

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Informatics

Applications of Large Language Models in Cybersecurity

Aleksandar Manev





SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Informatics

Applications of Large Language Models in Cybersecurity

Anwendungen von Großen Sprachmodellen in der Cybersicherheit

Author:	Aleksandar Manev
Supervisor:	Prof. Dr. Claudia Eckert
Advisors:	Alexander Wagner, Dr. Nicolas Müller
Submission Date:	December 15, 2023



I confirm that this bachelor's thesis is my own work and I have documented all sources and material used.

Munich, December 15, 2023 Aleksandar Manev

Acknowledgments

I want to acknowledge

- my supervisor, Prof. Dr. Claudia Eckert, for giving me the opportunity to delve into such an interesting and relevant topic.
- my advisors at Fraunhofer AISEC, Alexander Wagner and Dr. Nicolas Müller, for their invaluable support, guidance, and expertise. Their dedication and insights during our regular meetings significantly enriched my research process and contributed greatly to the success of this thesis.
- my friends and family, for believing in me and encouraging me throughout the whole journey. Their support has allowed me to focus my efforts entirely on this thesis, which I am extremely thankful for.

Abstract

Large Language Models (LLMs) have revolutionized the field of Natural Language Processing (NLP), demonstrating advanced capabilities in text interpretation and generation. In this thesis, we systematically explore their potential to enhance cybersecurity solutions, focusing on three key applications: text classification, data augmentation, and synthetic data generation. Within the context of spam detection and policy compliance checking, two representative NLP tasks in cybersecurity, we assess the effectiveness of state-of-the-art LLMs against various supervised Machine Learning (ML) techniques. Our findings reveal the proficiency of LLMs in text classification without the need for task-specific fine-tuning, making them particularly suitable in scenarios with limited data. However, we also identify some limitations of current LLMs, including prompt sensitivity and classification biases. Furthermore, LLMs demonstrate efficiency in augmenting existing text data and generating diverse synthetic datasets. Our experiments indicate that Deep Learning (DL) models benefit most from such data, which is a promising aspect, considering their high data demands. Nevertheless, these approaches also pose challenges such as bias propagation, stylistic shifts, and potential for malicious use. Future work should address these issues and explore a broader range of cybersecurity problems to fully harness the potential of LLMs. We conclude that despite the challenges they present, LLMs hold significant promise in advancing the field of cybersecurity.

Contents

Acknowledgments **iii**

Abstract **iv**

1 Introduction **1**

2 Background **3**
2.1 Large Language Models . . . 3
2.2 Spam Detection . . . 5
2.3 Policy Compliance Checking . . . 5

3 Related Work **7**
3.1 LLMs: Bridging Cybersecurity and General Applications . . . 7
3.2 Spam Detection . . . 8
3.3 Policy Compliance Checking . . . 9

4 Experimental Setup & Prerequisites **11**
4.1 General Approach . . . 11
4.2 Datasets . . . 12
4.2.1 Spam Detection . . . 12
4.2.2 Policy Compliance Checking . . . 12
4.3 Evaluation Framework . . . 14
4.4 Baseline Models . . . 16
4.4.1 Spam Detection . . . 16
4.4.2 Policy Compliance Checking . . . 19
4.5 Selected LLMs . . . 20
4.5.1 Llama 2 . . . 21
4.5.2 FLAN-T5 . . . 21
4.5.3 Implementation Details . . . 22

5 Experiments: LLMs for Text Classification **23**
5.1 Introduction . . . 23
5.2 Methodology . . . 24
5.3 Evaluation . . . 25
5.3.1 Spam Detection . . . 25
5.3.2 Policy Compliance Checking . . . 29
5.4 Discussion . . . 34

6 Experiments: LLMs for Data Augmentation **36**
6.1 Introduction . . . 36
6.2 Methodology . . . 37
6.3 Evaluation . . . 39
6.3.1 Spam Detection . . . 39
6.3.2 Policy Compliance Checking . . . 44
6.4 Discussion . . . 47

7 Experiments: LLMs for Data Generation **49**
7.1 Introduction . . . 49
7.2 Methodology . . . 50
7.3 Evaluation . . . 51
7.3.1 Spam Detection . . . 51
7.3.2 Policy Compliance Checking . . . 57
7.4 Discussion . . . 62

8 Conclusion **65**

Abbreviations **67**

List of Figures **69**

List of Tables **70**

Bibliography **71**

1 Introduction

Large Language Models (LLMs), powered by recent advancements in Machine Learning (ML) and deep neural networks, have significantly transformed the capabilities of Artificial Intelligence (AI) in understanding and generating natural language. These models, trained on extensive corpora of text data, are able to perform a variety of complex tasks that were previously intractable. From composing poetry to coding software, LLMs have demonstrated a remarkable proficiency that blurs the line between human and machine intelligence. Their ability to process, interpret, and produce text has opened up new frontiers in various domains such as healthcare, finance, and education.

Within the realm of cybersecurity, the implications of LLMs are twofold, presenting both challenges and opportunities. On the one hand, their sophistication introduces new vectors for potential threats. For instance, adversaries could leverage LLMs to craft highly convincing phishing emails or automate the generation of malicious code, thereby scaling their efforts with unprecedented efficiency. On the other hand, LLMs offer an array of opportunities to strengthen cybersecurity defenses. ML techniques have already been successfully applied in the field, e.g., for anomaly detection or spam detection. A key insight is that nearly all cybersecurity problems can be presented in text format, ranging from natural language emails and messages to server logs. Hence, advancements in Natural Language Processing (NLP) typically have a significant influence on cybersecurity strategies. Considering their proficiency in understanding and generating text, LLMs emerge as an attractive tool for advancing cybersecurity solutions.

This thesis delves into potential applications of LLMs in the realm of cybersecurity, focusing on three distinct but interconnected areas: text classification, data augmentation, and synthetic data generation. Text classification is fundamental in cybersecurity, as it underpins every detection and filtering task. The effectiveness of supervised ML models for text classification highly depends on the access to sufficient and high-quality data. However, obtaining such data is often challenging due to privacy concerns or inherent data scarcity. This necessitates the augmentation of existing text data for enhancing the robustness of cybersecurity systems. Furthermore, in situations where relevant data is entirely absent—a common issue in cybersecurity given the dynamic and constantly evolving nature of cyber threats—the generation of purely synthetic data stands out as a viable strategy.

In order to explore the efficacy of LLMs for the three mentioned tasks, we will dive into the contexts of spam detection and policy compliance checking. The two problems are representative for the application of NLP in cybersecurity, yet they cover different aspects of the natural language. Spam detection represents a constant challenge in digital communication – filtering out unsolicited emails or messages. Thus, it is often characterized by more informal or brief texts. Policy compliance checking, on the other hand, involves the automatic verification of

companies' privacy policies against compliance standards, dealing primarily with lengthy, standardized legal texts. Both tasks, though distinct in their nature, are rooted in processing natural language, a domain where LLMs excel. This makes them ideal candidates for our research.

Therefore, the primary objective of this thesis is to systematically evaluate the capabilities of LLMs within the field of cybersecurity. To achieve this, the research is centered around the following key applications of LLMs:

1. **Text Classification**: Assessing the reliability of LLMs in classifying text data.
2. **Data Augmentation**: Evaluating the effectiveness of LLMs in augmenting existing datasets for improved model training.
3. **Data Generation**: Investigating the potential of LLMs in generating synthetic, yet realistic data for training robust cybersecurity systems.

From these objectives, the following research questions arise:

- How do LLMs perform in classifying text data against other state-of-the-art ML techniques without being fine-tuned on the specific task?
- Can LLMs be effectively used to augment datasets, thereby enhancing the performance of cybersecurity models in text classification tasks?
- To what extent can LLMs generate new data that maintains the integrity and variability required for robust cybersecurity applications?

To address these questions, a comparative analysis will be conducted, exploring the potential of LLMs against various state-of-the-art ML techniques. In addition to the quantitative evaluation, we will perform qualitative assessments of the results to determine the applicability of LLMs in real-world scenarios.

The thesis is organized as follows: First, we delve into the essential theoretical concepts, laying the basis for further understanding. This is followed by a comprehensive review of existing literature on the application of ML in cybersecurity, with an emphasis on LLMs and our two use cases: spam detection and policy compliance checking. We then prepare the reader for our experiments by detailing the datasets used, introducing our evaluation framework, establishing baseline models for each use case, and selecting the LLMs for experimentation. Subsequent sections present each of the three applications, including a brief introduction, the methodology employed, and the results achieved. Next, we interpret the results and discuss the strengths and limitations of LLMs, as well as the broader implications of our findings. The thesis concludes by summarizing the research, highlighting the contributions, and proposing directions for future research.

2 Background

In this chapter, we provide the relevant background information for understanding the rest of the thesis.

2.1 Large Language Models

Over the last year, chatbots such as OpenAI's ChatGPT [42] or Google's Bard [22] have gained huge popularity. These systems are built upon LLMs – sophisticated models, trained on massive amounts of heterogeneous text data, e.g., web pages, news articles, social media content, digital books, as well as datasets tailored to specific tasks. By learning statistical patterns in natural language, such as syntax and semantics, LLMs can produce new, contextually relevant, and syntactically sound text in response to given input prompts. They can be further fine-tuned on specific datasets or prompts, making them highly adaptable to diverse NLP tasks, such as summarization, translation, text completion, and question answering [41]. Additionally, they exhibit *emergent abilities* [61], which allows them to show proficiency in tasks they have not been explicitly trained for, thanks to their extensive pattern memorization.

LLMs are primarily built upon a class of deep learning architectures, called **Transformers**. This groundbreaking concept, introduced by Google in 2017 [57], has seen remarkable growth in both complexity and capabilities in the past few years. Transformer-based models have become pivotal in the field of deep learning, particularly due to their flexibility and efficiency in processing sequential data. They can be categorized into three distinct types:

- **Encoder-Only Models**: These are designed to process input data and provide a rich contextual representation of that input. This makes them suitable for tasks that require understanding or classification of the input, such as sentiment analysis. Bidirectional Encoder Representations from Transformers (BERT) [11] and its improved version Robustly Optimized BERT Pretraining Approach (RoBERTa) [37] are prominent examples.
- **Decoder-Only Models**: These models are tailored for generating text by predicting subsequent tokens based on previous ones. Therefore, they excel at text generation and are ideal for generative tasks like creative writing and text completion. Key examples of decoder-only models include the Generative Pretrained Transformer (GPT) series, e.g., GPT-4 [43], and Meta AI's LLaMA family [54].
- **Encoder-Decoder Models**: These are also known as seq2seq (sequence-to-sequence) models. They consist of two parts: an encoder that processes the input sequence and creates a context representation, and a decoder that uses this representation to generate

an output sequence. This makes them suitable for tasks that involve transforming one sequence into another, such as translation, summarization, and question answering. Text-to-Text Transfer Transformer (T5) [47] and its advanced version, Fine-tuned LAnguage Net T5 (FLAN-T5) [7], exemplify this architecture.

Decoder-only models, such as OpenAI's GPT series, are specifically designed for generative tasks, which makes them particularly suitable for prompt-based interactions. Throughout the thesis, this is also how we will utilize the LLMs. By carefully crafting input prompts, we will steer them towards performing specific tasks, leveraging their pre-trained knowledge and capabilities. This prompt-based approach is particularly advantageous as it allows for a flexible application while bypassing the need for fine-tuning of the underlying model, which can be complex and resource-intensive.

Prompt Engineering

Prompt engineering is the key to the effective utilization of LLMs for specialized tasks. The process involves designing prompts that are clear, explicit, and tailored to leverage the model's extensive pre-training [6]. These prompts provide the instructions and specify the desired response format. Hence, their quality directly impacts the model's performance.

A key consideration in prompt engineering is balancing detail with clarity. Intuitively, adding more context to a prompt might seem beneficial for achieving better results. However, this often leads to increased complexity, which does not necessarily enhance the LLM's understanding of the task. In fact, when a prompt becomes overly complex, it can challenge the LLM's ability to identify relevant patterns from its training data, often resulting in less accurate responses. Thus, this underscores the importance of structuring the prompt in a way that optimally leverages the model's capabilities [63].

Two pivotal techniques in prompt engineering are the **Role-Task-Format (RTF)** framework [29] and the **Chain-of-Thought (CoT)** prompting [62]. The RTF framework focuses on defining the *Role* the model should adopt, the *Task* it needs to perform, and the *Format*, in which the output should be structured. The CoT technique, on the other hand, involves designing prompts that encourage the model to generate intermediate reasoning steps, thereby mirroring human problem-solving processes. This is particularly useful in preventing the model from jumping to conclusions, a known limitation [52], and enhancing transparency in its decision-making process. Both approaches help in guiding the model to understand the task at hand and generate useful and coherent responses, making them particularly relevant for cybersecurity applications where precision and reliability are paramount.

Furthermore, we differentiate between **zero-shot** and **few-shot** prompting. When giving instructions to the LLM, we have the choice to decide whether or not to include examples. In the zero-shot setting, we do not provide any examples and expect the LLM to respond correctly based solely on the provided explanation of the task. On the other hand, few-shot prompting implies including specific examples in the prompt. These examples then serve as a guide for the LLM when generating the output.

2.2 Spam Detection

Spam detection comprises the task of identifying and filtering out unsolicited, irrelevant, or inappropriate messages, emails and other forms of digital communication. ML techniques have already been widely used for this purpose. The task is typically formulated as a binary text classification problem where classifiers are trained to accurately distinguish between spam and non-spam (*ham*) content.

Despite the extensive research conducted in the field, the challenge remains significant. One major obstacle is the scarcity of labeled datasets, which are crucial for training effective ML models. In spite of the high volume of daily spam, obtaining labeled samples of fraudulent content is challenging, often due to privacy concerns. The issue is further exacerbated for sensitive, company-internal applications [34]. Consequently, this leads to the problem of *imbalanced learning*, wherein the lack of sufficient fraudulent samples hinders the model's ability to learn effectively, and thus results in poor detection performance.

Furthermore, the field of spam detection is continually challenged by the evolving nature of spam tactics. As spammers modify their strategies to avoid detection, the characteristics of spam content evolve, leading to shifts in the data patterns they exhibit. For instance, a specific set of keywords or phrases, once reliable indicators of spam, may become obsolete as spammers adjust their style of messaging. This constant evolution creates a mismatch between the data on which ML models are trained and the real-world data they encounter, a phenomenon known as *concept drift* [56, 31]. This drift challenges the foundational principles of supervised learning, which rely on the assumption that future data will mirror past data. To maintain the effectiveness of these models, it is essential to continuously update them with recent, relevant data that captures the current state of spam techniques.

Additionally, spammers are driven to bypass spam filters by the potential financial gains from successful fraud. This behavior, also known as *adversarial drift* [31], involves a range of strategies that aim to outsmart textual filters. For instance, spammers often employ obfuscation techniques to make the content of emails appear benign like using synonyms to replace common spam-indicative words, inserting irrelevant text, or altering the format of words to avoid keyword-based detection.

Our study aims to investigate the potential of LLMs to face these challenges, leveraging their advanced natural language understanding and ability to recognize complex patterns in text, even in cases of data scarcity and imbalance.

2.3 Policy Compliance Checking

Nowadays, we rely heavily on technology, and the digital world has become a key part of our lives, affecting how we work, spend our free time, and interact with others. At the heart of this digital transformation is the access to data, which is crucial for the continuous development of technological solutions. Consequently, data has become vital for many business strategies. While the large amount of data generated and collected opens up great opportunities for businesses, it also invites malicious actors, raising serious cybersecurity concerns.

To address these concerns, the European Union (EU) enacted the General Data Protection Regulation (GDPR) in 2018 [13]. GDPR serves as a comprehensive legal framework, unifying data protection and privacy laws across EU countries. It aims to protect individuals' personal data, ensuring their privacy rights are respected in an increasingly data-driven world. The regulation applies to all companies and businesses operating within the EU/EEA, or dealing with the data of EU citizens. Non-compliance with GDPR can lead to severe financial penalties, which highlights the importance of adherence.

Central to GDPR are several key principles designed to safeguard personal data, including:

- **Lawfulness, Fairness, and Transparency**: Data must be processed legally, openly, and fairly.
- **Purpose Limitation**: Data must be collected only for specific, explicit, and legitimate reasons.
- **Data Minimization**: Only the necessary data for the intended purpose should be collected and processed.
- **Accuracy**: Personal data must be accurate and kept up to date. Inaccurate data should be corrected or deleted promptly.
- **Storage Limitation**: Data should be stored only as long as necessary for its intended purposes.
- **Integrity and Confidentiality**: Data must be processed securely to prevent unauthorized access, loss, or damage.
- **Accountability**: Data controllers must demonstrate and be responsible for ensuring GDPR compliance.

While these principles establish a robust framework for data protection, ensuring compliance with them poses significant challenges. It requires a detailed examination of privacy policies, which can be extensive and intricate. Furthermore, the GDPR provides individuals with a tool to protect their rights against the entities that try to collect and monetize their personal data. Nonetheless, the mere presence of privacy policies does not inherently guarantee effective protection. GDPR's effectiveness relies on individuals' understanding and engagement with the policies. However, they are often time-consuming to read and their technical jargon makes them inaccessible to the average, non-technical user. Advancements in ML, and in NLP in particular, offer promising solutions to these challenges. NLP's capability to interpret and process human language aligns well with the task of examining privacy policies, which are text-based and designed for human comprehension. With the advent of LLMs, new avenues have opened up to automate the analysis of text data, thereby offering a potentially efficient, scalable, and accurate mechanism to assess policy compliance.

This thesis explores the potential of LLMs for this task. We aim to demonstrate their utility in bridging the gap between complex legal requirements and practical technological solutions, and thus pave the way for developing more sophisticated tools for legal compliance automation.

3 Related Work

3.1 LLMs: Bridging Cybersecurity and General Applications

The field of ML has been revolutionized with the rise of LLMs, offering exceptional capabilities in natural language understanding and learning from textual data. This chapter explores the application of LLMs in cybersecurity, focusing on the three key areas: text classification, data augmentation, and data generation.

The integration of LLMs in cybersecurity, particularly for text classification and cyber threat detection, has gained significant attention. In 2022, Aghaei et al. [2] introduced SecureBERT, a language model based on RoBERTa, fine-tuned on a large corpus of cybersecurity text, and thus tailored for critical cybersecurity applications. This model demonstrated promise in tasks like phishing detection, code and malware analysis, and intrusion detection. In 2023, Ferrag et al. [15] introduced the SecurityLLM model, built around two key elements: SecureBERT – used for cyber threat detection, and FalconLLM, which served as an incident response and recovery system. The model showed significant effectiveness in detecting various types of attacks, including Man-in-the-Middle (MITM), Cross-Site Scripting (XSS), SQL Injection, and Ransomware. Moreover, Ferrag et al. introduced SecureFalcon [14], a fine-tuned FalconLLM for cybersecurity applications, showcasing impressive accuracy in differentiating between vulnerable and non-vulnerable C code, and software vulnerabilities in general. Sultana et al. [51] highlighted the potential of LLMs in automating defensive cybersecurity tasks, including anomaly detection, incident response, and vulnerability mapping. Guastalla et al. [23] applied LLMs for the detection of Distributed Denial of Service (DDoS) attacks, demonstrating their ability to detect such threats with high accuracy and also elucidate their reasoning.

Researchers have also explored the use of LLMs for data augmentation, particularly in addressing class imbalances and data sparsity [33, 26, 18]. Edwards et al. [12] utilized GPT-2 for generating artificial training instances, leading to consistent classification improvements. Yoo et al. [66] proposed a data augmentation technique using LLMs like GPT-3 via prompt-based interaction, outperforming other existing text augmentation methods. Similarly, Sahu et al. [48] introduced a prompt-based approach to generate labeled training data for intent classification, demonstrating significant performance improvements. Nevertheless, most of the research on using LLMs for data augmentation is not directly cybersecurity-related. With our work, we aim to bridge this gap.

Some studies have also investigated the feasibility of generating purely synthetic datasets from scratch using LLMs [60, 39, 19, 53]. In 2022, Ye et al. [65] introduced ZEROGEN, a flexible zero-shot learning method using LLMs for dataset generation, achieving promising results in various NLP tasks, such as text classification and question answering. Hartvigsen et al. [26] made use of an LLM to create TOXIGEN – a large-scale toxic language dataset, which

was used for fine-tuning pre-trained hate classifiers and helped improving their performance on three popular human-generated toxicity datasets. Meng et al. [39] presented a zero-shot learning approach for Natural Language Understanding (NLU) tasks using LLMs, achieving strong performance across multiple classification tasks. In 2023, Li et al. [36] assessed the effectiveness of LLM-generated data in model training for 10 different classification tasks and addressed the factors that might moderate the potential of synthetic data in facilitating successful training, focusing on subjectivity of the task. Following this, Yu et al. [67] conducted an empirical study on biases, diversity, and efficiency in synthetic datasets generated by LLMs, offering insights into their challenges and potential. Chung et al. [8] addressed the challenge of diversification of LLM-based data generation, incorporating human interventions for higher data accuracy.

The current body of research on LLMs in cybersecurity demonstrates their transformative potential. While they have been successfully applied as classifiers for enhancing cyber threat detection, this is mainly achieved by fine-tuning them on the specific downstream task. However, their application as off-the-shelf classifiers without fine-tuning is not so widely explored. Another gap we identify is in their application for data augmentation and data generation. While LLM-based data augmentation is often found to outperform other data augmentation methods in boosting the model performance, the results are mixed regarding LLM-generated synthetic data. In addition, there is a limited number of studies that utilize cybersecurity-related data in their experiments. In this thesis, we address these gaps by exploring both the potential and limitations of LLMs for these tasks in a cybersecurity context.

3.2 Spam Detection

Spam, a persistent issue in digital communication since the late 1970s, has evolved and diversified across various platforms (e.g., email, instant messaging, social media), necessitating advanced detection techniques [16]. Initially, rule-based methods, focusing on specific patterns and keywords in the content, as well as conventional ML algorithms such as NB, SVM, LR, ensemble methods, and KNN were employed for spam detection [38, 24]. However, despite their efficiency, these methods often struggled to cope with the dynamic nature of spam due to their limitations in handling complex data [9].

The advancements in deep learning offered a promising alternative for handling complex data effectively [10]. In particular, Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and their variants like Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), significantly enhanced spam detection [50]. CNNs excelled in identifying patterns in data, a crucial feature of spam emails, while RNNs and variants effectively processed sequential text data, capturing contextual information.

The rise of LLMs has further transformed the field of NLP with their capabilities to process and interpret natural language. As an NLP problem, spam detection seems to benefit from their potential as well. Recent studies have shown that fine-tuning pre-trained LLMs, such as BERT, can outperform traditional classifiers like KNN and NB, as well as deep learning networks like BiLSTM in email spam detection [64]. In 2023, Jamal et al. [30] proposed

IPSDM, a specialized LLM-based model for detecting phishing and spam emails, which showed improved classification performance in both balanced and imbalanced datasets. Koide et al. [32] introduced another innovative approach, employing ChatGPT for phishing site detection without model fine-tuning, relying instead on prompt engineering. The results highlighted the potential of LLMs in enhancing cybersecurity measures in this domain.

