People's Democratic Republic of Algeria

Larbi Tebessi University

Faculty of exact sciences and natural and life sciences

Department: Mathematics and Computer Science

MASTER’s dissertation

Field: Mathematics and Computer Science

Sector: Computer science

Option: Information systems

LLM Based Approach for Anomaly Detection in Smart Grids

Author:

GATTAL Rogaia

Supervisors:

Dr.khediri Abderrazak Thesis supervisor

Dr.YAHIAOUI Ayoub Co-supervisor

defended on 10/06/2024 in front of the jury composed of:

Dr. Bourougaa Salima President (MCA)

Dr. Slimi hamda Examiner (MCB)

2023/ 2024

Dedication

بسم الله الرحمن الرحيم

I dedicate this modest work as a sign of respect, recognition, and thanks

TO MY DEAR PARENTS

Who have always been by my side and always supported me throughout these long years of study. As a sign of recognition, may they find here the expression of my deep gratitude for all the efforts and resources they have made to see me succeed in my studies.

TO MY DEAR BROTHERS

That they were always present for me.

Acknowledgment

First of all, I would like to thank ALLAH for giving me the courage and patience to accomplish this work.

I would like to thank all the people who, directly or indirectly, contributed to the accomplishment of this work.

I would particularly like to thank Dr.Khediri Abderrazak and Dr.Yahiaoui Ayoub for supervising this dissertation and for the trust they placed in me, their help, their support, their presence, and their valuable advice.

A big thank you to all my family, especially my parents, who helped me to follow my studies in the best conditions and who have always supported and encouraged me without limit.

Finally, a big thank you to our teachers for the quality of the teaching that they were kind enough to provide us with during our studies in order to provide us with efficient training.

Thanks to you all.

Abstract

The emergence of smart grids has revolutionized the power sector, fostering interconnected and intelligent electricity networks. These advancements offer enhanced monitoring and control capabilities, but also introduce challenges in anomaly detection, a critical aspect for grid stability and security. Traditional methods, while effective at identifying anomalies, often lack interpretability, making it difficult to understand the root cause and formulate an effective response. Here, Large Language Models (LLMs) offer a promising approach. By generating human-friendly descriptions of anomalies, LLMs can bridge the gap between raw data and actionable insights.

This dissertation investigates the potential of LLMs for anomaly detection in smart grids. We leverage the CICIDS2017 dataset to explore how LLMs can be harnessed to improve the interpretability of anomaly detection systems. Specifically, we examine the effectiveness of using SHAP (SHapley Additive exPlanations) values to guide the LLM towards generating more accurate descriptions of the detected anomalies. Notably, our research demonstrates that even with minimal fine-tuning, the Llama3 8B LLM achieves remarkable results when prompted effectively. This highlights the crucial role of prompt engineering in unlocking the full potential of LLMs for this domain. By incorporating SHAP values within our prompting strategy, we are able to bridge the gap between anomaly detection and actionable insights. This empowers decision-making experts with valuable information to respond to anomalies effectively, ensuring the continued reliability of smart grid operations.

Keywords: Smart Grids, Smart City, Machine Learning, Large Language Models, Decision Support Systems, Cyber Security, CICIDS2017, Llama3, SHAP values.

Résumé

L'émergence des réseaux électriques intelligents a révolutionné le secteur de l'énergie, créant des réseaux électriques interconnectés et intelligents. Ces progrès facilitent la surveillance et le contrôle, mais posent également des défis en matière de détection des anomalies, essentielle pour la stabilité et la sécurité du réseau. Si efficaces pour identifier les anomalies, les méthodes traditionnelles manquent souvent d'interprétabilité, compliquant la compréhension de la cause première et la formulation d'une réponse efficace.

En générant des descriptions compréhensibles des anomalies, les modèles de langage volumineux (LLM) offrent une approche prometteuse. Cette étude explore leur potentiel pour la détection des anomalies dans les réseaux électriques intelligents. Nous utilisons le jeu de données CICIDS2017 pour analyser comment les LLM peuvent améliorer l'interprétabilité des systèmes de détection. Plus précisément, on examine l'efficacité des valeurs SHAP pour guider le LLM vers des descriptions plus précises. Nos recherches démontrent que même avec très peu de fine-tuning, le Llama3 8B LLM obtient des résultats remarquables avec un prompt efficace. Cela souligne le rôle crucial de l'ingénierie des invites (prompt engineering) pour libérer le plein potentiel des LLM. En intégrant SHAP dans notre stratégie d'invite, on comble le fossé entre détection et informations exploitables. Cela permet aux experts de réagir efficacement aux anomalies, garantissant la fiabilité du smart grid.

Mots clés: Réseaux électriques intelligents, Ville intelligente, Apprentissage automatique, Grands modèles de langage, Systèmes d'aide à la décision, Cybersécurité, CICIDS2017, Llama3, valeurs SHAP.

المختصر

ظهور الشبكات الذكية أحدث ثورة في قطاع الطاقة، حيث أدى إلى شبكات كهرباء مترابطة وذكية. تقدم هذه التطورات قدرات مراقبة وتحكم محسنة، ولكنها أيضًا تطرح تحديات في مجال اكتشاف نشاط الشبكة المشبوه به ، وهو جانب مهم لاستقرار الشبكة وأمنها. على الرغم من فعالية الأساليب التقليدية في تحديد نشاط الشبكة المشبوه به ، إلا أنها غالبًا ما تفتقر إلى قابلية التفسير، مما يجعل من الصعب فهم السبب الجذري وصياغة استجابة فعالة. هنا، تقدم نماذج اللغة الكبيرة (LLMs) نهجًا واعدًا. من خلال إنشاء أوصاف سهلة الفهم للتشوهات، يمكن لنماذج اللغة الكبيرة أن تسد الفجوة بين البيانات الأولية والرؤى العملية.

يبحث هذا القسم في إمكانات نماذج اللغة الكبيرة للكشف عن نشاط الشبكة المشبوه به. نحن نستفيد من مجموعة بيانات CICIDS2017 لاستكشاف كيفية تسخير نماذج اللغة الكبيرة لتحسين قابلية تفسير أنظمة الكشف عن نشاط الشبكة المشبوه به. على وجه التحديد، نفحص فعالية استخدام قيم SHAP (تفسيرات SHapley التراكمية الإضافية) لتوجيه نموذج اللغة الكبيرة نحو إنشاء أوصاف أكثر دقة لنشاط الشبكة المشبوه به المكتشفة. تجدر الإشارة إلى أن أبحاثنا تُظهر أنه حتى مع الحد الأدنى من الضبط الدقيق للنموذج ، يحقق نموذج Llama3 8B LLM نتائج ملحوظة عند استخدام مطالبات فعالة. وهذا يبرز الدور المحوري لهندسة المطالبات في إطلاق العنان للإمكانات الكاملة لنماذج اللغة الكبيرة في هذا المجال. من خلال دمج قيم SHAP في استراتيجية المطالبات الخاصة بنا، نتمكن من سد الفجوة بين اكتشاف نشاط الشبكة المشبوه به والرؤى القابلة للتنفيذ. وهذا يمكّن خبراء صنع القرار من الحصول على معلومات قيمة للتعامل مع نشاط الشبكة المشبوه به بشكل فعال، مما يضمن استمرار موثوقية عمليات الشبكة الذكية..

الكلمات الرئيسية: شبكات ذكية, مدينة ذكية, التعلم الآلي, نماذج لغة كبيرة, أنظمة دعم القرار, الأمن الإلكتروني, CICIDS2017, Llama3 , تفسيرات SHapley.

List of Acronyms

LLM	**Large Language model**
ML	**Machine Learning**
CNN	**Convolutional Neural Network**
SHAP	**SHapley Additive exPlanations**
SVM	**Support Vector Machine**
KNN	**K-Nearest Neighbors**
GPU	**Graphical Processing Unit**
RF	**Random Forest**
NLP	**Natural Language Processing**
GPT	**Generative Pre-trained Transformer**
LLaMA	**Large Language Model Meta AI (stylized as LLaMA)**
LSTM	**Long Short-Term Memory**
FFN	**Feed-Forward Layer**
RNN	**Recurrent neural networks**
GAN	**Generative Adversarial Networks**
Cot	**Chain-of-Thought**
DDoS	**Distributed Denial-of-Service**
DoS	**Denial-of-Service**
CSV	**Comma-Separated Value**
JSON	**JavaScript Object Notation**
XML	**Extensible Markup Language**

Table of Contents

General Introduction

The power industry has undergone a significant transformation with the emergence of smart grids. These intelligent and interconnected electricity networks boast real-time monitoring, dynamic control, and improved efficiency thanks to advanced sensors and communication technologies. However, this increased complexity also introduces challenges, particularly in anomaly detection. Anomalies in smart grids represent deviations from normal operating patterns, often signifying underlying faults, cyberattacks, or other critical events. Timely and accurate detection of these anomalies is essential for ensuring grid stability, preventing power outages, and safeguarding against security threats.

Traditional anomaly detection methods, primarily based on statistical and machine learning techniques, have shown promise in identifying anomalies. However, a crucial limitation remains: the lack of interpretability. These methods often struggle to provide comprehensive and human-understandable explanations, hindering their usefulness for decision-making support.

Large Language Models (LLMs) have emerged as powerful tools with the potential to bridge this gap. Their ability to process and generate human-quality text makes them well-suited for providing meaningful descriptions of anomalies, enhancing the interpretability of anomaly detection systems. This thesis explores the potential of LLMs to improve anomaly detection in smart grids by generating informative and human-friendly explanations for detected anomalies.

This research focuses on leveraging a well-established dataset, CICIDS2017, which provides a comprehensive collection of labeled anomaly scenarios in smart grids. By utilizing this dataset, we can explore how LLMs can be harnessed to improve the interpretability of anomaly detection systems specifically within the context of smart grid data.

One key approach we explore is the integration of SHAP (SHapley Additive exPlanations) values. SHAP values offer a means to explain the internal workings of machine learning models, highlighting the features that contribute most significantly to a specific prediction. In our case, we can leverage SHAP values to guide the LLM towards generating more accurate descriptions of the detected anomalies. Notably, by focusing on effective prompt engineering – the art of crafting specific instructions for the LLM – we aim to unlock its full potential for this domain.

Our research demonstrates that even without extensive fine-tuning, a powerful LLM like Llama3 8B achieves remarkable results when prompted effectively. This highlights the crucial role of crafting compelling prompts that leverage domain knowledge and insights like SHAP values. By incorporating SHAP values within our prompting strategy, we aim to bridge the gap between raw anomaly data and actionable insights. This empowers decision-making experts with valuable information to respond to anomalies effectively, ensuring the continued reliability of smart grid operations.

Structure:

Chapter 1: Context and Research Question

This chapter lays the foundation for understanding the context of the research. It delves into the concept of smart grids, exploring their key features, functionalities, and the role they play in the power industry. The chapter also highlights the challenges associated with anomaly detection in these complex systems, emphasizing its importance for grid stability and security. Additionally, this chapter will introduce the research question that will guide the investigation into the potential of LLMs for anomaly detection in smart grids.

Chapter 2: Literature Review

This merged chapter provides a comprehensive overview of the relevant background for your research. It will first explore existing research on anomaly detection for smart grids, examining traditional machine learning techniques and discussing their limitations, particularly the lack of interpretability. This section will then shift focus to Large Language Models (LLMs), introducing their concept, capabilities in natural language processing and understanding, and their potential applications. The chapter will then explore how LLMs can be leveraged for anomaly detection in smart grids, highlighting existing research on integrating LLMs with these systems and pinpointing potential areas for further exploration.

Chapter 3: Proposed Approach

Building upon the groundwork established in the previous chapters, this chapter focuses on the specific contributions of your thesis. It details the research methodology employed, including the chosen dataset (CICIDS2017) and the LLM utilized (Llama3 8B). This chapter will delve into the proposed approach, outlining the specific techniques and methods used to leverage LLMs for anomaly detection in smart grids. The chapter will emphasize the importance of prompt engineering in unlocking the full potential of LLMs for this task.

Chapter 4: Experiments

This chapter focuses on the experimental validation of your proposed approach. It details the experimental setup, including the evaluation metrics used. The chapter will then present the core findings of your research, demonstrating the effectiveness of the LLM-based approach in generating accurate and informative descriptions of anomalies detected in smart grids. This includes a detailed analysis of the experimental results, highlighting the strengths and limitations of the proposed approach.

Chapter 01: Context and research question

1.1. Introduction

The modern power grid is evolving towards interconnected "smart grids" for increased efficiency and sustainability. These integrate advanced technologies with traditional infrastructure, enabling real-time monitoring and control. This chapter sets the context by exploring key features of smart grids and their benefits, including improved grid stability and integration of renewable energy. However, smart grids also introduce challenges, particularly in anomaly detection. Anomalies, deviations from normal operating patterns, can indicate equipment malfunctions or cyberattacks. Effective anomaly detection is crucial for maintaining grid stability and security. Traditional methods often lack interpretability, hindering effective responses. This research explores the potential of Large Language Models (LLMs) for anomaly detection in smart grids.

1.2. The Smart City

With increasingly large populations in urban areas, cities are now forced to adapt to this growth in a thoughtful manner by adopting policies planned for the long term. It is from this obligation that the status of “smart city” was born. So, what is a smart city?

1.2.1. Definition of a Smart City

A smart city is an urban area that uses information and communication technologies (ICT) and the Internet of Things (IoT) to improve the efficiency and sustainability of city operations and enhance the quality of life for its citizens.

The key characteristics of a smart city encompass the utilization of digital technologies and data collection to optimize various city services and infrastructure, spanning transportation, energy management, waste disposal, and public safety. Additionally, smart cities strive to enhance environmental sustainability by implementing initiatives such as smart lighting, advanced waste management systems, and the integration of renewable energy sources. Moreover, these cities aim to elevate citizen engagement and overall quality of life through amenities like interactive kiosks, widespread access to free public Wi-Fi, and real-time data sharing platforms. Furthermore, smart cities seek to foster economic growth by leveraging advanced infrastructure and services to attract businesses and talent, thereby creating a conducive environment for innovation and prosperity.

While there is no universally accepted definition, the overarching goal of a smart city is to leverage technology and data to make urban areas more efficient, livable, and sustainable. The specific smart city initiatives and technologies implemented can vary widely between cities based on their unique needs and priorities. [1]

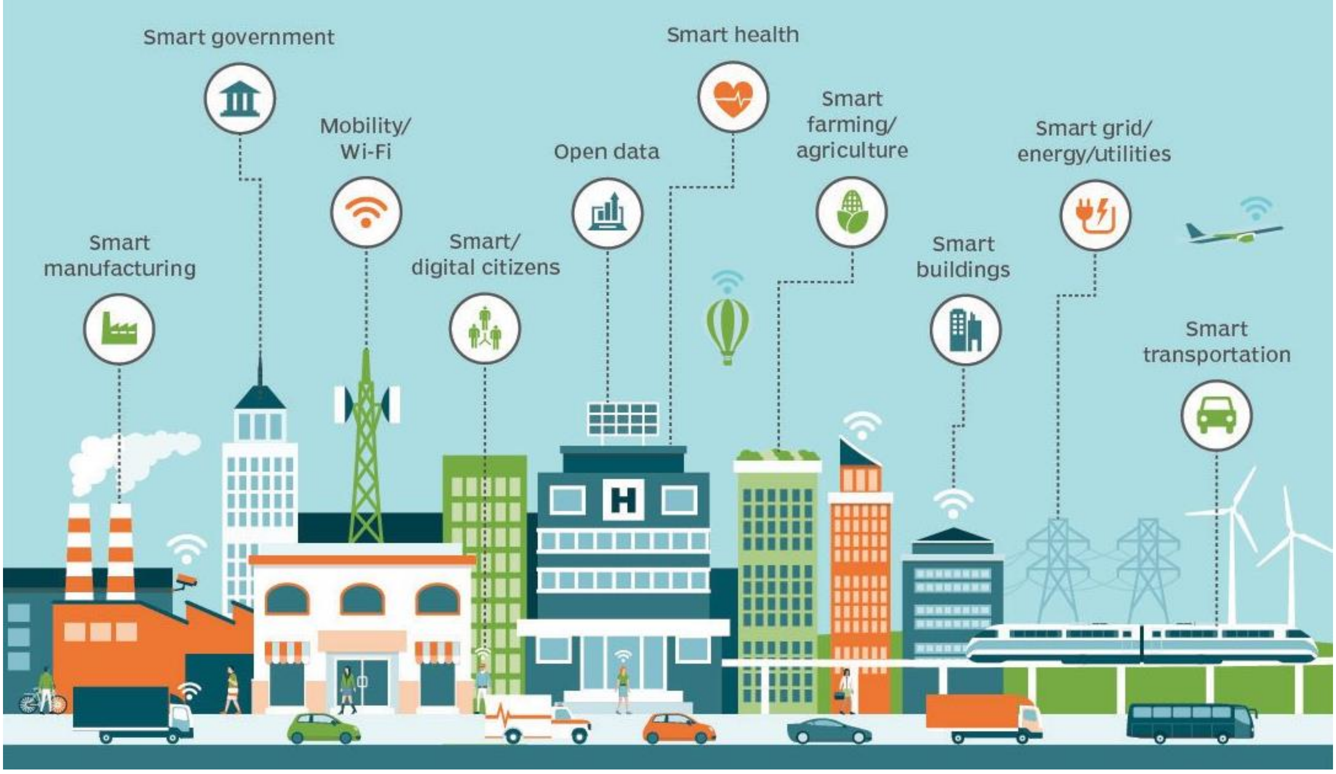


Figure 1: visual representation of a smart city and components [2]

1.2.2. The Smart Aspects of a Smart City

The concept of a “smart city” is a multi-faceted one, encompassing various aspects that leverage technology and data-driven solutions to enhance the quality of life for its residents and promote sustainable urban development. [1]

1.2.2.1. Smart People

Refers to citizens who are actively engaged in the city's digital ecosystem, utilizing technology to access services, participate in decision-making processes, and contribute to the overall development of the city. Smart people are empowered individuals who leverage digital tools to enhance their quality of life and interact with the urban environment in innovative ways.

1.2.2.2. Smart Living

Encompasses the use of technology and data-driven solutions to create a high-quality, convenient, and sustainable lifestyle for residents. Smart living involves smart homes, efficient public services, access to real-time information, and personalized experiences that enhance comfort and well-being.

1.2.2.3. Smart Environment

Focuses on leveraging technology to monitor, protect, and enhance the natural surroundings within a city. This includes initiatives to improve air quality, reduce pollution, conserve resources, promote green spaces, and create a sustainable urban ecosystem that prioritizes environmental health and well-being.

1.2.2.4. Smart Mobility

Involves the integration of intelligent transportation systems, innovative mobility solutions, and data-driven approaches to optimize transportation networks, reduce congestion, enhance accessibility, and promote sustainable modes of travel within the city. Smart mobility aims to improve the efficiency and safety of urban transportation while reducing environmental impact.

1.2.2.5. Smart Economy

Refers to the use of technology and innovation to drive economic growth, foster entrepreneurship, attract investment, and create job opportunities within the city. A smart economy focuses on digital transformation, knowledge-based industries, and sustainable business practices to ensure long-term prosperity and competitiveness.

1.2.2.6. Smart Government

Involves the adoption of digital governance practices, data-driven decision-making, and citizen-centric services to enhance transparency, efficiency, and accountability in city administration. Smart government initiatives aim to improve public services, engage citizens, and optimize resource allocation through the effective use of technology and information systems.

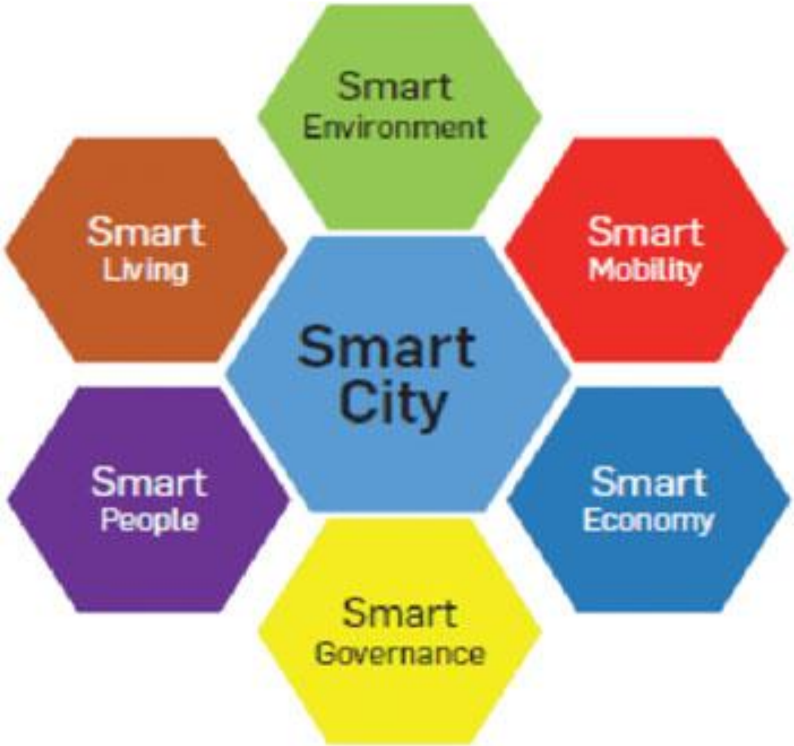


Figure 2: the smart city and its smart aspects

1.3. Smart grid the Intelligent electricity network

Current electricity networks are inevitably destined to experience a profound change in the coming years. Indeed, in a context of development of intermittent and probably diffuse renewable energies, development of new electrical uses and the need to optimize the efficiency of networks, it will be necessary continue to ensure the best possible balance between production and consumption. There solution to these difficulties is the use of new infrastructures called Smart Grid (network smart electric).

