors* 16.1 (2016), p. 115 (cit. on p. 139).

[192] P. F. Ovidiu Vermesan. *Internet of Things - Global Technological and Societal Trends From Smart Environments and Spaces to Green ICT*. River Publishers Series in Communications. River Publishers, 2011, p. 16 (cit. on pp. 59, 60, 174).

[193] D. E. Padilla, R. Brinkworth, and M. D. McDonnell. "Performance of a hierarchical temporal memory network in noisy sequence learning." In: *2013 IEEE International Conference on Computational Intelligence and Cybernetics (CYBERNETICSCOM)*. IEEE. 2013, pp. 45–51 (cit. on p. 47).

[194] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos. "LOCI: Fast outlier detection using the local correlation integral." In: *International Conference on Data Engineering*. IEEE. 2003, pp. 315–326 (cit. on pp. 41, 116, 119, 120).

[195] G. Papamakarios, T. Pavlakou, and I. Murray. "Masked autoregressive flow for density estimation." In: *Advances in Neural Information Processing Systems*. 2017, pp. 2338–2347 (cit. on p. 125).

[196] A.-S. K. Pathan. *The state of the art in intrusion prevention and detection*. Auerbach Publications, 2014 (cit. on p. 33).

[197] A. Patrizio. *Top Big Data Companies*. 2020. URL: `https://www.datamation.com/big-data/big-data-companies.html` (cit. on p. 3).

[198] X. Pei. "Emphysema classification using convolutional neural networks." In: *International Conference on Intelligent Robotics and Applications*. Springer. 2015, pp. 455–461 (cit. on p. 123).

[199] H. Pratt, F. Coenen, D. M. Broadbent, S. P. Harding, and Y. Zheng. "Convolutional neural networks for diabetic retinopathy." In: *Procedia Computer Science* 90 (2016), pp. 200–205 (cit. on p. 123).

[200] ProWebScraper. *7 Best Real-Life Example of Data Mining*. 2019. URL: `https://prowebscraper.com/blog/data-mining-examples/` (cit. on p. 3).

[201] S. Ramaswamy, R. Rastogi, and K. Shim. "Efficient algorithms for mining outliers from large data sets." In: *ACM SIGMOD Record*. Vol. 29. 2. ACM. 2000, pp. 427–438 (cit. on pp. 60, 65).

[202] M. T. Ribeiro, S. Singh, and C. Guestrin. “" Why should I trust you?" Explaining the predictions of any classifier.” In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, pp. 1135–1144 (cit. on p. 3).

[203] D. M. Rocke and D. L. Woodruff. “Identification of outliers in multivariate data.” In: *Journal of the American Statistical Association* 91.435 (1996), pp. 1047–1061 (cit. on p. 56).

[204] B. Rosner. “Percentage Points for a Generalized ESD Many-Outlier Procedure.” In: *Technometrics* 25.2 (May 1983), pp. 165–172 (cit. on p. 44).

[205] E. W. Saad and D. C. Wunsch II. “Neural network explanation using inversion.” In: *Neural Networks* 20.1 (2007), pp. 78–93 (cit. on p. 140).

[206] M. Sabokrou, M. Fayyaz, M. Fathy, Z. Moayed, and R. Klette. “Deep-anomaly: Fully convolutional neural network for fast anomaly detection in crowded scenes.” In: *Computer Vision and Image Understanding* 172 (2018), pp. 88–97 (cit. on p. 97).

[207] S. Sabour, N. Frosst, and G. E. Hinton. “Dynamic routing between capsules.” In: *Advances in neural information processing systems*. 2017, pp. 3856–3866 (cit. on pp. 4, 123).

[208] M. Sakurada and T. Yairi. “Anomaly detection using autoencoders with nonlinear dimensionality reduction.” In: *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*. 2014, pp. 4–11 (cit. on p. 51).