In SMS spam detection, Ghourabi et al. [20] combined LLMs, such as GPT-3, with ensemble learning to create robust classifiers. These classifiers, utilizing algorithms like SVM, KNN, CNN, and Light Gradient-Boosting Machine (LightGBM), leveraged the dense and meaningful representations of text messages produced by LLMs, and achieved remarkable accuracy. Further, Labonne et al. [34] introduced Spam-T5, a fine-tuned FLAN-T5 model for detection of different types of spam, and evaluated the effectiveness of LLMs from different families – BERT-like, Sentence Transformers, and Seq2Seq – against traditional ML techniques such as NB and SVM. Their results revealed the superiority of LLMs, especially in few-shot settings. However, the authors also noted the high computational costs involved in fine-tuning LLMs, indicating a significant trade-off between accuracy and resource utilization.

Our research contributes to this growing body of work by exploring the use of LLMs as out-of-the-box tools for spam detection. Since BERT has been released, the field has been improving on a daily basis. More recent models such as FLAN-T5 and Llama 2 have been trained on huge amounts of data. Their architecture also makes them suitable for prompt-based interactions. Therefore, they suggest the potential for high performance without task-specific training. We aim to examine their capabilities in direct classification via inference, as a data augmentation tool, and as a data generation tool. Hence, by building upon the baseline models proposed recently by Labonne et al. [34], we seek to extend their findings and provide insights into the full potential of LLMs in the domain of spam detection.

3.3 Policy Compliance Checking

This section presents a comprehensive overview of significant advancements in legal text processing using ML techniques, concerning the GDPR in particular. Aim of the research in this domain is automating the process of assessing privacy policies for compliance, as well as aiding organizations in achieving it.

One of the earliest efforts in this area was made by Müller et al. [40]. In 2019, they introduced a fully labeled dataset of privacy policies according to five key GDPR requirements, enabling analysis and detection of compliance. Moreover, they employed various classification algorithms, including SVM, LR, and Neural Networks (NNs), with different embedding strategies. Their best performing classifier, SVM with pre-trained Embeddings from Language Model (ELMo) embeddings, achieved F1 scores up to 0.71 across the different categories. Furthermore, testing it on unseen privacy policies confirmed the usefulness of the dataset for real-world applications. Their pioneering work was pivotal in policy compliance checking using GDPR as a legal text, setting a benchmark for subsequent research in the field.

Building on this foundation, in 2021, Sanchez et al. [49] developed a system to automatically assess how respectful a privacy policy is with the data protection goals stated by the GDPR.

Their system calculated and presented privacy scores to users, enabling them to immediately understand the risks and making them aware of the privacy threats hidden in privacy policies. With this novel approach of quantifying compliance, they made it possible to immediately compare and rank policies of different companies according to their GDPR compliance. Additionally, they also offered fine-grained ratings which help understand the reasons for the assessment, aligning with the concept of explainable AI. Nonetheless, due to the extremely limited training data at hand, they were restricted in the choice of ML techniques. Their system utilized SVM for multi-label text classification, hinting at the potential of expanding the training data and employing more advanced classifiers.

The necessity for GDPR compliance in organizations led to innovative ML applications as well. Goldsteen et al. [21] introduced a decision tree-based method to minimize the amount of personal data needed for ML models, thus adhering to GDPR's data minimization principle. This approach effectively balanced data reduction with model accuracy, offering a practical solution for existing systems without necessitating model retraining. Ferrara et al. [17] and Aberkane et al. [1] focused on integrating privacy consideration at the foundational level of software development. The first work emphasized the use of static analysis for generating GDPR compliance reports, customized for various stakeholders in the GDPR compliance process, while the latter highlighted automating manual tasks in requirements engineering using NLP, thereby embedding compliance from the early stages.

Since often the entity processing personal data (the processor) differs from the one collecting it (the controller), processing personal data is regulated by the GDPR through *Data Processing Agreements (DPAs)*. Amaral et al. [4] introduced DERECHA, an NLP-based method to assess and ensure the compliance of DPAs with GDPR, contributing to the compliance verification of whole software systems. By comparing Semantic Frames (SFs), representing GDPR requirements, with the textual content of DPAs, DERECHA not only assessed compliance but also suggested improvements, with an accuracy of 84.6%, which showed the method's potential to significantly reduce the manual effort in verifying DPA compliance.

Early LLMs have also been employed in the field. In 2021, Hamdani et al. [25] proposed a framework combining ML and rule-based methods to automate the compliance checking process, particularly with GDPR Articles 13 and 14. Their two-module system employed NLP in a text-to-text approach to extract key information from privacy policies, which was then used as input to a rule-based system to verify the presence of mandatory information. In particular, for the first module, they employed a fine-tuned T5-11B, which showed proficiency not only in classification, outperforming local classifiers such as CNN and XLNet, but also in extracting relevant text spans, which could be used to explain the predicted values.

Our study complements and extends the existing work by exploring the capabilities of state-of-the-art LLMs in the field. Focusing on the dataset introduced by Müller et al., we use LLMs for classifying sentences as GDPR-compliant or non-compliant. We further employ them in two novel ways – for data augmentation and synthetic data generation, addressing existing challenges such as limited data and class imbalance, with the aim of enhancing ML model training. This exploration not only contributes to the ongoing research in legal text processing, but also sets a new direction in utilizing the advanced capabilities of LLMs.

4 Experimental Setup & Prerequisites

4.1 General Approach

This section outlines the general approach we follow in order to examine the potential of LLMs across the three applications of interest. Our methodology comprises a structured process that combines both quantitative and qualitative evaluations.

Initially, for each use case (i.e., spam detection and policy compliance checking) we identify and set up existing ML models from related work. These models serve a dual purpose in our research. First, they establish a performance benchmark on the selected datasets, against which we can quantitatively assess the effectiveness of the LLMs across all three applications. Moreover, these baseline ML models are integral to our experiments in the contexts of data augmentation and synthetic data generation. We utilize them to train on data produced by LLMs, employing different strategies. By doing so, we can quantitatively evaluate the quality and the usability of the generated data, and thus understand its impact on enhancing or complementing existing ML solutions.

Our next step involves exploring the potential of LLMs as text classifiers without fine-tuning. We develop specific prompts and use them to evaluate a fully labeled dataset within each domain. The comparison of the predicted labels against the actual ones allows us to calculate performance metrics and draw conclusions. To ensure the robustness of our findings, we experiment with different prompt-crafting strategies.

Further, we delve into the use of LLMs for data augmentation. Again, without fine-tuning, LLMs are employed to expand existing datasets in a novel way, described in Chapter 6. We explore different augmentation strategies, including augmentation of the minority class only or both, as well as varying the proportion of augmented data. The different datasets resulted are then used to train our baseline models, and we compare their performance with the one achieved with the original datasets.

Our final investigation focuses on LLMs' potential in generating synthetic data. We instruct the LLMs to produce new data points, which are then utilized to train our baseline models. We employ different strategies to inspect the quality of the data, such as using solely synthetic data for training or mixing it with real-world. The performance achieved is benchmarked against the baseline. Additionally, we conduct a qualitative assessment of the generated data to determine its relevance and authenticity.

The following sections will detail the datasets selected for our study, outline the evaluation framework, and finally introduce the baseline ML models for each use case, as well as the specific LLMs employed in our experiments.

4.2 Datasets

In order to answer the research questions, we need reliable data for both use cases, which is fully labeled. In this section, we briefly describe the datasets we employ for our experiments and discuss the rationale behind their choice.

4.2.1 Spam Detection

For the spam detection use case, we leverage the **UCI SMS Spam Collection** [3], a public and widely recognized dataset, compiled by the Machine Learning Repository at the University of California, Irvine. This collection is specifically created for the purpose of spam detection research and has been employed extensively in the field of NLP due to its robustness and reliability.

The UCI SMS Spam Collection consists of 5,572 SMS messages in English, which are labeled as *ham* for legitimate and *spam* for unsolicited or misleading messages. The dataset exhibits an authentic representation of real-world text messaging, with a realistic distribution of spam and legitimate messages that mimics the typical prevalence observed in the daily SMS communication. Hence, this is an imbalanced dataset with a spam rate of approximately 13%. The dataset consists of two columns, one presenting the message in natural language, and the second presenting the binary label. The messages have been labeled by human annotators, ensuring a high degree of accuracy in the classification of spam and ham entries, crucial for training reliable spam detection models.

The UCI SMS Spam Collection serves as a standard benchmark in the domain of spam detection. Numerous studies have utilized it for experimenting with various classification models, ranging from traditional machine learning approaches to advanced deep learning techniques. Its widespread adoption facilitates comparative analysis and benchmarking, enabling our research to position its findings within the context of established works.

Considering the text-based nature of the dataset, it is particularly suited for using it with LLMs, which excel in understanding and generating human language patterns. The linguistic variety and complexity found in the dataset, including slang and shorthand, provide a challenging yet informative ground for training and evaluating the capabilities of LLMs in distinguishing spam from legitimate content.

4.2.2 Policy Compliance Checking

For our second use case, we utilize the dataset introduced by Müller et al. [40], a pioneering effort in automated compliance checking using GDPR as legal text of interest. The comprehensive and natural language-based content of this dataset makes it exceptionally suitable for in-depth evaluation of policy compliance.

The dataset consists of 18,397 sentences in total, derived from 250 real companies' privacy policies. In order to assess the GDPR compliance of each sentence, Müller et al. define five categories, representing core GDPR requirements: **Data Protection Officer (DPO)**, **Purpose**, **Acquired data**, **Data sharing**, and **Rights**. Table 4.1 summarizes the required content in

the privacy policy that they cover. Additionally, in their paper, they define a set of specific guidelines for each category in order to ensure unambiguous labeling. Each sentence has been then manually analyzed and labeled for its GDPR compliance according to the five categories. Hence, the dataset contains six columns, the first one presenting the sentence in natural language, followed by binary labels in the other five columns (i.e., '0' if the sentence contains no information related to the label or is non-compliant, and '1' if it discloses the required information for the respective category).

Table 4.1: The five GDPR requirements chosen to evaluate privacy policy compliance. [40]

No.	Category	Required content in privacy policy
1	DPO	Contact details for the data protection officer or equivalent authority.
2	Purpose	Disclosure of the purpose for which personal data is or is not used.
3	Acquired data	Disclosure that personal data is or is not collected, and/or which data is collected.
4	Data sharing	Disclosure if 3rd parties can or cannot access a user's personal data.
5	Rights	Disclosure of the user's right to rectify or erase personal data.

Privacy policies contain a lot of information, and individual requirements are often represented by a few sentences only. Thus, due to the nature of the task, the dataset exhibits high imbalance across categories. Table 4.2 presents the distribution of compliant and non-compliant instances across each category. This imbalance poses significant challenges in applying ML techniques. Among other problems, when the minority class has very few examples in comparison to the majority class, the model might not learn enough about its characteristics. This lack of representative data can then hinder the model's ability to recognize patterns that are specific to this class on unseen data, and lead to poor performance. We will further refer to this issue and how we tackle it throughout the thesis.

Table 4.2: Data point counts in the GDPR dataset.

Category	Compliant	Non-compliant
DPO	414	17,983
Purpose	980	17,417
Acquired data	565	17,832
Data sharing	830	17,567
Rights	251	18,146

Müller et al. have also applied a comprehensive set of supervised NLP algorithms in order to analyze the soundness of their dataset. Moreover, they have evaluated the trained models on a wide range of unseen policies, and thus have validated its effectiveness. The availability of benchmark models and the natural-language, fully labeled sentences make the dataset suitable for our study, aiming to assess the performance of LLMs with formal, legal text.

For our research, we will concentrate on the **DPO** category. This category, characterized by requiring specific and factual information such as contact details, presents a clear-cut binary classification challenge, ideal for assessing LLMs' capabilities. Moreover, it also contains the fewest labeling guidelines due to the clarity of the required content. To ensure a reliable assessment of LLMs' performance, it is crucial to eliminate ambiguity in the provided context through the prompts. Focusing on a well-defined category with clear requirements, which are "familiar" to the LLM even without explicit specifying, minimizes human error. This is especially critical in applications like text classification, where sentences are labeled based on the given prompt, and data generation, where the precision of the specified requirements for the generated data directly impacts the quality of the output, and thus the trained models' performance. By selecting the **DPO** category, we decrease the potential for misinterpretation, thus providing a more reliable and objective measure for evaluating LLMs. Nonetheless, this approach enables obtaining generic results that would be valid for any category.

4.3 Evaluation Framework

To draw reliable conclusions, it is imperative to establish a robust evaluation framework. This section outlines the metrics utilized to assess model performance, addressing both quantitative and qualitative aspects.

In the field of cybersecurity, where imbalanced datasets are typical, reliance on a single metric might be insufficient to capture model performance. Thus, considering a combination of metrics is crucial to assess it. In our research, we use the following set of metrics for quantitative evaluation:

- **Accuracy**: Measures the proportion of correct predictions (both true positives and true negatives) out of all predictions made: $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
- **Precision**: Measures the proportion of actual positives in the predictions labeled as positive: $Precision = \frac{TP}{TP+FP}$
- **Recall**: Measures the proportion of actual positives correctly identified by the model: $Recall = \frac{TP}{TP+FN}$
- **F1 Score**: Measures the harmonic mean of precision and recall: $F_1 = 2 \times \frac{Precision \times Recall}{Precision+Recall}$

where TP, TN, FP, FN stand for the number of true positives, true negatives, false positives, and false negatives, respectively.

Accuracy is the most intuitive metric, representing the proportion of correct predictions among the total number of predictions made. However, in the presence of imbalanced datasets, where one class significantly outnumbers the other, it can be misleading. For example, a model might predict the majority class for all instances and still achieve high accuracy, despite failing to correctly identify the minority class instances. Therefore, a combination of the above metrics is essential for a comprehensive understanding of the performance.

Precision quantifies the number of correct positive predictions out of all positive predictions made. It is particularly important when the cost of a false positive is significant, e.g., when an intrusion detection system alerts of a potential threat, it is important that it is indeed a threat, and not a false alarm.

Recall, or sensitivity, measures the ratio of actual positives that are correctly identified by the model. It plays an important role when the cost of missing a positive instance is high, e.g., when a malware detection system fails to identify a malware instance.

The **F1 score** provides a balance between precision and recall, taking into account both false positives and false negatives. Therefore, it is an important metric that gives a good understanding about model performance, particularly when there is an uneven class distribution.

In addition to quantitative metrics, qualitative evaluation is crucial across the different LLM applications. We will include subjective assessments of the model's output, such as relevance, naturalness, bias, and diversity.

This comprehensive evaluation framework aims to provide a thorough understanding of LLMs' effectiveness in enhancing cybersecurity solutions.

Spam Detection

Here, we shortly discuss the meaning of the different metrics in the context of spam detection. In the field of spam detection, the objective is twofold: to accurately identify as many spam instances (true positives) as possible, and to minimize the misclassification of legitimate ones (false positives). Therefore, precision and recall play particularly important roles.

Precision is paramount in spam detection because a high rate of false positives – where legitimate content is wrongly marked as spam – can lead to important information being missed by the users. For instance, if a spam detection system has a low precision, it may classify important business correspondence as spam, causing potential losses or communication issues. On the other hand, the **recall** is equally critical since it measures the system's effectiveness in capturing all spam instances. A low recall, signifying a failure to detect a substantial portion of spam, can clutter inboxes and potentially expose users to security risks like phishing or malware.

As discussed, since spam detection models usually have to deal with high class imbalance, **accuracy** can be less informative. In light of these considerations, the **F1 score** emerges as a crucial metric, providing a single measure to capture both false positives and false negatives.

Policy Compliance Checking

In the context of policy compliance assessment, we also aim for high precision and high recall simultaneously. **Precision** is crucial as it measures how many of the instances identified as compliant are indeed compliant. If it is low, the system might incorrectly flag non-compliant practices as compliant, which could result in an organization unknowingly violating GDPR regulations. This carries the risk of legal and financial penalties and damage to the organization's reputation. Moreover, for the data protection authorities, who are responsible for monitoring and enforcing compliance, poor precision could overlook critical violations,

posing privacy risks to individuals. On the other hand, the **recall** is equally important as it indicates the model's ability to identify all instances that are actually compliant. A low recall indicates that the model is missing compliant instances, which could lead to unnecessary legal reviews, wasted resources, and potentially modifying compliant procedures without need, disrupting business operations.

Hence, in this context, we also focus on the **F1 score**, a metric that harmonizes precision and recall. It provides a comprehensive view of a model's efficacy in legal text analysis. While we monitor **accuracy** as well, because it is an indicator of the model's overall correctness in distinguishing between compliance and non-compliance, it can be misleading in data with significant imbalance, which is also the case here. Thus, our emphasis will predominantly be on the other metrics.

4.4 Baseline Models

To perform a quantitative evaluation of LLMs' capabilities across the different applications, it is essential to establish a reference point. This reference is the state-of-the-art performance achieved by ML models in each specific problem area. In this section, we provide information about the baseline models chosen for each use case, which we refer to throughout the thesis.

4.4.1 Spam Detection

For spam detection, we reconstruct the baseline models from Labonne et al. [34]. This enables us to not only evaluate the results achieved across the different applications, but also to extend their findings and see the full picture of applying LLMs in this domain. We divide our dataset into two portions: 80% for training and 20% for validation. To ensure consistency and reproducibility, we set the *random_seed* to 42. Moreover, we employ stratification to maintain an equal distribution of spam and non-spam samples across sets. Thus, our baseline for this use case consists of the following ML models:

- **Naïve Bayes (NB)**, a probabilistic model, known for its efficiency in large datasets due to its assumption of feature independence.
- **Logistic Regression (LR)**, a linear model that uses a sigmoid function to predict binary outcomes.
- **K-Nearest Neighbors (KNN)**, a non-parametric model that classifies a data point by identifying the most common class among its k nearest neighbors.
- **Support Vector Machine (SVM)**, a model that constructs hyperplanes to optimally separate data into different classes, effective for both linear and non-linear classification through kernel methods.
- **eXtreme Gradient Boosting (XGBoost)**, a highly efficient gradient boosting method that builds decision trees in a level-wise manner, optimizing performance and accuracy by systematically correcting errors from previous trees.

- **Light Gradient-Boosting Machine (LightGBM)**, another gradient boosting framework, which uses a leaf-wise strategy for constructing decision trees and employs histogram-based algorithms for efficiency.

The preprocessing pipeline includes tokenization, stop word removal, and stemming via the Porter algorithm [46]. For feature extraction, we employ Term Frequency-Inverse Document Frequency (TF-IDF) encoding, with optimal number of features determined through a model-dependent grid search and stratified 5-fold cross-validation, performed by Labonne et al. [34]. The hyperparameters for training the different models are adopted directly from their study as well to ensure reproducibility of their results.

While conventional ML methods are typically faster to train and evaluate, they often lack the capacity to capture complex patterns inherent in the training data. To address this limitation and ensure a comprehensive analysis, particularly for the impact of data augmentation and synthetic data generation, we include two popular DL models to our baseline, **LSTM** and **BERT**. These DL models provide a contrast to the traditional ML techniques, offering more sophistication at the cost of increased training time. For the implementation of LSTM and BERT, we employ PyTorch [44], a widely-used open-source ML framework. Thanks to its intuitive interface and robust support for advanced neural network architectures, PyTorch enables efficient and dynamic model development. Its comprehensive ecosystem—especially in terms of pre-built functionalities for NLP tasks—and strong community support make it a suitable choice for our purposes. In the following, we outline the implementation details of the chosen DL models.

LSTM

To train the LSTM model for spam detection, we standardize our training set by following a series of preprocessing steps, including conversion to lowercase, stripping of URLs, and applying tokenization, lemmatization, as well as stop word removal. The training corpus is then used to create a vocabulary for converting messages into numerical tokens, considering padding and out-of-vocabulary words. This preprocessed data is handled by a custom **Dataset** class and batched using PyTorch's **DataLoader**, ensuring it is properly formatted for the LSTM model to process afterwards.

The architecture of the model consists of an embedding layer for encoding textual features, an LSTM layer for capturing sequential dependencies, and a linear output layer for binary classification. A dropout layer is employed to mitigate overfitting. The training is guided by the **BCEWithLogitsLoss** loss function, adjusted with class weights to address the imbalance. Moreover, we utilize the **Adam** optimizer with predefined learning rate and weight decay.

Further, a **ReduceLROnPlateau** scheduler dynamically adjusts the learning rate based on the validation loss, thus enhancing training efficiency. To avoid overfitting and facilitate computational efficiency, we employ early stopping to terminate training when there is no improvement in the validation loss within the predefined patience window. The model's optimal state is captured at the point of lowest validation loss, ensuring reproducibility and optimal performance during evaluation.

BERT

To provide a wide spectrum of baseline models and comprehensive results, in addition to the conventional ML models and the standard DL model LSTM, we include an early LLM, and namely BERT. BERT is an encoder-only LLM, which has shown state-of-the-art results for various NLP tasks. In particular, we employ the pre-trained **bert-base-uncased** model [11] from the Hugging Face Transformers library for sequence classification tasks, by fine-tuning it for spam detection.

Our text preprocessing pipeline consists of tokenization with the **BertTokenizer**, set to handle sequences up to 64 tokens while performing padding, truncation, and attention mask generation. The tokenized data is then converted into PyTorch tensors for input to the model. **DataLoaders** with batch size 32 manage the training and validation sets, employing a random sampler for the training set and a sequential sampler for the validation set to ensure a mixed representation of classes in each batch and consistent evaluation, respectively.

To fine-tune the BERT model, we use the **BertForSequenceClassification** class from Transformers, with a custom training loop encompassing the forward passes, loss calculation, backpropagation, and steps for optimizer and scheduler. The hyperparameters include a learning rate of $2e-5$ and epsilon of $1e-8$. The training spans up to 10 epochs with early stopping based on validation loss to prevent overfitting, saving the best model states at the lowest validation loss observed. We utilize the **CrossEntropyLoss** function as a loss function, with dynamically calculated class weights to address the class imbalances. Optimization is conducted using the **AdamW** optimizer. The entire training process is executed on a specified computing **device**, either CPU or GPU. In our case, a V100 GPU provided by Google Colab is employed for the task.

Baseline Results

To perform different experiments throughout the thesis uniformly, we introduce a systematic approach to manage data processing, model training, and evaluation. The data for training and evaluation is dynamically configured via a custom function, which is particularly useful for handling different portions of LLM-produced data and evaluating different strategies.

Following data configuration, we consecutively train our models and evaluate them on the validation set. Each model, once trained, generates binary labels as predictions, which we systematically collect and store. Our infrastructure is designed for multiple experimental runs to ensure result robustness, with a *run_id* to differentiate each run's outputs. This approach is especially important for evaluating DL models, because it helps account for variability in results due to random initialization of weights and biases. In this thesis, each model undergoes five independent runs. Upon completing them, we assess the predictions using the metrics outlined in Section 4.3, thereby deriving performance metrics for each model per run. We then calculate and report the average performance. Moreover, we include the standard deviation observed across the runs, which aims to give confidence in the findings.

Table 4.3 presents the results, achieved by our baseline models on the original spam dataset, thereby establishing a performance benchmark for this use case.

Table 4.3: Baseline performance for the spam detection use case (best results highlighted).