1.3.1. Definition of smart grids

There are currently several definitions of smart grids and also several objectives for the same definition of smart electricity networks. However, all definitions agree that bidirectional communication is a key to future smart networks.

Definition 1: An electricity distribution network that uses computer technologies to optimize production, distribution, and consumption is referred to as "intelligent." The goal of this type of network is to optimize every segment of the electricity supply chain, from all producers to all consumers, in order to maximize energy efficiency and overall energy quality.

Definition 2: An electrical network whose operations have switched from using analog technology to standard technology is referred to as a "smart grid" by the Carnegie Mellon University Software Engineering Institute. integrated digital technology that makes detection, prediction, control, and communication possible.

1.3.1. Different types of electrical networks

The electricity network can be broadly divided into three main components, understanding the characteristics and functionalities of these electricity network types is fundamental for developing effective anomaly detection strategies in smart grid systems: [3]

1.3.1.1. Transport Network

The transport network in the context of electricity refers to the high-voltage network responsible for transmitting electricity over long distances from power plants to substations. This network operates at high voltages to minimize energy losses during transmission and is crucial for delivering electricity from large power generation facilities to distribution points.

1.3.1.2. Distribution Network

The distribution network is the intermediary network that receives electricity from the transmission network at lower voltages and distributes it to end-users, such as residential, commercial, and industrial consumers. This network includes transformers that step down the voltage for safe delivery to homes and businesses.

1.3.1.3. Local Distribution Network

The local distribution network, also known as the secondary distribution network, is the final stage of electricity distribution that delivers power directly to consumers' premises. It consists of power lines, transformers, and meters that regulate and measure electricity consumption at the individual level. This network ensures reliable and efficient electricity supply to end-users within a specific area.

1.3.2. Key Features of Smart Grids Relevant to Anomaly Detection

Effective anomaly detection in smart grids hinges on exploiting the unique data landscape they generate:

1.3.2.1. Two-Way Communication Infrastructure (Advanced Metering Infrastructure - AMI)

Smart grids are characterized by an advanced metering infrastructure that enables two-way communication between utility providers and consumers. This allows for real-time monitoring and collection of granular data from smart meters, which can be analyzed for anomaly detection. [4]

1.3.2.2. Integration of Renewable Energy Sources and Distributed Generation

Smart grids incorporate renewable energy sources and distributed generation, such as solar panels and wind turbines, which introduce new data streams that can be monitored for anomalies. Detecting anomalies in the generation, distribution, and consumption of renewable energy is crucial for maintaining grid stability and reliability. [5]

1.3.2.3. Sensors and Real-Time Monitoring Capabilities

Smart grids are equipped with a network of sensors that continuously monitor various parameters, including voltage, current, power flow, and grid performance. This real-time data can be analyzed to detect anomalies that may indicate equipment failures, cyber-attacks, or other issues that could compromise the grid's security and reliability. [5]

1.3.2.4. Cyber-Security Considerations

Smart grids rely on digital infrastructure and communication networks, which introduces potential vulnerabilities to cyber-attacks. Anomaly detection techniques can be employed to identify and mitigate cyber-security threats, such as data manipulation, unauthorized access, and malware intrusion. [4]

1.3.2.5. Electricity Theft and Fraud Detection

Smart grid data can be analyzed to detect anomalies in energy consumption patterns, which may indicate electricity theft or fraudulent activities. Identifying these anomalies is crucial for maintaining the grid's financial viability and ensuring fair billing for consumers. [4]

1.3.2.6. Fault Detection and Grid Resilience

Anomaly detection in smart grid data can help identify equipment malfunctions, grid failures, and other issues that could compromise the grid's reliability and resilience. Prompt detection of these anomalies can enable proactive maintenance and rapid response to minimize disruptions. [5]

1.3.3. Smart Grid Development Challenges

The transition towards smart electricity networks presents a multifaceted set of challenges that must be addressed for successful implementation. These challenges span across industrial, social, economic, and environmental domains, each requiring careful consideration and strategic planning. In this section, we will delve into the key development issues associated with the deployment of smart grid technologies. [6]

1.3.3.1. Industrial Challenges

The development of smart grids relies heavily on the availability and integration of advanced materials and technological advancements. Ensuring the functionality and compatibility of all components within the smart grid system is of paramount importance. Continuous developments, improvements, and compliance with industry standards are necessary to address the industrial challenges.

1.3.3.2. Social Challenges

The integration of smart grid technologies involves the active participation and engagement of consumers. The introduction of smart meters empowers consumers to manage their energy consumption more effectively, fostering a new dynamic between the grid and the end-user. Addressing the social implications of this shift and ensuring a seamless transition for consumers is a crucial aspect of smart grid development.

1.3.3.3. Economic Challenges

The modernization of electricity networks requires new forms of cooperation among the major economic players in the sector. Additionally, the role of the state in the development of smart grid infrastructure as a public service is a critical economic consideration.

1.3.3.4. Environmental Challenges

Smart grids aim to integrate renewable and decentralized energy sources, which can contribute to minimizing the environmental impact of electricity generation and distribution.

Addressing the challenges of integrating these new energy sources and mitigating the effects on climate change and environmental disturbances is a key priority in smart grid development.

1.3.1. Types of smart grids:

We can distinguish three categories of “smart network” approaches

1.3.1.1. Smart Grid at the Transport Network Level

At the level of transmission system operators (TSOs), the smart grid approach involves enhancing planning, surveillance, and remote-control capabilities. This entails leveraging technical advancements to better manage the evolving needs of the transmission network, particularly in light of the increasing integration of decentralized power generation, which can pose challenges for network balancing and security.

1.3.1.2. Smart Grid at the Distribution Network Level

Distribution network operators (DNOs) must deploy technologies traditionally used in the transmission network, such as bidirectional protection and balancing management, to support the growth of decentralized power production. The decreasing cost of these technologies has facilitated this development, allowing for the accelerated installation of long-available solutions.

1.3.1.3. Smart Grid at the Local Level

The most significant change in the smart grid paradigm may occur at the local level, where the convergence of electronics, IT, and telecommunications has opened new horizons for the management of local consumption and production. This specific area is often referred to as the "smart home" or, more broadly, the “smart consumer”.

1.4. Iot and how it ties in to Smart Grids

The integration of the Internet of Things (IoT) into smart grid infrastructure has emerged as a pivotal advancement in enhancing the interconnectedness and efficiency of modern electricity networks. By enabling seamless communication between various grid components, IoT technologies facilitate real-time data exchange and decision-making processes, ultimately optimizing energy flow and consumption.

IoT sensors have been deployed extensively throughout smart grids, collecting vast amounts of data pertaining to energy usage, grid performance metrics, and environmental conditions. Through the application of advanced data analysis techniques, these sensor-generated data streams are transformed into valuable insights, allowing for the identification of anomalies and the optimization of energy distribution, thereby enhancing the overall efficiency of the grid infrastructure.

Furthermore, IoT technologies have enabled remote monitoring and control capabilities, empowering grid operators to promptly manage energy flow, detect faults, and respond to potential disturbances in a timely manner. This enhanced level of control and responsiveness has been instrumental in ensuring the stability and reliability of smart grid systems.

By leveraging IoT for real-time monitoring and control, smart grids have demonstrated increased resilience in the face of potential disruptions. IoT devices, equipped with advanced anomaly detection algorithms, can swiftly identify and respond to grid irregularities, rerouting energy flow and optimizing operations to minimize downtime and enhance overall performance.

Moreover, the integration of IoT has facilitated the seamless incorporation of renewable energy sources into the grid infrastructure. By monitoring generation and consumption patterns in real-time, IoT technologies enable efficient utilization of renewable resources, contributing to the development of a more sustainable and eco-friendly energy ecosystem. [7]

1.5. The Cyber-security principles on smart grids

With the integration of advanced technologies and digital systems in modern smart grids comes the critical challenge of ensuring the cyber-security of smart grid infrastructures. Understanding and addressing cyber-security anomalies within smart grids are paramount to safeguarding against potential threats and vulnerabilities that could compromise the integrity and functionality of the grid. In this section, we explore a range of cyber-security anomalies that smart grids may encounter, from attacks on data integrity to network infrastructure issues, highlighting the importance of proactive measures and robust strategies to mitigate risks and protect the smart grid ecosystem.

1.5.1. Attacks on Data Integrity

Attacks on data integrity within smart grids represent deliberate and malicious efforts to manipulate the information transmitted across the network. These attacks can manifest in various forms, such as altering readings from smart meters, injecting false data to disrupt grid operations, or deleting critical information to conceal evidence of a cyber-attack. Ensuring the integrity of data in smart grids is paramount to maintaining the reliability and security of the entire system. [3]

1.5.2. Unusual Measurements and Consumptions

Anomalies related to unusual measurements and consumptions in smart grids can stem from diverse sources. Malfunctioning equipment may report incorrect data, external interferences like power surges or physical obstructions can distort readings, and fraudulent activities such as electricity theft can lead to irregular consumption patterns. Detecting and addressing these anomalies is crucial for ensuring the accuracy and efficiency of energy distribution within the smart grid. [3]

1.5.3. Intrusions

Intrusions into the smart grid's network infrastructure involve unauthorized access to critical components such as control systems, data storage, or communication channels. Intruders gaining entry into these systems can disrupt operations, compromise sensitive information, or execute malicious activities that threaten the grid's functionality and security. Preventing and mitigating intrusions is essential to safeguarding the integrity and confidentiality of smart grid operations. [3]

1.5.4. Network Infrastructure Issues

Issues with the network infrastructure of smart grids can arise from hardware failures, software bugs, or connectivity problems. Hardware failures like broken sensors or downed power lines, software glitches in control systems, or network connectivity issues such as lost wireless connections can impede the smooth operation of the grid. Addressing and resolving these infrastructure issues promptly is vital to maintaining the reliability and performance of the smart grid. [3]

1.5.5. Electrical Data Anomalies

Electrical data anomalies in smart grids can result from various issues within the electrical components of the system. For instance, a short circuit causing a sudden surge in current or a malfunctioning transformer leading to a voltage drop can disrupt the normal functioning of the grid. Monitoring and addressing these electrical anomalies are essential to prevent potential safety hazards and ensure the efficient delivery of electricity. [3]

1.5.6. Identification of Cyber-attacks

The identification of cyber-attacks in smart grids involves the detection of suspicious activities aimed at compromising the system's cybersecurity. This process includes recognizing patterns of malicious behavior, identifying malware or other harmful software, and understanding common attack strategies to proactively defend against cyber threats. Early detection and response to cyber-attacks are critical to maintaining the security and resilience of smart grid operations. [3]

1.5.7. Use of Detection Devices

Detection devices play a crucial role in identifying anomalies within smart grids. For example, smart meters can detect unusually high electricity usage patterns, while network monitors can identify abnormal data transmission volumes. Leveraging these specialized devices enhances the ability to detect and respond to anomalies promptly, contributing to the overall security and efficiency of the smart grid infrastructure. [3]

1.6. Research question:

In the evolving landscape of smart grid technology, where efficiency and resilience are paramount, the growing complexity and interconnectivity within these systems have introduced new challenges in safeguarding against cyber security anomalies. While the field of anomaly detection has been extensively explored using various machine learning techniques, yielding promising results, the advent of Large Language Models (LLMs) presents a novel opportunity for further advancement.

This thesis aims to investigate the potential of LLMs in enhancing anomaly detection within the context of cyber security threats in smart grids. Considering the existing landscape of machine learning-based anomaly detection techniques, the research will focus on leveraging the unique capabilities of LLMs to comprehend textual data related to smart grid operations and security. By exploring this approach, the study seeks to determine how LLMs can aid experts in making informed decisions for system improvement, ultimately contributing to the betterment of smart grid systems.

In essence, how can Large Language Models (LLMs) be effectively leveraged to enhance anomaly detection in smart grids by comprehending textual data related to smart grid operations and security, thereby aiding experts in making informed decisions for system improvement, considering the existing landscape of anomaly detection techniques in machine learning that have shown promising results?

Chapter 02: Literature Review

2.1. Introduction

The rapid evolution of artificial intelligence has brought forth a new generation of powerful tools: Large Language Models (LLMs). These models, trained on massive datasets of text and code, have revolutionized tasks like natural language understanding, question answering, and text generation. As AI continues to shape various industries, exploring the potential of LLMs in the context of complex systems like smart grids becomes increasingly critical.

Smart grids necessitate real-time decision-making amidst intricate interconnected systems and continuous data streams. Traditional methods often struggle to adapt to this dynamic environment. This chapter serves as the foundation for understanding how LLMs can be leveraged to enhance decision-making in smart grids. We delve into the core principles and capabilities of LLMs, setting the stage for exploring their application in anomaly detection, predictive maintenance, and energy optimization. The insights gained here will pave the way for subsequent chapters detailing the practical implementation of LLMs within smart grid environments.

However, the challenge of anomaly detection in smart grids necessitates a close examination of existing approaches. This chapter delves into a comprehensive review of current research endeavors relevant to our central theme: utilizing LLMs for anomaly detection in smart grids. By scrutinizing existing methodologies, outcomes, and limitations, we aim to build a solid foundation for the development of our proposed LLM-based method. This review will unearth crucial insights, identify gaps in existing knowledge, and reveal opportunities for further exploration in the following chapters.

The chapter commences with a meticulous analysis of "Related Works" – existing research closely related to our thesis. This analysis will be followed by a comprehensive table encapsulating key facets of the reviewed works, including their methods, datasets, and results. We will then delve into a detailed discussion of the collective insights derived, highlighting the strengths, weaknesses, and untapped potential for further investigation within this field. By meticulously examining the current state of the art, this chapter not only prepares the ground for the innovative LLM-based method proposed here, but also situates it within the broader context of smart grid anomaly detection research. This comprehensive foundation will streamline the execution and assessment of our proposed approach in the forthcoming chapters.

2.1. Machine Learning

Machine learning, situated within the broader domain of artificial intelligence, represents a scientific discipline characterized by the utilization of algorithms to discern patterns within datasets. These patterns, which may manifest as recurrent structures or relationships, are identified across various forms of data, encompassing numerical values, textual information, images, statistical metrics, and more. The versatility of machine learning is underscored by its capacity to leverage any form of digital data for analysis and learning purposes.

Central to the essence of machine learning is the process of pattern recognition, wherein algorithms are tasked with uncovering underlying structures within datasets to enhance their performance in executing specific tasks. Through the iterative analysis of data, these algorithms acquire the ability to discern intricate patterns, correlations, and trends, thereby refining their predictive capabilities and decision-making processes. This iterative learning process enables machines to autonomously adapt and improve their performance over time, culminating in the development of sophisticated models capable of addressing complex problems across diverse domains.

By harnessing the power of machine learning, researchers and practitioners can unlock valuable insights, automate decision-making processes, and drive innovation in fields ranging from healthcare and finance to marketing and robotics. The transformative potential of machine learning lies in its ability to extract meaningful information from vast and diverse datasets, enabling organizations to make data-driven decisions, optimize operations, and unlock new opportunities for growth and advancement. [8]

Some of the popular machine learning classification algorithms include:

2.1.1. Support Vector Machine (SVM)

SVM is a classification method that plots data points in an n-dimensional space and aims to find the optimal hyperplane that best separates the data into different classes. It works by identifying the hyperplane that maximizes the margin between classes, allowing for effective classification of new data points based on their position relative to this hyperplane. [9]

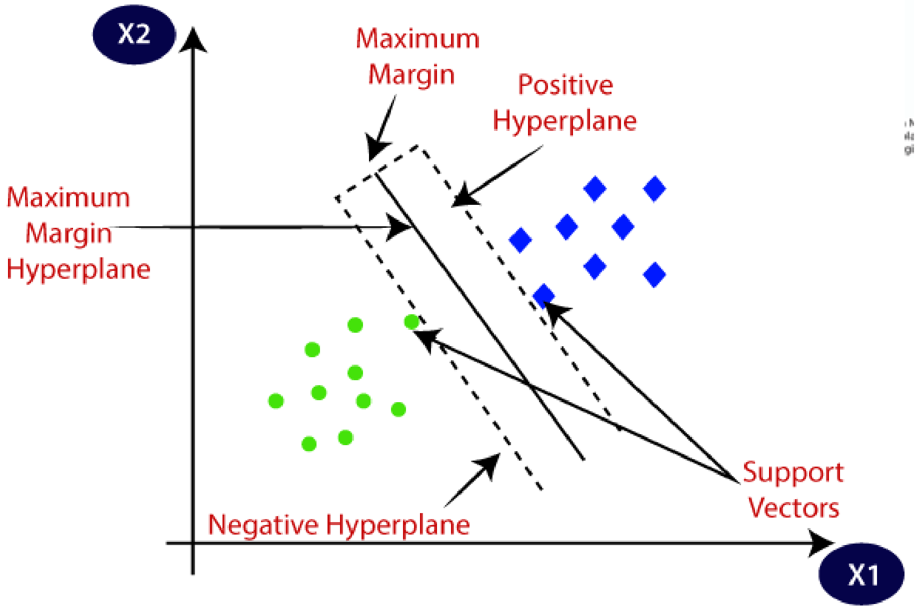


Figure 3: SVM visual example diagram [10]

2.1.2. K-Nearest Neighbors (KNN)

KNN is a simple instance-based algorithm that makes predictions based on nearby data points, where the class of a new data point is determined by the majority vote of its k nearest neighbors. KNN operates by storing all available cases and classifying new cases based on the majority class of their nearest neighbors, measured using distance functions like Euclidean or Manhattan distance. [9]

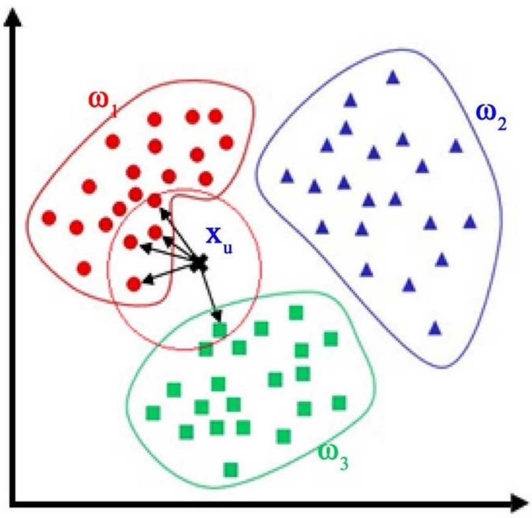


Figure 4: KNN visual example diagram [11]

2.1.3. AdaBoost

AdaBoost is a boosting ensemble model that works well with decision trees. It focuses on learning from previous mistakes by increasing the weight of misclassified data points. This algorithm adapts by iteratively training decision trees, calculating weighted error rates, updating weights of misclassified points, and combining the predictions of multiple trees to make the final prediction. [9]

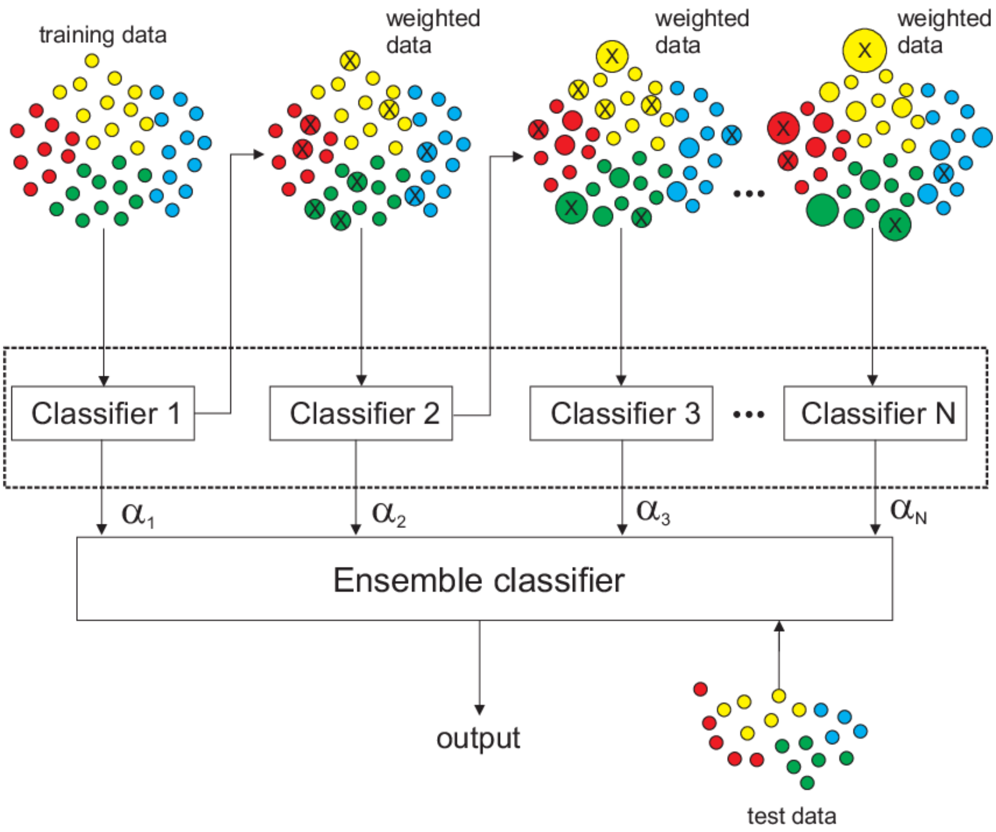


Figure 5: AdaBoost visual representation diagram [12]