[209] T. Salimans, D. P. Kingma, and M. Welling. “Markov chain monte carlo and variational inference: Bridging the gap.” In: *arXiv preprint arXiv:1410.6460* (2014) (cit. on p. 124).

[210] W. Samek, T. Wiegand, and K.-R. Müller. “Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models.” In: *arXiv preprint arXiv:1708.08296* (2017) (cit. on p. 3).

[211] SAP. *Seoul National University Bundang Hospital: Transforming Patient Care and Data Access with SAP HANA*. `http://www.microexcelhealthcare.com/pdf/Seoul-National-University-Case-Study.pdf`. Online; Accessed: 2016-06-15. 2015 (cit. on p. 15).

[212] SAP. *SAP Predictive Maintenance and Service*. `https://www.sap.com/mena/products/predictive-maintenance.html`. Online; Accessed: 2016-06-22 (cit. on p. 19).

[213] SAS Institute. *Manufacturing*. `http://www.sas.com/en_us/industry/manufacturing.html`. Online; Accessed: 2016-06-23 (cit. on p. 21).

[214] M. Schneider, W. Ertel, and F. Ramos. “Expected similarity estimation for large-scale batch and streaming anomaly detection.” In: *Machine Learning* 105.3 (2016), pp. 305–333 (cit. on pp. 91, 98, 108).

[215] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. “Estimating the support of a high-dimensional distribution.” In: *Neural computation* 13.7 (2001), pp. 1443–1471 (cit. on pp. 37, 46, 116).

[216] M. Schreyer, T. Sattarov, D. Borth, A. Dengel, and B. Reimer. “Detection of anomalies in large scale accounting data using deep autoencoder networks.” In: *arXiv preprint arXiv:1709.05254* (2017) (cit. on p. 51).

[217] M. Shepherd. *Is data the new gold?* 2018. URL: `https://www.ceotodaymagazine.com/2018/04/is-data-the-new-gold/` (cit. on p. 3).

[218] K. Shridhar. *A comprehensive guide to Bayesian CNN with variational inference : with implementation in PyTorch*. LAP Lambert Academic Publishing, 2019 (cit. on pp. 123, 124).

[219] K. Shridhar, F. Laumann, A. Llopart Maurin, and M. Liwicki. “Bayesian Convolutional Neural Networks.” In: *arXiv preprint arXiv:1806.05978* (2018) (cit. on pp. 124, 129, 132, 133, 135).

[220] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang. *A novel anomaly detection scheme based on principal component classifier*. Tech. rep. Miami University Coral Gables FL, 2003 (cit. on pp. 46, 98, 106, 116, 119, 120).

[221] S. A. Siddiqui, D. Mercier, M. Munir, A. Dengel, and S. Ahmed. “TSViz: Demystification of deep learning models for time-series analysis.” In: *IEEE Access* 7 (2019), pp. 67027–67040. DOI: `10.1109/ACCESS.2019.2912823` (cit. on pp. 139, 145, 150, 152).

[222] Siemens Healthcare GmbH. *CARE for Patients*. `https://www.siemens-healthineers.com/medical-imaging/low-dose/see-what-siemens-offers` (cit. on p. 15).

[223] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. “Mastering the game of Go without human knowledge.” In: *Nature* 550.7676 (2017), p. 354 (cit. on p. 157).

[224] K. Simonyan, A. Vedaldi, and A. Zisserman. “Deep inside convolutional networks: Visualising image classification models and saliency maps.” In: *arXiv preprint arXiv:1312.6034* (2013) (cit. on pp. 4, 139, 141).

[225] N. Singh and C. Olinsky. “Demystifying Numenta anomaly benchmark.” In: *Neural Networks (IJCNN), 2017 International Joint Conference on*. IEEE. 2017, pp. 1570–1577 (cit. on pp. 90, 91, 108).

[226] S. Singh and A. Majumdar. "Deep sparse coding for non–intrusive load monitoring." In: *IEEE Transactions on Smart Grid* 9.5 (2018), pp. 4669–4678 (cit. on p. 139).