Model	Accuracy	Precision	Recall	F1 Score
NB	97.66% ± 0.00	97.67% ± 0.00	84.56% ± 0.00	90.65% ± 0.00
LR	97.12% ± 0.00	99.16% ± 0.00	79.19% ± 0.00	88.06% ± 0.00
KNN	96.22% ± 0.00	94.96% ± 0.00	75.84% ± 0.00	84.33% ± 0.00
SVM	98.11% ± 0.00	97.06% ± 0.00	88.59% ± 0.00	92.63% ± 0.00
XGBoost	93.97% ± 0.00	91.84% ± 0.00	60.40% ± 0.00	72.87% ± 0.00
LightGBM	97.30% ± 0.00	95.42% ± 0.00	83.89% ± 0.00	89.29% ± 0.00
LSTM	98.10% ± 0.27	92.58% ± 1.84	93.29% ± 1.12	92.92% ± 0.98
BERT	98.87% ± 0.30	94.58% ± 2.27	97.18% ± 0.66	95.84% ± 1.06

4.4.2 Policy Compliance Checking

In this section, we establish baseline performance for the policy compliance checking use case. We build upon the work of Müller et al. Along with introducing the dataset, they evaluated it using three different classifiers (SVM, LR, and NN), in combination with three highly popular pre-trained word embedding techniques to map textual data to vector representation (Word2Vec, FastText, and ELMo). They employed a 3-fold cross-validated grid search to find the optimal set of hyperparameters and addressed class imbalance using SMOTE up-sampling for the NN, and balanced class weights for LR and SVM. Furthermore, they approached the problem as five separate binary classification tasks, which allows us to reproduce their work specifically for our category of interest, and namely **DPO**. After training the models, Müller et al. identified clear dominance of ELMo embeddings over FastText over Word2Vec. They also observed that SVM and NN yielded almost the same performance in most of the cases.

In this paper, we emulate their work in the following manner. First, due to the similarity in the performance between SVM and NN in their study and the fact that we will apply more advanced deep neural networks, we exclude NN's *keras* implementation from our baseline models. Moreover, while pre-trained ELMo embeddings demonstrated superior performance, they are based on a deep learning method (LSTM), which we employ in our work as a separate classifier. Hence, we opt for the TF-IDF-based implementations of SVM and LR from our spam detection use case by adjusting them with the optimal hyperparameters identified by Müller et al. Although less advanced, this approach shows comparable results and enables performing a wide range of experiments due to its computational efficiency. In addition, it is suitable for our purposes, aiming to investigate the impact of LLM-generated data.

We further incorporate the **LSTM** and **BERT** models, previously utilized for spam detection (see Subsection 4.4.1 for implementation details). Since the privacy policy checking task is essentially also a binary text classification problem, we adapt the models to classify according to the DPO criterion. Hence, our baseline for this use case comprises four models in total: LR, SVM, LSTM, and BERT. This selection provides a wide spectrum of models, ranging from traditional ML approaches to advanced DL techniques and early encoder-only LLMs. Again,

they serve a dual purpose in our work: establishing benchmark performance and facilitating training with varied datasets for augmentation and generation analyses.

As in the spam detection use case, we divide the original dataset into training and validation sets. However, due to the extremely limited amount of positive instances, we increase the validation set to 30% to ensure an adequate representation of the minority class. We apply stratification and a *random_seed* set to 42 for reproducibility. This results in training data of 290 compliant sentences and 12,578 non-compliant sentences, which makes it highly imbalanced.

Baseline Results

Similarly to the spam detection pipeline, each model undergoes five runs to ensure reliable results. We collect the predicted DPO labels from each model for each run and then evaluate them employing the metrics outlined in Section 4.3. In Table 4.4, we report the average performance alongside standard deviations to indicate performance stability across runs. As we can observe, despite the measures to mitigate class imbalance with class weights, the DL models exhibit a comparatively lower average performance than the simpler models. This highlights the inherent challenges for ML techniques posed by highly imbalanced datasets.

Table 4.4: Baseline performance for DPO compliance checking (best results highlighted).

Model	Accuracy	Precision	Recall	F1 Score
LR	96.45% ± 0.00	37.67% ± 0.00	88.71% ± 0.00	52.88% ± 0.00
SVM	98.10% ± 0.00	57.04% ± 0.00	62.10% ± 0.00	59.46% ± 0.00
LSTM	95.89% ± 0.72	34.67% ± 4.50	88.71% ± 1.69	49.67% ± 4.48
BERT	96.29% ± 0.57	37.58% ± 3.69	94.84% ± 0.82	53.72% ± 3.83

4.5 Selected LLMs

LLMs hold significant potential as proficient AI assistants that can perform sophisticated reasoning tasks in different domains. Considering the popularity they have gained among the general public since December 2022, advances in the field occur rapidly and on daily basis. Popular LLM-based applications such as OpenAI's ChatGPT [42] and Google's Bard [22] are built upon closed-source LLMs. However, more and more open-source LLMs get released that show promising performance. We focus on the suite of open-source models that are commonly used in academic papers and in industry applications. They have the advantage of publicly accessible code and weights, enabling a common platform for benchmarking and full transparency.

In the following, we outline the specific LLMs chosen for our experiments and provide a rationale for our choice. We have primarily focused on decoder-only LLMs, designed for prompt-based interactions and currently employed in a wide array of applications, which makes them highly relevant for our research.

4.5.1 Llama 2

In July 2023, Meta AI introduced the Llama 2 models [55], an advanced iteration of the Llama family of open-source LLMs [54]. This release was a significant milestone for the field of AI, as Llama 2's performance has been evaluated as comparable with leading closed-source LLMs, thereby promoting a more transparent and responsible approach to AI development. The Llama 2 suite encompasses models with varying parameter sizes—7B, 13B, and 70B—all pre-trained using a self-supervised learning approach on an extensive text corpus with two trillion tokens. They have been further refined through supervised fine-tuning and Reinforcement Learning with Human Feedback (RLHF), resulting in the Llama 2-Chat variants, tailored for dialogue-based use cases. Through comprehensive benchmark assessments, Llama 2-Chat models have demonstrated superior capabilities compared to other open-source models, and in human evaluations for helpfulness and safety, they have shown comparable performance to some prominent closed-source models like those underlying ChatGPT and Bard.

Our research employs Llama 2 70B Chat, the most proficient model of its size at the time of its release. As of October 2023, it remains among the top models on Hugging Face's leaderboard with all its derivatives. Llama 2 70B Chat, built upon a decoder-only architecture, excels in text generation and prompt-based interactions, which facilitates seamless integration into our experimental framework. The parameter size plays a crucial role in determining model accuracy, with larger sizes leading to more precise responses. This model's impressive parameter size enables nuanced understanding and generation of text, which is a critical feature. Moreover, the model's availability on Hugging Face greatly simplifies its integration into our study, allowing for direct inference without the need for additional fine-tuning. This is particularly advantageous given the significant computational resources required for handling such large models. These features of the model, in combination with its widespread adoption in both academic and industrial settings, make it a suitable choice for our purposes.

4.5.2 FLAN-T5

For the text classification task, we additionally employ Google's FLAN-T5 XXL model [7], a prominent representative of the encoder-decoder architecture. Based on T5 (2019) [47], FLAN-T5 is a family of models introduced in 2022, extending this architecture with significant enhancements. T5, an encoder-decoder Transformer, has been trained on a mix of supervised and unsupervised tasks, all having a text-to-text format. The supervised training has been performed on tasks from the General Language Understanding Evaluation (GLUE) [59] and SuperGLUE [58] benchmarks. The unsupervised and self-supervised tasks have involved reconstruction of text from corrupted inputs (e.g., by randomly removing 15% of tokens and replacing them with sentinel tokens). The FLAN-T5 models have undergone instruction-finetuning on over 1,800 language tasks, significantly improving their reasoning skills and promptability.

The FLAN-T5 XXL, as the name suggests, is among the largest variants within the FLAN-T5 family, offering a robust capacity for processing and generating nuanced text. For our specific research needs, the FLAN-T5 XXL model is chosen for its sophisticated encoder-decoder

structure and its proven proficiency in a wide array of language tasks. Although not explicitly trained for spam detection and policy compliance checking, its extensive training background and improved reasoning and prompt-based interaction capabilities make it a suitable choice for our text classification experiments and for evaluating the efficacy of LLMs as classifiers, particularly within the realm of cybersecurity.

4.5.3 Implementation Details

In the scope of this work, our primary focus is on utilizing LLMs for inference. We leverage the Hugging Face Hub [27], a comprehensive platform, which contains thousands of models and datasets, all open-source and publicly available. Its aim is to provide a platform where people can easily explore, experiment, and collaborate in building ML, and in general work together towards the goal of advancing it. A key feature of Hugging Face Hub is the *InferenceClient*, a unified interface that enables making inference via API requests with various models hosted on Hugging Face Hub. Hence, in this work we utilize the versions of FLAN-T5 XXL and Llama 2 70B Chat hosted by Hugging Face. The API-based inference allows us to use the largest available models of their respective families, without the need for specialized computational resources. Notably, Llama 2 70B Chat, further fine-tuned for dialogue interactions, is also available for prompt-based interaction via the *HuggingChat* [28] web interface, aligning well with our research needs.

The *InferenceClient* performs inference through the *text_generation* function, encompassing two distinct tasks: *text_generation* for decoder-only architectures and *text2text_generation* for encoder-decoder architectures.

- *text_generation*, also known as Causal Language Modeling, is the process of generating text that closely resembles human writing. It utilizes a decoder-only architecture and operates in a left-to-right context. This approach is ideal for tasks such as sentence completion or extending poetry when given a few lines as input. Models like the GPT or the Llama families exemplify this application, being widely used in chatbots, text completion, and content generation.

- *text2text_generation*, also known as Sequence-to-Sequence Modeling, involves transforming one piece of text into another. It relies on an encoder-decoder architecture and operates in both right-to-left and left-to-right contexts, making it suitable for tasks like translation or summarization, where both input and output are textual. T5 and BART [35] are notable examples.

In our research, we employ both methods. *text_generation*, represented by Llama 2 70B Chat, is what we are interested in for the applications **Data Augmentation** and **Data Generation**. For the **Text Classification** task, we employ the *text2text_generation*, represented by the FLAN-T5 XXL model. Nevertheless, we also experiment with Llama 2 70B Chat for text classification, since decoder-only models are widely used for chatbots nowadays and can be utilized for such tasks as well, even though it is not where the architecture excels. This would provide valuable insights and would allow us to see a full picture of the capabilities of LLMs.

5 Experiments: LLMs for Text Classification

5.1 Introduction

A fundamental task in the realm of cybersecurity is text classification, which underpins numerous applications – from filtering unwanted communications to ensuring adherence to regulatory frameworks and detecting attacks of any kind. Essentially, many cybersecurity challenges can be presented in textual form, ranging from natural language to intricate system or server logs. Therefore, the sophistication of NLP techniques becomes increasingly relevant for effectively addressing and mitigating cybersecurity threats.

As digitalization continues to evolve, so does the complexity of the cybersecurity challenges. While ML techniques have proven effective in this domain, they often rely on extensive, labeled datasets for accurate classification of new data instances. However, in the dynamic and rapidly evolving field of cybersecurity, acquiring such datasets becomes a significant bottleneck. On the one hand, this is due to privacy concerns – cybersecurity data often contains sensitive information, and anonymizing it without losing important details is a delicate and challenging process. The task of labeling the obtained data is often even harder, taking into account that it might be difficult, expensive, and time-consuming due to the nature and volume of the data involved. Moreover, attackers are typically aware of the existing protection mechanisms and continually try to outsmart them. For instance, in email spam detection, an attacker would try to craft content that does not get classified as spam, and thus reaches the target. This evolving nature of threats leads to the constant need for retraining the existing models and adjusting the training data. As a result, the ability to classify data effectively without a substantial amount of pre-labeled data has transitioned from a mere advantage to an essential requirement in modern cybersecurity. This shift calls for innovative approaches that can adapt swiftly to new threats and minimize reliance on extensive datasets.

Over the past year, LLMs have demonstrated remarkable capabilities in understanding and generating human language. They have been trained on diverse and extensive text corpora, enabling them to comprehend and categorize text in ways that were previously unattainable with traditional ML models. Interestingly, since LLMs are able to generate a response in natural language, they are also able to answer different kinds of questions, and hence perform under instructions. In our research, we leverage these advanced capabilities of LLMs and investigate the effectiveness of zero-shot and few-shot learning paradigms, which allow LLMs to classify text without or with minimal task-specific training data. These techniques are well-suited for scenarios where labeled data is scarce or where models need to be rapidly adapted to new tasks without exhaustive retraining.

In this thesis, we dive into the two use cases, outlined in Chapter 2. They both involve processing and analysis of natural-language data, which is where LLMs excel. Nevertheless,

they cover different aspects of the natural language: informal and colloquial versus formal and regulatory, which would allow us to extrapolate the findings to a broader range of applications. In the context of spam detection, we assess the ability of LLMs to differentiate between legitimate and unsolicited messages using the UCI SMS Spam Collection – a dataset reflecting a real-world communication scenario where spam is an ever-present concern. For policy compliance checking, we explore the potential of LLMs to identify whether a text conforms to particular requirements imposed by (in this case) the GDPR – a task that holds significant importance for both organizations—navigating the complexities of data privacy laws—and general users.

This chapter is structured to provide a comprehensive overview of the text classification process using LLMs. We begin by outlining the methodology employed in preparing the datasets and configuring the models. Following this, we present the results achieved and identify areas of strengths and weaknesses. We conclude with a discussion on the implications of our findings, reflecting on the role of LLMs in cybersecurity and their potential to transform the landscape of automated text classification.

5.2 Methodology

This section outlines our systematic approach for evaluating the capabilities of LLMs in classifying text data within cybersecurity context.

Our methodology employs an *InferenceClient* for running inference on LLMs, allowing us to make predictions for data points via API requests, and therefore assess the capabilities of LLMs as text classifiers. The success of an LLM in this context largely depends on the interaction between the user and the model. This interaction is facilitated through the prompts, which play an important role in guiding the model to comprehend and execute the desired task. Hence, crafting effective prompts is a key aspect of our methodology. What we need for the purpose is a fully labeled dataset and well-designed prompt templates. A prompt template is a pre-formatted string where instances from the dataset are inserted to form a coherent prompt that the LLM can understand and respond to appropriately. The datasets we are utilizing are in .csv format and contain a column with text in natural language (e.g., SMS messages), as well as a column, containing a binary label (e.g., labels *spam* and *ham*). Given the LLMs' robustness in handling raw data and that they are designed to deal with text in natural language, we use the feature column from the datasets without any preprocessing. This approach allows us to test the LLMs' ability to interpret and classify unmodified real-world data. Our procedure involves inserting a sentence from the dataset into the prompt template, sending it to the LLM via an API request, and then recording the model's response, which is expected to be a binary label (i.e., a prediction). This process is automated and applied to the entire dataset. Once the data collection is complete, we can evaluate the LLM's performance using the different metrics, detailed in Section 4.3.

For both models used for this task, we set the *temperature* parameter to 0.1. Generally, a lower temperature leads to less variation in the choice of words, thus pushing the model towards more deterministic outputs, which is desired in binary classification tasks like the

one at hand. Furthermore, we limit the *max_new_tokens* to 3 in order to ensure the outputs are brief and to the point, aligning with our binary label requirements. This also enhances computational efficiency, as generating fewer tokens generally requires less processing power. To manage large datasets efficiently and avoid API rate limits, we process the data in batches of 1000 entries. This allows processing the full datasets at once, using a rotation of different authentication tokens, and therefore maintaining consistency in classifications.

This sets the stage for an evaluation of the potential of LLMs in text classification tasks, particularly within the dynamic field of cybersecurity.

5.3 Evaluation

In this section, we go through the different experiments we performed for the two use cases and present the results achieved.

5.3.1 Spam Detection

Here, we focus on the experiments conducted to assess the effectiveness of LLMs in classifying spam messages, with an emphasis on understanding model responses to various prompting strategies.

Initially, we employed the RTF framework for creating prompt templates, aiming to elicit binary classifications from the LLMs. Figure 5.1 illustrates our basic prompt template, where each message from the dataset is inserted, expecting a predicted *spam* or *ham* label in response. Important to note is that our experiments involve no training data and 100% of the dataset is used for testing.

```
You are a security expert specialized in identifying spam SMS messages.

Your task is to analyze the content of the provided SMS message and determine whether it
is "spam" or "ham".

Message:  "{message}"
Classification:
```

Figure 5.1: Initial prompt template for spam detection.

The FLAN-T5 XXL model adapted very well to this strategy, achieving an F1 score of 74.34%, with a recall of over 97%, indicating that it captured almost all of the spam messages. Llama 2, however, failed to generate a label in the desired format. This could be explained with the model's inherent strengths in text generation and completion, rather than in solving specific tasks. As a result, it was unable to do what we expected based solely on descriptive instructions, without seeing the desired format.

To mitigate this problem, we tried another strategy and transitioned to a few-shot setting. This involved adjusting the prompt accordingly by providing an explicit example of what

the model was supposed to do immediately after explaining the task and before introducing the actual message. This is frequently done in the RTF framework for showing the expected format, rather than merely describing it. To avoid influencing the results by giving an example from the dataset, we generated a synthetic SMS message using an LLM. Given the nature of the example, this technique is still often considered zero-shot. Figure 5.2 shows our first modification, introducing an example for the positive class (*spam*), while keeping the rest of the prompt unchanged.

```
Message: "Congratulations! You've been selected to receive a $1000 Walmart Gift Card!
Claim your prize now!"

Classification: spam
```

Figure 5.2: Inserted example for the positive class.

In this setting, both models managed to produce the expected outputs. In fact, FLAN-T5 XXL's precision increased significantly from 59.98% to 84.89%, while the recall remained high, thereby achieving an F1 score of almost 90%. This way, it outperformed almost all of our baseline models, except for the SVM and the advanced DL models. Llama 2, when given the format of the output, also managed to produce the expected outputs and showed good performance: an F1 score of 72%, which is comparable with the initial result of FLAN-T5. Interestingly, for Llama 2, the precision and recall metrics were found to be equal.

Further experimentation involved adding an additional example for the negative class (*ham*), as depicted in Figure 5.3, in order to observe how it would influence the predictions. This change slightly improved FLAN-T5's recall with 2%, while keeping the precision unchanged. Therefore, the F1 score further increased to over 90%, an indicator of the model's stability. However, for Llama 2, it led to a substantial decrease in performance, with F1 score dropping to 38%. We further noticed that the amount of false positives drastically increased – from only 199 to 2,268, indicating that many more messages were classified as *spam*.

```
Message: "Lunch tomorrow at 1 pm? Let's try that new sushi place!"

Classification: ham
```

Figure 5.3: Inserted example for the negative class.

The performance shift in Llama 2 upon introducing an example for the negative class hinted at the fact that the LLM might have started following the pattern of the provided examples when predicting a label, rather than classifying the message based on its content. For instance, after seeing an alternating sequence between *spam* and *ham* messages in the prompt, it completes it in a pattern-consistent manner, e.g., by labeling the subsequent message as *spam* following a *spam-ham* sequence.

In order to check this assumption, we modified the prompt once again. This time, we removed the example for the positive *spam* class and the prompt remained only with an example for the *ham* class. What we expected was that the Llama 2 model would classify a lot of the messages as *spam*, in order to follow the pattern. The results achieved confirmed our hypothesis. The Llama 2 model achieved a recall of 99.60%, meaning almost all spam messages were identified. However, the precision dropped to less than 15% and the number of false positives doubled: 4,296. Therefore, this modification in the prompt resulted in nearly all messages being classified as spam, indicating the model's tendency to follow the example pattern over analyzing content. This was not the case for FLAN-T5 though, where despite the slight decrease in the performance, it remained high.

The observed problem with the Llama 2 model is termed *recency bias* – a known issue for LLMs [45], where the model is more likely to predict a label based on the pattern of the provided examples rather than their content, as desired. We discuss this limitation and its implications in detail in Section 5.4. Table 5.1 presents a summary of the results.

Table 5.1: Performance achieved by FLAN-T5 and Llama 2 for SMS spam detection.

Model	Prompt with	Accuracy	Precision	Recall	F1 Score
FLAN-T5	No example	90.95%	59.98%	97.72%	74.34%
	Example for positive class	97.04%	84.89%	94.78%	89.56%
	Examples for both classes	97.18%	84.79%	96.25%	90.16%
	Example for negative class	93.11%	66.52%	97.86%	79.20%
Llama 2	No example	-	-	-	-
	Example for positive class	92.59%	72.81%	71.25%	72.08%
	Examples for both classes	58.51%	23.66%	94.11%	37.82%
	Example for negative class	22.85%	14.76%	99.59%	25.71%

Qualitative Analysis

In exploring the qualitative aspects of text classification using LLMs, we employed the CoT technique to prompt the models for explanations of their classifications. This way, we could leverage the advantage that LLMs offer over conventional ML models, and namely: the ability to articulate their decision-making processes. This approach, illustrated in Figure 5.4, involved increasing the *max_new_tokens* to 256, allowing the LLMs to generate detailed explanations. Interestingly, while FLAN-T5 consistently failed to provide explanations beyond mere labels (either *spam* or *ham*), Llama 2 demonstrated an impressive capability to articulate reasoning behind its classifications – even without being given an example.

Therefore, we conducted an in-depth analysis using the Llama 2 model. For the purpose, we handpicked five data points from each classification category (FP, FN, TP, TN) using the model's predictions with the best-performing RTF prompt setup (i.e., when given one example for the positive class). This way, we aimed to examine the explanations that the same

```
You are a security expert specialized in identifying spam SMS messages.

Your task is to analyze the content of the following SMS message: "{message}".

Question: Should the message be classified as "spam" or "ham"? Explain why!
Answer:
```

Figure 5.4: Prompt template for spam detection using the CoT technique.

model would give, as well as to observe whether the assigned labels would remain consistent.

We found that the model's explanations were generally logical and aligned with human reasoning across all categories. Interestingly, the model remained consistent with the initial classifications made. There was only one message from the TN category, whose label got incorrectly changed: *"I am in office:)whats the matter..msg me now.i will call you at break:)"*. The explanation by the LLM, detailed in Figure 5.5, revealed that while the model's reasoning was valid in a broader context, it misinterpreted the cues in this specific message.

```
The message should be classified as "spam".

Explanation:

The message contains several red flags that indicate it may be spam:

   1. Use of emoticon: The use of the emoticon ":)" at the beginning of the message is
      a common tactic used by spammers to make their messages appear more friendly and
      personalized.
   2. Vague greeting: The greeting "I am in office" is quite vague and doesn't
      provide any specific information about the sender or the purpose of the message.
      Legitimate messages usually include a more detailed greeting or introduction.
   3. Request for response: The message asks the recipient to "msg me now" and promises
      a call back "at break". This could be a phishing attempt to get the recipient to
      respond with personal information or click on a malicious link.
   4. Poor grammar and punctuation: The message contains grammatical errors and lacks
      proper punctuation, which is a common trait of spam messages.

Therefore, based on these red flags, it is likely that the message is spam and should be
classified as such.
```

Figure 5.5: Explanation generated by the LLM for the classification of an SMS message.