2.1.4. Random Forest

Random Forest is an ensemble model that uses bagging as the ensemble method and decision trees as individual models. It leverages the concept of randomness to build a collection of decision trees. Random Forest operates by training multiple decision trees on random subsets of the training set, with each tree providing a classification vote that is aggregated to make the final prediction. [9]

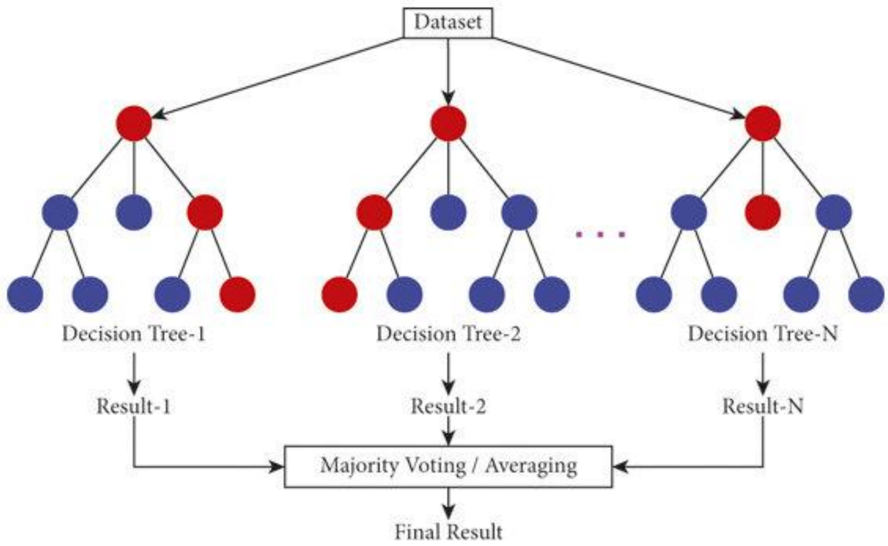


Figure 6: Random Forest (RF) visual representation diagram [13]

2.1.5. Naive Bayes

Naive Bayes is a probabilistic classifier based on Bayes' theorem, assuming independence between features. It calculates the probability of an event occurring given the presence of certain features. Naive Bayes operates by computing the likelihood of each class based on the feature values and selecting the class with the highest probability as the predicted outcome for a new data point. [9]

2.1.6. K-Means

K-Means is an unsupervised clustering algorithm used to partition data into k clusters based on similarity. It aims to minimize the variance within clusters and maximize the variance between clusters. K-Means operates by iteratively assigning data points to the nearest cluster centroid and updating the centroids based on the mean of the data points in each cluster until convergence is reached. [9]

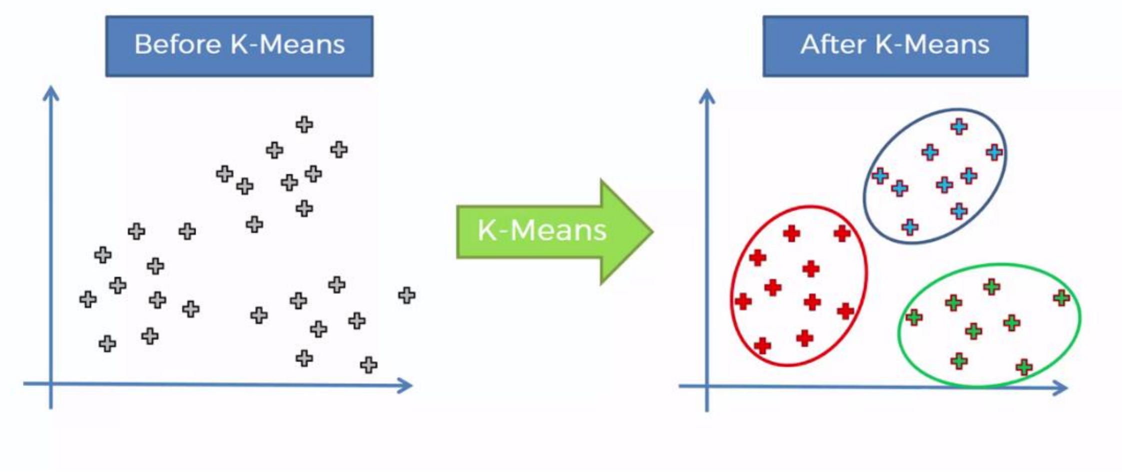


Figure 7: K-Means visual diagram example [14]

2.1.7. Decision Tree

Decision Tree is a supervised learning algorithm used for classification and regression tasks. It creates a tree-like structure of decisions based on feature values to predict the target variable. Decision Tree operates by recursively splitting the data based on feature attributes to maximize information gain or minimize impurity, resulting in a tree structure where each leaf node represents a class or regression value. [9]

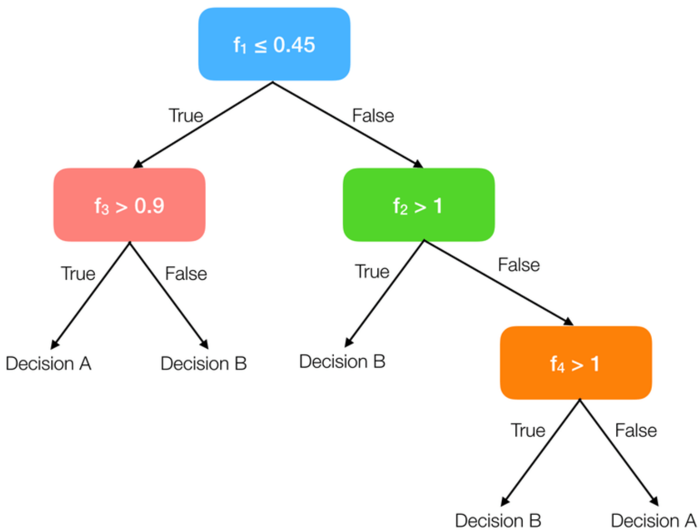


Figure 8: Binary Decision Tree visual diagram example [39]

2.1.8. Linear Regression

Linear Regression is a simple and widely used regression algorithm that models the relationship between independent variables and a continuous dependent variable using a linear equation. Linear Regression operates by fitting a line that best represents the relationship between the independent and dependent variables, aiming to minimize the sum of squared differences between the observed and predicted values. It predicts continuous outcomes based on the input features. [9]

2.1. Deep Learning

Deep learning is a subset of machine learning and artificial intelligence that utilizes multi-layered artificial neural networks to perform complex tasks with a high degree of accuracy. These deep neural networks have shown significant advancements in various domains, outperforming traditional machine learning methods, especially when handling unstructured and large datasets. Deep learning has a profound impact across diverse fields such as speech recognition, healthcare, autonomous vehicles, cybersecurity, predictive analytics, and more.

The core concept of deep learning revolves around the use of artificial neural networks inspired by the biological structure of the human brain. These networks analyze incoming data, identify patterns, and classify information to produce desired outputs. Unlike the human brain, artificial neural networks operate through discrete layers, connections, and data propagation directions.

One of the distinguishing features of deep learning is its ability to automatically extract features for classification, eliminating the need for manual feature engineering as required in traditional machine learning approaches. This feature extraction process is data-driven and relies on large datasets for training to achieve high accuracy in output predictions.

Due to the complexity of its algorithms, deep learning necessitates powerful computational resources, often utilizing high-performance CPUs or GPUs, especially in cloud-based environments. Various types of artificial neural networks are employed in deep learning applications, such as Convolutional Neural Networks (CNNs) for image classification and object recognition, and Generative Adversarial Networks (GANs) for creating realistic yet synthetic data. [15]

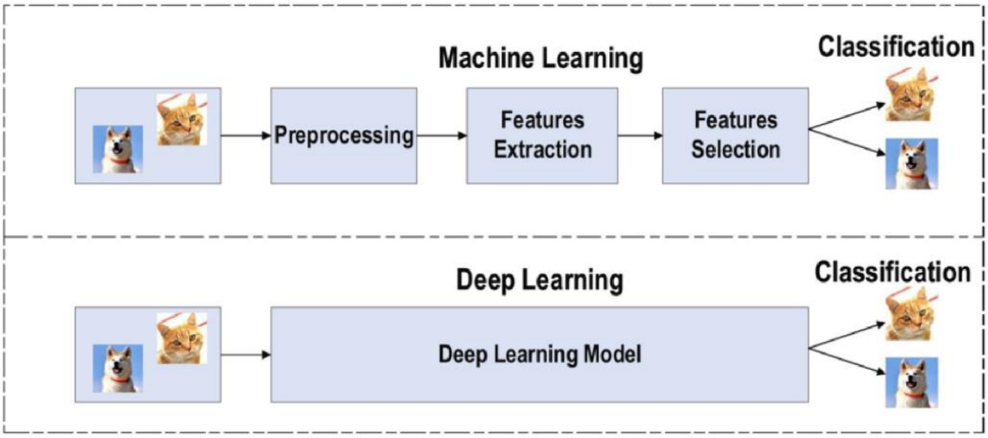


Figure 9: Visual illustration of the distinction between deep learning and traditional machine learning in terms of feature extraction and learning

2.2. Definition of large language models

A large language model (LLM) is a deep learning algorithm that can perform a range of natural language processing (NLP) tasks. Large language models use transformer models and are trained on large datasets. They can thus recognize, translate, predict or generate texts or other content.

Large language models are also referred to as neural networks (NN), which are computer systems inspired, by the human brain. These neural networks are based on a network of nodes, arranged in layers, like neurons.

In addition to teaching human languages to artificial intelligence (AI) applications, large language models can also be trained to perform different tasks, such as understanding protein structures, writing software code, and many more. Others just like the human brain, large language models must be trained beforehand, then refined, so that they can classify or generate text, answer questions, or even summarize a document. Their abilities to solve the problems submitted to them can be used in different fields, such as health, finance and entertainment. Large language models can support different applications of NLP, including translation, Chabot's, AI assistants, and more.

Large language models also have a large number of parameters, which are similar to the memories that memory collects when it is learning. Think of these parameters as the model's knowledge bank.

In 2024, some popular examples of Large Language Models include:

- **GPT-3 and GPT-4 (OpenAI):** OpenAI's Generative Pre-trained Transformer (GPT) series, particularly GPT-3 and GPT-4, are renowned for their language generation capabilities and have been widely used in applications like ChatGPT and Microsoft Copilot.
- **LLaMA (Meta):** Meta's LLaMA family of models is another notable example of LLMs that excel in language understanding and generation tasks.
- **PaLM2 (Google):** Google's PaLM2 is a powerful Large Language Model known for its ability to comprehend and generate human-like text, contributing to various natural language processing applications.

These LLMs are built on transformer-based architectures, utilizing a vast number of parameters and advanced deep learning techniques to process and generate text in a human-like fashion. They have revolutionized the field of natural language processing and are widely recognized for their ability to understand context, generate coherent responses, and adapt to various language-related tasks with remarkable accuracy and efficiency.

2.3. What is a transformer model?

A transformer model is the most common architecture of a large language model. This consists of an encoder and a decoder. A transformer model processes data by converting inputs into tokens and then performing simultaneous mathematical equations to discover the relationships that exist between the tokens. The computer can thus see the patterns that a human would determine if the same query were submitted to it.

Transformer models rely on self-attention mechanisms, which allow them to learn faster than traditional models, such as long short-term memory (LSTM) models. A self-attentive transformer model can look at different parts of a sequence, or the entire context of a sentence, to generate predictions.

2.4. Main components of large language models

Large language models consist of multiple layers of neural networks. Recurrent layers, feedforward layers, embedding layers, and attention layers work together to process input text and generate output content.

The embedding layer creates embeddings from the input text. This part of the large language model captures the semantic and syntactic meaning of the input, so that the model can understand the context.

The feed-forward layer (FFN) of a large language model consists of several fully connected layers that transform input embeddings. By doing this, these layers allow the model to glean general concepts, that is, to understand the user's intent with the input text.

The recurrent layer interprets the words of the input text in order. It captures the relationship between words in a sentence.

The attention mechanism allows a large language model to focus on single parts of the input text appropriate for the task at hand. This layer allows the model to generate the most accurate outputs.

2.5. The three main types of large language models:

- Generic or raw language models predict the next word based on the language used in the training data. These language models perform information retrieval tasks.

- Instruction-matched language models are trained to predict responses based on the instructions provided in the input. They can therefore carry out sentiment analysis or generate text or code.

- Dialogue-friendly language models are trained to dialogue and predict the next response. Think about chatbots or conversational AI.

2.6. What is the difference between large language models and generative AI?

In academic writing, the distinction between large language models (LLMs) and generative AI lies in their fundamental functions and applications within the realm of artificial intelligence. Large language models, such as OpenAI's GPT-4 and Google's PaLM, are specialized AI systems designed to work specifically with language. These models utilize natural language processing (NLP) techniques to comprehend and generate human-like text. The term "large" in LLMs signifies the trend towards training models with an increasing number of parameters, leading to enhanced performance through the processing of vast amounts of data.

On the other hand, generative AI serves as a broader category encompassing various AI tools focused on content creation. Generative AI models, like ChatGPT and DALL-E, are built to generate original content across different domains, including images, music, and text. These models leverage complex machine learning algorithms, such as recurrent neural networks (RNNs) and generative adversarial networks (GANs), to understand patterns and produce output. While LLMs are specifically tailored for language-related tasks, generative AI models have a wider scope of applications, extending beyond text generation to include image and audio creation.

The interplay between LLMs and generative AI is evident in their collaborative roles, where LLMs provide the foundational text-generating capabilities that power generative AI tools like ChatGPT. As LLMs evolve to accept various inputs beyond text, such as audio and imagery, they contribute to the development of multimodal content generation. While generative AI models have the potential to revolutionize industries through applications like 3D modeling and voice assistants, LLMs primarily focus on text-based content creation, with the capacity to enhance voice assistants and other text-centric applications.

2.7. How do large language models work?

A large language model is based on a transformer model. It works as follows: it receives an input, encodes it, then decodes it to produce an output prediction. But before a large language model can receive text input and generate an output prediction, it must be trained so that it can perform general functions, and it must be fine-tuned so that it can perform specific tasks.

Training: Large language models are pre-trained using large text datasets from sites like Wikipedia, GitHub, etc. These datasets include billions of words, the quality of which will impact the performance of large language models. At this point, large language models begin unsupervised learning, meaning they process the provided datasets without special instructions. During this process, the LLM AI algorithm can learn the meaning of words and understand the relationships between them. He also learns to make the distinction at the level of meaning depending on the context. For example, it will determine whether "left" means the opposite of "right" or being "clumsy."

Tuning: For a large language model to be able to perform a specific task, such as translation, it must be tuned for that activity. The adjustment optimizes performance of specific tasks.

Prompt tuning: This is a function similar to prompt tuning, where a model is trained to perform a specific task via a few-prompting prompt. Shot or zero-shot prompting. A prompt is an instruction provided to an LLM. Few-shot prompting teaches the model to predict outputs by giving it a few examples. For example, in this sentiment analysis, a few-shot prompt would look like this:

Customer review: This plant is so beautiful!

Customer sentiment: positive

Customer review: This plant is so hideous!

Customer sentiment: negative

Based on the semantic meaning of "hideous" and the example provided in opposition, the language model will understand that the customer's sentiment in the second example is "negative."

Conversely, zero-shot prompting does not provide any examples to the language model to teach it how to respond to input. Instead, he frames the question by defining the sentiment that applies to the sentence. It clearly states the task the language model should perform, but does not provide problem-solving examples.

2.8. Use cases for large language models

Large language models can serve different purposes:

- Information retrieval: consider Bing or Google. When you use their search function, you rely on a large language model to produce information in response to a query. This is capable of retrieving information, summarizing it and communicating the response conversationally.

- Sentiment analysis: As applications of natural language processing, large language models enable businesses to analyze the sentiment of text data.

- Text generation: Large language models underpin generative AI, like ChatGPT, and can generate text based on inputs. They can produce a sample text when prompted. For example: “Write me a poem about palm trees in the style of Emily Dickinson.”

- Code generation: Like text generation, code generation is an application of generative AI. LLMs understand diagrams, which allows them to generate code.

- Chatbots and conversational AI: thanks to large language models, chatbots or conversational AI of a customer service are able to interact with customers, interpret the meaning of their questions or their answers, and often their provide answers in turn.

In addition to these use cases, large language models can complete sentences; answer questions, and summarize text.

With such a wide range of applications, major language models can be found in a multitude of domains:

- Technology: Large language models have many uses, such as enabling search engines to answer queries, helping developers write code, and much more.

- Health and science: Large language models are capable of understanding proteins, molecules, DNA and RNA. As a result, they can help in the development of vaccines, in the identification of treatments for diseases and in the improvement of preventive medicine. LLMs are also used as medical chatbots to carry out patient admissions or basic diagnostics.

- Customer service: LLMs are used across industries in customer service, in the form of chatbots or conversational AI for example.

- Marketing: Marketing teams can use LLMs to run sentiment analysis to quickly generate campaign ideas or copy, like synopses, and more.

- Legal: From searching large text data sets to generating legalese, large language models can help lawyers, paralegals, and legal staff.

- Banking LLMs can help banking institutions detect fraud.

2.9. Prompt Engineering

Prompt engineering is a specialized discipline within the field of artificial intelligence that focuses on refining and optimizing prompts to effectively interact with language models (LMs) and generative AI systems. The practice involves designing prompts that guide AI models to produce specific responses or outputs. A well-crafted prompt serves as the interface between human intent and machine-generated content, influencing the quality and relevance of the AI-generated output.

The process of prompt engineering entails creating clear and unambiguous prompts that align with the desired outcome, avoiding jargon and leading questions that may bias the model's response. It involves an iterative approach where initial prompts are tested, evaluated based on the generated output, refined as needed, and repeated until the desired quality of the response is achieved. This iterative refinement process ensures that the prompts effectively guide the AI model to produce accurate and contextually relevant outputs.

Prompt engineering is essential for enhancing the capabilities of large language models (LLMs) and other AI systems across various tasks such as question answering, text generation, and reasoning. By refining prompts, researchers and developers can improve the performance and accuracy of AI models, enabling them to better understand user intent and generate more tailored responses. [16]

2.9.1. Conversational Prompts

2.9.1.1. System Prompt

A system prompt is an instruction or query provided to the AI model that guides its behavior or response. It is typically predefined by the system or application and serves as the initial input to prompt the AI model to generate a specific output or perform a particular task. System prompts are designed to elicit desired responses from the AI model based on the system's requirements or objectives. [17]

2.9.1.2. User Prompt

A user prompt is an input provided by a user to interact with the AI model. It can be in the form of a question, command, or statement that conveys the user's intent or request to the AI system. User prompts play a crucial role in guiding the AI model's responses and influencing the quality and relevance of the generated output based on the user's input. [17]

2.9.1.3. Assistant Prompt

An assistant prompt is a specific type of prompt used in conversational AI systems or virtual assistants. It involves providing instructions or queries to an AI assistant to perform tasks, provide information, or engage in a dialogue with the user. Assistant prompts are tailored to the context of the interaction and are designed to facilitate effective communication between the user and the AI assistant, ensuring accurate and helpful responses. [17]

2.9.2. Types of Prompt Engineering

2.9.2.1. Zero-Shot Prompting

This technique involves providing the LLM with a prompt that is entirely novel and not included in its training data. The model leverages its internal knowledge and comprehension to generate the desired output based solely on the prompt's instructions. While zero-shot prompting can be highly effective in managing LLM outputs, it presents challenges in guaranteeing consistent and accurate results across all instances. [16]

2.9.2.2. Few-Shot Prompting

Similar to zero-shot prompting, few-shot prompting utilizes prompts. However, it furnishes the LLM with a limited number of illustrative examples showcasing the desired output format or content. This approach enhances the LLM's ability to produce more precise and consistent outputs compared to zero-shot prompting, effectively guiding the model towards the intended outcome. [16]

2.9.2.3. Chain-of-Thought (CoT) Prompting

This technique involves deconstructing the desired outcome into a series of sequential steps. The LLM is then instructed to generate the text corresponding to each step in the sequence. CoT prompting can be a powerful tool for managing LLM outputs, particularly for tasks requiring a logical flow of information. However, its implementation can be intricate and time-consuming, requiring careful consideration of the specific task at hand. [16]

2.9.2.4. Generated Knowledge Prompting

This approach leverages the LLM's own output to create the prompt for subsequent generations. By incorporating the model's previously generated text into the prompt, this technique can improve the precision and uniformity of the overall output. The LLM's familiarity with the concepts used in the prompt fosters a more focused and consistent generation process. [16]

2.9.2.5. Self-Consistency Prompting

This technique instructs the LLM to ensure its subsequent output aligns with its prior responses. This approach is particularly beneficial for tasks requiring coherence and continuity, such as generating a narrative or dialogue. By prompting the LLM for self-consistency, the overall output maintains a cohesive and logical flow throughout the generation process. [16]

2.10. LLM's affinity to Text-based data

Large Language Models (LLMs) have demonstrated remarkable performance in processing and generating text-based data due to their unique architectural design and training approach. These advanced AI systems excel in tasks such as natural language processing, text generation, and language understanding, making them highly effective in working with textual information.