[227] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram, M. Patwary, M. Prabhat, and R. Adams. "Scalable bayesian optimization using deep neural networks." In: *International conference on machine learning*. 2015, pp. 2171–2180 (cit. on pp. 123, 125).

[228] S. Srivastava, I. Labutov, and T. Mitchell. "Joint concept learning and semantic parsing from natural language explanations." In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2017, pp. 1527–1536 (cit. on p. 142).

[229] Statista. *HVAC industry - Statistics Facts*. Apr. 2020. URL: `https://www.statista.com/topics/5225/hvac-industry/` (cit. on p. 59).

[230] S. B. Taieb, G. Bontempi, A. F. Atiya, and A. Sorjamaa. "A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition." In: *Expert systems with applications* 39.8 (2012), pp. 7067–7083 (cit. on pp. 162, 165).

[231] J. Tang, Z. Chen, A. W.-C. Fu, and D. W. Cheung. "Enhancing effectiveness of outlier detections for low density patterns." In: *PAKDD*. Springer. 2002, pp. 535–548 (cit. on pp. 37, 41, 116, 119, 120).

[232] Y. Tang and R. R. Salakhutdinov. "Learning stochastic feedforward neural networks." In: *Advances in Neural Information Processing Systems*. 2013, pp. 530–538 (cit. on p. 125).

[233] T. T. Tchrakian, B. Basu, and M. O'Mahony. "Real-time traffic flow forecasting using spectral analysis." In: *IEEE Transactions on Intelligent Transportation Systems* 13.2 (2012), pp. 519–526 (cit. on p. 83).

[234] N. Tishby and N. Zaslavsky. "Deep learning and the information bottleneck principle." In: *2015 IEEE Information Theory Workshop (ITW)*. IEEE. 2015, pp. 1–5 (cit. on pp. 4, 139).

[235] A. Torralba, R. Fergus, and W. T. Freeman. "80 million tiny images: A large data set for nonparametric object and scene recognition." In: *IEEE transactions on pattern analysis and machine intelligence* 30.11 (2008), pp. 1958–1970 (cit. on p. 130).

[236] G. G. Towell and J. W. Shavlik. "Knowledge-based artificial neural networks." In: *Artificial intelligence* 70.1-2 (1994), pp. 119–165 (cit. on pp. 157, 158).

[237] S. N. Tran and A. S. d. Garcez. "Deep logic networks: Inserting and extracting knowledge from deep belief networks." In: *IEEE transactions on neural networks and learning systems* 29.2 (2018), pp. 246–258 (cit. on p. 159).

[238] R. R. Trippi and E. Turban. *Neural networks in finance and investing: Using artificial intelligence to improve real world performance*. McGraw-Hill, Inc., 1992 (cit. on pp. 98, 139).

[239] J. D. Ullman. *Principles of database and knowledge-base systems*. Vol. 1. Computer Science Press, Incorporated, 1988 (cit. on p. 157).

[240] H. Uzunova, J. Ehrhardt, T. Kepp, and H. Handels. "Interpretable explanations of black box classifiers applied on medical images by meaningful perturbations using variational autoencoders." In: *Medical Imaging 2019: Image Processing*. Vol. 10949. International Society for Optics and Photonics. 2019, p. 1094911 (cit. on p. 140).

[241] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo. "OpenML: Networked Science in Machine Learning." In: *SIGKDD Explorations* 15.2 (2013), pp. 49–60. DOI: `10.1145/2641190.2641198`. URL: `http://doi.acm.org/10.1145/2641190.2641198` (cit. on pp. 54–56, 150, 151).

[242] V. Vapnik and A. Y. Chervonenkis. "A class of algorithms for pattern recognition learning." In: *Avtomat. i Telemekh* 25.6 (1964), pp. 937–945 (cit. on p. 46).