This observation held also for the FP and FN classes – valid augmentation, however, misinterpreted in the specific context. We noted that the model often flagged messages requiring any kind of information from the recipient as *spam* – a logical but contextually limited approach. This was observed in messages such as *"Send me your id and password"*, which, although legitimate, could indeed be considered dangerous. Requesting credentials

is generally ill-advised, and recipients should avoid sharing such sensitive information, especially via text messages. Nevertheless, also messages like *"So what u doing today?"* were classified as an attempt to *initiate conversation with the recipient and get information about their daily activities*. While this argument is valid and phishing messages indeed typically require providing personal data urgently, it is also a part of legitimate conversations. This highlights a key limitation of LLMs in this task: the need for more contextual information is crucial for accurate decision-making. For instance, having additional details such as whether the sender is an unknown number or related to the recipient might improve the performance a lot.

The problem was observed for the FN instances as well. Many unsolicited advertisements that a person would likely classify as spam were justified due to the fact that they had been sent from legitimate, well-known companies: *The message is offering a chance to win a prize for customers who purchase their products before a certain date. The message is not trying to deceive or manipulate the recipient in any way, and it is not attempting to sell anything illegal or unethical*. Nonetheless, in all of the examples, the LLM checked for the typical signs of spam and phishing messages, such as suspicious links, phone numbers, requests for personal information, unsolicited messages, commercial content, and inconsistent branding. This was also included in the explanation. Despite the misinterpretations, the LLM always checked thoroughly for typical signs of spam and explained its classifications reasonably. Without further context, some of the messages might even confuse a human.

What typically helps us classify correctly though is the additional information we have, such as the sender's number. Therefore, supplying the LLM with similar contextual details could enhance its performance. We can address this in a privacy-conscious manner, e.g., by providing solely binary information on whether the number is in the contact list, rather than the number itself. Regarding the messages containing advertisements, while they are labeled as *spam* in the benchmark dataset, some recipients might find them legitimate and informative. This indicates that the definition of spam is subjective and further fine-tuning of the models based on user feedback could be beneficial. Alternatively, the model could offer suggestions or warnings to users about potential spam or harmful content, instead of making direct classifications. The impacts of such approaches will be further discussed in Section 5.4.

5.3.2 Policy Compliance Checking

In this section, we present the experiments conducted to assess the effectiveness of LLMs in determining whether a privacy policy adheres to specific requirements. In our case, we focus on the GDPR and its DPO requirement, as described in Chapter 4. Unlike the traditional ML techniques, where a certain portion of data is reserved for training, we utilized 100% of our dataset exclusively for evaluation.

Mirroring our approach in the spam detection use case, we initially employed the RTF framework to create a prompt template that we could use for evaluating all data points within the DPO category of our selected dataset. However, there was a significant difference between the two use cases. The LLMs employed in this study both have information cutoffs after the enactment of GDPR, hence they possess inherent knowledge about the requirements we are interested in. However, policy compliance in our dataset has been measured based on five

handpicked requirements and for each of them, specific criteria and guidelines have been chosen. Therefore, to ensure that the LLMs had the same context for labeling the sentences as the human annotators, we had to provide all the guidelines that the creators of the dataset used. Consequently, we integrated the DPO-related guidelines outlined by Müller et al. [40] into our prompt template. It is noteworthy that the DPO category encompasses the most generic guidelines (without specific caveats) that would likely correspond to what the LLMs naturally assume about DPO compliance. An additional modification was the inclusion of the fact that sentences had been anonymized within the dataset, highlighting the absence of real contact details which might be expected. This setup allowed for a fair evaluation of LLMs' capabilities in this task, offering them the same contextual information a human annotator would have. Figure 5.6 presents the resulting prompt template.

```
You are a compliance officer specialized in checking sentences for GDPR compliance.

Your task is to analyze the content of the provided sentence and determine whether it
conforms to the Data Protection Officer (DPO) requirement.

Assign a binary label to each sentence:
- 0 if the sentence does not contain relevant DPO-related information.
- 1 if the sentence discloses DPO-relevant information and is compliant to the GDPR
requirement.

Consider a sentence compliant (assign label '1') if it either names the DPO or an
equivalent authority, or provides their contact details.
Sentences are also compliant if they refer to some list or enumeration of the required
information.  For example, the sentence "You can reach our data protection officer
via:" is considered compliant, even though the sentence itself does not contain the
contact details.

For confidentiality and privacy reasons, the sentences have been anonymized, i.e.,
numeric values have been randomized, and names, email addresses, companies, and URLs
have been substituted with generic placeholders (e.g., "company_42653").

Sentence:  "{sentence}"
Label:
```

Figure 5.6: Initial prompt template for GDPR compliance checking, category DPO.

In our initial prompt, we did not include an explicit example of the desired format. Contrary to our expectations that this might hinder Llama 2 from generating meaningful output, both Llama 2 and FLAN-T5 managed to produce the desired responses. The performance metrics revealed that FLAN-T5 achieved an F1 score of 56.43%, while Llama 2 47.84%. Interestingly, the models demonstrated nearly identical precision, suggesting that they are equally capable of identifying genuinely compliant sentences. However, the difference in the F1 scores came from the recall rate, where FLAN-T5 led by approximately 25%. This indicated FLAN-T5's enhanced ability to recognize a wider array of compliant sentences. Nonetheless, it is worth

mentioning that without any specific training data, FLAN-T5 outperformed significantly three out of our four baseline models. Llama 2, though slightly behind, still showed comparable performance. Both LLMs, in terms of precision, outperformed most of the baseline models, which were explicitly trained for this task.

Next, we modified our prompt template by incorporating an example for the positive class, which demonstrated to be beneficial for the LLMs in spam detection. Figure 5.7 shows the selected example. To maintain objectivity and ensure an unbiased evaluation, again, we chose not to use a data point from our dataset. Instead, we generated a representative sentence using Llama 2, thereby avoiding any potential influence on the result.

```
Sentence:  "For questions about our privacy and data protection policies, please
contact our data protection officer at the given address and contact details."

Label:  1
```

Figure 5.7: Inserted example for the positive class.

Upon evaluating all data points in the dataset with this modified prompt, we observed a notable improvement in the performance of both models, mirroring the improvements seen in the spam detection case. Specifically, the precision of both models increased, and for Llama 2, the recall improved as well. Consequently, the F1 score for FLAN-T5 reached 60.94%, outperforming all our baseline models, while Llama 2's F1 score hit 55.45%, outperforming most of the baseline models. These results are particularly impressive, considering that there was no dedicated training on the dataset. Another interesting aspect was seeing the change in the number of FPs and FNs across the models and the prompts, as detailed in Table 5.2. The inclusion of an example for the compliant class resulted in a decreased number of FPs for FLAN-T5, meaning the model became more robust in identifying genuinely compliant sentences. On the other hand, for Llama 2 it led to a decrease in the FNs, indicating a more refined detection of truly non-compliant sentences. The other metric remained relatively stable for both models. Thus, adding an example to the prompt had a distinct impact depending on the model, yet it consistently proved to be advantageous.

Table 5.2: Number of false positives and false negatives in DPO compliance checking.

Model	**Prompt with**	**False positives**	**False negatives**
FLAN-T5	No example	367	107
	Example for positive class	275	112
Llama 2	No example	254	204
	Example for positive class	256	157

It is also noteworthy that our initial statement about the potentially misleading nature of accuracy as a metric in imbalanced datasets was indeed validated by our findings. In every

scenario we examined, the accuracy consistently exceeded 97%, which is remarkably high. Across the different prompts and models, we observed only minor fluctuations within the range of 0.5. Nevertheless, this is due to the fact that the positive instances were only around 2% out of all sentences. Table 5.3 provides an overview of the performance achieved by the two models with each prompt variation.

Table 5.3: Performance achieved by FLAN-T5 and Llama 2 for DPO compliance checking.

Model	Prompt with	Accuracy	Precision	Recall	F1 Score
FLAN-T5	No example	97.42%	45.55%	74.15%	56.43%
	Example for positive class	97.90%	52.34%	72.95%	60.95%
Llama 2	No example	97.51%	45.26%	50.72%	47.84%
	Example for positive class	97.76%	50.10%	62.08%	55.45%

Qualitative Analysis

Here, we delve into the advanced capabilities of LLMs to elaborate on their decisions. Given the nature of privacy policies and their complexity, this could be particularly beneficial for those responsible for ensuring or checking for compliance. Similarly to our approach in spam detection, we adopted the CoT prompting strategy, which enabled us to trace the LLM's reasoning process more clearly. Additionally, we increased the *max_new_tokens* parameter to 256, facilitating the generation of more detailed explanations.

In order to explore interesting examples, we applied the following strategy when selecting them. We took into account all predictions made by the two models under the two different prompts. Our analysis focused on extracting four distinct sentence categories: FP, FN, TP, and TN, representing specific response types across all predictions. From each category, we selected five notable examples. The LLMs were then tasked to assess whether these examples were compliant or not, and to provide reasoned explanations for their classifications. This approach enabled us to investigate and understand the reasons for the consistent behaviour of the LLMs, e.g., why some sentences had been consistently misclassified. For the purpose, we modified the initial RTF prompt by restructuring it, while keeping the *Role* and the guidelines, which were relevant here as well. The *Task* was adapted as shown in Figure 5.8.

```
...
Your task is to analyze the content of the following sentence:  "{sentence}".

Question:  Should the sentence be classified as "compliant" or "non-compliant"?
Explain why!
Answer:
```

Figure 5.8: Prompt template for DPO compliance checking using the CoT technique.

Similarly to the case for spam detection, we observed that the FLAN-T5 model, despite being prompted to provide an explanation, tended to respond with simple classifications like *compliant* and *non-compliant*. Hence, we focused once again exclusively on the Llama 2 model, which managed to provide the content we required.

First, we examined the FP category. A notable pattern emerged here: nearly all misclassified sentences included anonymized contact information, such as email addresses, phone numbers, or physical addresses. We selected five representative examples and requested explanations from Llama 2. Interestingly, the model changed its response for the sentence *"we have appointed a data protection officer for our company."*, with explanation shown in Figure 5.9, which was entirely correct. However, the classifications for the other examples remained unchanged. A significant observation was that a lot of the sentences contained contact details for the Information Commissioner's Office (ICO), the UK's data protection authority. The Llama 2 model interpreted those as *provided contact details of an equivalent authority*, and hence the positive label. Another aspect was that many sentences presented contact details without any context, i.e., to whom they belong. The LLM interpreted this as a way to get in touch with the privacy policy's author. According to our guidelines, this makes the sentence DPO-compliant, even though the DPO is not explicitly named. Therefore, the LLM's argument was valid.

```
The sentence should be classified as non-compliant.  The sentence mentions the
appointment of a Data Protection Officer (DPO) but does not provide their contact
details or name.  It does not refer to a list or enumeration of required information
either.  Therefore, it does not meet the criteria for a compliant sentence under GDPR.
```

Figure 5.9: Explanation generated by the LLM for the classification of a policy sentence.

Next, we took a look at the FN category, where we also identified a pattern in the content of the misclassified sentences. Most of them had the structure of *"if you have any concerns, please contact us"*, however, they neither named the DPO or an equivalent authority, nor provided their contact details. Hence, the LLM justified its negative classification by claiming that even though the sentence referred to the possibility of contacting the company, it did not provide the necessary information to do so. Therefore, even with the CoT prompt, these labels remained consistent. Nevertheless, we also encountered sentences in the dataset containing general statements like *our email*, which had been classified as non-compliant by the human annotators, suggesting a certain inconsistency in the manual labeling. This highlighted that additional context would be beneficial to avoid misclassification by both humans and LLMs.

Moving to the TN and TP categories, the labels assigned by the LLM remained unchanged. In these cases, the LLM not only classified correctly but also provided detailed and meaningful explanations. The LLM's reasoning was logical and used elements from the sentences and the prompt to justify its decisions, which made it easy to understand the rationale behind the classification.

In general, we can draw several key conclusions from our analysis. First, there were indeed some misclassifications, particularly due to the treatment of the ICO as equivalent to a DPO.

However, other instances of 'misclassification' were reasonable given the limited context available. A notable observation was the inconsistency in our labeled samples. This indicates that given the information at hand, even human annotators might have misclassified certain sentences. Most importantly, the explanations generated by the LLM were consistently logical, tailored to the specific sentences, and allowed us to clearly understand the reasoning behind each label.

In line with our findings from the spam detection use case, it is evident that LLMs would benefit from more context. In the domain of privacy policies, for example, analyzing full paragraphs rather than fragmented sentences or sentence parts could significantly enhance the LLM's understanding and decision-making accuracy. Moreover, in the case of limited context, another strategy could be utilized. Instead of requesting a binary label, we could ask the LLM to provide a confidence score ranging from 0 to 1 on the compliance of a given sentence. We experimented with this method on sentences from our dataset and it proved effective, particularly when coupled with an explanation. Such an approach, especially in combination with human supervision, has the potential to greatly assist in both achieving and verifying GDPR compliance.

5.4 Discussion

The successful application of LLMs for text classification, particularly within the realm of cybersecurity, presents a wide range of opportunities. For instance, LLMs are often accessible to the general public as chatbots via user-friendly web interfaces (e.g., ChatGPT by OpenAI), which opens up new avenues for non-technical users in information verification. Furthermore, they could enhance existing solutions, e.g., by labeling data automatically, thus addressing the problem with insufficient labeled data. This section aims to discuss key observations from our experiments, highlighting the potential and limitations of current LLMs.

Our study compared FLAN-T5 XXL and Llama 2 70B Chat, two widely used LLMs, built upon distinct architectures. The performance differences observed between these models indicate that architecture and training methodology play a significant role in how well LLMs follow instructions and perform on specific tasks. FLAN-T5 with its fine-tuning for task-solving appeared to be particularly suitable for text classification and showed performance, comparable with specialized state-of-the-art NLP models. However, it consistently fell short in providing explanations about the decisions made. On the other hand, Llama 2 excelled in generating meaningful explanations every time it was asked to. This highlights the importance of selecting models and architectures aligned with the specific objectives.

Our experiments also revealed the critical role of prompt design in the effectiveness of LLMs. A key factor to consider is the *knowledge cutoff* of the LLMs employed for the task. While our selected models knew what spam characteristics to look for in a message without explicitly specifying them, for the policy compliance use case we needed to provide the specific GDPR guidelines so that we can ensure precise classification. Thus, the more specialized a task is, the more important the structure and the content of the prompt becomes for leveraging LLM capabilities.

The observations around the few-shot prompting and the addition of positive and negative class examples indicated the sensitivity of LLMs to the provided context. We observed that prompt engineering can "tune" these LLMs for various objectives, such as optimizing precision or recall. However, this could be a double-edged sword, leading to biased classifications, as seen with Llama 2 and its tendency to overfit to the pattern in the provided examples (e.g., by classifying most messages as *spam* after being given an example for a *ham* message). This limitation is also known as *recency bias* [45], where most recent examples in a prompt can disproportionately affect the output. It is particularly evident in models, optimized for text completion, as their aim is to complete the prompt in a reasonable way. This phenomenon is especially problematic in text classification tasks where we expect the models to analyze the content of the provided data point, not to return an arbitrary label. Robust training to mitigate such biases is therefore crucial.

Another important aspect in utilizing LLMs for text classification is the risk of 'hallucination' or arbitrary label assignment. This is particularly challenging with closed-source LLMs, which operate as 'black boxes' and are accessible only through APIs, thus significantly limiting our understanding of their internal decision-making processes. Therefore, it becomes particularly problematic if their decisions are not supervised by humans and have a direct external impact. Demanding reasoning from LLMs for their conclusions is a promising approach to mitigate this problem, as it allows for verification of the rationale behind model responses. This is a feature, which lacks in conventional ML models, and is one of the most important benefits of LLMs we should leverage.

Despite the differences in the proficiency of generating explanations, what we observed across all experiments was the need for more context. Like humans, LLMs need comprehensive context to make reliable decisions. They can often misinterpret a certain situation if not given enough information. For both use cases examined, we identified that the lack of context hindered LLM performance. Their ability to understand context is fascinating, regardless of whether in complicated privacy policies or with slang and informal text. Therefore, it is paramount to avoid ambiguity. Nevertheless, more context also implies more computational resource requirements. In particular, making inference with LLMs often demands significantly more computational power compared to training and deploying traditional ML models for similar tasks. This trade-off is a crucial factor to consider.

Overall, our research underscores the transformative potential of LLMs in enhancing cybersecurity solutions. Without being fine-tuned for the specific task, they already achieve performance similar to specialized ML models. This makes them particularly suitable for cybersecurity scenarios, characterized by limited data availability or evolving data (e.g., new spam techniques). Furthermore, class imbalance poses a huge challenge for both traditional ML models and advanced DL techniques, which struggle to learn the patterns of the minority class, unless tackled appropriately. With this approach, since there is no need for labeled datasets, class imbalance is also not relevant. Nevertheless, despite the promising strengths in text classification, their limitations in bias, context sensitivity, and prompt dependency necessitate further research. Addressing these challenges is key to fully leveraging LLMs in practical cybersecurity applications as text classifiers.

6 Experiments: LLMs for Data Augmentation

6.1 Introduction

As we have seen in the previous chapter, LLMs demonstrate remarkable performance as text classifiers in both zero-shot and few-shot settings. Our experiments showed that they can classify unseen data instances without the substantial cost of fine-tuning. Instead, we provide solely a task description and a few examples in the prompt and leverage the pre-trained LLM knowledge via inference. However, despite the impressive advantages of this approach, there are also several major drawbacks that need to be taken into account.

One major limitation is the maximum prompt length allowed. This constraint affects various aspects: it limits the complexity of task descriptions, which is particularly challenging for specialized tasks that require providing specific guidelines that need to be followed, e.g., in policy compliance checking. It also restricts the number of in-context training examples that can be included, which showed to be beneficial in our experiments. Additionally, the amount of context that can be provided for the data point to be classified is also constrained. As we have seen, this is a significant issue considering that LLMs benefit from more extensive context, such as paragraphs or full policies, rather than mere sentences in the policy compliance checking use case.

Another concern with the direct text classification approach is the necessity for real-time inference on LLMs, accessed via remote server APIs. This process can not only become costly over time but also allows the hosting organizations to access the data sent for inference, raising further security concerns. However, hosting an LLM with a substantial number of parameters locally could be even costlier due to the computational resources and memory required. These issues could potentially make the approach unscalable in real-world applications.

In contrast, other advanced deep learning methods have shown strong performance across a variety of supervised learning tasks, offering solutions to complex NLP problems with remarkable efficiency. Compared to LLMs, they require fewer resources and can be hosted locally, thus avoiding security concerns. However, these models depend on extensive labeled datasets for training to fully realize their potential. This makes it challenging to apply them in scenarios where collecting and labeling large amounts of high-quality data is prohibitive in terms of cost and effort. The data scarcity issue is particularly acute in cybersecurity context, where the rarity of certain malicious patterns can impede the development of high-performing predictive models. Even if there is a certain amount of labeled data, a common issue in cybersecurity contexts is high class imbalance, where due to the lack of minority class instances, ML models often fail to learn patterns, and thus recognize them in unseen data. The lack of diverse training data can also often lead to another major problem: overfitting. Powerful ML models memorize the patterns and correlations specific to the training dataset.

However, they might not hold in general, leading to very poor performance on unseen data.

To address these challenges, data augmentation emerges as a pivotal technique in machine learning. It involves modifying existing training data points to generate new instances, thereby artificially expanding the size and improving the quality of training datasets. Traditional methods for text data augmentation rely on techniques such as synonym replacement, random insertion, or back-translation. While effective to a degree, these approaches often fall short in adequately capturing the nuanced context inherent in textual data. However, as we have already seen, this is where LLMs excel – in understanding context. Moreover, they are also able to generate rich, nuanced text data that reflects the complex variability found in natural language.

This chapter aims to make use of these unique capabilities of LLMs. Instead of employing them as pre-trained models for fine-tuning or as backbone for prompt-based classification, we utilize them to enrich training sets with synthetic yet realistic text instances, which can be used for training various ML models. This approach allows targeting specific issues by employing different augmentation strategies. For instance, we can augment the minority class only to counteract class imbalance, or both classes in order to mitigate overfitting. The sophisticated context-understanding ability of LLMs suggests their potential for the task. In this thesis, we explore their efficacy within cybersecurity contexts, where data quality and balance are essential for building accurate detection systems. In particular, we focus on the spam detection and policy compliance checking use cases. This would allow us to not only investigate the capabilities of LLMs, but also to assess the efficiency of this approach in comparison to the direct classification one.

The subsequent section outlines the methodology employed in this study, followed by a detailed presentation of the experiments conducted for the two use cases. We include descriptions of the augmentation processes and in-depth evaluations of the results, encompassing both quantitative and qualitative analyses. Finally, we discuss the broader implications of our findings and identify potential directions for future research. This structured approach aims to offer a comprehensive understanding of the role of LLMs in augmenting datasets, and thus their potential to enhance cybersecurity solutions.

6.2 Methodology

In this section, we present the methodology we followed to evaluate the capabilities of LLMs for data augmentation in cybersecurity contexts, specifically spam detection and policy compliance checking.

In order to perform the desired experiments, we begin by preparing the data to be augmented. For this purpose, we utilize the training dataset from each use case, which has been previously employed for training the respective baseline models. The strategy for selecting data points for augmentation, as well as the quantity of augmentations per data points, varies depending on the specific use case and will be detailed in the subsequent sections.

A crucial aspect of the augmentation process is the creation of a prompt template that provides the LLM with the necessary information. This template is designed using the RTF

framework and includes a placeholder for the specific data instance to be augmented by the LLM. It clearly defines the task and the expected format with an LLM-generated example. In particular, we ask the LLM to rephrase the provided sentence while maintaining the original meaning, e.g., by restructuring it or using synonyms. The goal is to enrich the training data with new vocabulary and representations of the data points, which could enhance the model's robustness and ability to classify unseen data. An example of this prompt template for the spam detection use case is illustrated in Figure 6.1. Inserting a data point in it leads to a coherent prompt, ready to be sent to the LLM.

```
Your task is to generate a new version of the given message while retaining its original
meaning.

Example:
Original: "Congrats! You've won a $1000 gift card from Walmart. Click on this link
to claim now: www.offer.com"
Rephrased: "Congratulations! You have been awarded a $1000 Walmart gift card. Follow
this link to redeem your prize now: www.offer.com"

Original: "{message}"
Rephrased:
```

Figure 6.1: Prompt template used for augmenting training data for spam detection.

Now that we have the necessary data and prompt templates, we proceed with the augmentation itself. We employ the Llama 2 model from Hugging Face, selected for its advanced text generation capabilities. We interact with the LLM through the *InferenceClient*, mirroring our approach in Chapter 5. Again, we use the model off-the-shelf, without any task-specific fine-tuning. To configure the LLM for our needs, we adjust several parameters. We increase the *temperature* to 0.9 to encourage more creative, non-deterministic outputs, which is our primary goal: to get a diverse representation of the provided data instance. The *max_new_tokens* parameter is fixed at 128, which is appropriate for the length of the data points in our datasets. This not only helps for computational efficiency, but also guides the model in producing the expected output. Additionally, to ensure variability in the generated text, especially when creating multiple versions of the same data point, we set a unique seed in the *text_generation* function. For the purpose, we utilize a specially crafted function that uses a base seed and factors in the data point's position in the dataset and the variant number in order to generate a seed deterministically. On the one hand, this ensures that each data point has a unique seed for each version. Furthermore, it also ensures reproducibility of the augmentation – if we run the process multiple times, the same augmentations will be produced.