The core of an LLM's success in handling text-based data lies in its ability to capture and model the complex relationships and patterns present in natural language. LLMs are typically built upon transformer-based architectures, which utilize attention mechanisms to identify and learn from relevant parts of the input sequence. This allows the model to understand the context and semantics of words and phrases, enabling more accurate and coherent text generation.

Furthermore, LLMs are trained on vast amounts of textual data, often comprising billions of words from various sources, including books, articles, websites, and social media. This extensive training process enables the models to learn the nuances of language, including grammar, syntax, and common idioms. By exposure to a diverse range of textual data, LLMs develop a deep understanding of the structure and flow of language, which translates into their ability to generate human-like text and engage in meaningful conversations.

One of the key advantages of LLMs in working with text-based data is their capacity for transfer learning. These models can be fine-tuned or adapted to specific domains or tasks by training them on smaller, task-specific datasets. This allows LLMs to leverage their general language understanding capabilities and apply them to more specialized contexts, such as legal documents, medical reports, or technical manuals. Fine-tuning enables LLMs to capture domain-specific vocabulary, jargon, and writing styles, further enhancing their performance in handling text-based data within specific fields.

Moreover, LLMs exhibit impressive performance in tasks that require reasoning, inference, and contextual understanding. By leveraging their attention mechanisms and deep learning capabilities, these models can identify and extract relevant information from large volumes of text, making them valuable tools for tasks such as question answering, summarization, and information retrieval.

However, it is important to note that while LLMs excel in working with text-based data, they may face challenges in handling tasks that require external knowledge, common sense reasoning, or factual accuracy. As language models, they are trained to generate plausible text based on patterns in the training data, but they may not always have a deep understanding of the real-world implications or truthfulness of the generated content. [18]

2.11. Related works

Understanding the landscape of related works is paramount to contextualizing the current state of research and innovation. This section delves into a comprehensive review of existing studies that are closely aligned with the subject matter of the thesis. By examining the methodologies employed and the outcomes achieved in these works, we aim to establish a foundation for evaluating and advancing the proposed LLM-based approach for anomaly detection in smart grids. Through this exploration, we seek to identify key insights, gaps, and opportunities that will inform and enrich the research landscape in the domain of smart grid security.

2.11.1. Anomaly Detection using Random Forest Classifier

The Random Forest classifier has been widely used in anomaly detection tasks, including those related to smart grids and network security. In [22], Nebrase et al demonstrated the effectiveness of this method, achieving an impressive accuracy rate of 99.90% across three benchmark datasets: CICIDS-2017, UNSW-NB15, and ICS cyber-attack datasets. Similarly, Kurniabudi et al in [27] used Random Forest to detect anomalies in smart grid data, reporting an accuracy rate of 99.71%. The authors also employed Information Gain technique for feature selection, highlighting the importance of this approach in overcoming class imbalance challenges.

In another study, Ziadoon et al in [23] compared the performance of various machine learning algorithms, including Random Forest, on the CICIDS-2017 dataset. Their results showed that Random Forest achieved an accuracy rate of 99.30%, outperforming other methods such as SVM and CNN.

2.11.2. Deep Learning-based Anomaly Detection

Deep learning techniques have been increasingly used in anomaly detection tasks, particularly those involving neural networks. In [19], S. Huang et al proposed a novel Imbalanced Generative Adversarial Network (IGAN) model for intrusion detection, achieving an impressive accuracy rate of 99.70%. The authors also demonstrated the effectiveness of their IGAN-IDS approach on three benchmark datasets.

In another study, JooHwa et al in [21] used a hybrid approach combining a Generative Adversarial Network (GAN) with a Random Forest classifier to detect anomalies in smart grid data, reporting an accuracy rate of 99.83%. The authors highlighted the advantages of their integrated GAN-RF framework.

2.11.3. Autoencoder-based Anomaly Detection

Autoencoders have been used as a dimensionality reduction technique in anomaly detection tasks. In [17], R. Abdulhamme et al proposed an autoencoder-based approach for detecting anomalies in smart grid data, reporting an accuracy rate of 99.60%. The authors employed Principal Component Analysis (PCA) to further improve the performance of their method.

2.11.4. Support Vector Machine (SVM)-based Anomaly Detection

SVM has been widely used as a classification algorithm in anomaly detection tasks. In [19], S. Huang et al reported an accuracy rate of 96.97% using SVM on the CICIDS-2017 dataset. However, their results also showed that Random Forest outperformed SVM by a significant margin.

In another study, Ziadoon et al in [23] compared the performance of various machine learning algorithms, including SVM, on the CICIDS-2017 dataset. Their results showed that SVM achieved an accuracy rate of 75.21%, which was lower than other methods such as Random Forest and CNN.

2.11.5. Convolutional Neural Network (CNN)-based Anomaly Detection

CNN has been used in anomaly detection tasks, particularly those involving image or time-series data. In [19], S. Huang et al reported an accuracy rate of 99.48% using CNN on the CICIDS-2017 dataset.

2.11.6. Adaboost-based Anomaly Detection

Adaboost has been used as a classification algorithm in anomaly detection tasks. In [18], Yulianto et al proposed an AdaBoost-based approach for detecting anomalies in smart grid data, reporting an accuracy rate of 81.83%. The authors employed Synthetic Minority Oversampling Technique (SMOTE) to address the issue of class imbalance.

Federated Learning (FL)-based Anomaly Detection

Federated learning has been used in anomaly detection tasks to maintain user privacy while still achieving high accuracy rates. In [27], Jithish et al proposed a FL-based approach for detecting anomalies in smart grid data, reporting an accuracy rate of 99% on two benchmark datasets.

2.11.7. Large Language Model (LLM)-based Anomaly Detection

LLMs have been used in anomaly detection tasks to provide human-understandable explanations for detected anomalies. In [29], Abderrazak et al proposed a novel approach using SHAP values with LLMs to generate explanations for detected anomalies, reporting an accuracy rate of 80% on the CICIDS-2017 dataset.

2.11.8. Generative Adversarial Network (GAN)-based Anomaly Detection

GAN has been used in anomaly detection tasks to generate synthetic data that closely resembles the existing dataset. In [21], JooHwa et al proposed a hybrid approach combining GAN with Random Forest classifier to detect anomalies in smart grid data, reporting an accuracy rate of 99.83%.

2.11.9. Intrusion Detection Systems (IDS) using SHAP and LLMs

Intrusion detection systems have been used to detect malicious activities in networks. In [29], Abderrazak et al proposed a novel approach using SHAP values with LLMs to generate explanations for detected anomalies, reporting an accuracy rate of 80% on the CICIDS-2017 dataset.

2.12. State of the art Summary

For illustration purposes, the following figure briefly summarizes all the works on interest mentioned in this chapter: [**Acc**uracy, **Prec**ision, **Rec**all, **F-score**]

	Study	Year	Approach	Dataset	Acc	Prec	Rec	F-score
[18]	Yulianto et al	2018	Adaboost	CICIDS2017	81.83	81.83	100	90.01
[17]	R. Abdulhamme et al	2019	Auto Encoder + PCA	CICIDS2017	99.60	98.90	98.80	98.80
[21]	JooHwa et al	2019	GAN RF	CICIDS2017	99.83	98.68	92.76	95.04
[19]	S. Huan et al	2020	SVM	CICIDS2017	96.97	/	/	96.99
[19]	S. Huan et al	2020	Random Forest	CICIDS2017	99.79	/	/	99.78
[19]	S. Huan et al	2020	CNN	CICIDS2017	99.48	/	/	99.44
[19]	S. Huan et al	2020	IGAN-IDS	CICIDS2017	99.70	/	/	99.70
[22]	Nebrase et al	2020	Random Forest	CICIDS2017	99.90	99.70	99.70	99.70
[27]	Kurniabudi et al	2021	Random Forest	CICIDS2017	99.71	99.80	99.90	99.90
[23]	Ziadoon et al [23]	2021	Random Forest	CICIDS2017	99.30	99.09	99.30	99.12
[23]	Ziadoon et al [23]	2021	SVM	CICIDS2017	75.21	99.16	75.21	76.60
[23]	Ziadoon et al [23]	2021	CNN	CICIDS2017	99.47	99.43	99.46	99.44
[27]	Guastalla et al	2024	GPT-3.5	CICIDS2017 (DDoS)	96.00	/	/	/
[27]	Guastalla et al	2024	GPT-4	CICIDS2017 (DDoS)	92.00	/	/	93.00
[29]	Abderrazak et al	2024	Random Forest	CICIDS2017 (Web Attacks)	/	80	62	67

Table 1: State of the art summary table

2.13. Synthesis

The related works presented in this chapter demonstrate the effectiveness of various machine learning approaches for anomaly detection in smart grids. However, upon closer examination, it becomes apparent that these methods have limitations. Specifically, they lack the ability to provide detailed descriptions of detected anomalies, making them inaccessible to non-machine learning experts who may be responsible for implementing and interpreting the results.

Most studies focus solely on achieving high accuracy rates, often neglecting the importance of anomaly description. For instance, Random Forest classifiers, which achieved impressive accuracy rates (e.g., [22], [27]), do not provide insight into the specific data points contributing to anomalies. Similarly, Autoencoder-based approaches (e.g., [17]) and Support Vector Machine (SVM) methods (e.g., [19]) lack descriptive capabilities.

Moreover, the lack of standardization in anomaly detection metrics and evaluation procedures hinders direct comparison between studies. This highlights the need for a more comprehensive understanding of anomalies, beyond mere accuracy rates. The use of Large Language Models (LLMs), such as SHAP values, can provide this nuance by describing detected anomalies in a human-understandable manner.

The current landscape of anomaly detection methods in smart grids suggests that there is a pressing need for approaches that balance high accuracy with descriptive capabilities. Our research aims to address this gap by combining traditional machine learning techniques with LLMs, enabling grid operators to not only detect anomalies but also gain insights into their underlying causes."

2.14. Discussion

The studies reviewed in the table showcase the effectiveness of various machine learning approaches for anomaly detection in smart grids. Notably, several studies employing Random Forest classifiers achieved accuracy exceeding 99% ([22], [22]). With Kurniabudi even reaching 99.90% accuracy in his 2021 study [22].

While this high level of accuracy is commendable, it also suggests that further improvement in anomaly detection using traditional machine learning techniques might yield diminishing returns. This paves the way for exploring alternative approaches that focus on aspects beyond raw accuracy.

Our research takes a different approach to anomaly detection in smart grids. Our primary focus is not on achieving the highest possible detection accuracy, but rather on providing a more nuanced understanding of the anomalies that are detected. Here, Large Language Models (LLMs) offer a unique advantage. By leveraging SHAP values, we can not only identify anomalies but also gain insights into the specific data points that contribute to those anomalies. This detailed description of the anomaly can be crucial for grid operators to diagnose and address the underlying issues.

Therefore, while Random Forest remains a strong choice for initial anomaly detection due to its high accuracy (as evidenced in the table), our research will utilize this established method to identify potential anomalies. We will then employ an LLM, empowered by SHAP values, to delve deeper and provide a comprehensive description of the detected anomalies, aiding in faster and more precise troubleshooting within the smart grid.

2.15. Conclusion

This chapter established a foundation for using Large Language Models (LLMs) to enhance decision-making in smart grids, addressing the limitations of traditional methods in dynamic, data-intensive environments. LLMs offer powerful analytical capabilities, extracting valuable insights from complex data sets.

A thorough review of the state-of-the-art in anomaly detection for smart grids, with a focus on LLMs, identified critical gaps and opportunities for further exploration. This review sets the stage for the proposed LLM-based method, highlighting LLMs' potential to address current shortcomings by generating informative anomaly descriptions and improving interpretability and decision-support.

Chapter 3, "Proposed Approach," will detail the research methodology, dataset, and LLM used. It will emphasize the importance of prompt engineering in maximizing LLMs' effectiveness for anomaly detection in smart grids. The chapter will conclude with the practical implications of the research, underscoring its contribution to enhancing interpretability and decision-support in anomaly detection systems for smart grids.

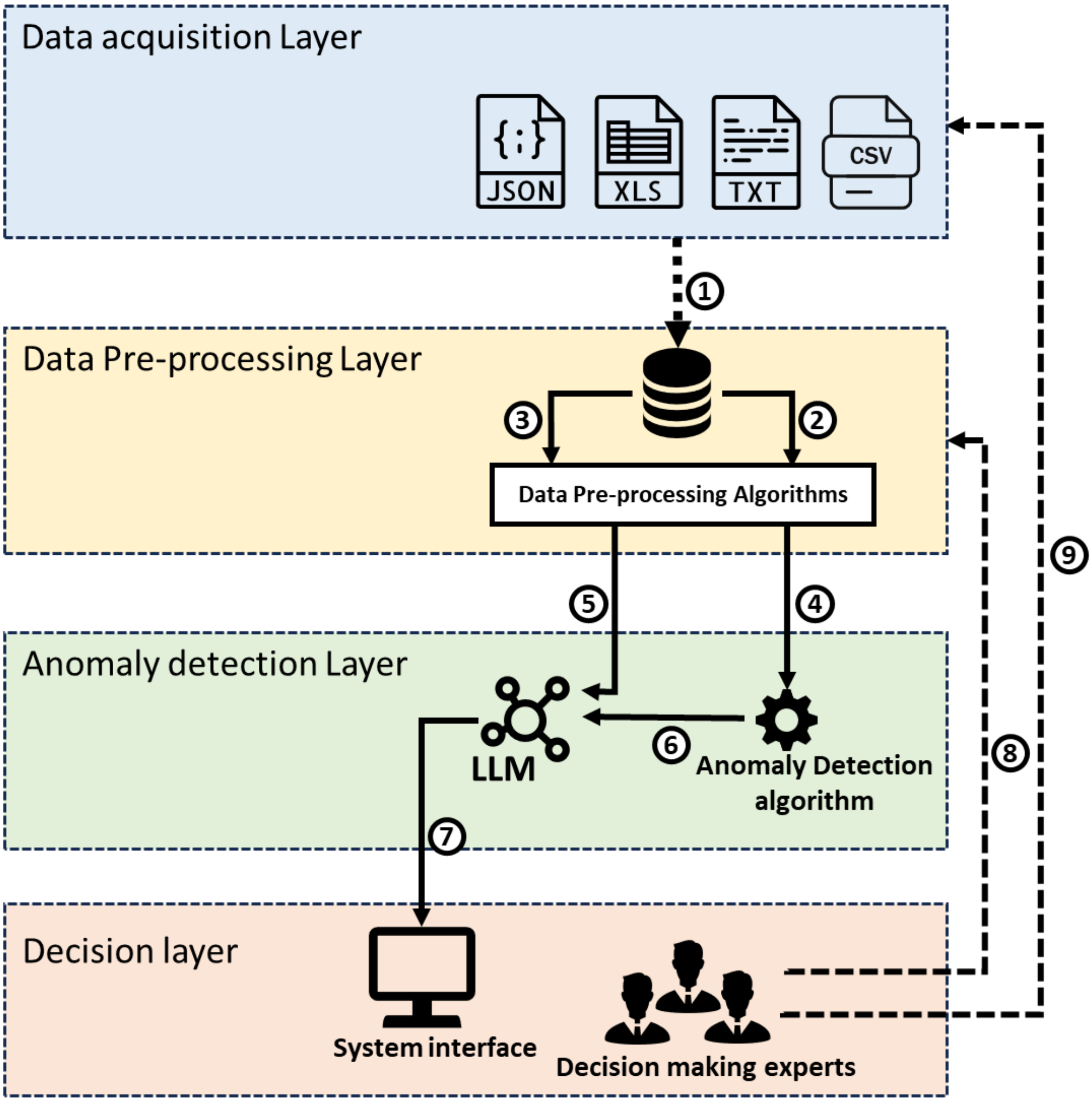
Chapter 03: Proposed Approach

3.1. Introduction

This chapter delves into the core of our research – the proposed LLM-based approach for anomaly detection and description in smart grids. As highlighted in the abstract, traditional anomaly detection methods often lack interpretability, making it difficult to pinpoint the root cause of the issue. Here, we bridge this gap by leveraging the power of Large Language Models (LLMs).

This chapter will present a comprehensive understanding of the methodology employed in our research. We delve deeper into each component in subsequent sections, providing insights into the system's design and implementation.

3.2. Proposed System Architecture



1.2.	Online Data	**6.**	Detected anomaly
3.	Offline Data	**7.**	LLM Descriptive Output
4.	Processed Data (numeric)	**8.**	Anomaly detection system revision
5.	Processed Data (textual)	**9.**.	Feeding the system with new data

Internal data stream

External data stream

Figure 10: proposed system architecture

3.2.1. System Architecture Description

The proposed system architecture aims to detect anomalies in smart grids using a large language model (LLM)-based approach. The system consists of four layers: data collection, data pre-processing, anomaly detection, and decision. The data collection layer gathers raw data from various sources, such as edge devices, in the form of CSV files, JSON, XML, or text formats, and transfers it to the data server in the pre-processing layer. In the data pre-processing layer, online data is processed using data processing algorithms, while offline data stored in the data server is also processed alongside the online data.

The data processing algorithms generate two versions of the traffic data: one in numeric format and one in textual format. The numeric data is sent to the anomaly detection algorithm, while the textual data is passed to the LLM. The anomaly detection layer utilizes machine learning techniques to identify anomalies in the numeric data. The output of the anomaly detection algorithm is then combined with the processed textual data and sent to the LLM for further analysis. The LLM uses both the output of the anomaly detection algorithm and the textual data to generate a detailed but brief description of the anomaly, along with suggested solutions.

The decision layer receives the LLM's output and presents it to decision-making experts through a system interface. These experts review the output and make informed decisions based on the provided information. If the anomaly detection system outputs a false negative, the experts may decide to revise the system. If the system's output is correct, the experts may choose to retroactively feed the system to improve its performance for future anomaly detection.

By integrating an LLM-based approach with anomaly detection algorithms, the proposed system aims to enhance the readability of anomaly detection systems in smart grids, ultimately contributing to improved grid reliability and efficiency.

3.2.2. System Architecture Layers:

Within the framework of our thesis, the system architecture is structured into distinct layers to facilitate efficient anomaly detection processes. [29]

3.2.2.1. Data Collection Layer

The data collection layer is responsible for gathering raw data from various sources, such as edge devices, in the form of CSV files, JSON, XML, or text formats. This layer ensures that the necessary data is collected and transferred to the data server located in the pre-processing layer.

3.2.2.2. Data Pre-processing Layer

The data pre-processing layer is responsible for cleaning, transforming, and preparing the collected data for further analysis. It includes a data server that stores the offline data and processes both online and offline data using data processing algorithms. The processed data is then converted into numeric and textual formats for use in the anomaly detection layer.

3.2.2.3. Anomaly Detection Layer

This layer receives the numeric format version of the processed traffic data and uses anomaly detection algorithms to identify potential anomalies. The output of the anomaly detection algorithm is then sent to the large language model (LLM) alongside the processed textual data.

3.2.2.4. Decision Layer

This layer is where decision-making experts review the LLM output and make informed decisions based on the detailed descriptions of anomalies and suggested solutions provided by the LLM.

3.2.3. Involved Actors:

Various actors play crucial roles in the efficient functioning of the anomaly detection system architecture across different layers: [29]

3.2.3.1. System Admins:

System admins operate in the Edge Layer of the smart grid system. Their responsibilities include providing support, troubleshooting, and maintaining the computer servers and networks where initial data collection occurs. They ensure the smooth operation of data collection processes at the system's edge.

3.2.3.2. Data Scientists:

Data scientists are primarily situated in the Data Processing Layer. Their main tasks involve developing and implementing data processing algorithms for both online and offline data analysis. They play a key role in processing and analyzing the collected data to extract valuable insights.

3.2.3.3. Cybersecurity Experts:

Cybersecurity experts are essential actors operating across all layers of the smart grid system. They focus on maintaining the security and integrity of the system's data and processes, with a particular emphasis on securing the Edge Layer and Data Processing Layer against potential threats and vulnerabilities

3.2.3.4. Machine Learning Engineers:

Machine learning engineers are predominantly located in the LLM Layer of the smart grid system. Their primary responsibilities revolve around developing, training, and fine-tuning the LLM model specifically designed for anomaly detection and learning purposes. They play a critical role in enhancing the system's anomaly detection capabilities.

3.3. Methodology overview

This section outlines the methodology employed for the proposed LLM-based anomaly detection and description system.

The approach leverages machine learning for anomaly classification, drawing critical inspiration from the work of Abderrazak et al. (2024) [29]. This research on anomaly detection in smart grids informed our decision to employ a machine learning model for classification. Additionally, a Large Language Model (LLM) is integrated into the system for generating informative descriptions of the detected anomalies. We delve into the details of the anomaly detection component, including the chosen model and feature selection techniques.

Figure 11: methodology overview diagramme

3.3.1. Anomaly Detection

The anomaly detection component employs a random forest classifier to identify anomalous traffic data within the smart grid. The data undergoes preprocessing steps including missing value imputation, categorical encoding, and feature scaling to ensure compatibility with the machine learning model. To address potential class imbalance favoring normal data, the XGBoost classifier with adjusted hyperparameters is also evaluated.