[243] S. Venugopalan, L. A. Hendricks, R. Mooney, and K. Saenko. "Improving LSTM-based video description with linguistic knowledge mined from text." In: *arXiv preprint arXiv:1604.01729* (2016) (cit. on pp. 157, 159).

[244] R. Viereckl, D. Ahlemann, A. Koster, and S. Jursch. *Connected Car Study 2015: Racing ahead with autonomous cars and digital innovation*. `http://www.strategyand.pwc.com/reports/connected-car-2015-study`. Online; Accessed: 2016-06-14. Sept. 2015 (cit. on p. 21).

[245] G. Vilone and L. Longo. "Explainable Artificial Intelligence: a Systematic Review." In: *arXiv preprint arXiv:2006.00093* (2020) (cit. on p. 140).

[246] N. D. K. Vy and D. T. Anh. "Detecting Variable Length Anomaly Patterns in Time Series Data." In: *International Conference on Data Mining and Big Data*. Springer. 2016, pp. 279–287 (cit. on p. 61).

[247] C. Wang, K. Viswanathan, L. Choudur, V. Talwar, W. Satterfield, and K. Schwan. “Statistical techniques for online anomaly detection in data centers.” In: *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*. IEEE. 2011, pp. 385–392 (cit. on pp. 91, 98, 108).

[248] X. Wang, K. Smith, and R. Hyndman. “Characteristic-based clustering for time series data.” In: *Data mining and knowledge Discovery* 13.3 (2006), pp. 335–364 (cit. on p. 146).

[249] D. A. Wangler. “Aktivitätsmessung zur Brunsterkennung – Möglichkeiten und Nutzen.” In: *Landeskontrollverband für Leistungs- und Qualitütsprüfung Sachsen-Anhalt e.V.* (Nov. 2009) (cit. on pp. 24, 26).

[250] P. Whittle. *Hypothesis Testing in Time Series Analysis. Peter Whittle*. Almqvist och Wiksells boktryck, 1951 (cit. on p. 98).

[251] Wikipedia. *The Internet of Things*. `https://psu.pb.unizin.org/ist110/chapter/3-1-applications-of-the-iot/` (cit. on p. 10).

[252] World Economic Forum. *Data is the new gold. This is how it can benefit everyone – while harming no one*. 2020. URL: `t.ly/XH2G` (cit. on p. 3).

[253] A. Wu, S. Nowozin, E. Meeds, R. E. Turner, J. M. Hernández-Lobato, and A. L. Gaunt. “Deterministic variational inference for robust bayesian neural networks.” In: *arXiv preprint arXiv:1810.03958* (2018) (cit. on p. 4).

[254] Y. Wu, W. Wu, Z. Li, and M. Zhou. “Knowledge Enhanced Hybrid Neural Network for Text Matching.” In: *arXiv preprint arXiv:1611.04684* (2016) (cit. on p. 160).

[255] J. Xu, Z. Zhang, T. Friedman, Y. Liang, and G. V. d. Broeck. “A semantic loss function for deep learning with symbolic knowledge.” In: *arXiv preprint arXiv:1711.11157* (2017) (cit. on p. 160).

[256] A. H. Yaacob, I. K. Tan, S. F. Chien, and H. K. Tan. “ARIMA based network anomaly detection.” In: *Communication Software and Networks, 2010. ICCSN'10. Second International Conference on*. IEEE. 2010, pp. 205–209 (cit. on p. 98).

[257] Yahoo! Labs. *Yahoo! Webscope Dataset Ydata-Labeled-Time-Series-Anomalies*. `https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70`. 2010 (cit. on pp. 52, 53).

[258] K. Yamanishi, J.-i. Takeuchi, and G. Williams. “On-line Unsupervised Outlier Detection Using Finite Mixtures with Discounting Learning Algorithms.” In: *PROCEEDINGS OF THE SIXTH ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING*. ACM Press, 2000, pp. 320–324 (cit. on p. 55).