Using this setup, we go through the selected data points and generate new versions. The process involves sending coherent prompts to the LLM via an API request and receiving a rephrased sentence in return. Pattern matching techniques are employed to extract the rephrased sentence from the response. Since the augmentation process is entirely automated, we have also implemented error handling mechanisms. It is possible that the LLM fails to

understand the format required and does not produce a satisfactory output. For this case, we have implemented a retry mechanism, allowing up to five attempts. To ensure that each attempt would result in different responses, we increment the *seed* with each attempt. If the model persistently fails to generate a valid output, the data point is skipped. Additionally, a brief pause is introduced between requests to manage cases when the model is overloaded.

At the end of the augmentation process, the generated data is stored in .csv files, categorized based on the amount and strategy of augmentation. We then merge the various augmented datasets with the original training data, and after thorough shuffling, we proceed to train the baseline models. The validation set remains unchanged. This setup allows us to systematically assess the impact of LLM-based augmentation on model performance using the predefined evaluation metrics, and thus draw reliable conclusions. Furthermore, we conduct a qualitative analysis of the generated instances to better understand the observed results.

6.3 Evaluation

6.3.1 Spam Detection

In this section, we describe the experiments conducted for the spam detection use case, as well as the results achieved.

Augmentation Process

The dataset we utilized for spam detection presented an inherent challenge due to its high class imbalance. To address this issue, our first augmentation strategy aimed at increasing the amount of data points in the minority class, thereby potentially enhancing overall model performance. The initial step involved segmenting the training data, corresponding to 80% of the original dataset, into two categories based on the labels *spam* and *ham*. This resulted in one set exclusively containing spam messages and another with only legitimate messages. With this setup, we applied our first augmentation strategy: incrementally generating 1 to 5 new variants for each spam message. This approach yielded five separate datasets, each containing 1 to 5 new versions, respectively. Essentially, the total number of positive data points was low, allowing for the efficient generation of more versions. However, given the length of the SMS messages, we capped at five to ensure diversity and prevent from generating duplicates.

The relatively limited size of the original training dataset prompted us to explore another augmentation strategy: equal augmentation of both the *spam* and *ham* messages. This approach was designed to maintain the inherent data distribution. To this end, we incrementally augmented random selections of 5%, 10%, 20%, and so on, up to 100% of messages from both classes, with each percentage point representing the proportion of messages selected for augmentation. Consequently, at the 100% augmentation level, a new version of every message in the dataset was generated by the LLM.

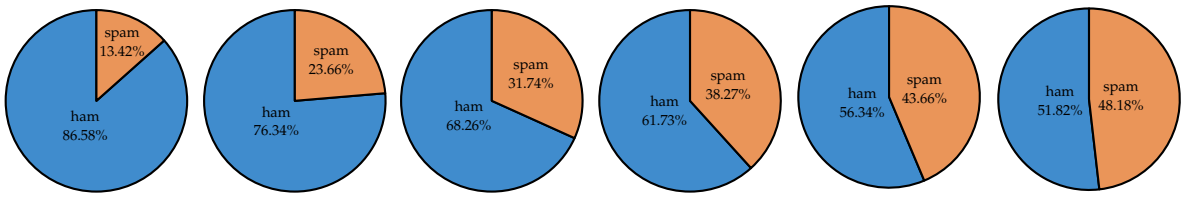
During the augmentation process, we observed that the employed model managed to generate new versions as required within the given amount of trials. Additionally, we also checked for duplicate entries in the augmented datasets to ensure diversity. As a result, at

the end of the augmentation pipeline, we had a wide variety of augmented data: augmented spam only (1 to 5 new versions of each message), as well as balanced augmentation across both classes (5% to 100% of the training data). Each dataset was stored in a separate .csv file. In the subsequent sections we evaluate this data quantitatively and qualitatively.

Results

Having described the process of obtaining augmented data, we now analyze the performance achieved by our baseline models across various levels of augmentation. We follow a systematic approach to assess the quality of the augmentation quantitatively.

First, we investigated the impact of augmenting solely the minority class, which is *spam* in this context. For the purpose, we employed the following approach: to the training data, we added incrementally and consecutively 1 to 5 new versions of the *spam* messages. Figure 6.2 illustrates how the distribution of the two classes changed in each version of the dataset. Initially, the training set contained 3,859 *ham* messages and 598 *spam* messages, i.e., 86.58% *ham* and 13.42% *spam*. After adding 5 additional versions of the spam messages, we observed that the dataset was nearly balanced.



(a) Original (b) +1 version (c) +2 versions (d) +3 versions (e) +4 versions (f) +5 versions

Figure 6.2: Class distribution of spam and ham within the different datasets.

The idea behind this augmentation strategy was that by including more versions of the *spam* messages, we would enrich the training data and also eliminate the huge imbalance that was present between the classes. This balance could allow the model to learn from a more equal representation of both classes, potentially enhancing overall performance. Our expectation was improvement in detecting spam due to the increased variation in the spam examples. From a performance metric perspective, this would lead to a higher recall for the positive class, as the model would become better at identifying various spam characteristics.

Indeed, what we observed was that augmenting the *spam* class generally led to an improvement in recall for the conventional baseline models, with varying degrees of success, i.e., on average more of the spam messages were correctly identified. Across the different augmentation levels, all models achieved a recall of at least 90%, except for KNN and XGBoost. Nevertheless, they had the lowest initial recalls and still showed significant improvements: 7% for KNN and 17% for XGBoost, the latter being the most notable increase. However, for all these models, this improvement in recall came with a decrease in precision. On the other hand, for the DL models like LSTM and BERT, the opposite happened. The precision increased, while the recall remained relatively unchanged, leading to an improved F1 score in

comparison to the initial. Furthermore, the positive impact of the augmentation was evident from the first version added. These observations suggest that augmenting the minority class affects model performance differently depending on the model employed and highlights the importance of selecting a model in alignment with the overall goals.

The other strategy we employed was augmenting both classes equally, maintaining their original distribution to prevent introducing bias towards any class. Again, we consecutively added different portions of augmented data to the training set, from 5% to 100%. This resulted in *spam* messages increasing from 598 to 1,196 and *ham* messages from 3,859 to 7,718 across training sets, while keeping the class ratio unchanged. Afterwards, we trained the selected baseline models. Our hypothesis was that this type of augmentation would lead to a balanced improvement in classification for both classes due to the diverse linguistic representation introduced, thus expecting an increase in the F1 score.

Our analysis, based on the average performance across five runs, showed that each model managed to surpass its initial F1 score with augmented data, as illustrated in Figure 6.3. Interestingly, this improvement was primarily due to a rise in precision, while recall remained consistent. This trend suggests that equal augmentation of both classes reduces the misclassification of *ham* messages as *spam*, a critical objective in spam detection. Most models, except for LR, observed an increase in their precision, with many exceeding 99%. LR shows a slight decrease, however, it had the highest precision of 99.16% with the original training set already.

Another important aspect proved to be the quantity of data augmentation. While some models, such as BERT, benefited consistently from augmentation, others like SVM, KNN, and XGBoost showed more variable responses, sometimes even underperforming compared to the original training dataset. This highlighted the importance of choosing an appropriate level of data augmentation based on the characteristics of the model at hand.

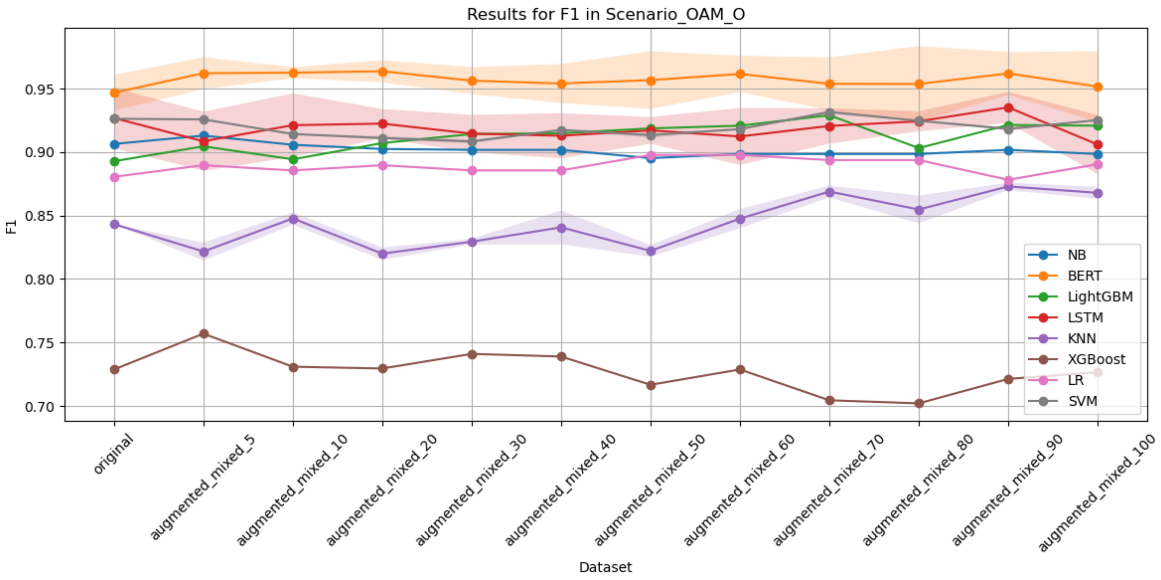


Figure 6.3: F1 scores achieved by the baseline models with different portions of augmentation.

Finally, in order to check the standalone quality of the augmented data, we replaced the entire training dataset with 100% augmented data points for both classes. In other words, rather than using the original messages, we trained the models entirely on rephrased versions by the LLM. This kept both the size of the training dataset and the class distribution the same. The validation set, again, also remained unchanged. The underlying assumption was that if the augmentation was of high quality, the models would still perform adequately on the original validation data, suggesting that augmented data could potentially replace real-world data in training scenarios.

Table 6.1 displays the achieved results. Even though the performance of all models trained solely with rephrased data points was inferior compared to those with the original dataset (see Table 4.3), they still performed decently. Furthermore, this was an anticipated outcome, as it is challenging for rephrased text to capture every nuance present in the original version. Interestingly, BERT managed to achieve almost equivalent performance to the baseline, highlighting its advanced capabilities in effectively handling augmented data. These findings lead us to conclude that LLMs can indeed serve as a powerful tool for text augmentation.

Table 6.1: Results achieved by training the baseline models with rephrased messages only.

Model	**Accuracy**	**Precision**	**Recall**	**F1 Score**
NB	96.49% ± 0.00	96.61% ± 0.00	76.51% ± 0.00	85.39% ± 0.00
LR	93.44% ± 0.00	94.19% ± 0.00	54.26% ± 0.00	68.94% ± 0.00
KNN	92.55% ± 0.07	91.07% ± 0.47	49.26% ± 0.33	63.94% ± 0.35
SVM	96.13% ± 0.00	94.92% ± 0.00	75.17% ± 0.00	83.90% ± 0.00
XGBoost	90.83% ± 0.00	88.52% ± 0.00	36.24% ± 0.00	51.43% ± 0.00
LightGBM	93.88% ± 0.00	90.91% ± 0.00	60.40% ± 0.00	72.58% ± 0.00
LSTM	95.89% ± 0.89	83.52% ± 5.18	86.85% ± 2.15	85.04% ± 2.75
BERT	98.73% ± 0.52	94.15% ± 3.72	96.64% ± 0.42	95.34% ± 1.82

Qualitative Analysis

In this section, we explore the qualitative aspects of data augmentation using LLMs. Our focus is on evaluating the quality of the rephrased messages in terms of semantic integrity, stylistic variations, and their potential impact on model performance.

Throughout the augmentation process, we closely monitored the newly generated versions. Our analysis revealed that the LLM consistently produced high-quality rephrasings. The rephrased messages not only preserved the original message's meaning but also introduced meaningful variations in structure and style, demonstrating a level of sophistication comparable to human rephrasing.

A key strength of the employed LLM proved to be its ability to retain the semantic content of the original message. For instance, the message: *"Hi Jon, Pete here, Ive bin 2 Spain recently & hav sum dinero left, Bill said u or ur rents mayb interested in it, I hav 12,000pes, so around £48,*

tb, James" was rephrased to: *"Hi Jon, it's Pete. I recently returned from Spain and have some leftover currency. Bill thought you or your parents might be interested in it. I have 12,000 pesetas, which is equivalent to around £48. Take care, James"*. This example highlights the LLM's ability to produce a new version of the given message while preserving the original's meaning. Moreover, it is written in proper English – from vocabulary and grammar to punctuation.

The model also demonstrated impressive capabilities in altering sentence structure and word choice. It adeptly used synonyms, modified sentence lengths, and changed grammatical structures while keeping the meaning intact. However, this variation sometimes led to a shift in style from informal to formal, which, while maintaining the message's essence, altered its original tone. To illustrate this observation more clearly, consider the examples in Table 6.2.

Table 6.2: Examples of original and rephrased SMS messages by an LLM.

Original message	**Rephrased message**	**Comments on changes**
"It's ok i wun b angry. Msg u aft i come home tonight."	"Don't worry, I won't be angry. I'll message you after I get home tonight."	The rephrasing maintains the meaning but shifts from informal to slightly more formal tone.
"Yo dude guess who just got arrested the other day."	"Greetings! I'm writing to inform you that someone you know has recently been taken into custody by law enforcement."	The informal tone is completely transformed into a formal one, changing the style significantly.

These examples show how the LLM can sometimes fail to maintain the original message's stylistic context, particularly when informal language is used. Considering the context of SMS messages, this is often the case since slang and colloquial expressions are common. The tendency to formalize informal content could impact the augmented data's effectiveness, especially when the real-world data predominantly consists of informal language.

Hence, the efficacy of LLM-augmented data may not be fully apparent when tested on a fixed set, extracted from the same dataset as the training data (as in our case). Such a testing set often mirrors the training data's style and contains similar vocabulary, e.g., abbreviations like *'msg'* for *'message'* or *'he din c me'* for *'he didn't see me'* would likely be common in both. However, in real-world scenarios characterized by diverse linguistic expressions, the variation in style and introduction of new tokens by the LLMs could be highly advantageous. The enriched vocabulary during training could potentially make the model more robust to various linguistic patterns, improving its performance. We further discuss this aspect in Section 6.4.

Overall, LLMs demonstrate a high capability in rephrasing sentences, preserving semantic content, and introducing structural variations. However, the tendency to alter the original style, especially in informal contexts, presents an area for further refinement. Future enhancements in LLMs should focus on better recognizing and preserving the original message's stylistic nuances to maximize the effectiveness of augmentation in diverse linguistic scenarios.

6.3.2 Policy Compliance Checking

In this section, we dive into the experiments we performed to evaluate the capabilities of LLMs for data augmentation in the realm of policy compliance checking.

Augmentation Process

As described in Chapter 4, in this use case we allocated 30% of the whole dataset for validation purposes, instead of 20% as in spam detection. The decision was driven by the extremely limited number of positive instances. This resulted in a training dataset of 290 compliant and 12,758 non-compliant sentences, making it highly imbalanced: for every positive instance, there were over 43 negative ones. This imbalance posed a significant challenge for ML models. To establish a baseline performance, we initially applied class weights to compensate for the uneven distribution but still observed the considerable impacts on the DL models.

A common strategy for mitigating such imbalances is data augmentation, wherein additional samples of the minority class are synthetically generated. This approach directly addresses the imbalance and can potentially enhance the model's proficiency in recognizing patterns within minority data. Following the methodology applied in spam detection, we employed an LLM to rephrase compliant sentences. The goal was to generate realistic LLM-modified sentences that retain the meaning of the originals (thus remaining *compliant*) while introducing sufficient variability to the training data. Given the dataset's severe imbalance, we specifically refrained from augmenting the negative class. This could not only exacerbate the imbalance, but also demand extensive resources due to the significant number of instances.

Similarly to our spam detection approach, we extracted the positive instances from the training data. Afterwards, for each sentence, we consecutively generated 1 to 5 new versions. The decision to limit the number to 5 was influenced by the brevity of the sentences, which often made it challenging, even for humans, to create multiple meaningful rephrasings without changing the core meaning. This cap ensured minimal repetition and prevented the introduction of nearly identical sentences, which could lead to model overfitting. Overfitting emerges as a significant risk when augmenting training data, because the model could become too tailored to the specific patterns within that data, thereby losing its ability to generalize and perform well on new, unseen data.

For each augmentation level, we stored the produced sentences in a data frame, maintaining the original dataset's format and labeling them as DPO-compliant. The augmentation pipeline yielded five distinct datasets in total, each containing 1 to 5 variations of the positive training sentences. We made sure to remove potential duplicates from each dataset to prepare them for experimentation. The subsequent sections describe the results achieved with these datasets and provide insights into the quality of the augmentation obtained through this process.

Results

In our experiments with the five datasets containing rephrased versions of the compliant sentences, we follow a systematic approach to evaluate the impact of varying augmentation proportions on model performance.

Our strategy involved consecutively adding 1 to 5 rephrased versions of the compliant sentences to the original training data, followed by shuffling the resulting datasets. This process effectively increased the number of positive instances by up to six times, altering the positive-to-negative ratio from approximately 1:43 to 1:7. In theory, we could have aimed for an even more balanced ratio. However, our method's dependency on the available sentences imposed a natural limitation. Beyond a certain threshold of generated rephrasings, the risk of encountering repetitive and lower quality augmented sentences increases, which could potentially harm the trained model's performance. It was, therefore, crucial to find a balance between augmenting the dataset and maintaining the integrity and diversity of the augmented content, to ensure meaningful results. In general, by exposing the models to more possible compliant sentences during training, we expected that we would help them identify patterns that make a sentence *compliant*.

The results we observed after training the baseline models on these datasets varied. LR did not show a consistent improvement with augmentation. In fact, its performance even progressively declined with the addition of more LLM-rephrased versions, suggesting a potential mismatch between the model's learning mechanism and the newly introduced data. On the other hand, exactly the opposite happened for BERT: it exhibited a steady performance increase with augmentation, peaking at an F1 score of 60.84% (±0.83 standard deviation) with five versions. This indicates BERT's proficiency in leveraging the nuanced variations in the augmented data, likely due to its deep understanding of language context.

Similarly, the LSTM model also showed performance gains with augmentation, though the degree of improvement varied depending on the amount of augmented data. Nevertheless, it managed to improve its initial F1 score with up to 6.66% with four augmented versions, indicating the positive impact of the LLM-produced data. The SVM model, which already performed best with the original dataset, continued to exhibit improvements at different levels of augmentation as well, although to a lesser extent. Interestingly, the most significant boost was observed with the addition of just one augmented version, achieving the highest F1 score of 61.59% across all baseline models. This suggests that even minimal augmentation can substantially benefit SVM's comprehension of the positive class.

Table 6.3 summarizes the average performance metrics for each model across the different portions of augmentation. We observe a general increase in the F1 scores for SVM, LSTM, and BERT. This trend supports the hypothesis that augmented data can enhance model performance, highlighting the potential of LLMs for this task.

Table 6.3: Average results with augmentation of the positive class for DPO compliance checking. Difference to the baseline shown in absolute percentage points.

Model	Accuracy	Precision	Recall	F1 Score
LR	96.03% ± 0.00 ↓ 0.42%	34.65% ± 0.00 ↓ 3.02%	86.61% ± 0.00 ↓ 2.10%	49.50% ± 0.00 ↓ 3.38%
SVM	97.95% ± 0.00 ↓ 0.15%	53.42% ± 0.00 ↓ 3.62%	70.16% ± 0.00 ↑ 8.06%	60.63% ± 0.00 ↑ 1.17%
LSTM	96.55% ± 0.51 ↑ 0.66%	38.83% ± 3.71 ↑ 4.16%	85.22% ± 2.43 ↓ 3.49%	53.05% ± 3.23 ↑ 3.38%
BERT	96.96% ± 0.55 ↑ 0.67%	42.78% ± 4.19 ↑ 5.20%	91.58% ± 2.75 ↓ 3.26%	58.01% ± 3.62 ↑ 4.29%

These results also emphasize the crucial role of model selection in contexts involving augmented data. Notably, advanced DL models appear to benefit more from LLM-augmented data, which could be attributed to their architectural sophistication. This finding is particularly relevant considering the higher data demands of these models. In contrast, conventional ML models show varying responses with augmented data, aligning with their generally lower data requirements. The observation that models with higher data needs also show greater improvements with LLM-augmented data is a promising aspect, suggesting a synergistic relationship between advanced models and LLM-driven data augmentation.

Qualitative Analysis

In this section, we delve into the characteristics of the sentences rephrased by the LLM. To investigate the quality of the produced sentences, we focused on the dataset containing all five rephrased versions of each original compliant sentence. Our augmentation process was designed in a cumulative manner, with each containing all the versions of its predecessor. For instance, the dataset with three versions encompassed the two-version dataset plus an additional new version for each compliant sentence. Therefore, an analysis of the five-version dataset was sufficient for comprehensive insights.

A key observation was that the LLM consistently managed to generate sentences in the desired format within the allocated number of attempts. Furthermore, these sentences varied from one another while retaining the original meaning, which was a crucial aspect of our augmentation strategy. To illustrate this, consider the compliant sentence *"if you wish to do this, please use the contact details at the end of this policy."* from our original training dataset. The LLM produced the following five distinct versions:

1. *"If you desire to pursue this, kindly utilize the contact details provided at the conclusion of this policy."*
2. *"if you want to accomplish this, please make use of the contact information provided at the end of this guideline."*
3. *"If you desire to take such action, please reach out to us using the contact information provided at the bottom of this guideline."*
4. *"If you desire to perform this action, please refer to the contact information provided at the bottom of this document."*
5. *"If you're interested in doing so, kindly make use of the contact details provided at the end of this policy."*

Contrary to what we observed in the spam detection case, where the LLM sometimes altered the style of the SMS message, here the LLM consistently preserved the style of the sentence. This can be attributed to the formal nature of the text in privacy policies, characterized by its adherence to grammatical rules and absence of slang. Hence, no shift in the style was observed. The rephrased sentences maintained a professional tone and could

seamlessly replace the original in a privacy policy without altering its essence, demonstrating that they are natural and appropriate.

These observations underscore the LLM's effectiveness for text augmentation, particularly in the context of formal, official documents where a neutral tone is paramount. They suggest that LLMs are well-suited for tasks involving restructuring of formal text, potentially playing a significant role in enhancing cybersecurity solutions where the accuracy and consistency of language are critical.

6.4 Discussion

Applying machine learning techniques to cybersecurity problems is often challenged by the scarcity of extensive, labeled datasets and significant class imbalances in the available ones. The employment of LLMs for data augmentation presents a promising approach to address these issues. By introducing variability, LLM-augmented data can enhance model robustness and lead to improved recognition of specific threat patterns, thereby advancing cybersecurity.