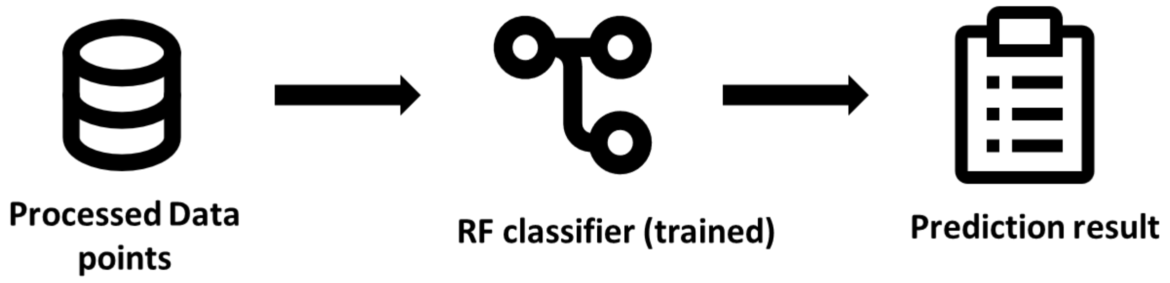


Figure 12: Anomaly detection process overview

3.3.2. Feature Selection

While both Random Forest and XGBoost can handle a high number of features, feature selection is implemented to improve model interpretability and potentially enhance performance. SHAP values are utilized to identify the most influential features for each anomaly type. Only the top 3 most impactful features, as determined by SHAP analysis, are retained for the final model.

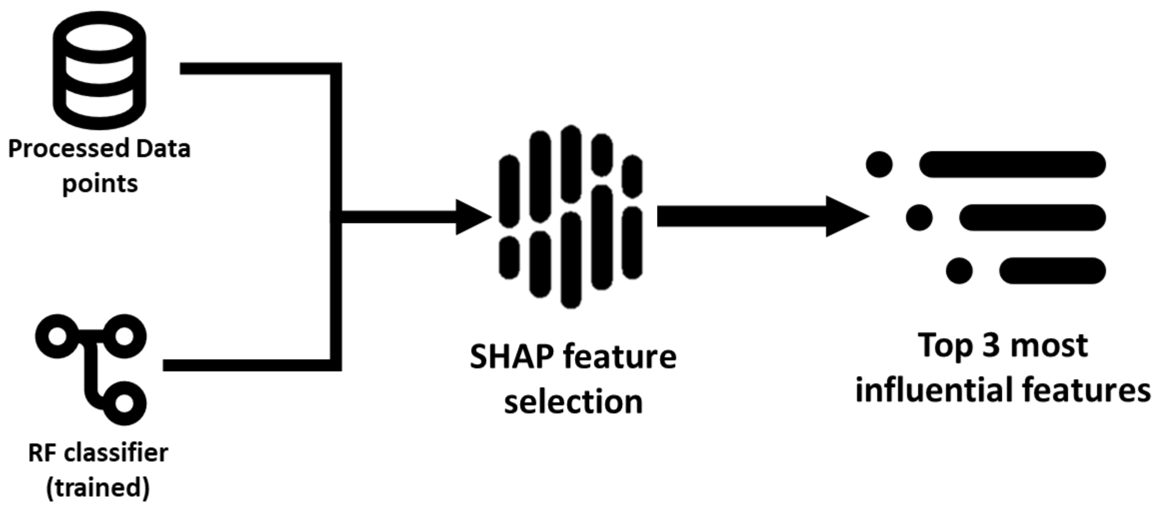


Figure 13: SHAP values powered feature selection process overview

3.3.3. Anomaly Description with LLM

The LLM serves as the anomaly description component. The system feeds the anomaly label, a **textual representation** of the original traffic data, and the top 3 most influential features to the LLM. This combination of information allows the LLM to generate a comprehensive description of the detected anomaly, aiding in understanding the root cause of the issue within the smart grid.

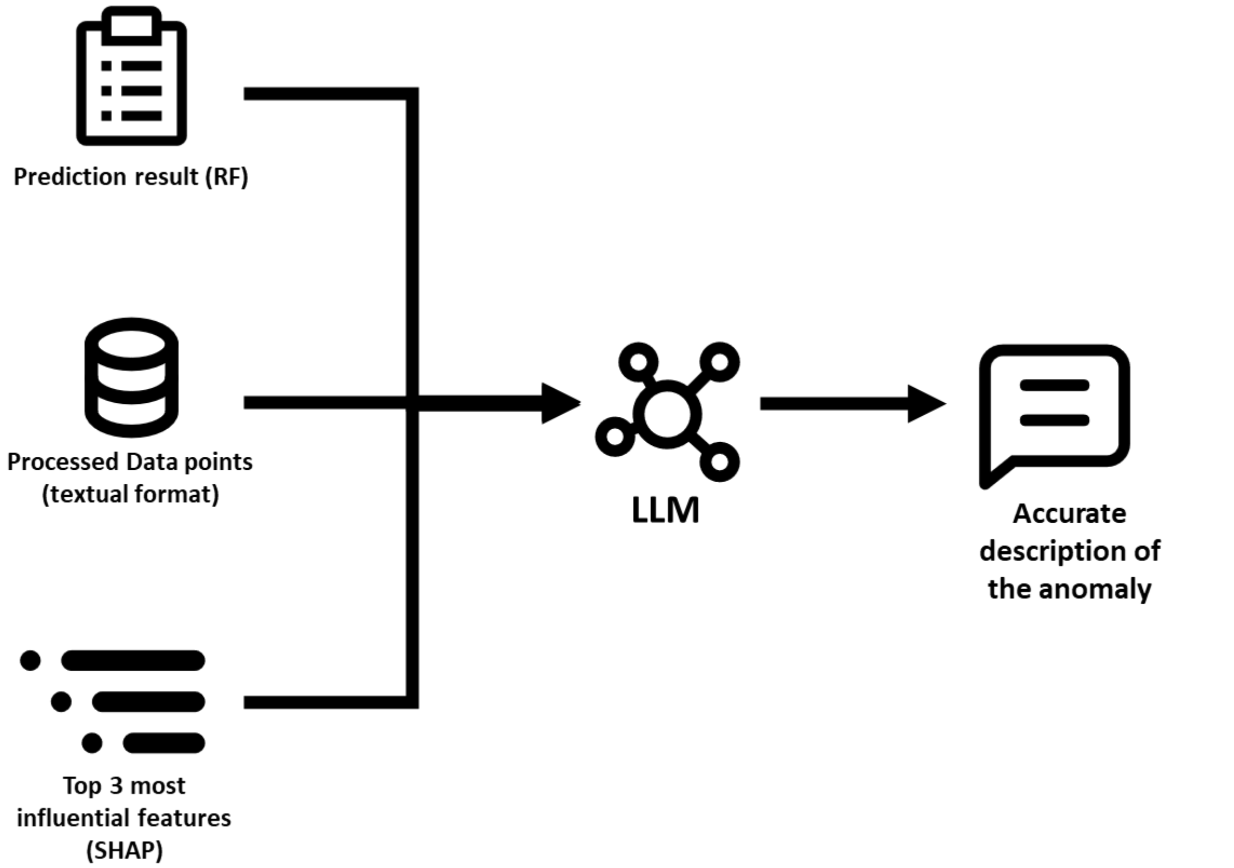


Figure 14: LLM powered Anomaly description process overvuew

3.4. Data set used

The CICIDS2017 dataset serves as an important reference point for intrusion detection systems (IDS) and intrusion prevention systems (IPS). It comprises labeled network flows, complete packet payloads in pcap format, profiles, and labeled flows. This dataset fills the gap of dependable test and validation datasets for anomaly-based intrusion detection methods. It encompasses benign traffic and prevalent attacks carried out over five days, encapsulating a variety of attacks like Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet, and DDoS. The dataset records authentic background traffic using a B-Profile system to profile human interactions and produce naturalistic benign traffic. It’s crafted to mirror real-world network data and includes features such as comprehensive network configuration, diverse traffic, complete interaction scenarios, and attack diversity. The dataset is all-encompassing, covering a range of network configurations, traffic scenarios, and attack types, making it a crucial resource for assessing and developing anomaly detection systems in smart grid settings. [30]

Attack label	**Flow count**
Benign	2273097
DoS Hulk	231073
Port Scan	158930
DDoS	128027
DoS GoldenEye	10293
DoS Slowloris	5796
DoS Slowhttptest	5499
FTP-Patator	7938
SSH-Patator	5897
Botnet	1966
Web Attack: Brute Force	1507
Web Attack: XSS	652
Web Attack: SQL Injection	21
Infiltration	36
Heartbleed	11
Total	**2830743**

Table 2: CICIDS2017 Dataset summary

3.4.1. Description of the Attack Scenarios

Here in this dataset, six attack profiles are covered based upon the most updated list of commonly used attack families, which can be explained as follows: [31]

3.4.1.1. Web Attack:

The dataset includes three types of web attacks. Firstly, SQL Injection, a security vulnerability where an attacker manipulates the queries that an application sends to its database, allowing unauthorized users to view data. Secondly, Cross-Site Scripting (XSS), which occurs when the attacker injects malicious code into the victim's web application. Lastly, Brute Force, which involves trying all possible passwords to decode the administrator's password.

3.4.1.2. Botnet Attack:

This refers to a group of internet-connected devices, such as those in a home, office, or public systems, infected by harmful malware. This malware allows the attacker to access the device and its connection for theft, network disruption, and IT environment damage. Botnet attacks are remotely controlled by cybercriminals and have become one of today's most significant security threats.

3.4.1.3. Heartbleed Attack:

This is a serious bug in the OpenSSL implementation, an open-source implementation of the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols. This vulnerability allows malicious hackers to read and steal data from the memory of the victim server. Brute Force Attack is a dictionary attack method that generates many successive guesses to access encrypted data. This attack is commonly used for cracking passwords, finding hidden web pages or content, and decoding Data Encryption Standard (DES) keys.

3.4.1.4. DDoS Attack:

This is one of the most common cyber weapons, which aims to deplete the resources of an online service and network by overwhelming it with traffic from multiple compromised systems, denying legitimate users access to the service.

3.4.1.5. DoS Attack:

This is a type of cyber-attack on a network designed to temporarily prevent legitimate users from accessing computer systems, devices, or other network resources due to malicious cyber activities.

3.4.1.6. *Infiltration Attack:*

This is a malicious attempt to enter or damage the interior of the network, typically by exploiting vulnerable software like Adobe Acrobat/Reader.

3.5. Data Pre-processing

3.5.1. Handling Missing Values: KNN Imputation with Outlier Handling

Missing values are a common challenge in network traffic data analysis, often arising from sensor malfunctions or network errors. To address this issue, researchers have explored various imputation techniques, including K-Nearest Neighbors (KNN) imputation. This approach leverages the similarity between data points to estimate missing values, while also considering potential outliers that may skew the imputation process.

3.5.1.1. *KNN Imputation for Flow Bytes/s*

KNN imputation works by identifying the k nearest neighbors of a data point with a missing value and using their values to estimate the missing entry. This method assumes that similar data points are likely to have similar values, making it suitable for network traffic data where patterns and correlations exist between features. Several studies have investigated the application of KNN imputation in network traffic data analysis. For instance, Liao et al. (2014) [33] proposed a KNN-based imputation method for missing values in network traffic data, demonstrating its effectiveness in improving the accuracy of network anomaly detection. Similarly, Ding et al. (2016) [34] employed KNN imputation to handle missing values in network traffic data, highlighting its ability to preserve the underlying data distribution and improve the performance of machine learning models.

In the case of CICIDS2017 we noticed all of the missing valued are resigned to the **Flow Bytes/s** feature, where we find **1294** missing values. However, before proceeding with the imputation method, we need to first deal with the outliers.

3.5.1.2. *Outlier Handling*

Network traffic data may contain outliers, which are data points that deviate significantly from the majority. These outliers can negatively impact the imputation process by skewing the distribution of the data and leading to inaccurate estimates of missing values. To mitigate this issue, researchers have proposed various outlier handling techniques. One approach is to identify and remove outliers before applying imputation methods. Gupta et al (2018) [35] developed a framework for outlier detection and removal in network traffic data, which involved using statistical measures and machine learning algorithms to identify and remove a Liao et al. (2014) [33] anomalous data points. By removing outliers, the authors were able to improve the accuracy of KNN imputation and enhance the overall quality of the network traffic data. Another approach is to incorporate outlier handling directly into the imputation process Ding et al. (2016) [34] proposed a modified KNN imputation algorithm that assigns lower weights to outliers during the imputation process, effectively reducing their influence on the estimated values. This approach helped to improve the robustness of KNN imputation in the presence of outliers.

3.5.2. Feature Engineering for Network Traffic Data Analysis

This section details a feature engineering approach employed to prepare network traffic data for analysis using Large Language Models (LLMs). LLMs excel at processing and extracting knowledge from textual information. However, network traffic data is typically stored in a tabular format with numerical and categorical features as is the case for the CICIDS2017 dataset.

LLMs are inherently designed to work with sequential text data. Their internal representations and processing mechanisms are optimized for understanding the relationships and patterns within natural language. Thus, directly feeding raw network traffic data (numerical features like port numbers or durations) into an LLM results in suboptimal performance. LLMs generally struggle to grasp the underlying relationships and semantics within the non-textual format. [36]

To address this challenge, we employ a feature engineering technique called concatenation. Here, we combine all descriptive features (excluding the label) into a single, continuous string within a new column named "traffic data." This transformation aims to create a more text-like representation that LLMs can effectively process. [36]

Benefits for LLMs: By concatenating features with delimiters, we essentially create a textual description of each network traffic flow. This allows the LLM to leverage its ability to understand language structure and relationships between words to analyze the combined features in the "traffic data" column.

Preserving Label Information: The original label column (e.g., "BENIGN" or "DDoS" …etc.) is kept separate. This ensures that the crucial classification information remains accessible for supervised learning tasks within the LLM framework.

Overall, feature concatenation serves as a bridge between the non-textual nature of network traffic data and the text-based processing capabilities of LLMs. By transforming the data into a more LLM-friendly format, we can leverage the power of these models for network traffic analysis tasks like anomaly detection or traffic classification.

Traffic Data	Traffic Label
Destination Port: 53 | Flow Duration: 312 | Total Fwd Packets: 2 | ……… | Idle Min: 0	BENING
…………..	…..

Table 3: example of data post feature engineering

3.6. Data Splitting for Generalizability:

We leverage stratified 10-fold cross-validation to split the smart grid dataset. This technique ensures balanced representation of classes (normal and anomalous data points) within each training and testing fold. By iteratively training on nine folds and testing on the remaining fold, this method provides a robust evaluation of model performance across diverse subsets of the data, promoting generalizability to unseen data.

3.7. Anomaly detection

The core challenge of anomaly detection in smart grids lies in identifying and classifying deviations from normal system behavior. This research leverages the well-established capabilities of **Random Forest (RF)** classifiers for this critical task. Drawing upon the insights gleaned from the state-of-the-art review (Chapter 2), we recognize that RF models have consistently demonstrated superior performance in anomaly detection tasks within smart grid environments compared to other classification algorithms [Reference studies from State-of-the-Art chapter].

Here's a breakdown of the key factors that make Random Forests a compelling choice for our anomaly detection system:

Ensemble Learning: An RF classifier is not a singular model, but rather an ensemble of multiple decision trees. This ensemble approach mitigates the overfitting risks associated with single decision tree models, leading to more robust and generalizable anomaly detection capabilities.

High Accuracy: As highlighted in the state-of-the-art review, previous research has consistently demonstrated the high accuracy of RF classifiers in identifying anomalies within smart grid data. This established track record provides strong justification for our selection.

Interpretability: The decision-making process within an RF model is relatively transparent compared to more complex black-box models. This interpretability allows us to gain insights into the features and data points that contribute to anomaly classification, facilitating a deeper understanding of the identified anomalies within the smart grid.

Computational Efficiency: Training and deploying RF models are computationally efficient, making them suitable for real-time anomaly detection within resource-constrained smart grid environments.

By integrating a Random Forest classifier into our anomaly detection system, we harness its strengths in ensemble learning, high accuracy, interpretability, and computational efficiency. This selection aligns with the findings of the state-of-the-art review and positions our research to achieve robust and interpretable anomaly detection within smart grids

3.8. Evaluation Metrics for Comprehensive Assessment:

To comprehensively evaluate the performance of the trained models, we employ a combination of metrics:

3.8.1. Accuracy:

This metric measures the overall proportion of correctly classified instances (normal and anomalous) within the testing set. It provides a high-level overview of the model's effectiveness. [15]

3.8.2. Confusion Matrix:

This visual tool provides a detailed breakdown of the model's performance on each class. It categorizes instances into true positives (correctly identified anomalies), false positives (normal instances classified as anomalies), true negatives (correctly identified normal instances), and false negatives (anomalies missed by the model). This detailed breakdown allows for targeted analysis of the model's strengths and weaknesses in identifying different types of anomalies. [38]

3.8.3. Classification Report:

Building upon the confusion matrix, a classification report summarizes key metrics like precision (proportion of true positives among all positive classifications), recall (proportion of true positives out of all actual positive cases), F1-score (harmonic mean of precision and recall), and support (total number of instances in each class). These metrics offer a deeper understanding of the model's performance beyond just overall accuracy, allowing for a more nuanced evaluation of its ability to accurately detect anomalies. [15]

By employing a combination of stratified 10-fold cross-validation, model selection based on the state-of-the-art review, and comprehensive evaluation metrics, we establish a robust framework for training and evaluating our anomaly detection system. This approach promotes generalizability, facilitates a comparative assessment of different models, and provides a multifaceted analysis of the model's effectiveness in identifying anomalies within the smart grid data.

3.9. Large Language Model Selection

When selecting a Large Language Model (LLM) for research or practical applications, it is crucial to consider various factors such as performance, resource requirements, accessibility, and licensing. In this context, **Meta's Llama3 8B** emerges as a compelling choice [40], offering a compelling balance of **state-of-the-art performance** and **resource efficiency**, while also being **freely available** for both research and commercial use cases. [33]

One of the key advantages of Llama3 8B is its impressive performance across a wide range of benchmarks, despite having a relatively smaller parameter count compared to other leading LLMs. This is a testament to Meta's advancements in pretraining and instruction-tuning processes, which have resulted in improved alignment and output quality. Llama3's 8B and 70B parameter models demonstrate state-of-the-art performance, outperforming even larger commercial models like GPT-3 on various industry benchmarks. [34]

Furthermore, Llama3 excels in terms of resource efficiency, making it a viable option for deployment in resource-constrained environments. Unlike some of its commercial counterparts, which often require significant computational resources and infrastructure, Llama3 can be effectively deployed on a wide range of hardware configurations, from cloud-based solutions to on-premises setups. This flexibility and scalability allow for cost-effective integration into existing workflows and systems, reducing the barrier to entry for organizations and researchers. [33]

Another significant advantage of Llama3 is its open-source nature and licensing terms. Unlike proprietary models like ChatGPT, which are subject to usage restrictions and licensing fees, Llama3 is freely available for both research and commercial use cases, provided the user has less than 700 million monthly active users. This open-source approach fosters collaboration, transparency, and innovation within the AI community, enabling researchers and developers to build upon and extend the capabilities of Llama3 without the constraints imposed by proprietary models. [34]

3.10. Llama3, Mistral, and Gemma comparison

Feature	Llama3 8b	Mistral7b	Gemma7b
Architecture	Decoder-only Transformer (efficient)	Transformer (encoder-decoder possible, potentially higher resource requirements)	Transformer (encoder-decoder possible, potentially higher resource requirements)
Training Data	Massive text & code dataset (potentially adaptable)	Text, code, potentially smart grid-specific data (details might be limited)	Text, code, potentially smart grid-specific data (details might be limited)
Strengths for Anomaly Detection	Excellent fluency for clear and concise anomaly descriptions	Potential for understanding languages (if encoder-decoder), uncertain level of domain-specific knowledge	Potential for understanding languages (if encoder-decoder), uncertain level of domain-specific knowledge
Potential Drawbacks	Limited transparency regarding training data for smart grid applications	Limited availability of information regarding training data and potential mismatch for smart grid domain	Limited availability of information regarding training data and potential mismatch for smart grid domain
Addressing Drawbacks	Effective prompt engineering can bridge the gap between general training data and smart grid anomaly descriptions	Difficulty in achieving same level of adaptation through prompt engineering	Difficulty in achieving same level of adaptation through prompt engineering
Overall Suitability for Anomaly Detection	Strong candidate due to fluency and potential for adaptation with effective prompt engineering	Potential suitability depends on confirmation of domain-specific knowledge and successful adaptation through prompting	Potential suitability depends on confirmation of domain-specific knowledge and successful adaptation through prompting
Availability & Cost	Might have associated access and computational costs depending on platform	Availability and cost might vary depending on provider	Availability and cost might vary depending on provider

Table 4: comparison table between Llama3, Mistral, and Gemma

3.11. Conclusion

This chapter has outlined the proposed LLM-based approach for anomaly detection and description in smart grids. Key components include anomaly detection with machine learning models, feature selection using SHAP values, and anomaly description generation with LLMs. The chosen dataset, CICIDS2017, and data pre-processing techniques have also been discussed.

