Explainable AI for Log-Based Anomaly Detection in Security Monitoring: Reasoning Pipelines and Cross-Dataset Evaluation.

University Of Oulu
Faculty of Information Technology and Electrical Engineering
Master's Thesis

Emmanuel Ikwunna
13th November 2025

Abstract

Abstract is needed to sum the master's thesis up. The abstract is to be uploaded into Optima before the final grading of the thesis. Please find the current information about the format given in Optima.

The guide includes instructions for students. It is written keeping in mind the idea that the user may utilise it e.g. by pasting his or her text on the current text. The contents include information about formatting the text, positioning tables and figures, among other things. In addition, the use of proper literature is instructed. Even if there is no strict structure for the thesis, a recommendation is offered in this guideline.

One important guideline for the text is that do not write too short paragraphs. For instance, if there is only one sentence in a paragraph, the sentence must be really important and influential to form a paragraph of its own.

It is not possible to provide information in a guideline like this for all issues related to master's thesis. For example, the research process, ways to acquire research material and its analysis are excluded in the guideline. On the other hand, a structure for a research plan is provided in the appendices.

Keywords
first keyword, second keyword, other keywords

Supervisor
Title, position First name Last name

Foreword

The foreword is not instructed by the supervisors. In other words, the student may write in this section what she or he wants to share with readers. However, it is a custom to thank all those who have contributed to the research somehow. When acknowledging people, their affiliations are given (e.g. Professor, University Lecturer, Adjunct Professor, Mrs.) This guideline is based on the previous version that was written in Finnish and finalised by Dr. Lasse Harjumaa in January 2007. This version is to replace the earlier version. I want to thank all those people who have contributed to the earlier versions and this newest version, the first written in English. Hopefully this guideline will serve both students and faculty with its instructions that include both formal and informal regulations and recommendations. In the first phase, the constructive comments are received with pleasure by raija.halonen@oulu.fi. Oulu, January 10, 2011

Raija Halonen Oulu, March 10, 2020

Contents

1 Introduction

In the thesis we follow the style introduced by The American Psychological Association (APA). The APA style can be found easily in the Internet and some sites provide a quick guide, too. E.g. `http://www.waikato.ac.nz/library/learning/g_apaguide.shtml` and `http://owl.english.purdue.edu/owl/resource/560/01/` are useful links.

It is important to follow given instructions. In academic theses, not only the content but also the format is important. Generally every academic publication forum requires that the publications follow their guidelines. In the theses accepted in the Department of Information Processing Science the format is APA. Currently there are several editions published from APA. The general rule is that the latest available edition is applied. Currently the newest edition is 6th. If a thesis is already in process it is not needed to transfer it into a newer edition of APA. Whichever you apply, do it consistently.

In addition to teach the students to follow given formal instructions, the guideline aims to unify and standardise the outlook of the theses made in the department. The guideline also enables the supervisors to focus on the content of the theses as the students already consider the outlook and format themselves. In this sense, it is a question of available resources for supervision and guidance.

The use of language and grammar cannot be discussed in detail in this kind of guide. However, the writing style should meet the general academic writing styles in the sense that no causeries are accepted or other lightweight texts such as jokes or rumblings. In other words, in academic theses all writing must be appropriate and reasonable. There are several guidebooks for academic writing available in the Oulu University Library, for example, and in the Internet. For those who write their thesis in Finnish there are books such as Tieteellinen kirjoittaminen. The style reference by APA (American Psychological Association, 2010) offers fruitful practical hints for writing thesis in English.

As the guideline is written according to the instructions, it enables the students to copy their text (without format) on the document and thus get their text into the right format. The format is to be used in the Bachelor's Theses and in the Master's Theses. In case of other theses, essays or reports it is recommended that the students inquire their teachers if the guideline is to be followed or not.

The structure of the guideline is as follows. The formal instructions for different topics are presented next. This is followed by examples of references and their use. After that the structure of theses and its writing style is discussed briefly. The guideline ends with a summary.

2 Background

2.1 Log-Based Anomaly Detection

Log files contain fundamental information for monitoring the stability and security of networked systems; the logs are generated ubiquitously [1, 2]. Logs are valuable because they provide detailed and chronological records of system events, which can include warnings, errors, or information on system changes, as well as user intentions.[3, 4]. As system complexity continues to grow, these logs have become a very important asset for operations such as performance monitoring, security auditing, transaction tracing, and fault diagnosis [5]. The primary purpose of log anomaly detection is to protect digital infrastructures by identifying abnormal activities, such as network intrusions, from the enormous volumes of event logs. In this context, anomalies represent log patterns that significantly deviate from the expected behavior of the system. Detecting these deviations is crucial for maintaining system reliability and preventing severe disruptions or financial losses, as global cybercrime costs are estimated to reach trillions of euros annually [6].