[259] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. "Understanding Neural Networks Through Deep Visualization." In: *Deep Learning Workshop, International Conference on Machine Learning (ICML)*. 2015 (cit. on pp. 4, 139, 141).

[260] Q. Yu, L. Jibin, and L. Jiang. "An improved ARIMA-based traffic anomaly detection algorithm for wireless sensor networks." In: *International Journal of Distributed Sensor Networks* (2016) (cit. on p. 98).

[261] M. D. Zeiler and R. Fergus. "Visualizing and understanding convolutional networks." In: *European conference on computer vision*. Springer. 2014, pp. 818–833 (cit. on pp. 4, 139, 141).

[262] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. "Understanding deep learning requires rethinking generalization." In: *arXiv preprint arXiv:1611.03530* (2016) (cit. on pp. 4, 139, 141).

[263] Z. Zhang, F. S. Yin, J. Liu, W. K. Wong, N. M. Tan, B. H. Lee, J. Cheng, and T. Y. Wong. "Origa-light: An online retinal fundus image database for glaucoma analysis and research." In: *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*. IEEE. 2010, pp. 3065–3068 (cit. on p. 130).

[264] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu. "Convolutional neural networks for time series classification." In: *Journal of Systems Engineering and Electronics* 28.1 (2017), pp. 162–169 (cit. on p. 49).

[265] W. X. Zhao, J. Jiang, J. Weng, J. He, E.-P. Lim, H. Yan, and X. Li. "Comparing twitter and traditional media using topic models." In: *European conference on information retrieval*. Springer. 2011, pp. 338–349 (cit. on p. 160).

[266] Y. Zhao and M. K. Hryniewicki. "XGBOD: improving supervised outlier detection with unsupervised representation learning." In: *IJCNN*. 2018 (cit. on pp. 47, 116, 119, 120).

[267] Y. Zhao, Z. Nasrullah, and Z. Li. "PyOD: A Python Toolbox for Scalable Outlier Detection." In: *Journal of Machine Learning Research* (2019) (cit. on pp. 116, 119, 120).

[268] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao. "Time series classification using multi-channels deep convolutional neural networks." In: *International Conference on Web-Age Information Management*. Springer. 2014, pp. 298–310 (cit. on pp. 49, 79, 99, 123, 139, 150).

[269] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling. "Visualizing deep neural network decisions: Prediction difference analysis." In: *arXiv preprint arXiv:1702.04595* (2017) (cit. on p. 143).

[270] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. “Learning transferable architectures for scalable image recognition.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 8697–8710 (cit. on pp. 4, 123).

[271] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. “Learning Transferable Architectures for Scalable Image Recognition.” In: *(CVPR)*. June 2018 (cit. on p. 157).

INDEX

AD, 83, 85, 94, 100, 102
ADM, 44, 45, 81
AE, 50, 51, 110
AI, 4, 35, 49, 117, 169
AIC, 99
ANN, 48, 94, 96, 133, 134, 145
AR, 42, 43, 48
ARIMA, 6, 43–45, 94–99, 103, 104, 149, 151, 159
ARMA, 43, 94
ATM, 157
AUC, 5, 58, 66–70, 86, 89, 99–102, 109, 111
AWS, 53

BI, 10

CBLOF, 41, 66–69, 166
CDR, 124
CDW, 14, 15
CIF, 157
CNN, v, 6, 49, 61, 74–78, 80, 83, 90, 95–99, 103, 104, 117, 118, 120, 123, 125–127, 135, 142, 143, 153–156, 166
COF, 37, 41, 110, 111
CPC, 53
CPM, 53
CPU, 53
CSV, 64
CV, 124

DBSCAN, 47, 48
DeepAnT, v, 6, 8, 74–77, 80–83, 85, 86, 89, 90, 94, 95, 100, 102, 110, 166, 168
DeepEX, 149, 150, 153–155, 157–160, 168
DNN, v, 3–6, 8, 37, 48, 73, 74, 95, 107, 108, 110–112, 133–139, 143, 149, 150, 155, 157, 165, 167–169
DTW, 49, 61, 65