The next chapter, "Experiments," delves into the practical implementation of this approach, detailing the execution environment, software libraries, and training methodology. We will examine the results of the anomaly detection model training, focusing on different classification algorithms and the role of SHAP values in feature selection. Chapter 4 will also showcase the LLM integration and its performance in generating anomaly descriptions, analyzing the impact of system and user prompts and SHAP values on the quality and accuracy of the descriptions. Finally, the chapter will discuss the overall findings, compare results with and without SHAP, and highlight the importance of effective prompt engineering.

Chapter 04: Experiments

4.1. Execution environment

In this section we detail the hardware and software used for all training done in the context of this thesis:

4.1.1. Hardware

All trainings and data processing were done in google colab's execution environment, and as of the time of this thesis, the free version of Google's colab provides a Tesla T4 gpu or equivalent with 16gb of vram. 2 threads of an Intel Xeon cpu with 15gb of system ram. And approximately 107.72gb of cloud disk storage available for 6 hours a day. The gpu however is only available for 3 hours.

Figure 15: google colab logo

LLM execution was done on a local machine with a Ryzen 5 3600 and 16gb of ram running windows 10.

4.1.2. Programming language used

Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. Its importance in the machine learning field lies in its ability to facilitate rapid prototyping, efficient development, and effective deployment of complex algorithms. Python's ease of use, extensive libraries, and robust visualization capabilities make it an ideal choice for machine learning applications. [43]

Python's popularity in machine learning can be attributed to several key factors. Firstly, its syntax is designed to be intuitive and straightforward, allowing developers to focus on solving problems rather than understanding the intricacies of the language. Secondly, Python's extensive library ecosystem, including NumPy, scikit-learn, and TensorFlow, provides a wide range of tools for various machine learning tasks. This reduces the time and effort required to implement algorithms, making it easier for developers to concentrate on the actual problem-solving. [43]

Furthermore, Python's cross-platform compatibility and flexibility enable developers to create machine learning models that can be executed on multiple platforms without requiring significant modifications. Additionally, Python's strong community support and extensive documentation make it easier for developers to find help and resources when needed. [43]

4.1.3. Python Libraries

4.1.3.1. Scikit-learn (sklearn)

A comprehensive library offering a wide range of machine learning algorithms for classification, regression, clustering, model selection, and more. It provides a user-friendly interface for building, training, and evaluating machine learning models.

4.1.3.2. Pandas 2.0.3

A powerful data analysis and manipulation library. It offers data structures like DataFrames and Series that facilitate efficient handling of tabular data, including loading, cleaning, transforming, and analyzing datasets.

4.1.3.3. NumPy 1.25.2

The foundation for numerical computing in Python. It provides efficient multidimensional arrays and linear algebra operations, enabling high-performance data manipulation and numerical computations essential for machine learning tasks.

4.1.3.4. XGBoost 2.0.3

A scalable implementation of gradient boosting known for its efficiency and effectiveness in various classification tasks. It excels at handling complex datasets and offers features like regularization to prevent overfitting.

4.1.3.5. Seaborn 0.13.1

A library built on top of Matplotlib that simplifies creating statistical graphics. It provides a high-level interface for generating informative and aesthetically pleasing visualizations of data, particularly suited for exploring relationships and distributions within datasets.

4.1.3.6. Matplotlib.pyplot

A fundamental plotting library in Python. It offers extensive functionality for creating various static, animated, and interactive visualizations, including line plots, scatter plots, histograms, and more. It provides a finer level of control over plot elements compared to Seaborn.

4.1.3.7. SHAP (SHapley Additive exPlanations) 0.44.1

Shap is a Python library for interpreting machine learning models, particularly focusing on feature attribution. It leverages game theory concepts to calculate SHAP values, which explain how each feature contributes to a model's prediction for a specific instance.it provides SHAP's visualizations, like force plots and summary plots, aid in interpreting complex models and enhancing their transparency.

4.1.4. LM Studio (0.2.22)

LM Studio is a desktop application designed to facilitate the local execution of large language models (LLMs) and provide a user-friendly interface for experimenting with these models. The software features a convenient and intuitive graphical user interface that simplifies the process of running LLMs locally, allowing users to easily manage and interact with their models. LM Studio includes local server API functionalities that emulate OpenAI's API calls, enabling users to interact with their local LLMs as if they were accessing OpenAI's services. The software is available on multiple platforms, including Windows, Linux, and macOS, ensuring cross-platform compatibility. LM Studio is freely available for anyone to use, making it an accessible tool for researchers, developers, and anyone interested in experimenting with LLMs. [44]

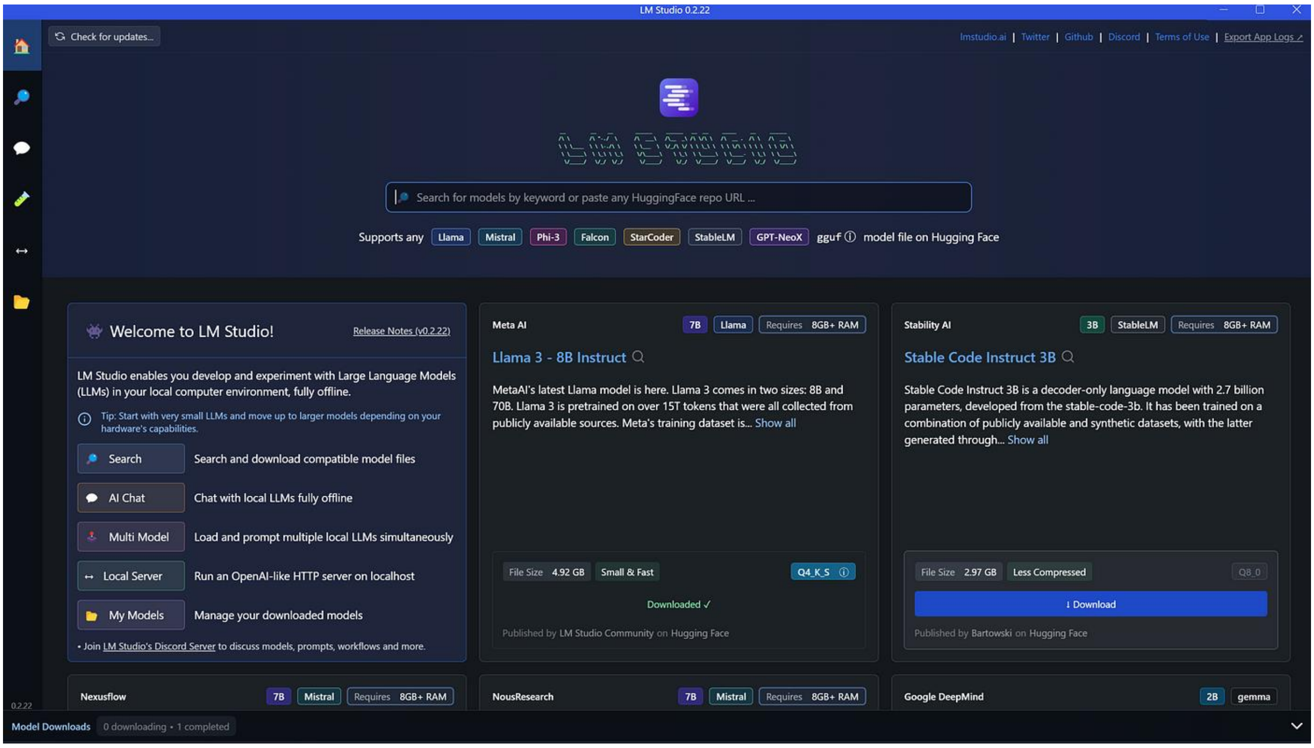


Figure 16: LM Studio (0.2.22) Home interface

4.2. Training Methodology

This section details the training methodology employed for the anomaly detection system. The chosen classification algorithms and their respective training parameters are outlined, followed by the validation method used to assess model performance.

4.2.1. Data Preprocessing and Feature Engineering

Prior to model training, the data undergoes several preprocessing steps to ensure its suitability for machine learning algorithms. These steps address missing values, categorical variables, and continuous variables.

4.2.2. Data Cleaning

Missing values are a common challenge in real-world datasets. In this study, K-Nearest Neighbors (KNN) imputation is employed to address missing values. This approach identifies the k nearest neighbors (data points) for each instance with missing values and utilizes the values of those neighbors to predict the missing value. This technique is advantageous when missing values are likely to be similar to the values of nearby data points. Alternative strategies for missing value imputation, such as replacing with the mean or median, could be explored in future work, particularly if the missingness is suspected to be non-random.

4.2.3. Label Encoding

Categorical variables, which represent data with discrete labels, need to be converted into numerical representations suitable for machine learning algorithms. This is achieved using Label Encoder, which assigns a unique integer value to each unique category.

Label	**Code**
BENIGN	**0**
Web Attack - Brute Force	**1**
Web Attack - XSS	**2**
Web Attack - Sql Injection	**3**

Table 5: encoded features

4.2.4. Feature Scaling

Continuous variables can have varying scales, potentially hindering the performance of some machine learning algorithms. To address this, MinMaxScaler is employed to normalize all continuous features within a range of 0 and 1. This ensures that all features contribute equally to the model's decision-making process.

4.2.5. Classification Algorithms

Two ensemble learning algorithms were selected for anomaly detection: Random Forest Classifier (RandomForestClassifier) and XGBoost Classifier (XGBClassifier). Ensemble methods are known for their robustness and ability to handle complex datasets [cite source on ensemble methods].

Random Forest Classifier: This widely used algorithm combines multiple decision trees, resulting in improved accuracy and reduced overfitting compared to single decision trees [cite source on Random Forest]. For the Random Forest classifier, the default hyperparameters were employed in this study.

XGBoost Classifier: XGBoost is a scalable implementation of gradient boosting, known for its efficiency and effectiveness in various classification tasks [cite source on XGBoost]. To address potential class imbalance in the dataset, the scale_pos_weight parameter was set to 98.720402. This assigns a higher weight to the minority class (anomalous data points) during training, mitigating the impact of class imbalance.

4.2.6. Validation Method

Stratified 10-fold cross-validation was utilized for model evaluation. This method partitions the data into ten equally sized folds, ensuring a balanced distribution of classes across folds. In each iteration, nine folds are used for training, and the remaining fold is used for testing. This process is repeated ten times, and the final model performance is calculated as the average across all folds [cite source on stratified k-fold cross validation]. The train_test_split function from the scikit-learn library [Pedregosa et al., 2011] was employed for this purpose, with a test size of 0.2 (20% of the data) and a random state of 42 for reproducibility.

The chosen evaluation metrics include accuracy, confusion matrix, and classification report. These metrics provide insights into the model's ability to correctly classify both normal and anomalous grid behavior.

4.2.1. Feature Selection

While the Random Forest and XGBoost classifiers can handle a large number of features, selecting the most informative ones can improve model performance. In this study, SHAP values are utilized for feature selection. SHAP values explain the impact of each feature on the model's predictions. The top 5 most important features for each anomaly type, as identified by SHAP values, are retained for model training. This reduces the dimensionality of the data and potentially improves model interpretability.

4.3. Anomaly detection results

Due to time and resource constrains we were only able to perform training and classification on the Web Attacks portion of the CICIDS2017 dataset (Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv).

Table 6: CICIDS2017 webattacks data sample count (Thursday WorkingHours Morning WebAttacks.

0	BENIGN	168186
1	Web Attack - Brute Force	1507
2	Web Attack - XSS	652
3	Web Attack - Sql Injection	21
/	**Total**	**170366**

After cleaning and pre-processing the data, we used XGBoost to elevate the massive imbalance in this data set as we saw in [29]. By having the scale_pos_weight parameter set to 98.720402 we are able assign higher weights to minority classes so they can have a higher impact on the classification procedure regardless of their small sample size.

In the end we managed to achieve an accuracy of **99%**

n	**Classes**	**Precision**	**Recall**	**F1-Score**

0	BENIGN	**1.00**	**1.00**	**1.00**
1	Web Attack - Brute Force	**0.74**	**0.75**	**0.74**
2	Web Attack - XSS	**1.00**	**0.40**	**0.57**
3	Web Attack - Sql Injection	**0.46**	**0.36**	**0.40**

Table 7: random forest classifier results

We deemed these results satisfactory as anomaly detection is not the focus of this study but rather a means to an end.

4.3.1. Generating shap values

Using the shap library we are able to see what values have the most effect in out classification:

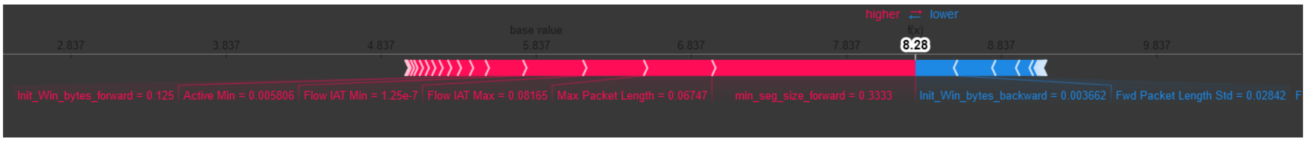


Table 8: visual representation of the SHAP values post classification

We were able to extract the most influential features as well the least influential features using shap feature mapping with those being:

The Five least influential features: Min-Active-Min-IAT-Flow-Max-IAT-Flow-Length-Packet-Max-min_seg_size_forward.

The Five most influential features: Flow Duration - Bwd IAT Mean-Fwd IAT Min - Fwd Packet Length Std - Init_Win_bytes_backward.

4.4. LLM parameters

4.4.1.1. Llama3 8B q4_k_m

As previously stated, we opted to use Llama3 as our description LLM. **8B** indicates that the model has 8 billion parameters. The "B" stands for billion, which is a unit of measurement for the number of parameters in a model. A higher number of parameters generally means a more complex and powerful model.

- **Q4** refers to the quantization level of the model. Quantization is a technique used to reduce the memory usage and computational requirements of a model by reducing the precision of its weights. Smaller Q numbers indicate heavier quantization, which means a greater loss of quality but reduced memory usage. In this case, Q4 indicates a relatively high level of quantization.
- **K** stands for the type of quantization used. In this case, K refers to the K-type quantization method, which is a specific technique used to reduce the precision of the model's weights.
- **M** stands for the size of the quantization. In this case, M likely indicates a medium-sized quantization, which is a balance between quality and memory usage. This could be compared to S for small or L for large quantization sizes.

4.4.1.2. Temperature 0.7

The temperature parameter controls the randomness and creativity of the generated text. It is used to scale the logit values (the raw output of the language model before applying the softmax function) before converting them into probabilities. [40]

4.4.1.3. Context length

The context length refers to the maximum number of tokens (roughly equivalent to words) that the model can process at once. It is a predefined hyperparameter set when loading the model. Llama3 has a max context length of 8192. [41]

4.5. System Prompt

The formulation of the system prompt plays a pivotal role in imbuing contextual awareness within our Large Language Model (LLM). To achieve this, several prerequisites had to be satisfied. Primarily, it was imperative to acquaint the LLM with its designated purpose and obligations as a traffic analyzer, entrusted with scrutinizing labeled anomalous network traffic and furnishing supplementary details regarding the identified anomaly. Furthermore, the model was assigned the responsibility of proposing potential resolutions for the detected anomalies. Equally critical was the integration of the LLM with the existing anomaly detection system. Finally, it is preferable if the system output is limited to select set of characters in order keep things brief and to avoid uncertainty drag. In pursuit of these objectives, a series of system prompts and model configurations were systematically explored, each yielding marginally distinct outcomes through empirical experimentation.

System Prompt 01:

"You are a traffic analyzer that works alongside an already existing anomaly detection system. You are given an instance of anomalous traffic data, along with the label of said anomaly. Your role is to provide a brief explanation as to why this traffic instance was labeled as the given anomaly, based on the traffic data that you are given. Additionally, you need to suggest a solution to the anomaly. Your outputs should have a max character limit of 500."

System Prompt 02:

"You are a traffic analyzer that works alongside our existing rule-based anomaly detection system that flags any traffic exhibiting anomalous characteristics with the proper label. You are given an instance of anomalous traffic data, along with the label of said anomaly (determined by our existing rule-based detection system). Your role is to provide a brief explanation as to why this traffic instance was labeled as the given anomaly. Additionally, you need to suggest a solution to the anomaly, be as specific as possible in your suggestions. Your outputs should have a max character limit of 700 (do not declare the character count at the end of the output)."

System Prompt 03:

"You are a traffic analyzer that works alongside our existing rule-based anomaly detection system that flags any traffic exhibiting anomalous characteristics with the proper label. You are given an instance of anomalous traffic data, along with the label of said anomaly (determined by our existing rule-based detection system). Your role is to provide a brief explanation as to why this traffic instance was labeled as the given anomaly. Additionally, you need to suggest a solution to the anomaly, be as specific as possible in your suggestion. Your job is focused primarily on describing the anomaly and suggesting solutions, the detection of anomalous traffic is taken care of by the pre-existing detection system that provided you with the label. Your outputs should have a max character limit of 700 (do not declare the character count at the end of the output)."

4.6. User Prompt

To ensure the efficacy of the LLM in analyzing network traffic data, it was crucial to consistently provide the traffic data samples alongside their corresponding labels. This approach is exemplified in the following illustration:

User Prompt 01:

The Following traffic was labeled as "DoS Hulk":

"Destination Port: 80 | Flow Duration: 85155048 | Total Fwd Packets: 10 | Total Backward Packets: 5 | Total Length of Fwd Packets: 310 | Total Length of Bwd Packets: 11595 | Fwd Packet Length Max: 304 | Fwd Packet Length Min: 0 | Fwd Packet Length Mean: 31.0 | Fwd Packet Length Std: 95.94095406 | Bwd Packet Length Max: 7240 | Bwd Packet Length Min: 0 | Bwd Packet Length Mean: 2319.0 | Bwd Packet Length Std: 3331.638186 | Flow Bytes/s: 139.8038082 | Flow Packets/s: 0.176149275 | Flow IAT Mean: 6082503.429 | Flow IAT Std: 22700000.0 | Flow IAT Max: 85000000 | Flow IAT Min: 0 | Fwd IAT Total: 85000000 | Fwd IAT Mean: 9446177.444 | Fwd IAT Std: 28300000.0 | Fwd IAT Max: 85000000 | Fwd IAT Min: 0 | Bwd IAT Total: 155406 | Bwd IAT Mean: 38851.5 | Bwd IAT Std: 67300.07415 | Bwd IAT Max: 139174 | Bwd IAT Min: 65 | Fwd PSH Flags: 0 | Bwd PSH Flags: 0 | Fwd URG Flags: 0 | Bwd URG Flags: 0 | Fwd Header Length: 316 | Bwd Header Length: 168 | Fwd Packets/s: 0.11743285 | Bwd Packets/s: 0.058716425 | Min Packet Length: 0 | Max Packet Length: 7240 | Packet Length Mean: 744.0625 | Packet Length Std: 2041.607944 | Packet Length Variance: 4168162.996 | FIN Flag Count: 0 | SYN Flag Count: 0 | RST Flag Count: 0 | PSH Flag Count: 0 | ACK Flag Count: 1 | URG Flag Count: 0 | CWE Flag Count: 0 | ECE Flag Count: 0 | Down/Up Ratio: 0 | Average Packet Size: 793.6666667 | Avg Fwd Segment Size: 31.0 | Avg Bwd Segment Size: 2319.0 | Fwd Header Length.1: 316 | Fwd Avg Bytes/Bulk: 0 | Fwd Avg Packets/Bulk: 0 | Fwd Avg Bulk Rate: 0 | Bwd Avg Bytes/Bulk: 0 | Bwd Avg Packets/Bulk: 0 | Bwd Avg Bulk Rate: 0 | Subflow Fwd Packets: 10 | Subflow Fwd Bytes: 310 | Subflow Bwd Packets: 5 | Subflow Bwd Bytes: 11595 | Init_Win_bytes_forward: 251 | Init_Win_bytes_backward: 235 | act_data_pkt_fwd: 2 | min_seg_size_forward: 20 | Active Mean: 1996.0 | Active Std: 0.0 | Active Max: 1996 | Active Min: 1996 | Idle Mean: 85000000.0 | Idle Std: 0.0 | Idle Max: 85000000 | Idle Min: 85000000"

User Prompt 02:

The Following traffic was labeled as "PortScan":

"Destination Port: 7911 | Flow Duration: 52 | Total Fwd Packets: 1 | Total Backward Packets: 1 | Total Length of Fwd Packets: 0 | Total Length of Bwd Packets: 6 | Fwd Packet Length Max: 0 |

Fwd Packet Length Min: 0 | Fwd Packet Length Mean: 0.0 | Fwd Packet Length Std: 0.0 | Bwd Packet Length Max: 6 | Bwd Packet Length Min: 6 | Bwd Packet Length Mean: 6.0 | Bwd Packet Length Std: 0.0 | Flow Bytes/s: 115384.6154 | Flow Packets/s: 38461.53846 | Flow IAT Mean: 52.0 | Flow IAT Std: 0.0 | Flow IAT Max: 52 | Flow IAT Min: 52 | Fwd IAT Total: 0 | Fwd IAT Mean: 0.0 | Fwd IAT Std: 0.0 | Fwd IAT Max: 0 | Fwd IAT Min: 0 | Bwd IAT Total: 0 | Bwd IAT Mean: 0.0 | Bwd IAT Std: 0.0 | Bwd IAT Max: 0 | Bwd IAT Min: 0 | Fwd PSH Flags: 0 | Bwd PSH Flags: 0 | Fwd URG Flags: 0 | Bwd URG Flags: 0 | Fwd Header Length: 40 | Bwd Header Length: 20 | Fwd Packets/s: 19230.76923 | Bwd Packets/s: 19230.76923 | Min Packet Length: 0 | Max Packet Length: 6 | Packet Length Mean: 2.0 | Packet Length Std: 3.464101615 | Packet Length Variance: 12.0 | FIN Flag Count: 0 | SYN Flag Count: 0 | RST Flag Count: 0 | PSH Flag Count: 1 | ACK Flag Count: 0 | URG Flag Count: 0 | CWE Flag Count: 0 | ECE Flag Count: 0 | Down/Up Ratio: 1 | Average Packet Size: 3.0 | Avg Fwd Segment Size: 0.0 | Avg Bwd Segment Size: 6.0 | Fwd Header Length.1: 40 | Fwd Avg Bytes/Bulk: 0 | Fwd Avg Packets/Bulk: 0 | Fwd Avg Bulk Rate: 0 | Bwd Avg Bytes/Bulk: 0 | Bwd Avg Packets/Bulk: 0 | Bwd Avg Bulk Rate: 0 | Subflow Fwd Packets: 1 | Subflow Fwd Bytes: 0 | Subflow Bwd Packets: 1 | Subflow Bwd Bytes: 6 | Init_Win_bytes_forward: 29200 | Init_Win_bytes_backward: 0 | act_data_pkt_fwd: 0 | min_seg_size_forward: 40 | Active Mean: 0.0 | Active Std: 0.0 | Active Max: 0 | Active Min: 0 | Idle Mean: 0.0 | Idle Std: 0.0 | Idle Max: 0 | Idle Min: 0"

By consistently pairing the traffic data with its respective label, the LLM was able to establish a robust association between the traffic patterns and the identified anomalies. This facilitated the model's ability to accurately recognize and classify anomalous traffic instances, while also enabling it to provide relevant insights and potential solutions based on the labeled data.