Log analysis, and consequently anomaly detection, faces several significant challenges primarily driven by the nature and scale of the data. First, the volume of system logs is enormous, as they are collected in real time [7]. The sheer volume of logs has grown rapidly, often reaching 50 GB (120–200 million lines) per hour for large-scale services, making manual inspection and traditional processing infeasible [4]. Second, the variety and complexity of logs further complicate analysis. Logs are typically unstructured or semi-structured text files generated by logging statements in source code [2]. Because developers are allowed to write free-text messages, the format and semantics of logs vary significantly across systems, leading to high-dimensional features with complex interrelationships. This complexity and diversity increase the difficulty of accurate anomaly detection [8]. Finally, for anomaly detection to be useful, it must be timely, requiring decisions to be made in a streaming fashion to allow users to intervene in ongoing attacks or performance issues. Offline methods that require multiple passes over the entire log data are unsuitable for real-time security monitoring [2]. Due to the challenges of volume and complexity, the adoption of automated log analysis has become imperative to efficiently process and interpret vast corpora of logs.

2.1.1 Traditional Methods

Early log anomaly detection efforts relied heavily on human expertise [5]. As the volume of logs grew, research shifted toward automated, data-driven methods, broadly categorized into rule-based systems and statistical approaches, many of which depend on logs first being converted into a structured format through a process known as log parsing [8]. Log parsing is a critical precursor step where raw, unstructured log messages are transformed into structured data, typically by extracting a constant part, called the log template (or log key), and identifying the variable parts (parameters). The parser Spell, for example, is an online streaming parser that utilizes the Longest Common Subsequence (LCS) technique to dynamically identify and update log patterns. Tools like DeepLog rely on log parsing methods like Spell to generate log templates for their inputs [2, 9].

Rule-based methodologies were among the first attempts to automate log analysis to reduce human error. These methods typically rely on explicitly defined rules, patterns, or known indicators of abnormal behavior, often requiring specific domain knowledge from human experts. Early rule-based systems focused on matching specific keywords (e.g.,

”error,” ”failed”) or using regular expressions to flag anomalous log entries [10]. However, relying solely on keywords or structural features often prevents a large portion of log anomalies from being detected and can lead to unnecessary alarms (alarm fatigue) if the system constantly evolves [10, 11]. Furthermore, manually designing and maintaining regular expressions is prohibitive given the rapid increase in log volume and frequent system updates [8]. Invariant mining is another traditional approach that captures co-occurrence patterns between different log keys [2, 11]. This method defines a window (time or session based) and detects whether certain mined quantitative relationships, or invariants, hold true within that window (e.g., ensuring that the count of ”file open” logs equals the count of ”file close” logs in a normal condition) [11]. IM is typically characterized as an unsupervised offline method [2].

2.1.1.1 Statistical Methods (Clustering, PCA)

Statistical methods leverage mathematical principles to identify normal patterns and flag deviations statistically likely to be anomalous. These methods typically operate on numeric vector representations of logs, often using only log keys and their counts rather than parameter values [2]. Most rely on initial log parsing to convert raw messages into structured, numeric representations such as event count vectors [12, 2].

PCA is a linear transformation technique that converts correlated variables into uncorrelated principal components [7, 13]. In log analysis, PCA operates at the session level, where log entries are grouped by identifiers (e.g., block_id in HDFS) and converted into event count vectors recording log key frequencies [12]. It then projects this high-dimensional counting matrix into a lower-dimensional space by identifying components capturing the most variance [5]. PCA produces a normal space (S_n) from the first k principal components and an anomaly space (S_a) from the remaining dimensions. Anomalies are detected by measuring each session’s projection length onto S_a using the Squared Prediction Error (SPE) [12, 10, 2].

Clustering algorithms group similar data instances to identify patterns or outliers [14, 13]. LogCluster is an unsupervised method that groups textually similar log messages using IDF-scaled vectorization to build a knowledge base of normal and abnormal clusters [12, 5]. Sequences are classified as anomalous if their distance to the nearest cluster centroid exceeds a threshold [5]. Density-based approaches like SLCT group frequently occurring messages into clusters representing common patterns, treating entries outside these clusters as potential anomalies [15]. HDBSCAN extends this approach and is sometimes used to provide pseudo-labels for semi-supervised learning [5].