ECG, 3, 15, 38, 39, 50, 52, 61–63, 66, 68–70, 72, 125
EER, 67
EGADS, 44, 81, 83
ELBO, 118, 121
ERP, 18, 19
ESD, 44

FC, 121, 125, 126
FN, 57
FP, 56, 57
FPR, 57, 58, 67
FuseAD, v, 6, 8, 93–95, 100–104, 110, 151, 166

GE, 15, 19
GPS, 11
GRU, 152

HBOS, 37, 42, 66–72, 110, 112
HTM, 47
HVAC, 5, 6, 8, 28, 52, 59–64, 66–72, 166

ICU, 14
IIoT, 16, 19
INFLO, 41
IoT, 10, 11, 13, 19, 22, 24, 26, 59, 61, 93, 107, 117, 165, 166
IT, 26

KBANN, 150
KEHNN, 152
KINN, 152, 157, 159
KL, 139, 140, 152
kNN, 40, 66, 67, 69, 70, 72, 110–112
KS, 149, 150, 154

LDA, 152
LOCI, 41, 110, 112
LOF, 37, 40, 41, 89, 90, 94, 100, 110, 111
LRD, 41
LSTM, 6, 50, 74, 75, 80, 83, 94, 95, 136, 151, 166

MA, 42, 43, 45
MAD, 44
MAE, 79, 96
MAF, 119
MC-DCNN, 49
MCD, 46
MCMC, 118
MCNN, 49
MLE, 125
MLP, 48–50, 95, 117, 119
mV, 52

NAB, 53, 54, 84–86, 101–103, 109, 112, 169
NNAR, 48
NYC, 54

OC-SVM, 46, 66, 67, 70, 89, 90, 94, 100, 110
OOD, 119

PA, 11
PC, 26, 27
PCA, 46, 51, 94, 100, 110, 112
PPC, 22
PR, 58, 109, 111, 112

RBF, 89
real NVP, 119
ReLU, 77, 78, 142
RFID, 25
RMSE, 95, 151
RNN, 50
ROC, 58, 66, 67, 86, 99–101, 109, 111, 112
rPCA, 46, 66, 67, 70, 166

S-H-ESD, 44
SBN, 119
seq2seq, 150
SGD, 77
SMAPE, 157–159
STL, 153–155
SVM, 46

TMM, 44, 45, 81
TN, 57
TP, 56, 57
TPR, 57, 58, 67
TSViz, v, 134, 137, 138, 167
TSXplain, v, 134, 137, 143, 168, 169

VI, 118

XAI, 35, 169, 170
XGBOD, 47, 110
XGBoost, 46, 47, 61, 110, 111

CURRICULUM VITAE: MOHSIN MUNIR

email mohsin.munir@dfki.de
phone +49 631 20575 1421

EDUCATION

2015-2021 Technische Universität Kaiserslautern, Germany

Specialization: Anomaly Detection, Time-series Analysis, DNN Explainability
Topic: “A Hybrid Framework for Time-series Analysis: From Anomaly Detection to Explainability and Confidence Estimation”

Doctor of Philosophy

2012-2015 Technische Universität Kaiserslautern, Germany

Specialization: Artificial Intelligence, Data Mining, Computer Vision

Master of Computer Science

2006-2010 FAST-NU Lahore, Pakistan

Specialization: Software Engineering

Bachelor of Computer Science

PUBLICATIONS

Conference Publications

Pattern-Based Contextual Anomaly Detection in HVAC Systems. IEEE International Conference on Data Mining Workshops (ICDMW). 18 Nov 2017.

TSXplain: Demystification of DNN Decisions for Time-Series using Natural Language and Statistical Features. International Conference on Artificial Neural Networks (ICANN). 17 Sep 2019.

DeepEX: Bridging the Gap Between Knowledge and Data Driven Techniques for Time Series Forecasting. International Conference on Artificial Neural Networks (ICANN). 17 Sep 2019.