Our experiments have revealed that the overall impact of LLM-augmented data is highly model-dependent. Furthermore, factors such as the amount and strategy of augmentation can also be crucial. We have seen that inappropriate augmentation can, in fact, even harm model performance. Nonetheless, this approach showed to be particularly beneficial for deep learning models, which is promising considering that they typically require a significant volume of data to demonstrate their full capabilities. This can be attributed to the high quality of the LLM-rephrased data points, which consistently preserved the original content's core meaning while introducing new vocabulary.

A potential limitation we identified though was a shift in the style of the text after augmentation, which was especially evident in the spam detection scenario involving SMS messages. The LLM-augmented messages maintained strict adherence to grammatical rules, reflecting a high level of language proficiency. However, this is not always the case in everyday messaging, which tends to be more casual, containing abbreviations and slang. This issue was further exacerbated in the dataset used, reflecting messaging styles from over a decade ago, when people often shortened words due to character limitations in SMS messages. In more contemporary contexts, this might be less of an issue due to the evolution of messaging styles. The problem was also not observed with formal, regulatory texts, such as those found in privacy policies. However, future work in the area should consider the specific characteristics of the original dataset and ensure that the LLMs are able to generate sentences that preserve the original style.

Moreover, strict supervision during the augmentation process is important. LLMs, particularly those optimized for text completion, may produce content that, while format-compliant, diverges from the intended meaning or context. This necessitates careful review and validation of the augmented data, possibly beyond automated checks.

Another consideration is that the approach often involves online inference. Given their significant computational requirements, LLMs are commonly hosted on remote servers and accessed through APIs. Typically, users send queries to these servers, which are processed in

real-time, and a response is generated immediately. This operational model introduces several challenges. First, augmenting large datasets can be costly, though generally less so than text classification, as it is a one-time process prior to system training. More critically, there is the issue of data privacy. For cybersecurity tasks, where data is often sensitive, ensuring anonymity is paramount. Since LLMs are typically hosted externally and accessed via APIs, there is a risk of training data exposure to external entities. This raises the need for either carefully anonymizing the data before augmentation or considering local hosting of LLMs, which, however, might not be feasible given the resource requirements.

Further important aspect is the efficiency of the data augmentation process. Our approach largely depends on the inference speed of the model being used. In particular, the LLMs we have employed are capable of generating responses within seconds, thereby demonstrating a high level of efficiency for such applications. However, inference speed can vary significantly based on the model complexity, the length of the input provided and the output expected, as well as the computational resources available. Since our approach involves the sequential processing of individual data points (i.e., processing data points one by one), the response time per data point is quick, but the total time will add up with the number of data points. Consequently, the time required to process the entire dataset tends to increase linearly with the dataset's size, assuming that each data point takes approximately the same amount of time to process. It is important to consider, however, that if the complexity of the data points varies significantly, this can lead to inconsistent processing times, thereby impacting the total time needed. Additionally, given that LLMs are often hosted on remote servers, they are subject to rate limits that could affect the efficiency of the augmentation process, especially as the number of requests increases. This becomes a notable challenge when augmenting exceptionally large datasets. As a potential improvement of the approach, batch processing could be considered (i.e., processing several data points in a single request). This approach would significantly decrease the number of API requests required. Yet, it also introduces a complication: the handling of more complex inputs, and thus outputs, which could lead to longer processing times per request. Future work should therefore carefully examine the viability and effectiveness of this method for large datasets in comparison to the current approach, while also considering the quality of augmentation. This is especially interesting to observe in scenarios where multiple versions of a data point should be generated within a single batch.

Lastly, the inherent limitation of this approach must be taken into account. Data augmentation can be beneficial, but its effectiveness is constrained in scenarios with extreme class imbalances or severely limited original data. Over-augmentation could lead to redundancy, failing to significantly enhance, or even potentially worsening, the learning capabilities of the model. Therefore, careful evaluation of the specific context and nature of the data beforehand is essential.

In conclusion, while LLMs for data augmentation in cybersecurity demonstrate significant potential, there are limitations and challenges that must be addressed to harness the benefits of the approach. These considerations offer important directions for future research.

7 Experiments: LLMs for Data Generation

7.1 Introduction

In the dynamic field of cybersecurity, the development of robust and effective ML models is essential in addressing various challenges. The success of these models depends heavily on the access to diverse, high-quality data, which, however, is not always readily available. As discussed in the previous chapter, data augmentation, particularly through the use of LLMs, presents a potential solution by enriching existing text datasets. However, this approach has significant limitations, as we have observed.

On the one hand, it requires providing real-world data to the LLM in the prompt, which has to be augmented. Given the computational demands of hosting LLMs, this task is often outsourced to external organizations. Consequently, the provided data in the prompts can be accessed by external entities, which can be problematic. Cybersecurity data often involves dealing with sensitive personal or business information, which cannot be easily anonymized without stripping away critical details. In combination with strict privacy laws and ethical considerations, sharing such data becomes a non-viable option.

Another inherent limitation of this approach is its dependency on the quality and quantity of the available text data. Augmentation can typically only be performed to a certain extent before its quality decreases and starts to negatively impact model performance. Furthermore, in some cybersecurity scenarios, labeled data might be completely absent, which hinders the approach from even being applied. This is often due to the rapidly evolving nature of cyber threats or the rarity of specific incidents, leading to a scarcity of relevant data. Additionally, after obtaining such data, the process of labeling it is also challenging, as it requires expert knowledge and is often time-consuming.

Generating purely synthetic data, which does not rely on existing datasets, stands out as a pivotal solution to this bottleneck. Nevertheless, traditional text generation methods, such as rule-based systems and templating, often struggle to produce diverse and contextually nuanced text. Their tendency to yield repetitive and predictable outputs fails to reflect the complexities of natural human language, thereby limiting their effectiveness in training robust ML models.

This is where LLMs come into play. The key characteristic of LLMs is their generative capability, enabling them to produce novel and contextually relevant content. This is rooted in their advanced architectures, which allow for the generation of large volumes of text data that can mimic the nuances of natural language. Moreover, their prompt-based nature allows for targeted generation of specific types of data, addressing specific scenarios. This opens doors for generating text datasets for training ML models in various domains. Particularly in the context of cybersecurity, the demand for extensive and diverse datasets is paramount.

The ability of LLMs to produce datasets tailored to specific cybersecurity scenarios, thereby facilitating the training of resilient cybersecurity models, could become invaluable. This is what we will explore in the following.

In this chapter, we delve into the application of LLMs for data generation, focusing on their role in mitigating the shortage of quality data in cybersecurity contexts. We assess whether LLMs can offer a scalable, efficient, and compliant way to generate synthetic data for training cybersecurity systems. Through this exploration, we aim to provide a deepened understanding of the capabilities and limitations of LLMs in generating relevant cybersecurity data. Moreover, we seek to offer insights into the practical integration of these models into cybersecurity workflows, setting the stage for future advancements in this area. As a field that heavily relies on data, cybersecurity stands to benefit significantly from advances in data generation techniques. Hence, the sophisticated generative abilities of LLMs present an opportunity to simulate diverse cyber threats and scenarios, which could directly impact the development of systems, capable of detecting various malicious activities.

The structure of this chapter is systematic. We begin with an in-depth explanation of how we employed LLMs for synthetic data generation, followed by a comparative analysis of various ML models trained with LLM-generated data. We focus once again on our two key use cases: spam detection and policy compliance checking, aiming to provide insights about the general performance of LLMs, as well as in comparison to the other applications explored in this thesis. We conclude the chapter with a discussion of the findings and their broader implications for the field of cybersecurity.

7.2 Methodology

In this section, we present the methodology used to evaluate the capabilities of LLMs for synthetic data generation in the context of cybersecurity.

Similarly to the data augmentation approach, our generation process begins by constructing a prompt, which provides the necessary information to the LLM and instructs it to perform the task. However, unlike data augmentation, the synthetic data generation does not rely on pre-existing training data. In fact, despite the availability of real-world data for our use cases, we assume that no such data in the form of text-label pairs exist. This allows us to explore the capabilities of LLMs to generate relevant data in a *zero-shot* scenario, without prior real-world data exposure.

In order to obtain synthetic training data, we employ Llama 2 70B Chat, excelling in text generation. As described in Chapter 4, for interacting with this model, Hugging Face also provides a user-friendly web interface – *HuggingChat*. We adopt this approach in our study for several reasons. First, given the fact that we do not need different data points from our original datasets, the initial prompt remains unchanged for the whole generation process. Moreover, conversational context is managed well within the GUI, allowing the LLM to accurately recall previously mentioned details, and provide more relevant responses that build upon earlier interactions. Furthermore, *HuggingChat* also allows us to set a system prompt, which establishes the context for the further interaction with the LLM. This gives us

an opportunity to use the system prompt for providing all the rules that the LLM needs to consider when generating a data point. This is crucial, since the quality of the synthetic data heavily relies on this setup, and hence its impact on the performance too. We employ the RTF framework once again for defining the *Role*, the *Task* with all the rules, and the desired *Format* for the generated instances. This approach is particularly useful, because within the conversation, we do not need to specify the rules again.

Once the context is set, a second prompt, i.e., an actual data generation prompt, is provided to the LLM, instructing it to produce a certain amount of data points. We do this in batches of 50, which allows for having a good oversight of the produced content, and also enables efficient generation of large number of instances. In order to further enhance the diversity of the synthetic data, after generating a certain volume, we provide a *diversity* prompt to the LLM. In this prompt, we request generation of more diverse examples in comparison to the previously generated, aiming to overall increase the synthetic data's diversity. By generating data points within the same context, we ensure that the produced instances are indeed distinct. For our two use cases, spam detection and policy compliance checking, we employ different generation strategies regarding which class should be enriched and with what amount. The specific details are outlined in the subsequent sections.

The process repeats until the target data volume is reached and all generated data points are stored in the respective files, according to the strategy. We then eliminate duplicates and form datasets with varying amount of synthetic data (detailed later). Ultimately, we end up with a wide range of datasets for each use case, composed exclusively of synthetic data in the same format as the original training dataset. To assess the quality of the synthetic data and its impact on ML models, we experiment on our baseline models with different strategies, e.g., incrementally adding synthetic data to real-world data or entirely replacing real-world data with synthetic. This approach allows for thorough evaluation. Additionally, we empirically review the generated data points, aiming to understand the observed trends and pinpoint limitations or future research directions.

7.3 Evaluation

7.3.1 Spam Detection

In this section, we delve into the experiments conducted with LLM-generated data for the spam detection use case.

Generation Process

In order to assess the efficacy of LLM-produced synthetic data in spam detection, we applied the following strategy. We instructed the LLM to generate batches of spam and non-spam (*ham*) SMS messages, with each batch containing 50 messages per category.

Given the LLM's inherent understanding of SMS spam characteristics, we avoided imposing explicit content guidelines in the system prompt. Due to the universality of the task at hand, we relied on the LLM's pre-trained knowledge to generate messages that not only exhibit

the usual structure of an SMS message, but also reflect typical spam tactics and legitimate content found in such messages. This approach granted the LLM considerable autonomy in the data generation process, and it successfully managed to generate messages that distinctly represented both categories – *spam* and *ham* – while adhering to the required format.

An interesting observation during this process was the need to clarify our intent in the prompt itself. When prompted to generate spam messages, a task potentially mirroring malicious intents, the LLM initially declined, citing ethical concerns and legal implications. Hence, we had to explicitly clarify the educational and research purposes behind our request, in order to obtain the necessary messages. This response underscores the ethical considerations programmed into the LLM, but also the fact that such measures can be bypassed, and thus spam content can still be generated. We delve deeper into this aspect in Section 7.4.

Throughout the generation phase, we carefully monitored the messages being generated and ensured they were in the desired format. We continuously stored the generated data points of each category in a separate .txt file. Subsequently, we randomly mixed the messages in a consistent 50/50 split of spam and ham, but with varying total quantities, storing them in different .csv files with the respective label (*spam* or *ham*). This process yielded datasets ranging from 350 to 5,000 messages per category, with a balanced representation of spam and ham. The datasets were structured with *message* and *label* columns, mirroring the format of the original dataset, which allowed direct integration with the existing models, as well as mixing with the real-world data at hand.

Although the generation process was manually conducted to observe trends in smaller datasets, the methodology is scalable and can be automated for larger data requirements. This flexibility demonstrates the utility of LLMs in generating extensive datasets for ML applications. Next, we inspect the quality of the generated messages through experiments on the baseline models and discuss our observations for their content.

Results

In this phase, we evaluate the impact of synthetic data on the performance of the eight baseline models for the spam detection use case, outlined in Chapter 4.

Initially, our goal was to assess how enriching the original training data with synthetic data would influence the performance. We progressively added varying amounts of synthetic data to the initial training dataset, increasing its size by more than three times at the highest volume (with 5,000 messages per class). This approach with increasing the amount of synthetic data aimed at identifying the threshold, where additional data stops improving, or even worsens, model performance due to overfitting or introduction of noise. By adding the same number of messages for each class, we preserved the class distribution at each step, thus mirroring a real-world scenario. Throughout all experiments, the validation set remained unchanged.

Our hypothesis was that high-quality synthetic data would enhance model generalization by introducing diverse examples, absent in the original training dataset, and therefore helping the model learn more general patterns. This could potentially increase its robustness in identifying existing and evolving types of spam. To test this hypothesis, we evaluated the baseline models using the unchanged validation data and illustrated the results in Figure 7.1.

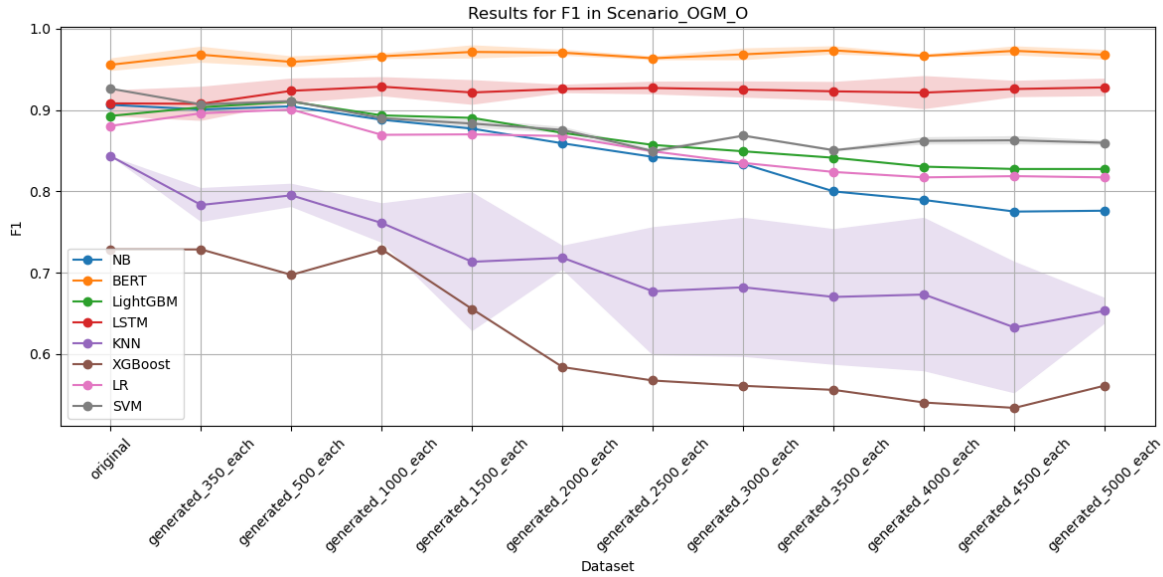


Figure 7.1: F1 scores achieved by the spam detection models when trained with different portions of LLM-generated synthetic data added to the original training data.

What we observed was that adding synthetic data to real-world training data had a different impact on the performance, depending on the model that was being trained. In particular, we identified distinct trends between DL models and conventional ML models:

- **DL Models**: Both LSTM and BERT demonstrated consistently improved performance with the addition of synthetic data, peaking between 1,000 and 3,500 synthetic messages per class. Moreover, BERT's F1 score surpassed 97% with several portions of synthetic data. Thus, it outperformed all the three fine-tuned LLMs in [34], where the highest F1 score achieved is 96%. These results suggest that DL models benefit from the diversity and volume provided by LLM-generated data, which is a promising aspect, considering their substantial data requirements.

- **Conventional ML Models**: Contrarily, models like XGBoost and KNN experienced performance drops of nearly 20% across the different portions of synthetic data added. LR and LightGBM initially showed improvements with up to 500 synthetic messages, but their performance also declined thereafter. In general, all models in this category perform (at least) slightly worse when more synthetic data is added to the training set. This pattern indicates that they are more sensitive to the variability or noise introduced by the synthetic data, suggesting it may not be beneficial to supplement them with such.

To further assess the quality of the synthetic messages, we employed the following strategy: we substituted the entire training dataset with synthetic data while keeping the validation set, once again, unchanged. We trained our baseline models using progressively larger amounts of data. This approach was designed to provide insight into how well the different quantities

of LLM-generated data points could represent the real-world data distribution. Essentially, the experiment aimed to determine whether we could rely on synthetic data only for training models that are to be used in practical, real-world applications – an outcome that could prove highly advantageous. Although we expected that the overall performance in this scenario was likely to be lower compared to using the original messages, our primary interest laid in quantifying and understanding the exact nature of this impact.

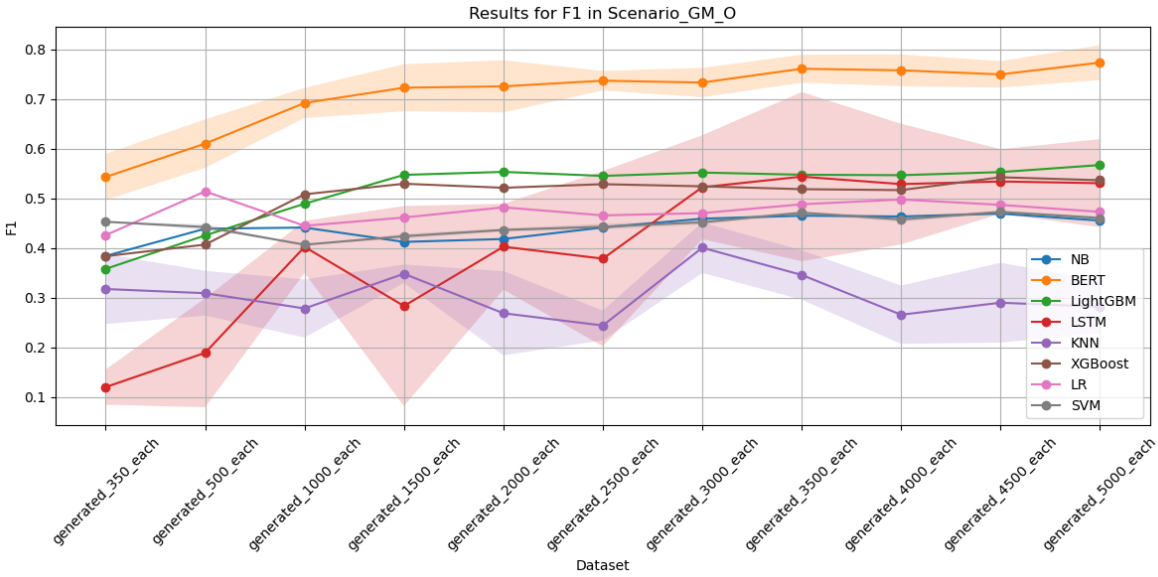


Figure 7.2: F1 scores achieved by the spam detection models when trained with different portions of LLM-generated synthetic data only.

Figure 7.2 illustrates the F1 scores achieved by all baseline models when trained with increasing amounts of synthetic data. The experiments showed that, although with notable differences, most models generally benefited from more synthetic data:

- **SVM**: Achieved an F1 score of 45.26% with only 350 messages per class and increasing the amount of training data points led to 2% improvement at most, suggesting a quick adaptation to the synthetic data's characteristics and no need of large amounts of data.
- **LR**: Showed varying performance, with F1 score starting from 42.56% at 350 training messages per class. Interestingly, it achieved its maximum F1 score of 51.38% with just 500 messages per class. Beyond this point, the F1 score fluctuated within this range, indicating the importance of identifying the optimal quantity of training messages.
- **KNN**: Exhibited inconsistent performance, with F1 score ranging between 24% and 40%, indicating a potential mismatch between its learning mechanism and the synthetic data's characteristics.
- **NB**: Improved moderately with more data but plateaued at higher volumes, suggesting a limit of its capacity to leverage synthetic data.

- **LightGBM and XGBoost**: Despite starting with a lower F1 score, LightGBM consistently outperformed XGBoost as more synthetic messages were added, showing an overall increase of over 20%. Its F1 score trend displayed a logarithmic curve, indicating greater gains initially. XGBoost, while exhibiting slightly more modest improvements, mirrored this logarithmic progression in its performance curve.
- **LSTM and BERT**: Demonstrated significant improvements. LSTM managed to increase its performance from 11% to over 54%. Nevertheless, this was the starting performance of BERT and it managed to achieve 77% at 5,000 synthetic messages, surpassing baseline performances with the original training set of some models. This indicated once again the proficiency of DL models to learn from synthetic data.

In conclusion, the effectiveness of LLM-generated data in training is highly model-dependent and sensitive to the quantity used. While DL models seem to benefit more from increasing amount of synthetic data, there are also differences among them. BERT, in particular, seems well-suited for leveraging LLM-generated data. Due to its extensive pre-training, with the least amount of synthetic data it can achieve performance, comparable to the best one for many other models. In addition, it improves consistently with more synthetic data. On the other hand, LSTM shows very poor performance with small amounts of data, while SVM, for instance, achieves close to its highest performance with the same training set. In general, the results highlight the importance of model selection when using synthetic data.

Qualitative Analysis

This section evaluates the qualitative aspects of LLM-generated messages. We compare these with real-world ones to assess their suitability for ML model training in spam detection tasks.

Overall, the quality of the synthetic messages was notably high. According to our observations, both types of generated messages were clearly distinguishable and could be accurately classified by a human annotator. Upon analysis, we identified the following characteristics in our most extensive dataset, which contained 5,000 messages per class:

- **Ham Messages**: These were simple and often informal, covering a wide range of daily interactions in personal and professional contexts. Typical topics included social check-ins (*"Hey, how have you been?"*), reminders (*"Reminder: Dentist appointment tomorrow at 3pm"*), and personal notes (*"I miss you. Let's meet up soon"*). There was a clear pattern of short, direct sentences, often with a specific action or information conveyed, aligning with common legitimate messages one might receive in everyday life. There appeared to be no ambiguity in the messages; they were straightforward and to the point.
- **Spam Messages**: They typically contained offers or alerts designed to grab attention, such as winning prizes (*"Congratulations! You've won a $1000 gift card!"*), exclusive deals (*"70% off our premium products"*), or urgent warnings (*"Warning! Your computer has a virus!"*). A consistent pattern of offering something extraordinary or alarming was identified, which is a common tactic in spam to induce quick, often impulsive action from recipients. In general, they reflected typical spam content seen in real-world data.

Figure 7.3 provides a visual overview of the most common words in both classes, which confirms that the vocabulary used in each type corresponds well to real-world data. Therefore, the synthetic data seem appropriate for training ML models.