2.1.2 Machine Learning Approaches

Following traditional statistical and rule-based methods, log anomaly detection quickly adopted general Machine Learning (ML) algorithms. These approaches rely heavily on the preceding log parsing and vectorization steps to transform unstructured log files into numerical features [1]. Traditional ML techniques are generally categorized into supervised models, which require labeled data, and unsupervised models, which are crucial given that log data is overwhelmingly unlabeled in real-world scenarios [10, 7].

Traditional machine learning algorithms require structured, vectorized representations of logs, typically event count matrices derived from parsed log templates. Support Vector Machines (SVM) is a flexible model capable of classification and outlier detection. One-Class SVM (OCSVM) variant is trained only on normal data to identify anomalies

that fall outside a learned boundary [16, 13]. Random Forests (RF) is an ensemble method that builds multiple decision trees on random subsets of features and data. RF models are effective with minimal parameter tuning and often outperform other classifiers in log anomaly detection [17, 18]. Isolation Forest (iForest) is a tree-based unsupervised algorithm that isolates observations distinct from the remaining data. Unlike clustering methods that seek dense regions, iForest forms an ensemble of decision trees where outliers are isolated more quickly than normal instances [19, 7, 13].

2.1.3 Deep Learning Models

Deep learning approaches use neural networks to automatically learn representations from log sequences, treating logs analogously to natural language. These models typically follow a pipeline of preprocessing, parsing, vectorization, and neural network classification [20]. DeepLog, proposed by Du et al. (2017), uses Long Short-Term Memory networks to learn normal execution patterns through a forecasting-based self-supervised approach. The model predicts the probability distribution of the next log key given a history of preceding keys derived from parsing tools like Spell [2, 9, 10]. An incoming log key is classified as normal if it falls within the top g candidates with the highest predicted probabilities, enabling detection at the granular per-log-entry level [2, 5]. LogBERT, proposed by Guo et al. (2021), applies Bidirectional Encoder Representations from Transformers (BERT) to log anomaly detection. It encodes each log key using bidirectional self-attention, capturing dependencies across entire sequences [21]. LogBERT is trained solely on normal log data using two self-supervised objectives: Masked Log Key Prediction (MLKP), which predicts masked log keys to learn normal patterns, and Volume of Hypersphere Minimization (VHM), which encourages embeddings of normal sequences to cluster closely in latent space [5, 7, 21]. This approach consistently outperforms LSTM-based methods due to BERT's stronger contextual understanding [7].

Attention mechanisms, introduced by Vaswani et al. (2017) in *"Attention Is All You Need"*, enable models to dynamically weigh the importance of different elements within input sequences [22]. In Transformer models like LogBERT, multi-head self-attention enables each log key to attend to every other key, producing context-aware embeddings that capture semantic and sequential relationships [21]. Beyond Transformers, attention has been integrated into LSTM-based models like LogRobust for improved robustness, and hybrid architectures like LogCTBL that combine convolutional, recurrent, and attention layers [10, 5]. Attention weights can also provide interpretability by revealing which tokens influenced predictions, though their reliability as explanatory tools remains debated [23].

Deep learning models consistently outperform traditional statistical methods and classical ML approaches, with superior F1 scores attributed to their ability to learn semantic embeddings and model complex sequential structures [1]. Table X compares representative models across benchmark datasets.

Model	Architecture Type	Log Parser Dep.	HDFS F1	BGL F1	T-bird F1
LogCTBL [5]	Hybrid (CNN-TCN-Bi-LSTM + BERT)	Yes (Drain3)	-	0.9987	0.9978
LAnoBERT [7]	Transformer (BERT-MLM, Parser-free)	No	0.9645	0.8749	0.9990
OneLog [20]	End-to-End HCNN (Character-based)	No	0.9900	0.9900	0.9900
LogFiT [1]	Transformer (Fine-tuned BERT)	No	0.9497	0.9122	0.9414
DeepLog [2]	LSTM-based (Forecasting)	Yes (Spell)	0.7734	0.8612	0.9308
LogBERT [21]	Transformer (BERT-MLKP/VHM)	Yes (Drain)	0.8232	0.9083	0.9664
LogFormer [4]	Transformer (Pre-train/Tuning)	Yes (Drain)	0.9800	0.9700	0.9900

Table 1: Comparison of representative log anomaly detection models.