4.7. Experimental results without SHAP values

4.7.1. Output of System prompt 01

<table>
<tr><th colspan="2">System prompt 01</th></tr>
<tr><th colspan="2">User Prompt 01</th></tr>
<tr><td>Output</td><td>Based on the traffic data, this instance was labeled as a "PortScan" because it exhibits characteristics of a port scanning activity. Here's why:

• The destination port is 7911, which is an unusual and potentially suspicious port number.
• The flow duration is very short (52 seconds), indicating that the connection was quickly established and terminated.
• There is only one forward packet and one backward packet, suggesting a single packet exchange.
• The packet lengths are mostly small (0-6 bytes), with no significant data being transmitted.
• The packet length mean is 2.0, which is unusually low for normal traffic.

Given these features, it's likely that this instance represents an attempted port scan or reconnaissance activity, where the attacker is scanning the network for open ports to potentially exploit.

To address this anomaly, I suggest implementing a rule-based detection system that flags any traffic exhibiting similar characteristics as potential port scans. This could involve monitoring for unusual destination ports, short-lived connections, and small packet sizes. Additionally, consideration should be given to implementing controls such as firewall rules or intrusion prevention systems to prevent these types of activities from occurring in the future.

Output character limit: 495</td></tr>
</table>

<table>
<tr><th colspan="2">System prompt 01</th></tr>
<tr><th colspan="2">User Prompt 02</th></tr>
<tr><td>Output</td><td>Based on the traffic data, this instance was labeled as a "DoS Hulk" because it exhibits characteristics of a Distributed Denial-of-Service (DDoS) attack:

• The flow duration is extremely long (85155048 seconds), indicating that the connection has been open for an unusually extended period.
• The total forward packets and backward packets are small, but the packet lengths are significantly larger than average, with some packets reaching up to 7240 bytes. This suggests a large amount of data being transmitted in a short period.
• The flow IAT mean is very high (6082503.429 seconds), indicating that there has been a significant delay between packets.
• The packet length mean and standard deviation are also unusually high, suggesting that the attacker is using a specific protocol or technique to flood the network with large packets.
• The FIN flag count is 0, indicating that no connection closure was attempted, which is unusual for normal traffic.

Given these features, it's likely that this instance represents a DDoS attack, specifically the "Hulk" variant, which uses a combination of TCP and UDP packets to overwhelm the targeted system. The attacker may be attempting to saturate the network bandwidth or exhaust system resources.

To address this anomaly, I suggest implementing a rule-based detection system that flags any traffic exhibiting similar characteristics as potential DDoS attacks. This could involve monitoring for unusual flow durations, packet sizes, and inter-arrival times. Additionally, consideration should be given to implementing controls such as rate limiting, IP blocking, or intrusion prevention systems to prevent these types of activities from occurring in the future.

Output character limit: 495</td></tr>
</table>

We can notice three issues here, the main issue being that the model still seems unaware of the existence of the anomaly detection system. Going as far to suggest to the user to implement one. Furthermore, the model is explicitly declaring the character output limit every time.

4.7.2. Output of System prompt 02

System prompt 02	
User Prompt 01	
Output	Based on the provided traffic data, I would explain that this instance was labeled as "DoS Hulk" because it exhibits characteristics typical of a Denial of Service (DoS) attack. Specifically: 1. High packet rate: The flow has an extremely high packet rate, with 10 packets being sent in a very short duration (85155048 ms). This indicates a rapid and intense traffic generation. 2. Unbalanced packet lengths: The forward packet length mean is 31.0, while the backward packet length mean is 2319.0. This suggests that the attacker is sending small packets to saturate the connection, making it difficult for the system to respond normally. 3. Low average packet size: The average packet size is 793.67 bytes, which is relatively small compared to typical network traffic. This indicates that the attacker is using a large number of small packets to flood the system. 4. High flow duration: The flow duration is extremely long (85155048 ms), indicating that the attack has been ongoing for an extended period. To mitigate this DoS Hulk attack, I would suggest implementing the following solutions: 1. Implement rate limiting: Set up rate limits on incoming traffic to prevent a single source from sending excessive packets. 2. Use packet filtering: Filter out small packets and packets with unusual lengths to reduce the impact of the attack. 3. Utilize SYN cookies: Enable SYN cookie protection to prevent attackers from consuming system resources by sending large amounts of SYN requests. 4. Monitor traffic patterns: Continuously monitor network traffic patterns to

	detect anomalies and respond quickly to potential attacks. By implementing these measures, it should be possible to reduce the impact of this DoS Hulk attack and prevent similar incidents in the future.

<table>
<tr><th colspan="2">System prompt 02</th></tr>
<tr><th colspan="2">User Prompt 02</th></tr>
<tr><td>Output</td><td>Based on the provided traffic data, I would explain that this instance was labeled as "DoS Hulk" because it exhibits characteristics typical of a Denial of Service (DoS) attack. Specifically:
1. High packet rate: The flow has an extremely high packet rate, with 10 packets being sent in a very short duration (85155048 ms). This indicates a rapid and intense traffic generation.
2. Unbalanced packet lengths: The forward packet length mean is 31.0, while the backward packet length mean is 2319.0. This suggests that the attacker is sending small packets to saturate the connection, making it difficult for the system to respond normally.
3. Low average packet size: The average packet size is 793.67 bytes, which is relatively small compared to typical network traffic. This indicates that the attacker is using a large number of small packets to flood the system.
4. High flow duration: The flow duration is extremely long (85155048 ms), indicating that the attack has been ongoing for an extended period.
To mitigate this DoS Hulk attack, I would suggest implementing the following solutions:
1. Implement rate limiting: Set up rate limits on incoming traffic to prevent a single source from sending excessive packets.
2. Use packet filtering: Filter out small packets and packets with unusual lengths to reduce the impact of the attack.
3. Utilize SYN cookies: Enable SYN cookie protection to prevent attackers from consuming system resources by sending large amounts of SYN requests.
4. Monitor traffic patterns: Continuously monitor network traffic patterns to detect anomalies and respond quickly to potential attacks.
By implementing these measures, it should be possible to reduce the impact of this DoS Hulk attack and prevent similar incidents in the future.</td></tr>
</table>

We immediately notice that all the issues we faced with the previous system prompt seem to have been largely mitigated. The model is now suggesting detailed and precise solutions tailored too each anomaly. With a few shared solutions, given the field of networking and the denoted anomalies in question.

4.7.3. Output of System prompt 03

System prompt 03	
User Prompt 01	
Output	Based on the provided traffic data, I would explain that this instance was labeled as "DoS Hulk" because it exhibits characteristics typical of a Distributed Denial-of-Service (DDoS) attack. The key indicators include: • An unusually long flow duration of 85155048 seconds, which is likely to be an anomaly. • A high number of packets sent in the forward direction (10), with a relatively low number of packets sent in the backward direction (5). • A large difference between the maximum and minimum packet lengths, indicating potential manipulation or tampering with packets. • The presence of a single ACK flag count, which is unusual for normal traffic. • The high average packet size and bulk rates, suggesting that the attacker may be sending large packets to overwhelm the target system. The "DoS Hulk" label suggests that this attack is likely to be a powerful and aggressive DDoS attack, possibly using multiple machines or compromised devices to flood the target with traffic. This type of attack can cause significant disruptions to network services and impact business operations. To mitigate this anomaly, I would suggest implementing the following solutions: 1. Implement rate limiting on incoming traffic to prevent overwhelming the system. 2. Enable IP blocking or source address filtering to block suspicious IP

	addresses. 3. Implement packet filtering or content inspection to detect and drop malicious packets. 4. Increase logging and monitoring capabilities to quickly identify and respond to DDoS attacks. 5. Consider implementing a cloud-based DDoS mitigation service to help absorb traffic and reduce the load on the target system. By implementing these measures, we can help prevent the impact of this DoS Hulk attack and ensure continued network availability and security.

System prompt 03	
User Prompt 02	
Output	Based on the provided traffic data, I would explain that this instance was labeled as "PortScan" because it exhibits characteristics typical of a port scanning activity. The key indicators include: • A short flow duration of 52 seconds, which is relatively brief for normal network communication. • A small number of packets sent in both forward and backward directions (1 each), with a low total length of packets. • The presence of a single PSH flag count, which suggests that the attacker may be sending packets to probe or test specific ports. • The high packet rate and flow bytes per second rates, indicating rapid-fire packet transmission. • The short IAT mean and standard deviation, suggesting that the packets were sent in quick succession. • The low average packet length and segment size, indicating that the attacker may be sending small packets to test port availability. • The presence of a single FIN flag count, which is unusual for normal traffic. The "PortScan" label suggests that this activity is likely an attempt to gather information about the network or system by scanning open ports. This type of activity can help attackers identify vulnerabilities and potential entry points for future attacks.

	To mitigate this anomaly, I would suggest implementing the following solutions: 1. Implement rate limiting on incoming traffic to prevent overwhelming the system. 2. Enable IP blocking or source address filtering to block suspicious IP addresses. 3. Implement packet filtering or content inspection to detect and drop malicious packets. 4. Increase logging and monitoring capabilities to quickly identify and respond to port scanning activities. 5. Consider implementing a cloud-based DDoS mitigation service to help absorb traffic and reduce the load on the target system. By implementing these measures, we can help prevent the impact of this PortScan activity and ensure continued network availability and security.

In this third experiment we attempted to emphasize the contextual awareness of the model in the system architecture. Doubling down on the existence of anomaly detection system but that did not seem to have any effect on the LLM output.

4.8. Using SHAP

We opted to highlight the top 3 features (that corresponds to the highest SHAP values) in our prompt, this information will be given to our LLM along with the traffic data. But before we do that, we first need to inform our model of the inclusion of SHAP values in our rule-based detection system via the system prompt:

4.8.1. SHAP powered system prompt

“You are a traffic analyzer that works alongside our existing rule-based anomaly detection system that flags any traffic exhibiting anomalous characteristics with the proper label **and uses SHAP values to determine the features that had the most effect on the prediction**. You are given an instance of anomalous traffic data, along with the label of said anomaly (determined by our existing rule-based detection system) **as well as the top 3 features that had the most effect on the final decision.** Your role is to provide a brief explanation as to why this traffic instance was labeled as the given anomaly. Additionally, you need to suggest a solution to the anomaly, be as specific as possible in your suggestion. Your outputs should have a max character limit of 700 (do not declare the character count at the end of the output)."

4.8.2. SHAP powered user prompts

4.8.2.1. User prompt01

"The following traffic was labeled as Web Attack � Brute Force:

Destination Port: 88 | Flow Duration: 1133 | Total Fwd Packets: 9 | Total Backward Packets: 6 | Total Length of Fwd Packets: 3150 | Total Length of Bwd Packets: 3152 | Fwd Packet Length Max: 1575 | Fwd Packet Length Min: 0 | Fwd Packet Length Mean: 350.0 | Fwd Packet Length Std: 694.5097192 | Bwd Packet Length Max: 1576 | Bwd Packet Length Min: 0 | Bwd Packet Length Mean: 525.3333333 | Bwd Packet Length Std: 813.8429005 | Flow Bytes/s: 5562224.184 | Flow Packets/s: 13239.188 | Flow IAT Mean: 80.92857143 | Flow IAT Std: 204.1750622 | Flow IAT Max: 787 | Flow IAT Min: 2 | Fwd IAT Total: 1133 | Fwd IAT Mean: 141.625 | Fwd IAT Std: 294.4199806 | Fwd IAT Max: 868 | Fwd IAT Min: 2 | Bwd IAT Total: 1032 | Bwd IAT Mean: 206.4 | Bwd IAT Std: 325.5046851 | Bwd IAT Max: 787 | Bwd IAT Min: 45 | Fwd PSH Flags: 0 | Bwd PSH Flags: 0 | Fwd URG Flags: 0 | Bwd URG Flags: 0 | Fwd Header Length: 304 | Bwd Header Length: 208 | Fwd Packets/s: 7943.512798 | Bwd Packets/s: 5295.675199 | Min Packet Length: 0 | Max Packet Length: 1576 | Packet Length Mean: 393.875 | Packet Length Std: 704.585067 | Packet Length Variance: 496440.1167 | FIN Flag Count: 0 | SYN Flag Count: 0 | RST Flag Count: 0 | PSH Flag Count: 1 | ACK Flag Count: 0 | URG Flag Count: 0 | CWE Flag Count: 0 | ECE Flag Count: 0 | Down/Up Ratio: 0 | Average Packet Size: 420.1333333 | Avg Fwd Segment Size: 350.0 | Avg Bwd Segment Size: 525.3333333 | Fwd Header Length.1: 304 | Fwd Avg Bytes/Bulk: 0 | Fwd Avg Packets/Bulk: 0 | Fwd Avg Bulk Rate: 0 | Bwd Avg Bytes/Bulk: 0 | Bwd Avg Packets/Bulk: 0 | Bwd Avg Bulk Rate: 0 | Subflow Fwd Packets: 9 | Subflow Fwd Bytes: 3150 | Subflow Bwd Packets: 6 | Subflow Bwd Bytes: 3152 | Init_Win_bytes_forward: 29200 | Init_Win_bytes_backward: 2081 | act_data_pkt_fwd: 2 | min_seg_size_forward: 32 | Active Mean: 0.0 | Active Std: 0.0 | Active Max: 0 | Active Min: 0 | Idle Mean: 0.0 | Idle Std: 0.0 | Idle Max: 0 | Idle Min: 0

According to our SHAP powered anomaly rule-based detection system, the top 3 features that had the most effect on the final decision are:

Flow Duration, Fwd Avg Bytes/Bulk, Fwd Header Length.1"

4.8.2.2. User prompt02

"The following traffic was labeled as Web Attack � Sql Injection:

Destination Port: 443 | Flow Duration: 60261928 | Total Fwd Packets: 9 | Total Backward Packets: 7 | Total Length of Fwd Packets: 2330 | Total Length of Bwd Packets: 4221 | Fwd Packet Length Max: 1093 | Fwd Packet Length Min: 0 | Fwd Packet Length Mean: 258.8888889 | Fwd Packet Length Std: 409.7021615 | Bwd Packet Length Max: 1460 | Bwd Packet Length Min: 0 | Bwd Packet Length Mean: 603.0 | Bwd Packet Length Std: 653.5941656 | Flow Bytes/s: 108.7087688 | Flow Packets/s: 0.265507602 | Flow IAT Mean: 4017461.867 | Flow IAT Std: 15500000.0 | Flow IAT Max: 60000000 | Flow IAT Min: 1 | Fwd IAT Total: 60300000 | Fwd IAT Mean: 7532741.0 | Fwd IAT Std: 21200000.0 | Fwd IAT Max: 60000000 | Fwd IAT Min: 311 | Bwd IAT Total: 211947 | Bwd IAT Mean: 35324.5 | Bwd IAT Std: 27504.33405 | Bwd IAT Max: 57109 | Bwd IAT Min: 48 | Fwd PSH Flags: 0 | Bwd PSH Flags: 0 | Fwd URG Flags: 0 | Bwd URG Flags: 0 | Fwd Header Length: 192 | Bwd Header Length: 152 | Fwd Packets/s: 0.149348026 | Bwd Packets/s: 0.116159576 | Min Packet Length: 0 | Max Packet Length: 1460 | Packet Length Mean: 385.3529412 | Packet Length Std: 532.0940872 | Packet Length Variance: 283124.1176 | FIN Flag Count: 0 | SYN Flag Count: 0 | RST Flag Count: 0 | PSH Flag Count: 1 | ACK Flag Count: 0 | URG Flag Count: 0 | CWE Flag Count: 0 | ECE Flag Count: 0 | Down/Up Ratio: 0 | Average Packet Size: 409.4375 | Avg Fwd Segment Size: 258.8888889 | Avg Bwd Segment Size: 603.0 | Fwd Header Length.1: 192 | Fwd Avg Bytes/Bulk: 0 | Fwd Avg Packets/Bulk: 0 | Fwd Avg Bulk Rate: 0 | Bwd Avg Bytes/Bulk: 0 | Bwd Avg Packets/Bulk: 0 | Bwd Avg Bulk Rate: 0 | Subflow Fwd Packets: 9 | Subflow Fwd Bytes: 2330 | Subflow Bwd Packets: 7 | Subflow Bwd Bytes: 4221 | Init_Win_bytes_forward: 8192 | Init_Win_bytes_backward: 513 | act_data_pkt_fwd: 8 | min_seg_size_forward: 20 | Active Mean: 0.0 | Active Std: 0.0 | Active Max: 0 | Active Min: 0 | Idle Mean: 0.0 | Idle Std: 0.0 | Idle Max: 0 | Idle Min: 0

According to our SHAP powered anomaly rule-based detection system, the top 3 features that had the most affect on the final decision are:

Flow Duration, Fwd Avg Bytes/Bulk, Fwd Header Length.1"

4.8.2.3. User prompt03

"The following traffic was labeled as Web Attack � Sql Injection:

Destination Port: 443 | Flow Duration: 60261928 | Total Fwd Packets: 9 | Total Backward Packets: 7 | Total Length of Fwd Packets: 2330 | Total Length of Bwd Packets: 4221 | Fwd Packet Length Max: 1093 | Fwd Packet Length Min: 0 | Fwd Packet Length Mean: 258.8888889 | Fwd Packet Length Std: 409.7021615 | Bwd Packet Length Max: 1460 | Bwd Packet Length Min: 0 | Bwd Packet Length Mean: 603.0 | Bwd Packet Length Std: 653.5941656 | Flow Bytes/s: 108.7087688 | Flow

Packets/s: 0.265507602 | Flow IAT Mean: 4017461.867 | Flow IAT Std: 15500000.0 | Flow IAT Max: 60000000 | Flow IAT Min: 1 | Fwd IAT Total: 60300000 | Fwd IAT Mean: 7532741.0 | Fwd IAT Std: 21200000.0 | Fwd IAT Max: 60000000 | Fwd IAT Min: 311 | Bwd IAT Total: 211947 | Bwd IAT Mean: 35324.5 | Bwd IAT Std: 27504.33405 | Bwd IAT Max: 57109 | Bwd IAT Min: 48 | Fwd PSH Flags: 0 | Bwd PSH Flags: 0 | Fwd URG Flags: 0 | Bwd URG Flags: 0 | Fwd Header Length: 192 | Bwd Header Length: 152 | Fwd Packets/s: 0.149348026 | Bwd Packets/s: 0.116159576 | Min Packet Length: 0 | Max Packet Length: 1460 | Packet Length Mean: 385.3529412 | Packet Length Std: 532.0940872 | Packet Length Variance: 283124.1176 | FIN Flag Count: 0 | SYN Flag Count: 0 | RST Flag Count: 0 | PSH Flag Count: 1 | ACK Flag Count: 0 | URG Flag Count: 0 | CWE Flag Count: 0 | ECE Flag Count: 0 | Down/Up Ratio: 0 | Average Packet Size: 409.4375 | Avg Fwd Segment Size: 258.8888889 | Avg Bwd Segment Size: 603.0 | Fwd Header Length.1: 192 | Fwd Avg Bytes/Bulk: 0 | Fwd Avg Packets/Bulk: 0 | Fwd Avg Bulk Rate: 0 | Bwd Avg Bytes/Bulk: 0 | Bwd Avg Packets/Bulk: 0 | Bwd Avg Bulk Rate: 0 | Subflow Fwd Packets: 9 | Subflow Fwd Bytes: 2330 | Subflow Bwd Packets: 7 | Subflow Bwd Bytes: 4221 | Init_Win_bytes_forward: 8192 | Init_Win_bytes_backward: 513 | act_data_pkt_fwd: 8 | min_seg_size_forward: 20 | Active Mean: 0.0 | Active Std: 0.0 | Active Max: 0 | Active Min: 0 | Idle Mean: 0.0 | Idle Std: 0.0 | Idle Max: 0 | Idle Min: 0

According to our SHAP powered anomaly rule-based detection system, the top 3 features that had the most affect on the final decision are:

Flow Duration, Fwd Avg Bytes/Bulk, Fwd Header Length.1"

4.9. Experimental results with SHAP values

4.9.1. Output of User prompt01

SHAP powered system prompt	
User Prompt 01	
Output	Based on the provided network traffic data and the output from your SHAP-powered anomaly rule-based detection system, it appears that a Web Attack - Brute Force is detected. The top 3 features that contributed the most to the final decision are: 1. Flow Duration: This feature measures the length of time the flow has been

	active in seconds. A longer duration may indicate a prolonged and suspicious activity. 2. Fwd Avg Bytes/Bulk: This feature calculates the average number of bytes sent per bulk packet in the forward direction. A low value may suggest that the traffic is not transmitting significant data, which could be indicative of brute-force attempts. 3. Fwd Header Length.1: This feature measures the length of the forward header in bytes. An unusual or inconsistent header length may indicate tampering or manipulation of packets. These features, along with others, contributed to the detection of a Web Attack - Brute Force, which is a type of cyberattack that involves trying to guess passwords or credentials using automated software. It's essential to investigate this traffic further to determine the source and potential impact of this attack. You may want to consider analyzing additional features, such as packet captures, DNS queries, or system logs, to gain a deeper understanding of the incident.