Figure 7.3: Word clouds for synthetic ham and spam SMS messages.

Nevertheless, it is important to examine how the synthetic data relate to the real-world data we utilized. We observed notable differences, particularly in language usage. The synthetic messages, despite their informal tone and use of slang (e.g., *"Sup? Wanna hit the gym together later?"* or *"Bro, that new game is lit! You tried it?"*), consistently exhibit correct spelling and punctuation. This contrasts with the messages from the UCI SMS spam dataset, which reflect the casual and often hurried nature of text messaging, where speed and ease of typing are prioritized. Thus, using shortened forms of words and phrases is common there – e.g., *"gudnite"* for *"good night"*, *"i tot he din c me"* for *"I thought he didn't see me"*, or *"do ü noe wat time"* for *"do you know what time"*. Omitting punctuation is also a common practice in the messages from the real-world dataset.

Furthermore, we analyzed the full synthetic dataset (5,000 messages per class) and the real-world dataset statistically; the results are presented in Table 7.1. The comparison of message lengths revealed significant differences. A distinctive feature between real-world spam and ham messages was namely the length: ham messages were, on average, twice as short as the spam messages. However, this characteristic was completely lost in the synthetic messages, where the two types were not only much shorter, but also revealed no significant variation in length. This resulted in almost twice as few words in an average synthetic ham message and almost three times as few for spam. This significant difference in length and complexity suggests that synthetic data might not adequately represent the nature of real-world messages used, potentially leading to misalignment in the model's learning process.

Interestingly, the synthetic data showed a slightly higher lexical diversity for both types of messages. This aspect of synthetic data is advantageous as it exposes models to a wider range of vocabulary, potentially aiding in better generalization. However, this does not compensate for the fundamental differences in message structure and complexity. These findings highlight the importance of aligning synthetic data closely with key characteristics of the real-world data it is intended to augment. Generally, this can be achieved with LLMs, suggesting their potential in enhancing ML models for such cybersecurity tasks.

Table 7.1: Statistical analysis of real-world and synthetic SMS messages.

Data type	Category	Average length	Average word count	Lexical diversity
Synthetic	Ham	43.01 characters	7.81 words	0.995
	Spam	49.41 characters	8.00 words	0.997
Real-world	Ham	71.48 characters	14.31 words	0.962
	Spam	138.67 characters	23.91 words	0.957

7.3.2 Policy Compliance Checking

In this section, we explore the experiments performed to assess the effectiveness of LLMs in generating data for enhancing policy compliance checking.

Generation Process

While establishing the baseline performance for the policy compliance checking use case, we faced a significant challenge due to the substantial imbalance within our dataset, which comprised only 290 positive instances against over 12,500 negative instances. This imbalance was particularly problematic for our DL models, which typically require a more balanced dataset for effective learning. To mitigate this issue, our strategy when employing LLMs for data generation focused on increasing the amount of positive instances in the training data.

The task of generating negative instances in this case was significantly more ambiguous than generating positive ones. Unlike the clear definition of GDPR *compliance*, the concept of *non-compliance* is inherently broad and diverse. Non-compliant instances could range from sentences that provide relevant information in an incorrect manner, to those discussing unrelated GDPR topics, or even containing completely irrelevant content. Given the complexity of accurately generating such diverse negative instances and the substantial volume already present, we decided to focus exclusively on enriching our dataset with positive instances. This approach later proved to be an effective strategy for answering our research questions.

A critical aspect of our data generation process was the development of a precise and clear system prompt. Unlike the spam detection use case, where the LLM had inherent knowledge about the characteristics of SMS spam, here, we had to explicitly specify the guidelines that the LLM should take into account. Drawing parallels to our use of LLMs for text classification, the model first needed a clear understanding of the compliance criteria before it could generate appropriate content. Hence, the specificity of our task underscored the necessity for clarity and precision in our prompt construction in order to ensure high-quality synthetic data. We employed the RTF framework for this purpose, incorporating the DPO-relevant guidelines defined by Müller et al. [40].

Once the prompt was finalized, we proceeded with the actual generation of synthetic sentences. In line with our previous experiments, we adopted a strategy of incrementally adding synthetic data to our training set. Starting with the initial 290 positive instances, we aimed to add 300 new synthetic sentences at each step, subsequently training and evaluating

the baseline models against the original validation dataset after each addition. This method allowed us to progressively increase the proportion of positive instances in the training set, aiming towards a more balanced distribution. We continued this process until we reached approximately 3,300 distinct positive instances, effectively reducing the initial imbalance from a ratio of 1:43 to less than 1:4 in favor of negative sentences. We targeted at this ratio, because too many synthetic instances might introduce noise, and thus harm model performance.

By the end of this process, we had created 10 separate datasets, each progressively including an additional 300 synthetic sentences, up to a total of 3,000. Each dataset was cumulative, meaning it included all previously generated sentences. These datasets were saved in .csv files, maintaining the format of the original dataset, and were then used to train the baseline models. Employing this incremental approach allowed us to observe the impact of varying levels of balance on model performance. This methodology provided valuable insights into the efficacy of synthetic data, which we discuss in more detail in the following section.

Results

We experimented with the available synthetic datasets in the following manner: we consecutively looped through each synthetic dataset, merging it with the original training data and then shuffling the resulting set. Then, with each of these combined datasets, we trained the four baseline models and reported their performance on the unchanged validation data.

Our expectation, based on the previous experiments, was that the performance for SVM and LR would initially improve with the addition of synthetic data, especially in terms of recall for the minority class. However, their performance might plateau or even degrade with the incorporation of more data. On the other hand, being more sophisticated and capable of understanding complex language patterns, BERT and LSTM were anticipated to benefit more substantially from the additional data, particularly in terms of generalization to unseen data. We hypothesized that as more synthetic sentences were added, there might be a point of diminishing returns. Beyond that threshold, further additions of synthetic data might not yield improvements.

The achieved F1 scores by our baseline models are shown in Figure 7.4. Interestingly, we observed three distinct trends that aligned with our expectations:

- **DL Models**: Both BERT and LSTM exhibited a consistent improvement in performance. Interestingly, BERT's performance continuously increased with adding more data. In contrast, LSTM began to plateau after the incorporation of 1,200 synthetic sentences. However, after 2,100 synthetic sentences, it started improving again, surpassing 60%, though with higher variance. Overall, both models demonstrated an increase of over 12% in F1 score compared to their original performance. The improvement pattern for both models resembled a logarithmic curve, with more substantial benefits observed initially and diminishing returns as more data was added.

- **SVM**: A different trend could be observed for SVM. The model's performance generally improved with the addition of up to 2,400 synthetic sentences. Beyond this point, a decline in performance was noted, resulting in an F1 score lower than the baseline.

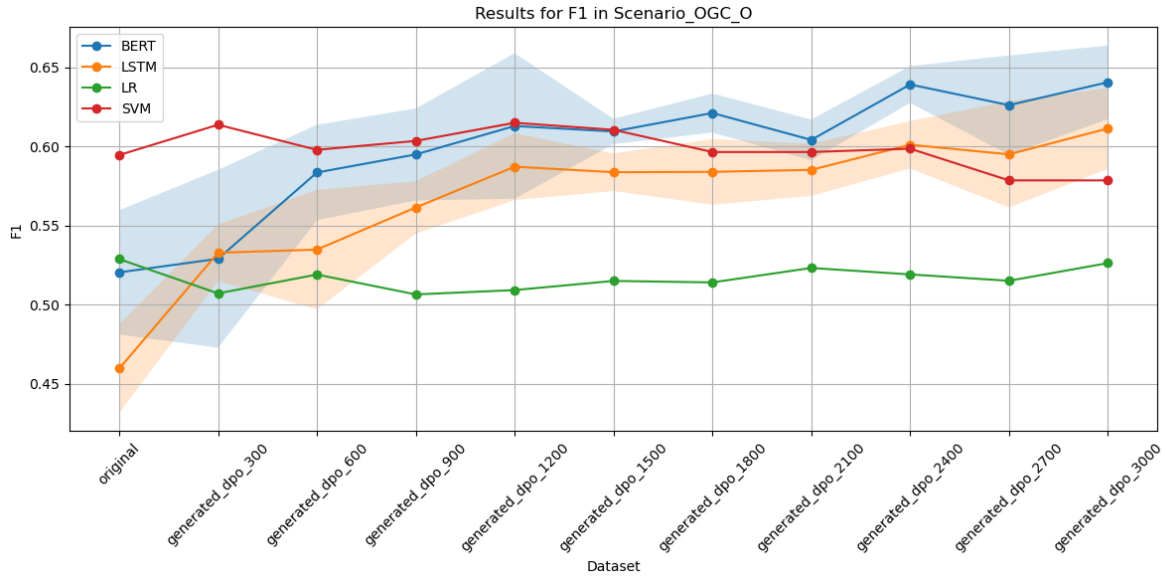


Figure 7.4: F1 scores achieved by the different models with various portions of synthetic data.

Furthermore, the increase in the performance exhibited fluctuations and certain amounts of synthetic data yielded better outcomes than others. Hence, this pattern implies that the quantity of synthetic data plays a significant role for the model's performance.

- **LR**: The F1 score of the LR classifier generally remained consistent across the different portions, however, a subtle trend of decline was observed, as the baseline F1 score was never surpassed. Nevertheless, its accuracy improved, rising from 96.45% to 97.23%.

It is important to note that the increase in the F1 score for BERT and LSTM was due to a consistent increase in precision across portions, as illustrated in Figure 7.5. Notably, after adding 1,200 synthetic sentences to the original training dataset, the precision of both models aligned closely for subsequent portions. This convergence suggested that beyond a certain threshold of synthetic data, both models adapted similarly in distinguishing GDPR-compliant sentences. However, Figure 7.6 shows that the recall for both models continuously decreased. This trend indicated that while the models became more robust at accurately recognizing compliant sentences with more synthetic data, they generally classified fewer sentences as such. As a result, they became more precise in their classifications, suggesting a trade-off between precision and recall with increasing amount of synthetic positive instances.

For SVM, on the other hand, the recall increased by up to 8% across the different portions of synthetic data, while the precision continuously declined, reaching 51.92% at 3,000 synthetic sentences (approximately 6% lower than the baseline). Hence, the SVM model started identifying more compliant sentences but became less precise in its classifications. Interestingly, the precision and recall trends for LR mirrored those of the deep learning models. The recall of LR dropped similarly to that of the LSTM model, and its precision increased, though not as significantly as LSTM's. Consequently, this affected the resulting F1 score of the LR model.

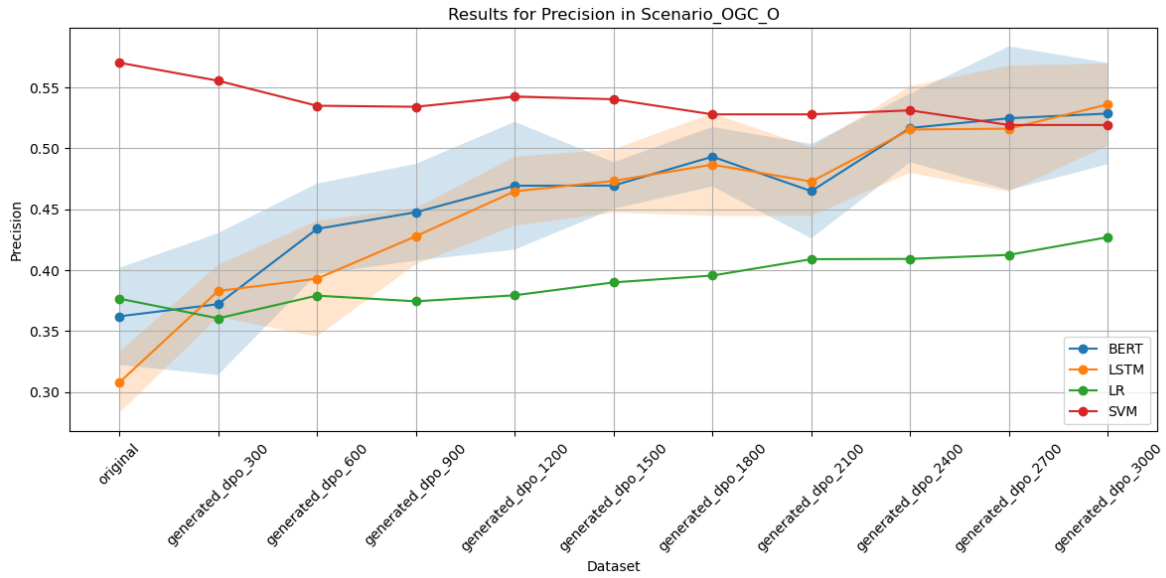


Figure 7.5: Precision achieved by the different models with various portions of synthetic data.

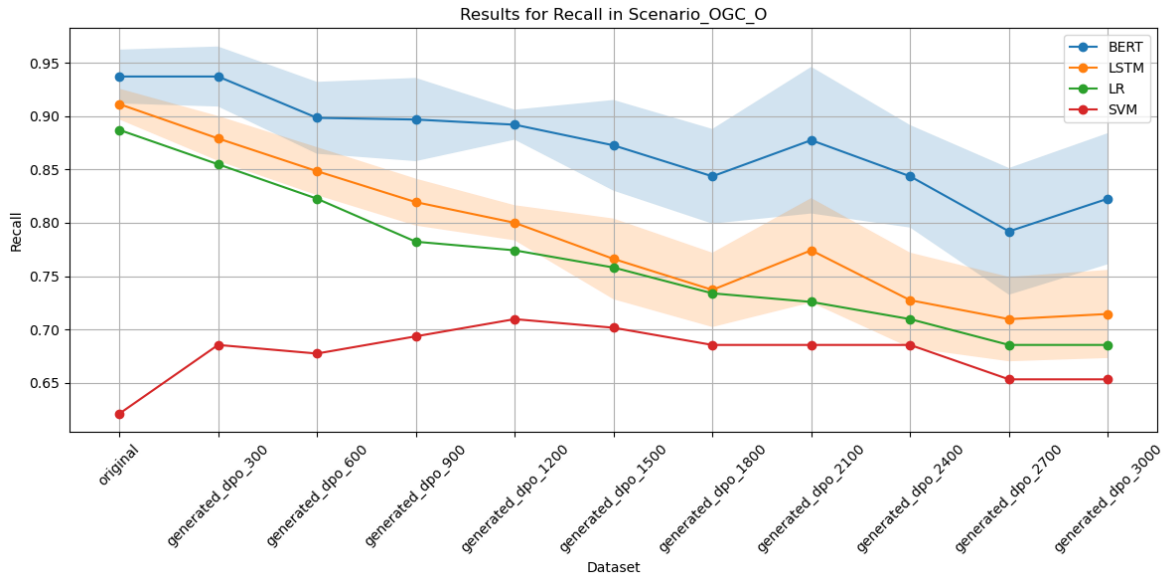


Figure 7.6: Recall achieved by the different models with various portions of synthetic data.

In general, the integration of LLM-produced synthetic data for tasks such as policy compliance checking appears to be advantageous. The quantity of added data plays a significant role. Moreover, it is crucial to align the model selection with the main goal of the task at hand. As observed, diverse models respond differently to the incorporation of synthetic data. While some increased their precision, recall, or overall accuracy, others exhibited a decrease in performance. Consistent with findings from our earlier experiments, DL models notably gained the most significant benefits from the inclusion of LLM-generated data. This suggests that the synthetic data is generally of high quality and effective. Bearing this in mind, our next step involves a qualitative analysis of the synthetic sentences to check this assumption.

Qualitative Analysis

The effectiveness of our approach heavily depends on the quality of the synthetic data. If the LLM-generated texts are realistic and closely resemble authentic policy-compliance scenarios, they can be highly beneficial. From the observations during the generation process we had, we could conclude that the generated sentences adhered well to the specified requirements, and a human annotator would likely classify them as *compliant* as well. Moreover, given that these sentences typically employ formal language, their style was consistent with that of actual privacy policies, allowing for their seamless integration into real-world scenarios.

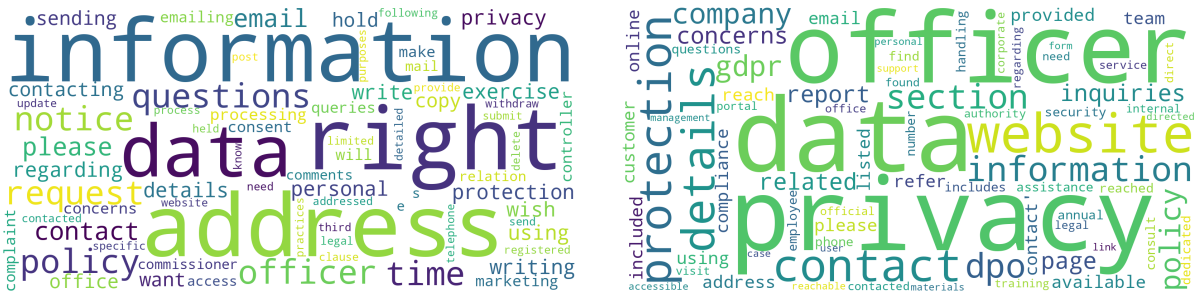


Figure 7.7: Word clouds for real-world and synthetic DPO-compliant sentences.

Figure 7.7 presents word clouds of the most frequent words in the compliant sentences of the original dataset and the largest synthetic dataset. A key observation is that there is an overlap in essential GDPR-related terms across the two sets. Words such as *data*, *information*, *privacy*, *rights* are present in both, indicating that the synthetic data successfully captures relevant terminology. Additionally, there are also noticeable differences in word frequencies, as well as presence of terms unique to the synthetic dataset. Nevertheless, these terms are all central to the subject matter of GDPR and privacy policies, which highlights that the synthetic data comprises relevant content that could potentially enrich the original dataset with additional context or variations on key themes. This is a positive sign for its quality, suggesting that it could provide valuable augmentation for data related to policy compliance, particularly in areas that may be underrepresented in real-world data.

7.4 Discussion

The application of LLMs for generating synthetic text data to train ML models emerges as a highly promising strategy in the field of cybersecurity. As we have seen in our experiments, LLMs are capable of producing a diverse range of data without explicit guidance. This diversity is critical for the development of robust ML models. Furthermore, the significant volume of data that LLMs can generate is particularly beneficial for advanced DL models, which often require extensive datasets to fully realize their potential. In our observations, DL models demonstrated a strong ability to learn from synthetic data, highlighting the synergy between the data needs of these models and the capabilities offered by LLMs.

Beyond the sheer volume and diversity of the data, the ability to customize LLM-generated text is another key advantage. The process can be finely tuned to meet specific requirements or simulate various scenarios within the cybersecurity domain, simply by specifying instructions in the prompt. This flexibility allows for training robust ML models and developing systems capable of identifying a wide range of existing, but also emerging cyber threats, thus enhancing their effectiveness in real-world applications.

Additionally, the synthetic data generation approach addresses significant limitations of data augmentation, as it does not rely on pre-existing datasets. This aspect is particularly advantageous in niche areas of cybersecurity, where real data may be scarce or challenging to obtain. Moreover, this method eliminates the risks associated with handling sensitive or personal information present in real-world datasets, which is often the case for cybersecurity data. It also mitigates the potential for data leakage and generally minimizes the privacy concerns, while also being cost-effective, especially in scenarios where collecting and labeling real data is prohibitively expensive.

Nevertheless, there are important considerations to take into account when utilizing LLMs for data generation. First, it is essential to consider the underlying model being used. Our experiments have shown that different models benefit differently from synthetic data. This also guides the choice of strategy, such as addressing class imbalance or overfitting. Further, the quantity of synthetic data generated is a pivotal factor. While some models perform better with smaller datasets, others require larger volumes. An inappropriate amount of synthetic data can, in fact, harm model performance.

Another significant aspect is the nature of the real-world data available. For instance, in our spam detection experiments, we used the UCI SMS spam dataset, which is a collection of real-world SMS messages from over a decade ago. Over time, there have been considerable shifts in language usage and communication patterns, reflected in both spam and legitimate messages. Technological advancements have enabled faster typing, reducing the need for excessive abbreviation and resulting in more proper language usage in modern messaging. While the LLM-generated synthetic data mirrored these more contemporary messaging styles, it showed notable differences from the style observed in the UCI dataset, which was characterized by extensive use of abbreviations, also due to the character limitations of SMS at that time. Moreover, the vocabulary has evolved. For example, the spam message *"Yo, be the king of TikTok. Viral secrets inside."* or the ham message *"Lol, that TikTok you sent. Made my day!"*, both synthetic, are about a video platform widely used nowadays, but released only in 2018.

Adding them to the training data introduces a new word in the vocabulary: *tiktok*. However, this can be irrelevant or even misleading when testing the model against messages from the older UCI dataset, as such terms were not part of the communication landscape a decade ago. While such up-to-date synthetic data would be beneficial in contemporary real-world scenarios, in cases like ours, it necessitates careful tailoring to match the characteristics of the existing data in order to achieve optimal results. Otherwise, a discrepancy between the training data and the context it aims to represent could occur.

In contrast, in privacy compliance checking, which typically involves more formal and standardized language, the issue of stylistic shift was less pronounced. However, this scenario highlighted another critical aspect: the importance of carefully specifying the classification guidelines for the LLM to generate relevant data. This requirement was particularly evident in this use case due to the specificity of the task. Given our intention to use the approach in cybersecurity contexts, where tasks are often very specific, this indicates the need for precise prompt creation to leverage the full potential of LLMs in generating useful synthetic data.

Another crucial aspect for the quality of the generated data is the employed LLM itself. As ML models, their general capabilities are strongly dependent on their training data, since they learn and replicate patterns from it in their responses. The source and nature of this data are thus critical. However, LLMs are often used off-the-shelf and pre-trained on extensive datasets, making it difficult to control their exposure to specific content. On the one hand, low-quality training data could be problematic, as we will see in the following. Nevertheless, a significant limitation arises when the LLM has not been exposed to relevant content, especially in fields like cybersecurity. It may prove challenging to create specific synthetic data, e.g., highly specialized technical content, such as log data. In such scenarios, the reliance shifts heavily onto the prompt provided to the LLM. However, this approach has its constraints, as the length of the prompt is limited, restricting the amount of guidelines or examples that can be included to direct the LLM. Additionally, current LLMs are characterized by knowledge cutoffs, implying that their training encompasses data only up to a certain point in time. They lack factual information on anything that occurred afterwards. Therefore, selecting an LLM that has been trained with the most recent data becomes critical when the goal is to generate content that accurately reflects current scenarios for effective training.

The ethical implications of using LLMs for data generation are also significant. Given the amount of data the LLM has seen during training, there is a risk of unintentionally responding with real data scraped from the internet, believing it to be synthetic or anonymized. There have been instances where data thought to be *artificial*, was actually discovered to be real. More concerning is that malicious actors could also manage to obtain such data through specialized methods [5]. This raises concerns about data integrity and privacy. For instance, generating DPO-compliant sentences with LLMs might include contact details of "non-existing individuals", though there is a risk that this data is not purely artificial. Additionally, the training data of LLMs, often sourced from public domains and not explicitly checked due to its volume, might contain biases. These biases could be inherited and replicated in the synthetic data generated by LLMs. Especially if the data is then directly used for training ML models, it can lead to biased systems, a significant concern in cybersecurity.