2.2 Explainable AI (XAI) Techniques

Explainable Artificial Intelligence (XAI) encompasses methods that help users interpret and understand how machine learning models make their decisions, thereby improving transparency [24, 20]. The goal of XAI is to build comprehension regarding the influences on a model, how that influence occurs, and where the model succeeds or fails [24]. The need for explainability is paramount in high-stake security applications such as anomaly detection, vulnerability detection, and malware detection [25, 20]. Deep learning models, while achieving high accuracy in security domains, often function as black boxes, obscuring the features and factors that lead to their predictions. This opacity leads to uncertainty and distrust, especially when an incorrect prediction carries severe consequences [25].

A fundamental challenge in implementing AI, particularly in high-stakes domains, is managing the tension between model accuracy and interpretability [26]. Deep learning models typically achieve the highest accuracy on complex, large datasets, but their complexity makes their decisions difficult to understand [26]. Conversely, highly interpretable models, such as decision trees, may be easy for humans to understand but often struggle to match the performance of complex black-box models, especially when dealing with unstructured raw data like system logs [25]. Consequently, achieving a usable XAI solution requires finding a balance where the interpretability gained justifies the complexity introduced. Developers must choose between designing intrinsically interpretable models (ante-hoc explanation) or applying post-hoc explanation methods to analyze the decisions of highly accurate black-box models.

2.2.1 SHAP and Attention Mechanisms

SHapley Additive exPlanations (SHAP) is a post-hoc method that generates local explanations for the predictions of any ML model [25] by viewing the explanation process through the lens of game theory, specifically utilizing Shapley values [26]. SHAP introduces a group of methods that explain a model's prediction by adding up the effects of each feature, in which the explanation model is a linear function of binary variables indicating feature presence or absence [24, 26]. SHAP values uniquely satisfy desirable properties: local accuracy (the explanation model matches the original

prediction), missingness (features missing in the input have no impact), and consistency. SHAP assigns an importance value (the Shapley value) to each feature, quantifying its contribution to the model's output by averaging its marginal contribution across all possible subsets of other features [26]. Although SHAP is computationally slower than LIME, its foundation in game-theoretic principles generally leads to more robust explanations [25]. Deep SHAP is a model-specific approximation method designed to leverage the compositional nature of deep networks to rapidly estimate SHAP values [26].

Transformer models rely on attention mechanisms that allow models to selectively focus on different parts of the input sequence to compute representations [25, 24]. The attention function maps a query and a set of key-value pairs to an output, calculated as a weighted sum of values, where weights are derived from the compatibility score between the query and corresponding key [22, 13]. These attention weights can be leveraged as a form of explanation by quantifying the importance assigned by the model. When used for explainability, the magnitude of the attention weight assigned to a specific token or segment indicates its perceived significance to the model's prediction [24].

Several tools have been developed to map attention scores back onto input data. BERTViz [27] is an open-source tool for visualizing multi-head self-attention in BERT-based models, providing views at the attention-head, model, and neuron levels [28, 27]. Other visualizations, such as Attention-Viz, highlight relationships between query and key embeddings for global analysis of patterns across sequences [22]. In log analysis using Transformer-based models, attention visualization can show which tokens or templates receive high attention weights during anomaly detection tasks [25]. The use of attention weights as explanations is controversial, with researchers debating their faithfulness and stability [25]. The argument that ”Attention is not Explanation” [23] claims that if alternative sets of attention weights can be found that produce near-identical model predictions, then the original attention distribution cannot be the exclusive, faithful explanation for that prediction [23]. If the goal is to achieve explainability by providing a reasonable justification for a model's prediction, the presence of alternative explanations does not necessarily reduce the value of the original one.

2.2.2 Brief Overview of LIME

LIME (Local Interpretable Model-agnostic Explanations) provides explanations for individual model predictions by learning a simple, interpretable surrogate model locally around the specific prediction instance [29]. The core process involves perturbing the input instance to generate new data samples, obtaining predictions from the black-box model for these perturbed samples, weighting them by their proximity to the original instance, and training a simple, interpretable model (often sparse linear regression) on the weighted samples. The weights of this local surrogate model serve as the explanation, indicating the degree to which each input feature contributes to the prediction. LIME aims to minimize a function that balances local fidelity (how accurately the simple model approximates the black-box prediction locally) and the simplicity of the explanation [24]. While LIME is efficient, its explanations can sometimes be unreliable or susceptible to adversarial manipulation, partly because it assumes feature independence [29].