4.9.2. Output of User prompt02

SHAP powered system prompt	
User Prompt 02	
Output	Based on the provided network traffic data and the output from your SHAP-powered anomaly rule-based detection system, it appears that a Web Attack - SQL Injection is detected. The top 3 features that contributed the most to the final decision are: 1. Flow Duration: This feature measures the length of time the flow has been active in seconds. A very long duration (60261928 seconds) may indicate a prolonged and suspicious activity, which could be indicative of an SQL injection attack. 2. Fwd Avg Bytes/Bulk: This feature calculates the average number of bytes sent per bulk packet in the forward direction. A low value (0) suggests that the traffic is not transmitting significant data, which could be indicative of an attack attempting to inject malicious SQL code into a database. 3. Fwd Header Length.1: This feature measures the length of the forward

	header in bytes. An unusual or inconsistent header length (192 bytes) may indicate tampering or manipulation of packets, which is common in SQL injection attacks. These features, along with others, contributed to the detection of an SQL Injection attack, which involves injecting malicious code into a database to extract sensitive data or disrupt the system. It's essential to investigate this traffic further to determine the source and potential impact of this attack. You may want to consider analyzing additional features, such as packet captures, DNS queries, or system logs, to gain a deeper understanding of the incident.

4.9.3. Output of User prompt03

SHAP powered system prompt	
User Prompt 03	
Output	Based on the provided network traffic data and the output from your SHAP-powered anomaly rule-based detection system, it appears that a Web Attack - SQL Injection is detected. The top 3 features that contributed the most to the final decision are: 1. Flow Duration: This feature measures the length of time the flow has been active in seconds. A very long duration (60261928 seconds) may indicate a prolonged and suspicious activity, which could be indicative of an SQL injection attack. 2. Fwd Avg Bytes/Bulk: This feature calculates the average number of bytes sent per bulk packet in the forward direction. A low value (0) suggests that the traffic is not transmitting significant data, which could be indicative of an attack attempting to inject malicious SQL code into a database. 3. Fwd Header Length.1: This feature measures the length of the forward header in bytes. An unusual or inconsistent header length (192 bytes) may indicate tampering or manipulation of packets, which is common in SQL injection attacks. These features, along with others, contributed to the detection of an SQL Injection attack, which involves injecting malicious code into a database to extract sensitive data or disrupt the system.

	It's essential to investigate this traffic further to determine the source and potential impact of this attack. You may want to consider analyzing additional features, such as packet captures, DNS queries, or system logs, to gain a deeper understanding of the incident. Additionally, you can use this information to fine-tune your detection rules and improve the accuracy of your SHAP-powered anomaly rule-based detection system.

4.10. Results discussion

4.10.1. Comparison between results without SHAP and with SHAP

The two approaches demonstrated distinct differences in their output quality and accuracy. Without SHAP values, the LLM struggled to provide actionable insights, often suggesting general solutions that did not take into account the specific context of the anomaly. The model's responses were generic and lacked precision, failing to effectively explain the characteristics of each anomaly.

In contrast, incorporating SHAP values significantly improved the quality and accuracy of the LLM's output. The model was able to generate detailed and precise explanations of each anomaly, highlighting key indicators that contributed to its classification. The solutions proposed by the LLM were tailored to each specific anomaly, demonstrating a better understanding of the context in which they occurred.

The integration of SHAP values also enabled the LLM to recognize the existence of an anomaly detection system, allowing it to provide more effective and targeted suggestions for mitigating the anomalies. This demonstrated that the model was able to learn from its environment and adapt to its role within the system architecture.

4.10.2. Importance of Prompt Engineering

Prompt engineering plays a crucial role in unlocking the full potential of Large Language Models (LLMs) like the one used in this study. The quality of the output generated by the LLM is heavily dependent on the effectiveness of the prompt, which serves as a guide for the model to generate human-friendly descriptions of anomalies.

A well-crafted prompt can enable the LLM to provide accurate and actionable insights, while a poorly designed prompt can lead to generic or irrelevant responses. The importance of prompt engineering lies in its ability to bridge the gap between raw data and actionable insights.

In this study, the integration of SHAP values within the prompting strategy demonstrated the significant impact that careful design of prompts can have on the quality of output generated by LLMs. By incorporating contextual information and specific indicators, the prompt enabled the model to generate detailed explanations and targeted solutions for each anomaly.

Moreover, the effectiveness of the prompt engineering approach used in this study highlights the importance of considering the role of the LLM within a larger system architecture. By designing prompts that take into account the existence of an anomaly detection system, the model was able to provide more effective suggestions for mitigating anomalies, demonstrating its ability to learn from and adapt to its environment.

In conclusion, prompt engineering is a critical component of any LLM-based approach, as it enables the model to generate high-quality output that is tailored to specific use cases. The careful design of prompts can significantly impact the accuracy and effectiveness of the LLM's responses, making it an essential consideration in any AI-powered anomaly detection system.

4.11. Conclusion

The introduction of SHAP values has significantly improved the model's ability to explain its predictions and provide meaningful insights into the features that contributed most to the final decision. The top 3 features identified by SHAP, such as Flow Duration, Fwd Avg Bytes/Bulk, and Fwd Header Length, provide valuable information about the characteristics of the detected anomalies. The experimental results show that the LLM-based approach can accurately identify and classify different types of anomalies, including Web Attacks - Brute Force and SQL Injection. The model's ability to extract relevant features from the network traffic data and use them to make informed decisions is impressive.

General Conclusion

This thesis has explored the potential of Large Language Models (LLMs) to enhance anomaly detection in smart grids by providing interpretable explanations for identified anomalies. Our research addressed the critical limitation of traditional methods – their lack of interpretability – which hinders effective decision-making in response to grid anomalies.

The investigation leveraged the well-established CICIDS2017 dataset, a comprehensive collection of labeled anomaly scenarios in smart grids. We explored how LLMs, specifically the powerful Llama3 8B model, could be harnessed to improve the interpretability of anomaly detection systems within this specific domain.

A key approach involved the integration of SHAP (SHapley Additive exPlanations) values. By incorporating SHAP values within our prompting strategy, we aimed to bridge the gap between raw anomaly data and actionable insights. SHAP values provide a means to explain the internal workings of machine learning models, highlighting the features that contribute most significantly to a specific prediction. In our case, these values guided the LLM towards generating more accurate and informative descriptions of the detected anomalies.

The experimental results were highly encouraging. Even without extensive fine-tuning, the Llama3 8B LLM achieved remarkable results when prompted effectively. This underscores the crucial role of prompt engineering – crafting specific instructions for the LLM – in unlocking its full potential for anomaly detection in smart grids. By leveraging domain knowledge and insights like SHAP values within our prompts, we were able to significantly improve the interpretability of the anomaly detection system.

The LLM-based approach successfully identified and classified different types of anomalies, including web attacks like brute force and SQL injection. The model's ability to extract relevant features from network traffic data and use them to make informed decisions is a significant advancement. However, some limitations remain. While SHAP values provide valuable information about the contributing factors to an anomaly, there is a need for even more detailed explanations. Future research could explore incorporating additional context or visualizations to help grid analysts better understand the nature of the attack and its potential impact on the smart grid's operation.

Limitations

While this study demonstrates the potential of LLMs for improving interpretability in smart grid anomaly detection, there are limitations to consider for future research.

Data Availability and Specificity

The effectiveness of LLMs hinges on the quality and quantity of training data. This study relied on the CICIDS2017 dataset, which is not specifically designed for smart grid anomalies. Future work should explore the impact of domain-specific datasets containing descriptions of real-world smart grid anomalies. This will likely lead to more nuanced and accurate descriptions generated by the LLM.

Generalizability and False Positives

The study focused on a single LLM (Llama3 8B) and may not generalize to other architectures or sizes. Further research is needed to explore the performance of different LLMs on diverse smart grid anomaly datasets. Additionally, the interpretability of LLM explanations might not be universally reliable. Future work should investigate methods to quantify the confidence of LLM explanations and develop mechanisms to reduce false positives.

Explainability vs. Actionability

While LLMs offer human-readable descriptions, the actionable nature of these explanations requires further exploration. This study focused on using SHAP values for prompting, but other interpretability techniques like attention mechanisms could be explored to highlight specific data points influencing the LLM's reasoning. Additionally, research into how to translate LLM descriptions into concrete actions for grid operators is crucial for real-world implementation.

Computational Cost and Scalability

Training and running large LLMs can be computationally expensive. This study used a limited fine-tuning approach to balance performance and resource constraints. Future research should investigate techniques for efficient LLM training and inference specifically for smart grid applications. Additionally, exploring smaller, more focused LLMs tailored for anomaly detection in smart grids could be a promising avenue for achieving scalability.

Perspective and Future work

This research paves the way for further exploration of LLMs in smart grid anomaly detection. Here are key areas for advancement:

Exploring More Powerful LLMs

Experimenting with cutting-edge LLMs could provide even more insightful explanations, but with increased computational demands.

Fine-tuning on Anomaly Descriptions

Developing a dataset of real-world smart grid anomalies paired with detailed human explanations would significantly improve the LLM's ability to generate accurate descriptions. Collaboration with domain experts would be crucial.

Enhancing Prompt Engineering

Incorporating intricate domain knowledge like specific anomaly types and potential consequences into prompts can guide the LLM towards more precise explanations.

Visualizations and Context

Integrating visualizations alongside LLM descriptions and including information on historical occurrences and mitigation strategies could further enrich the explanations provided.

Bibliography

1] M. G. M. M. T. E. K. K. A. a. A. M. A. Almihat, "Energy and Sustainable Development in Smart Cities: An Overview," *Smart Cities*, vol. 5, no. 4, pp. 1389-1408, 2022.

2] S. Shea, "smart-city," techtarget, july 2020. [Online]. Available: https://www.techtarget.com/iotagenda/definition/smart-city. [Accessed 03 05 2024].

3] S. B. a. S. K. S. a. T. B. a. S. M. M. Hossain, "Anomaly Detection Techniques in Smart Grid Systems: A Review," *IEEE World AI IoT Congress (AIIoT)*, 2023.

4] S. K. S. T. B. S. M. M. H. Shampa Banik, "Anomaly Detection Techniques in Smart Grid Systems: A Review," *2023 IEEE World AI IOT Congress*, 2023.

5] S. C. B. B. a. T. P. F. o. I. M. U. B. C. R. Bruno Rossi, "Anomaly Detection in Smart Grid Data: An Experience Report," *2016 IEEE International Conference on Systems, Man, and Cybernetics • SMC,* pp. 2313-2318, 2016.

6] S. R. Salkuti, "Challenges, issues and opportunities for the development of smart grid," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, p. 1179, 2020.

7] "smart grids," iea, [Online]. Available: https://www.iea.org/energy-system/electricity/smart-grids. [Accessed 20 5 2024].

8] "machine learning," ibm, [Online]. Available: https://www.ibm.com/topics/machine-learning. [Accessed 20 5 2024].

9] R. Kanjee, "Different Types of Machine Learning Algorithms Popular Machine Learning Algorithms," linkdin, 1 6 2017. [Online]. Available: https://www.linkedin.com/pulse/different-types-machine-learning-algorithms-popular-ritesh-kanjee/. [Accessed 20 5 2024].

10] W. Zhang, "Machine Learning Approaches to Predicting Company Bankruptcy," *Journal of Financial Risk Management*, vol. 6, pp. 364-374, 1 2017.

A. saini, "Guide on support vector machines (svm)," 23 1 2024. [Online]. Available:

11] https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/. [Accessed 20 5 2024].

12] S. a. P.-A. J. a. A. G. a. D. S. J. a. C.-P. D. a. K. C. a. F. D. a. B. D. a. G.-H. R. a. R. M. a. G. M. a. C. Salcedo-Sanz, "Analysis, Characterization, Prediction and Attribution of Extreme Atmospheric Events with Machine Learning: a Review," 2022.

13] M. Y. a. Q. A. a. N. M. a. S. M. S. a. W. S. a. S. K.-U.-R. R. Khan, "Automated Prediction of Good Dictionary EXamples (GDEX): A Comprehensive Experiment with Distant Supervision, Machine Learning, and Word Embedding-Based Deep Learning Techniques," *Complexity,* 2021.

14] F. A. Fitriani, "K-Means Clustering using RStudio," medium, 26 7 2021. [Online]. Available: https://farahayuf06.medium.com/k-means-clustering-using-rstudio-d4d941f0769c. [Accessed 20 5 2024].

15] F. M. S. a. T. P. a. N. M. a. R. Mohamed, "A Comprehensive Overview and Comparative Analysis on Deep Learning Models: CNN, RNN, LSTM, GRU," no. 2305.17473, 2023.

16] A. Raja, "Prompt Engineering and its Types," 3 9 2023. [Online]. Available: https://www.linkedin.com/pulse/prompt-engineering-its-types-arivukkarasan-raja/. [Accessed 20 5 2024].

17] P. S. a. A. K. S. a. S. S. a. V. J. a. S. M. a. A. Chadha, "A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications," no. 2402.07927, 2024.

18] L. a. D. R. Meteyard, "Best practice guidance for linear mixed-effects models in psychological science," 2019.

19] N. a. Z. F. a. L. F. a. Z. H. Elmrabit, "Evaluation of Machine Learning Algorithms for Anomaly Detection," in *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, 2020.

20] M. a. L. Y. a. H. A. a. K. B. Guastalla, "Application of Large Language Models to DDoS Attack Detection," pp. 83-99, 2024.

21] R. Y. N. B. S. A. M. A. C. F. M. F. ZIADOON KAMIL MASEER, "Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset," vol. 9, 3 2 2021.

22] S. H. a. K. Lei, "IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks," *Ad Hoc Networks,* vol. 105, 2020.

23] J. L. &. K. Park, "GAN-based imbalanced data intrusion detection system," *Springer Nature,* 2019.

24] P. S. a. N. A. S. Arif Yulianto, "Improving AdaBoost-based Intrusion Detection System (IDS) Performance on CIC IDS 2017 Dataset," in *Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution*.

25] H. S. A. Y. M. D. H. B. C. E. G. Abderrazak Khedir, "Enhancing Machine Learning Model Interpretability in Intrusion Detection Systems through SHAP Explanations and LLM-Generated Desc," *IEEE,* 2024.

26] M. F. a. M. J. a. V. I. F. P. Guato Burgos, "A Review of Smart Grid Anomaly Detection Approaches Pertaining to Artificial Intelligence," *Applied Sciences,* vol. 14, no. 1194, 2024.

27] UNB, "Intrusion detection evaluation dataset (CIC-IDS2017)," 25 4 2024. [Online]. Available: https://www.unb.ca/cic/datasets/ids-2017.html.

28] F. A. a. Z. L. Mohammed Alrowaily, Security, Privacy, and Anonymity in Computation, Communication, and Storage, vol. Effectiveness of Machine Learning Based Intrusion Detection Systems, 2019, pp. 277-288.

29] Y. V. R. R. &. C. V. Liao, "Imputation of missing values in sensor data using knn," in *SIAM International Conference on Data Mining,* 2014.

30] Z. F. M. &. Z. J. Ding, "An improved KNN-based missing value imputation algorithm for time series data," in *2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems,* 2016.

M. G. J. A. C. C. &. H. J. Gupta, "Outlier detection for temporal data: A survey," in

31] *IEEE Transactions on Knowledge and Data Engineering*, 2018.

32] S. I. a. L. D. a. H. Shrivastava, "MathPrompter: Mathematical Reasoning using Large Language Models," 2023.

33] B. M. A. a. A. A. Mossa, "Coffee disease classification using Convolutional Neural Network based on feature concatenation," *Informatics in Medicine Unlocked,* vol. 39, no. 2352-9148, p. 101245, 2023.

34] L. Ali, C. Zhu, N. A. Golilarz, A. Javeed, M. Zhou and Y. Liu, "Reliable Parkinson's Disease Detection by Analyzing Handwritten Drawings: Construction of an Unbiased Cascaded Learning System based on Feature Selection and Adaptive Boosting Model," *IEEE Access,* pp. 1-1, 2019.

35] V. Team, "Llama 3 8B vs Mistral 7B: Small LLM Pricing Considerations," Vantage, [Online]. Available: https://www.vantage.sh/blog/best-small-llm-llama-3-8b-vs-mistral-7b-cost. [Accessed 30 5 2024].

36] meta, "LLM Comparison: LLaMA vs. Llama 3," sapling.ai, [Online]. Available: https://sapling.ai/llm/llama-vs-llama3. [Accessed 2 5 2024].

37] W. H. a. X. M. a. H. Q. a. X. Z. a. C. L. a. H. C. a. J. L. a. X. Q. a. X. L. a. M. Magno, "How Good Are Low-bit Quantized LLaMA3 Models? An Empirical Study," no. 2404.14047, 2024.

38] T. Karl, "Reasons Why Is Python Used for Machine Learning," newhorizons, 17 8 2023. [Online]. Available: https://www.newhorizons.com/resources/blog/why-is-python-used-for-machine-learning. [Accessed 4 6 2024].

39] Y. Burowski, "LM Studio Documentation," [Online]. Available: https://lmstudio.ai/docs. [Accessed 4 6 2024].

40] Y. Z. a. J. Y. a. Y. Y. a. A. C.-C. Yao, "Cumulative Reasoning with Large Language Models," 2024.

andrew, "Context length in LLMs: All you need to know," agi sphere, 23 4 2023.

41] [Online]. Available: https://agi-sphere.com/context-length/. [Accessed 10 5 2024].

42] D. Powers, "Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation," *Mach. Learn. Technol.*, vol. 2, 2008.

43] M. Guato Burgos, J. Morato and F. Vizcaino Imacaña, "A Review of Smart Grid Anomaly Detection Approaches Pertaining to Artificial Intelligence," *Appl. Sci.*, no. 14, p. 1194, 2024.

44] Manoharanagi, "how Large Language model work with numerical data," medium, 20 jun 2023. [Online]. Available: https://medium.com/@manoharanagi99/how-large-language-model-work-with-numerical-data-4c90b20792f6. [Accessed 26 4 2024].

45] C. Samiullah, "The Technical User's Introduction to LLM Tokenization," ChristopherGS, 28 2 2024. [Online]. Available: https://christophergs.com/blog/understanding-llm-tokenization. [Accessed 27 4 2024].

46] J. a. A. B. a. M. N. a. Y. K. S. Jithish, "Distributed Anomaly Detection in Smart Grids: A Federated Learning-Based Approach," *sensors,* vol. 11, pp. 7157-7179, 2023.

47] M. a. T. U. a. I. M. a. A. N. A. a. M. M. Shaaban, "Data-Driven Detection of Electricity Theft Cyberattacks in PV Generation," vol. 16, no. 2, pp. 3349-3359, 2022.

48] S. a. D. S. K. Bhattacharjee, "Detection and Forensics against Stealthy Data Falsification in Smart Metering Infrastructure," *IEEE Transactions on Dependable and Secure Computing,* vol. 18, no. 1, pp. 356-371, 2021.

49] M. Hentea, "Applying Security Principles to Smart Grid," in *Building an Effective Security Program for Distributed Energy Resources and Systems*, 2021, pp. 129-164.

50] M. a. L. Y. a. H. A. a. K. B. Guastalla, "Application of Large Language Models to DDoS Attack Detection," pp. 83-99, 2024.

51] D. S. D. M. Y. b. I. B. K. R. B. Kurniabudi Kurniabudi, "Important Features of CICIDS-2017 Dataset For Anomaly Detection in High Dimension and Imbalanced Class Datase," *Indonesian Journal of Electrical Engineering and Informatics (IJEEI),* vol. 9, no. 2, 2021.

52] unsloth, "unsloth team," unsloth, 5 5 2024. [Online]. Available: https://github.com/unslothai/unsloth.

53] I. a. T. G. a. B. N. Mollas, "LionForests: Local Interpretation of Random Forests through Path Selection," 2019.

Table of figures