Furthermore, the way LLMs are utilized makes them potential targets for malicious actors. In this approach, we interact with the LLM by sending a prompt via an API request and get a response back. However, the black-box nature of proprietary LLMs provides no insight into their response generation processes, which creates potential vulnerabilities. For instance, if the API used to communicate with an LLM is compromised, it might be challenging to detect, as long as responses remain in the expected format; users typically just expect a proper answer and might not realize they are receiving data from an unauthorized source. Alternatively, attackers could intercept communication and manipulate the prompts sent to legitimate LLMs, resulting in synthetic data that does not meet the intended requirements. Due to the volume of such data, as this is often the aim, or the nature, e.g., technical or log data, human oversight becomes challenging. It could be very problematic if this data is then directly used for training ML models, as we are propagating compromised data and facilitate the success of further attacks. For example, if manipulated synthetic data is used to train a malware detection system, it could significantly undermine the system's effectiveness, leaving intentionally undetected threats. These scenarios highlight the necessity of integrating robust security measures at all stages of data generation, especially when applied in cybersecurity.

While advancements in LLMs bring the potential for generating relevant cybersecurity content and simulating various cyber threats, they simultaneously pose risks by potentially assisting attackers. For example, LLMs are able to generate realistic spam content to aid spam detection efforts. However, they also enable spammers to craft sophisticated, hard-to-detect spam. Even though LLMs would not intentionally generate malicious content, techniques like prompt manipulation can be used to circumvent security measures. Particularly in the spam use case, we observed that this can be done by specifying research purposes in the prompt. In general, LLMs can be used to produce high-quality, personalized spam very efficiently. This is also a reason for shifting the nature of spam from grammatically flawed to more sophisticated and natural-sounding. This observation demonstrates the dual role of LLMs as both a tool and a challenge in cybersecurity.

In conclusion, while LLMs offer significant advantages in generating synthetic data for ML model training, particularly in cybersecurity, it is crucial to use them with awareness of the associated challenges. These include maintaining data relevance, safeguarding against misuse, and ensuring ethical and secure applications. As we move forward, the continued evolution of LLMs promises further advancements. The increasing amount of LLM-generated data in real-world applications suggests a future where such data could entirely replace traditional training datasets. This could be particularly beneficial in cybersecurity for simulating potential threats. However, this also introduces numerous risks that must be addressed in order to harness the full potential of LLMs in shaping the future of cybersecurity.

8 Conclusion

LLMs have revolutionized the field of NLP, demonstrating advanced capabilities in text interpretation and generation. Their rise has had a significant impact on various domains. Within the realm of cybersecurity, their implications are twofold, presenting both challenges and opportunities. In this thesis, we systematically explored the transformative potential of LLMs to enhance cybersecurity solutions, focusing on three key applications: text classification, data augmentation, and synthetic data generation. We leveraged the extensive pre-trained knowledge of state-of-the-art LLMs without additional task-specific fine-tuning.

Our study centered on two cybersecurity problems, representative for applying NLP techniques in the field: spam detection and policy compliance checking. They provided a comprehensive perspective on natural language and allowed us to assess the capabilities of LLMs thoroughly. We began by identifying relevant labeled datasets for each use case and reconstructing state-of-the-art supervised ML techniques, ranging from conventional ML models to advanced DL techniques. To establish a performance benchmark and uniformly assess the effectiveness of LLMs in each application throughout the thesis, we adopted a common evaluation framework. In addition, we performed empirical analyses to understand the results achieved and identify the strengths and limitations of LLMs.

Initially, we employed LLMs for text classification, a task of significant importance in cybersecurity. The capabilities of decoder-only (Llama 2) and encoder-decoder (FLAN-T5) architectures were investigated using different prompting strategies. Moreover, we explored their ability to provide explanations for their classifications. Our findings revealed that current LLMs, without any training data, outperformed various conventional ML models, showcasing their potential in scenarios with limited data available and resilience against common issues like class imbalance. Nevertheless, we observed that the architecture plays a significant role in how well LLMs follow instructions and perform on tasks. While FLAN-T5 showed superior performance quantitatively, Llama 2 excelled in providing explanations for its predictions, aligning with the idea of explainable AI and enhancing transparency. However, we noted limitations like potential classification bias and high prompt sensitivity, particularly in terms of examples provided, suggesting directions for future work.

A general drawback of the text classification approach was the constant reliance on externally hosted LLMs for classification. Driven from the fact that other advanced and efficient ML models often cannot be applied for cybersecurity tasks due to lack of relevant data, we delved into our second application. We employed Llama 2 to augment the initial training datasets with different strategies, addressing various problems such as class imbalance. The augmented data's impact was assessed by training our baseline models and evaluating them against the original validation set. LLMs demonstrated advanced capabilities in augmenting text, maintaining the core meaning while introducing new vocabulary and structural varia-

tions. The enhancement in model performance, however, varied depending on the specific ML model used, as well as the amount and strategy of augmentation. Inappropriate augmentation proved to even harm model performance. Our observations showed that DL models benefit significantly from LLM-augmented data, which is particularly relevant considering their higher data demands. Nevertheless, challenges such as potential stylistic shifts, especially in casual language due to LLMs' tendency to formalize content, were noted. This indicates the need for altering the augmentation technique by providing specific instructions in the prompt or future enhancements in LLMs to better recognize and preserve the original stylistic nuances, in order to maximize the effectiveness of augmentation. Despite these limitations, LLMs demonstrated their potential as powerful tools for text augmentation.

To address the dependence on available real-world data in the data augmentation approach, we turned to synthetic data generation using Llama 2. This way, we also reduce the risks associated with sharing real-world training data with externally hosted LLMs. This method allowed for tailoring data to specific cybersecurity needs, simulating cyber threats, and producing diverse data in desired quantities. In our experiments, we created various synthetic datasets using different strategies to train our baseline models, revealing the impact of LLM-generated data on model performance. Key factors appeared to be the underlying model, the volume of synthetic data, and the nature of the task at hand. DL models particularly benefited from the diversity and volume provided by LLM-generated data, whereas the effectiveness for conventional ML models depended highly on the amount used. The synthetic data was generally of high quality and reflective of diverse real-world scenarios. However, we observed that depending on the specific task it might be necessary to modify the generation technique to take into account the characteristics of available data, which emerged as a key consideration for future research. Besides the need for maintaining data relevance, this approach also presents challenges like potential bias propagation and malicious use. Thus, ensuring ethical and secure application of LLM-generated synthetic data is imperative and highlights the need for robust security measures throughout the whole data generation process. Addressing these issues is crucial to fully harness the capabilities of LLMs in generating relevant synthetic data for cybersecurity applications.

Looking ahead, it is essential to expand our scope of research to include a broader range of cybersecurity challenges, such as anomaly detection and fake news detection. Furthermore, leveraging more advanced LLMs, like the high-performing GPT-4, could yield deeper insights into the full spectrum of their capabilities. Additionally, our systematic approach for evaluating the effectiveness of LLMs in cybersecurity scenarios could also be further enhanced by fine-tuning baseline models with LLM-augmented and synthetic data. Adjusting the initial parameter settings to the specific data used could lead to improved outcomes.

In conclusion, LLMs hold significant promise in enhancing cybersecurity solutions, particularly when coupled with expert oversight. They have the potential to transform cybersecurity practices, adapting to the continuously evolving cyber threats. The rapid advancements in AI, while presenting new cybersecurity challenges, also open up vast opportunities. Therefore, consistently and proactively harnessing them is essential for sustaining effective cybersecurity measures in our dynamic digital world.

Abbreviations

AI Artificial Intelligence

BERT Bidirectional Encoder Representations from Transformers

CNN Convolutional Neural Network

CoT Chain-of-Thought

DDoS Distributed Denial of Service

DL Deep Learning

DPA Data Processing Agreement

DPO Data Protection Officer

ELMo Embeddings from Language Model

EU European Union

FLAN-T5 Fine-tuned LAnguage Net T5

GDPR General Data Protection Regulation

GPT Generative Pretrained Transformer

GRU Gated Recurrent Unit

ICO Information Commissioner's Office

KNN K-Nearest Neighbors

LightGBM Light Gradient-Boosting Machine

LLM Large Language Model

LR Logistic Regression

LSTM Long Short-Term Memory

MITM Man-in-the-Middle

ML Machine Learning

NB Naïve Bayes

NLP Natural Language Processing

NLU Natural Language Understanding

NN Neural Network

RLHF Reinforcement Learning with Human Feedback

RNN Recurrent Neural Network

RoBERTa Robustly Optimized BERT Pretraining Approach

RTF Role-Task-Format

SF Semantic Frame

SVM Support Vector Machine

T5 Text-to-Text Transfer Transformer

TF-IDF Term Frequency-Inverse Document Frequency

XGBoost eXtreme Gradient Boosting

XSS Cross-Site Scripting

List of Figures

5.1 Initial prompt template for spam detection. 25
5.2 Inserted example for the positive class. 26
5.3 Inserted example for the negative class. 26
5.4 Prompt template for spam detection using the CoT technique. 28
5.5 Explanation generated by the LLM for the classification of an SMS message. . 28
5.6 Initial prompt template for GDPR compliance checking, category DPO. 30
5.7 Inserted example for the positive class. 31
5.8 Prompt template for DPO compliance checking using the CoT technique. . . . 32
5.9 Explanation generated by the LLM for the classification of a policy sentence. . 33

6.1 Prompt template used for augmenting training data for spam detection. 38
6.2 Class distribution of spam and ham within the different datasets. 40
6.3 F1 scores achieved by the baseline models with different portions of augmentation. 41

7.1 F1 scores achieved by the spam detection models when trained with different portions of LLM-generated synthetic data added to the original training data. 53
7.2 F1 scores achieved by the spam detection models when trained with different portions of LLM-generated synthetic data only. 54
7.3 Word clouds for synthetic ham and spam SMS messages. 56
7.4 F1 scores achieved by the different models with various portions of synthetic data. 59
7.5 Precision achieved by the different models with various portions of synthetic data. 60
7.6 Recall achieved by the different models with various portions of synthetic data. 60
7.7 Word clouds for real-world and synthetic DPO-compliant sentences. 61

List of Tables

4.1 The five GDPR requirements chosen to evaluate privacy policy compliance. . . 13
4.2 Data point counts in the GDPR dataset. 13
4.3 Baseline performance for the spam detection use case. 19
4.4 Baseline performance for the DPO compliance checking. 20

5.1 Performance achieved by FLAN-T5 and Llama 2 for SMS spam detection. . . . 27
5.2 Number of false positives and false negatives in DPO compliance checking. . . 31
5.3 Performance achieved by FLAN-T5 and Llama 2 for DPO compliance checking. 32

6.1 Results achieved by training the baseline models with rephrased messages only. 42
6.2 Examples of original and rephrased SMS messages by an LLM. 43
6.3 Average results with augmentation of the positive class for DPO compliance checking. 45

7.1 Statistical analysis of real-world and synthetic SMS messages. 57

Bibliography

[1] A.-J. Aberkane, G. Poels, and S. V. Broucke. "Exploring automated GDPR-compliance in requirements engineering: A systematic mapping study." In: *IEEE Access* 9 (2021), pp. 66542–66559.

[2] E. Aghaei, X. Niu, W. Shadid, and E. Al-Shaer. "SecureBERT: A Domain-Specific Language Model for Cybersecurity." In: *International Conference on Security and Privacy in Communication Systems*. Springer. 2022, pp. 39–56.

[3] T. Almeida and J. Hidalgo. *SMS Spam Collection*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5CC84. 2012.

[4] O. Amaral, M. I. Azeem, S. Abualhaija, and L. C. Briand. "NLP-based automated compliance checking of data processing agreements against GDPR." In: *IEEE Transactions on Software Engineering* (2023).

[5] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, A. Oprea, and C. Raffel. *Extracting Training Data from Large Language Models*. 2021. arXiv: 2012.07805 [cs.CR].

[6] B. Chen, Z. Zhang, N. Langrené, and S. Zhu. "Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review." In: *arXiv preprint arXiv:2310.14735* (2023).

[7] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, et al. "Scaling instruction-finetuned language models." In: *arXiv preprint arXiv:2210.11416* (2022).

[8] J. J. Y. Chung, E. Kamar, and S. Amershi. "Increasing Diversity While Maintaining Accuracy: Text Data Generation with Large Language Models and Human Interventions." In: *arXiv preprint arXiv:2306.04140* (2023).

[9] E. G. Dada, J. S. Bassi, H. Chiroma, A. O. Adetunmbi, O. E. Ajibuwa, et al. "Machine learning for email spam filtering: review, approaches and open research problems." In: *Heliyon* 5.6 (2019).

[10] K. Debnath and N. Kar. "Email Spam Detection using Deep Learning Approach." In: *2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON)*. Vol. 1. 2022, pp. 37–41. DOI: 10.1109/COM-IT-CON54601.2022.9850588.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[12] A. Edwards, A. Ushio, J. Camacho-Collados, H. de Ribaupierre, and A. Preece. "Guiding generative language models for data augmentation in few-shot text classification." In: *arXiv preprint arXiv:2111.09064* (2021).

[13] European Parliament and Council. *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance)*. Accessed: November 18, 2023. 2016.

[14] M. A. Ferrag, A. Battah, N. Tihanyi, M. Debbah, T. Lestable, and L. C. Cordeiro. "SecureFalcon: The Next Cyber Reasoning System for Cyber Security." In: *arXiv preprint arXiv:2307.06616* (2023).

[15] M. A. Ferrag, M. Ndhlovu, N. Tihanyi, L. C. Cordeiro, M. Debbah, and T. Lestable. "Revolutionizing Cyber Threat Detection with Large Language Models." In: *arXiv preprint arXiv:2306.14263* (2023).

[16] E. Ferrara. "The history of digital spam." In: *Communications of the ACM* 62.8 (2019), pp. 82–91.

[17] P. Ferrara, N. F. Spoto, et al. "Static analysis for GDPR compliance." In: *CEUR Workshop Proceedings*. Vol. 2058. CEUR Workshop Proceedings. 2018, pp. 1–10.

[18] R. A. Frick. "Fraunhofer SIT at CheckThat! 2023: Can LLMs be Used for Data Augmentation & Few-Shot Classification? Detecting Subjectivity in Text Using ChatGPT." In: *Working Notes of CLEF* (2023).

[19] J. Gao, R. Pi, L. Yong, H. Xu, J. Ye, Z. Wu, W. ZHANG, X. Liang, Z. Li, and L. Kong. "Self-Guided Noise-Free Data Generation for Efficient Zero-Shot Learning." In: *The Eleventh International Conference on Learning Representations*. 2022.

[20] A. Ghourabi and M. Alohaly. "Enhancing Spam Message Classification and Detection Using Transformer-Based Embedding and Ensemble Learning." In: *Sensors* 23.8 (2023), p. 3861.

[21] A. Goldsteen, G. Ezov, R. Shmelkin, M. Moffie, and A. Farkash. "Data minimization for GDPR compliance in machine learning models." In: *AI and Ethics* (2021), pp. 1–15.

[22] Google. *Bard: A Conversational AI Service by Google*. `https://bard.google.com`. Accessed: November 15, 2023. 2023.

[23] M. Guastalla, Y. Li, A. Hekmati, and B. Krishnamachari. *Application of Large Language Models to DDoS Attack Detection*.

[24] T. S. Guzella and W. M. Caminhas. "A review of machine learning approaches to spam filtering." In: *Expert Systems with Applications* 36.7 (2009), pp. 10206–10222.

[25] R. E. Hamdani, M. Mustapha, D. R. Amariles, A. Troussel, S. Meeùs, and K. Krasnashchok. "A combined rule-based and machine learning approach for automated GDPR compliance checking." In: *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*. 2021, pp. 40–49.

[26] T. Hartvigsen, S. Gabriel, H. Palangi, M. Sap, D. Ray, and E. Kamar. "ToxiGen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection." In: *arXiv preprint arXiv:2203.09509* (2022).

[27] Hugging Face. *Hugging Face Hub Documentation*. Accessed: November 16, 2023. 2023.

[28] *HuggingChat*. `https://huggingface.co/chat/`. Accessed: November 16, 2023. Hugging Face.

[29] A. Istwal. *Mastering ChatGPT Prompting: The RTF Framework*. `https://medium.com/@aciddun/mastering-chatgpt-prompting-the-rtf-framework-2db7465c6500`. Accessed: November 15, 2023. 2023.

[30] S. Jamal and H. Wimmer. "An Improved Transformer-based Model for Detecting Phishing, Spam, and Ham: A Large Language Model Approach." In: *arXiv preprint arXiv:2311.04913* (2023).

[31] F. Jáñez-Martino, R. Alaiz-Rodríguez, V. González-Castro, E. Fidalgo, and E. Alegre. "A review of spam email detection: analysis of spammer strategies and the dataset shift problem." In: *Artificial Intelligence Review* 56.2 (2023), pp. 1145–1173.

[32] T. Koide, N. Fukushi, H. Nakano, and D. Chiba. "Detecting Phishing Sites Using ChatGPT." In: *arXiv preprint arXiv:2306.05816* (2023).

[33] V. Kumar, A. Choudhary, and E. Cho. "Data augmentation using pre-trained transformer models." In: *arXiv preprint arXiv:2003.02245* (2020).

[34] M. Labonne and S. Moran. "Spam-T5: Benchmarking Large Language Models for Few-Shot Email Spam Detection." In: *arXiv preprint arXiv:2304.01238* (2023).

[35] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. 2019. arXiv: `1910.13461 [cs.CL]`.

[36] Z. Li, H. Zhu, Z. Lu, and M. Yin. "Synthetic Data Generation with Large Language Models for Text Classification: Potential and Limitations." In: *arXiv preprint arXiv:2310.07849* (2023).

[37] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. "RoBERTa: A Robustly Optimized BERT Pretraining Approach." In: *arXiv preprint arXiv:1907.11692* (2019).

[38] J. R. Méndez, F. Fdez-Riverola, F. Díaz, E. L. Iglesias, and J. M. Corchado. "A comparative performance study of feature selection methods for the anti-spam filtering domain." In: *Advances in Data Mining. Applications in Medicine, Web Mining, Marketing, Image and Signal Mining: 6th Industrial Conference on Data Mining, ICDM 2006, Leipzig, Germany, July 14-15, 2006. Proceedings 6*. Springer. 2006, pp. 106–120.

[39] Y. Meng, J. Huang, Y. Zhang, and J. Han. "Generating training data with language models: Towards zero-shot language understanding." In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 462–477.

[40] N. M. Müller, D. Kowatsch, P. Debus, D. Mirdita, and K. Böttinger. "On GDPR compliance of companies' privacy policies." In: *Text, Speech, and Dialogue: 22nd International Conference, TSD 2019, Ljubljana, Slovenia, September 11–13, 2019, Proceedings 22*. Springer. 2019, pp. 151–159.

[41] B. Nicula, M. Dascalu, T. Arner, R. Balyan, and D. S. McNamara. "Automated Assessment of Comprehension Strategies from Self-Explanations Using LLMs." In: *Information* 14.10 (2023), p. 567.

[42] OpenAI. *ChatGPT: Optimizing Language Models for Dialogue*. `https://openai.com/blog/chatgpt/`. Accessed: November 15, 2023. 2022.

[43] OpenAI. *GPT-4 Technical Report*. 2023. arXiv: `2303.08774 [cs.CL]`.

[44] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. "PyTorch: An Imperative Style, High-Performance Deep Learning Library." In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[45] A. Peysakhovich and A. Lerer. "Attention Sorting Combats Recency Bias In Long Context Language Models." In: *arXiv preprint arXiv:2310.01427* (2023).

[46] M. F. Porter. "An algorithm for suffix stripping." In: *Program* 14.3 (1980), pp. 130–137.

[47] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. "Exploring the limits of transfer learning with a unified text-to-text transformer." In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 5485–5551.

[48] G. Sahu, P. Rodriguez, I. H. Laradji, P. Atighehchian, D. Vazquez, and D. Bahdanau. "Data augmentation for intent classification with off-the-shelf large language models." In: *arXiv preprint arXiv:2204.01959* (2022).

[49] D. Sánchez, A. Viejo, and M. Batet. "Automatic assessment of privacy policies under the GDPR." In: *Applied Sciences* 11.4 (2021), p. 1762.

[50] G. Shahariar, S. Biswas, F. Omar, F. M. Shah, and S. B. Hassan. "Spam review detection using deep learning." In: *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE. 2019, pp. 0027–0033.

[51] M. Sultana, A. Taylor, L. Li, and S. Majumdar. "Towards Evaluation and Understanding of Large Language Models for Cyber Operation Automation." In: *2023 IEEE Conference on Communications and Network Security (CNS)*. 2023, pp. 1–6. DOI: `10.1109/CNS59707.2023.10288677`.

[52] A. Tamkin, M. Brundage, J. Clark, and D. Ganguli. "Understanding the capabilities, limitations, and societal impact of large language models." In: *arXiv preprint arXiv:2102.02503* (2021).

[53] R. Tang, X. Han, X. Jiang, and X. Hu. "Does synthetic data generation of LLMs help clinical text mining?" In: *arXiv preprint arXiv:2303.04360* (2023).

[54] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971 [cs.CL].

[55] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. "Llama 2: Open foundation and fine-tuned chat models." In: *arXiv preprint arXiv:2307.09288* (2023).

[56] A. Tsymbal. "The problem of concept drift: definitions and related work." In: *Computer Science Department, Trinity College Dublin* 106.2 (2004), p. 58.

[57] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. "Attention is all you need." In: *Advances in neural information processing systems* 30 (2017).

[58] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. "SuperGLUE: A stickier benchmark for general-purpose language understanding systems." In: *Advances in neural information processing systems* 32 (2019).

[59] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. "GLUE: A multi-task benchmark and analysis platform for natural language understanding." In: *arXiv preprint arXiv:1804.07461* (2018).

[60] Z. Wang, A. W. Yu, O. Firat, and Y. Cao. "Towards zero-label language learning." In: *arXiv preprint arXiv:2109.09193* (2021).

[61] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, et al. "Emergent abilities of large language models." In: *arXiv preprint arXiv:2206.07682* (2022).

[62] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. 2023. arXiv: 2201.11903 [cs.CL].

[63] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt. *A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT*. 2023. arXiv: 2302.11382 [cs.SE].

[64] Q. Yaseen et al. "Spam email detection using deep learning techniques." In: *Procedia Computer Science* 184 (2021), pp. 853–858.

[65] J. Ye, J. Gao, Q. Li, H. Xu, J. Feng, Z. Wu, T. Yu, and L. Kong. "Zerogen: Efficient zero-shot learning via dataset generation." In: *arXiv preprint arXiv:2202.07922* (2022).

[66] K. M. Yoo, D. Park, J. Kang, S.-W. Lee, and W. Park. "GPT3Mix: Leveraging large-scale language models for text augmentation." In: *arXiv preprint arXiv:2104.08826* (2021).

[67] Y. Yu, Y. Zhuang, J. Zhang, Y. Meng, A. Ratner, R. Krishna, J. Shen, and C. Zhang. "Large language model as attributed training data generator: A tale of diversity and bias." In: *arXiv preprint arXiv:2306.15895* (2023).