Feature	LIME	SHAP
Methodology	Perturbation-based local approximation.	Game-theoretic (Shapley values); additive attribution.
Scope	Local explanation for a single instance.	Local explanation, but derived from a framework designed for global coherence.
Model Dependency	Model-agnostic.	Model-agnostic (using approximations).
Theoretical Basis	Minimizes local fidelity and complexity loss.	Unique solution satisfying local accuracy, missingness, and consistency.
Computational Cost	Generally faster for local explanations.	Computationally slower (though approximations exist).

Table 2: Comparison of LIME and SHAP methodologies.

2.2.3 Application to Security Domains

XAI techniques are applied in security domains such as anomaly detection using system logs. Modern log anomaly detection models like LogFiT and LAnoBERT often use fine-tuned, pre-trained BERT-based language models to learn the linguistic structure and sequential patterns of normal log data [7, 30]. In this context, explanations serve to validate alerts and understand why a sequence of logs was flagged as anomalous. LIME and SHAP are used to assign relevance scores to individual log events within an anomalous sequence, helping analysts evaluate model-generated alarms. DeepAID takes a contrastive approach by identifying the nearest normal sample corresponding to the anomalous input, highlighting the specific event that caused the anomalous prediction by showing differences between the anomaly and the reference normal log [31]. In hybrid systems, specialized models extract attention weights from Transformer layers to visualize which log tokens or sequences were most critical to the anomaly prediction [32].

HuntGPT, an intrusion detection dashboard, integrates a Random Forest classifier with SHAP and LIME to explain predictions, coupled with a GPT-3.5 Turbo conversational agent to deliver insights in natural language format [18]. Similarly, the AnomalyExplainerBot framework uses conversational AI along with visualization tools like BERTViz and Captum (which provides feature attribution) to explain LLM-based anomaly detection decisions on log data [28].

References

[1] Crispin Almodovar, Fariza Sabrina, Sarvnaz Karimi and Salahuddin Azad. ‘Can Language Models Help in System Security? Investigating Log Anomaly Detection using BERT’. In: *Proceedings of the 20th Annual Workshop of the Australasian Language Technology Association*. Ed. by Pradeesh Parameswaran, Jennifer Biggs and David Powers. Adelaide, Australia: Australasian Language Technology Association, Dec. 2022, pp. 139–147. URL: https://aclanthology.org/2022.alta-1.19/.

[2] Min Du, Feifei Li, Guineng Zheng and Vivek Srikumar. ‘DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning’. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. CCS ’17. Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 1285–1298. ISBN: 9781450349468. DOI: 10.1145/3133956.3134015. URL: https://doi.org/10.1145/3133956.3134015.

[3] Amazon Web Services. *What are Log Files? - Log Files Explained - AWS*. https://aws.amazon.com/what-is/log-files/. Accessed: 2025-11-11. 2025.

[4] Hongcheng Guo, Jian Yang, Jiaheng Liu, Jiaqi Bai, Boyang Wang, Zhoujun Li, Tieqiao Zheng, Bo Zhang, Junran peng and Qi Tian. *LogFormer: A Pre-train and Tuning Pipeline for Log Anomaly Detection*. 2024. arXiv: 2401.04749 [cs.LG]. URL: https://arxiv.org/abs/2401.04749.

[5] Hong Huang, Wengang Luo, Yunfei Wang, Yinghang Zhou and Weitao Huang. ‘LogCTBL: a hybrid deep learning model for log-based anomaly detection’. In: *The Journal of Supercomputing* 81 (Jan. 2025). DOI: 10.1007/s11227-025-06926-3.

[6] European Parliament. *Cybercrime in the EU: Threats, Trends and Policy Responses*. Tech. rep. According to an EU briefing, the annual global cost of cybercrime was estimated at approximately €5.5 trillion in recent years. Accessed November 9, 2025. European Parliamentary Research Service, 2024. URL: https://www.europarl.europa.eu/RegData/etudes/BRIE/2024/760356/EPRS_BRI(2024)760356_EN.pdf.