A Comparative Analysis of Traditional and Deep Learning-based Anomaly Detection Methods for Streaming Data. IEEE International Conference on Machine Learning and Applications (ICMLA). 16 Dec 2019.

Journal Publications

DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. IEEE Access. 19 Dec 2018.

TSViz: Demystification of Deep Learning Models for Time-Series Analysis. IEEE Access. 23 Apr 2019.

Confident Classification Using a Hybrid Between Deterministic and Probabilistic Convolutional Neural Networks. IEEE Access. 23 Jun 2020.

FuseAD: Unsupervised Anomaly Detection in Streaming Sensors Data by Fusing Statistical and Deep Learning Models. Sensors. 19 Jan 2019.

Fi-Fo Detector: Figure and Formula Detection Using Deformable Networks. Sensors. 16 Sep 2020.

Book Chapter

Data Analytics: Industrial Perspective Solutions for Streaming Data. Data Mining in Time Series and Streaming Databases; Last, M., Kandel, A., Bunke, H., Eds. 1 Jan 2018.

FELLOWSHIPS AND EXPERIENCE

2015–Present Ph.D. Researcher at Technische Universität Kaiserslautern, Germany and German Research Center for Artificial Intelligence (DFKI)

Ingenieurgesellschaft Auto und Verkehr (IAV)

As a part of the IAV-FLAP, I am supporting IAV in different use-cases in the automotive sector. I applied different state-of-the-art generic peak detection methods on the real data set to detect peaks and trends in the given streaming data. In addition to that, I have analyzed one of their engine data set and applied the latest deep learning methods to predict the efficient engine setting.

Hitachi

In the context of the Smart Factory, several sensors are generating bulk of streaming data. Based on these sensors data, an anomaly detection system was developed to detect if there was an issue in the manufacturing process. With the help of the interactive dashboard, the engineers were also able to know the exact point where an anomalous activity occurred.

Villeroy and Boch

As a proof of concept, I developed an intelligent system that was able to provide early warnings to the engineers regarding the faulty products. After analyzing an individual component produced on a different assembly line, the system predicts if the final product will be faulty or not. By detecting the faulty components in the early stages of production, the system saved the company's material, efforts, and energy.

BMW BigData

As a proof of concept, I followed the standard data analysis procedure to understand the BMW car assembly belt data. After pre-processing, I used deep learning-based methods to predict the error codes for the different kinds of faulty products.

BMBF Proposal

I worked on a feasibility study in which I analyzed the earthquake simulation data provided by the FB-Bauingenieurwesen, TU-KL. The target was to forecast the level of damage an earthquake can cause to a particular type of building. I also worked on proposal writing.

2016–2017 Visiting Researcher

During this research visit, I worked on the Person data collected at Kyushu University. In the context of the Smart City, different poles installed at the campus collected different information about the person passing by. I used that data to predict the crowd anomalies so that an automatic notification to the concerned authorities can be generated for an unexpected event.

Kyushu University, Japan

2014–2015 Internship and MS Thesis

The main task of my internship was to develop a process that automatically transfers field data to a structured format, which is further used for Data Analysis and Data Mining processes. In the thesis, I presented a novel technique to detect anomalies and unwanted trends in operational data (time series) of heating systems.

BOSCH, Germany

2013-2014 Visiting Researcher

The primary goal of the project was to develop an efficient and low-cost eye tracking and gaze estimation system based on computer commodities for a desktop environment. Common eye-tracking system restrictions like static head assumption, lightening conditions and external hardware were aimed to be minimized.

RICOH, Japan

2010-2012 Software Engineer

NetSol is one of the two CMMI level 5 companies in Pakistan. I worked on an Administrator tool (desktop application) for an insurance domain USA-based company. My role in the team was to code component level and application level services and to hook application services from UI. It was also part of my job to create and maintain the specification documents of these components and application services.

NetSol Technologies, Pakistan