[7] Yukyung Lee, Jina Kim and Pilsung Kang. ‘LAnoBERT: System log anomaly detection based on BERT masked language model’. In: *Applied Soft Computing* 146 (2023), p. 110689. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2023.110689. URL: https://www.sciencedirect.com/science/article/pii/S156849462300707X.

[8] Pinjia He, Jieming Zhu, Zibin Zheng and Michael R. Lyu. ‘Drain: An Online Log Parsing Approach with Fixed Depth Tree’. In: *2017 IEEE International Conference on Web Services (ICWS)*. 2017, pp. 33–40. DOI: 10.1109/ICWS.2017.13.

[9] Min Du and Feifei Li. ‘Spell: Streaming Parsing of System Event Logs’. In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. 2016, pp. 859–864. DOI: 10.1109/ICDM.2016.0103.

[10] Harold Ott, Jasmin Bogatinovski, Alexander Acker, Sasho Nedelkoski and Odej Kao. *Robust and Transferable Anomaly Detection in Log Data using Pre-Trained Language Models*. 2021. arXiv: 2102.11570 [cs.AI]. URL: https://arxiv.org/abs/2102.11570.

[11] Weibin Meng, Ying Liu, Yichen Zhu, Shenglin Zhang, Dan Pei, Yuqing Liu, Yihao Chen, Ruizhi Zhang, Shimin Tao, Pei Sun and Rong Zhou. 'LogAnomaly: Unsupervised Detection of Sequential and Quantitative Anomalies in Unstructured Logs'. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, July 2019, pp. 4739–4745. DOI: 10.24963/ijcai.2019/658. URL: https://doi.org/10.24963/ijcai.2019/658.

[12] Shilin He, Jieming Zhu, Pinjia He and Michael R. Lyu. 'Experience Report: System Log Analysis for Anomaly Detection'. In: *2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE)*. 2016, pp. 207–218. DOI: 10.1109/ISSRE.2016.21.

[13] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. 2nd. Sebastopol, CA: O'Reilly Media, 2019. ISBN: 978-1-492-03264-9. URL: https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/.

[14] Jay Alammar and Maarten Grootendorst. *Hands-On Large Language Models: Language Understanding and Generation*. Sebastopol, CA: O'Reilly Media, 2024. ISBN: 978-1-492-08828-3. URL: https://www.oreilly.com/library/view/hands-on-large-language/9781492088283/.

[15] Risto Vaarandi. 'Mining event logs with SLCT and LogHound'. In: May 2008, pp. 1071–1074. DOI: 10.1109/NOMS.2008.4575281.

[16] Mennatallah Amer, Markus Goldstein and Slim Abdennadher. 'Enhancing one-class Support Vector Machines for unsupervised anomaly detection'. In: Aug. 2013, pp. 8–15. DOI: 10.1145/2500853.2500857.

[17] Andreas C. Müller and Sarah Guido. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. Sebastopol, CA: O'Reilly Media, 2016. ISBN: 978-1449369415.

[18] Tarek Ali. 'Next-Generation Intrusion Detection Systems with LLMs: Real-Time Anomaly Detection, Explainable AI, and Adaptive Data Generation'. Master's thesis. Oulu, Finland: Faculty of Information Technology and Electrical Engineering, University of Oulu, 2025.

[19] Dong Xu, Yanjun Wang, Yulong Meng and Ziying Zhang. 'An Improved Data Anomaly Detection Method Based on Isolation Forest'. In: Dec. 2017, pp. 287–291. DOI: 10.1109/ISCID.2017.202.

[20] Sayedshayan Hashemi Hosseinabad. 'Data-Driven Software System Log Anomaly Detection'. Acta Univ. Oul. A 808. Doctoral dissertation. Oulu, Finland: University of Oulu, Faculty of Information Technology and Electrical Engineering, 2025. ISBN: 978-952-62-4502-7.

[21] Haixuan Guo, Shuhan Yuan and Xintao Wu. *LogBERT: Log Anomaly Detection via BERT*. 2021. arXiv: 2103.04475 [cs.CR]. URL: https://arxiv.org/abs/2103.04475.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[23] Sarah Wiegreffe and Yuval Pinter. *Attention is not not Explanation*. 2019. arXiv: 1908.04626 [cs.CL]. URL: https://arxiv.org/abs/1908.04626.

[24] Melkamu Mersha, Khang Lam, Joseph Wood, Ali K. AlShami and Jugal Kalita. 'Explainable artificial intelligence: A survey of needs, techniques, applications, and future direction'. In: *Neurocomputing* 599 (Sept. 2024), p. 128111. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2024.128111. URL: http://dx.doi.org/10.1016/j.neucom.2024.128111.

[25] Dipkamal Bhusal, Rosalyn Shin, Ajay Ashok Shewale, Monish Kumar Manikya Veerabhadran, Michael Clifford, Sara Rampazzi and Nidhi Rastogi. 'SoK: Modeling Explainability in Security Analytics for Interpretability, Trustworthiness, and Usability'. In: *Proceedings of the 18th International Conference on Availability, Reliability and Security*. ARES 2023. ACM, Aug. 2023, pp. 1–12. DOI: 10.1145/3600160.3600193. URL: http://dx.doi.org/10.1145/3600160.3600193.

[26] Scott Lundberg and Su-In Lee. *A Unified Approach to Interpreting Model Predictions*. 2017. arXiv: 1705.07874 [cs.AI]. URL: https://arxiv.org/abs/1705.07874.

[27] Jesse Vig. 'BertViz: A Tool for Visualizing Multi-Head Self-Attention in the BERT Model'. In: (May 2019).

[28] Prasasthy Balasubramanian, Dumindu Kankanamge, Ekaterina Gilman and Mourad Oussalah. *AnomalyExplainer Explainable AI for LLM-based anomaly detection using BERTViz and Captum*. 2025. arXiv: 2509.00069 [cs.LG]. URL: https://arxiv.org/abs/2509.00069.

[29] Marco Tulio Ribeiro, Sameer Singh and Carlos Guestrin. *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. 2016. arXiv: 1602.04938 [cs.LG]. URL: https://arxiv.org/abs/1602.04938.

[30] Crispin Almodovar, Fariza Sabrina, Sarvnaz Karimi and Salahuddin Azad. 'LogFiT: Log Anomaly Detection Using Fine-Tuned Language Models'. In: *IEEE Transactions on Network and Service Management* 21.2 (2024), pp. 1715–1723. DOI: 10.1109/TNSM.2024.3358730.

[31] Dongqi Han, Zhiliang Wang, Wenqi Chen, Ying Zhong, Su Wang, Han Zhang, Jiahai Yang, Xingang Shi and Xia Yin. 'DeepAID: Interpreting and Improving Deep Learning-based Anomaly Detection in Security Applications'. In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. CCS '21. ACM, Nov. 2021, pp. 3197–3217. DOI: 10.1145/3460120.3484589. URL: http://dx.doi.org/10.1145/3460120.3484589.

[32] Runqiang Zang, Hongcheng Guo, Jian Yang, Jiaheng Liu, Zhoujun Li, Tieqiao Zheng, Xu Shi, Liangfan Zheng and Bo Zhang. *MLAD: A Unified Model for Multi-system Log Anomaly Detection*. 2024. arXiv: 2401.07655 [cs.SE]. URL: https://arxiv.org/abs/2401.07655.

[33] Chiung Ko, Jintaek Kang, Chaejun Lim, Donggeun Kim and Minwoo Lee. 'Application of Machine Learning Models in the Estimation of Quercus mongolica Stem Profiles'. In: *Forests* 16.7 (2025). ISSN: 1999-4907. DOI: 10.3390/f16071138. URL: https://www.mdpi.com/1999-4907/16/7/1138.

A Structure for the research plan

A research plan can be reported according to the next structure. The order of the items is important.

Introduction

The topic is introduced on general level. The context of the research is described and the research problem is explained and justified. The problem is situated in its larger environment. Note references when needed. The researcher may reason the topic also by describing his or her personal motivation.

Research problem and research methods

The problem under study is explained as explicitly as possible. The research problem can be divided into sub problems or presented as hypotheses. The research methods and analysis are described.

Limitations

The planned limitations and known shortcomings are reported. The reasons for them – if known – are explained from the viewpoint of the current research.

Preliminary earlier research

The prior literature is presented briefly with full sentences. All required references are included. Its relevance in the current research is described and limitations recognised in prior research are identified if possible. List of main prior literature in relation to the background theory Main background references are listed in the required format (APA).

Lähteet

Timetable

A plan to describe the planned research related to calendar time. It is recommended that the plan is discussed with supervisor to ensure enough milestones for checking thoroughly the status of the thesis.

Preliminary structure of contents

1. Introduction
2. Glossary
3. Prior research
   (a) First
   (b) Second
   Subsecond
4. Sources