DTU Compute

Department of Applied Mathematics and Computer Science

Structuring Free Text for Process Mining Purposes via LLMs

Master Thesis

Structuring Free Text for Process Mining Purposes via LLMs

Master Thesis
March, 2024

By
Qiannan Liu

Cover photo: Vibeke Hempler, 2012
Published by: DTU, Department of Applied Mathematics and Computer Science, Richard Petersens Plads, Building 324, 2800 Kgs. Lyngby Denmark
https://www.compute.dtu.dk
ISSN: [0000-0000] (electronic version)
ISBN: [000-00-0000-000-0] (electronic version)

ISSN: [0000-0000] (printed version)
ISBN: [000-00-0000-000-0] (printed version)

Approval

This thesis, titled ”Structuring Free Text for Process Mining Purposes via LLMs,” has been prepared at the Department of Applied Mathematics and Computer Science at the Technical University of Denmark, DTU, over six months. It represents a partial fulfillment of the requirements for the degree of Master of Science in Computer Science and Engineering, MSc Eng.

It is assumed that the reader has a fundamental understanding of natural language processing and process mining.

Qiannan Liu - s212427

.....

Signature

.....

Date

Abstract

This study investigates the application of Large Language Models (LLMs) for transforming unstructured text from insurance claims into structured data for process mining, aiming to automate process categorization and enhance decision-making in the insurance sector. Employing a mixed-methods approach, the research assesses the effectiveness of LLMs in classifying text according to the American Productivity & Quality Center (APQC) framework through prompt engineering and instruction fine-tuning techniques.

The implementation framework details the selection of LLMs for their efficiency and adaptability to instruction-based tasks, the configuration of computational resources, and the processing of data for model training and evaluation. Results demonstrate the potential of optimized LLMs in accurately processing and categorizing unstructured text, offering significant improvements in process mining applications.

Despite limitations like small dataset sizes and computational resource constraints, the study provides insights into the integration of natural language processing in business process management. Future research directions include expanding dataset sizes, exploring other unstructured data types, and applying advanced training methods to enhance LLM performance. This research contributes to the field by showcasing the utility of LLMs in process analysis and setting the stage for further advancements in the integration of LLMs in business process optimization.

Acknowledgements

I wish to express my gratitude to my supervisor, **Andrea Burattin**, for his meticulous guidance and substantial support. My thanks to the cooperating companies **Breakawai** and **Topdanmark** are also indispensable; their provision of data and resources was crucial for my research. I am deeply grateful to my family, especially my husband, **Dong**, for his hard work, trust, and care for our children, allowing me to dedicate myself fully to my studies. Lastly, I thank my children, **Noah and Yan**, who provide me with the strength and motivation to face difficulties, making me stronger and enabling me to persevere.

Contents

1 Introduction

This chapter aims to provide a comprehensive introduction to this study. First, the background section discusses the background within which this study is situated. After that, the problem statement section clearly defines the specific issues that this research aims to address. The objectives section details the specific goals that the study intends to achieve. The research significance section highlights the importance of this work within the field. Finally, the report structure section provides an overview of the overall architecture of this report.

1.1 Background

From the emergence of process mining technology in the late 1990s to its current landscape, there has been a story of constant evolution and adaptation. It acts like an experienced detective, capable of dissecting the mysteries within complex business workflows. By examining event logs, it identifies problems and bottlenecks within business operations, thereby offering powerful suggestions for process optimization. Due to the exceptional capabilities of process mining, it rapidly gained recognition, with many enterprises and organizations now employing process mining tools to improve efficiency, reduce costs, and enhance customer satisfaction.

However, process mining techniques predominantly rely on structured event logs, yet in the current vast universe of data, businesses and organizations possess an abundance of unstructured textual data such as customer claim requests, medical diagnostic records, and work order processing notes. These text data are rich with process information but are difficult to directly utilize in process mining, akin to a multitude of case clues scattered everywhere, waiting to be pieced together into a complete picture.

The study of Large Language Models (LLMs) can be traced back to the early stages of Natural Language Processing (NLP), but it was the development of deep learning techniques and significant improvements in computing power that marked true breakthroughs in the past decade. In 2018, Google's research team unveiled the BERT (Bidirectional Encoder Representations from Transformers) model, which became a milestone in LLM research. The model demonstrated the profound capabilities of deep bidirectional networks for language comprehension and achieved

unprecedented accuracy across various language understanding tasks.

Subsequently, the development of Large Language Models (LLMs) has progressed rapidly. The release of the Generative Pretrained Transformer (GPT) series by OpenAI has demonstrated remarkable achievements in tasks such as text generation, semantic understanding, and fine-grained text classification. The emergence and successful application of these models have attracted more attention from researchers and organizations. LLMs act like linguists, capable of comprehending the semantics and patterns of text, and possessing the ability to transform obscure clues into clear evidence.

Despite the immense potential of LLMs in processing unstructured textual data, the challenge of applying these models in process mining, especially in transforming unstructured data into structured information suitable for process mining analysis, remains unresolved. This study aims to facilitate the collaboration of the two domain experts, exploring the application potential of LLMs in process mining. It seeks to provide innovative approaches for process optimization across various industries, assisting enterprises and organizations in enhancing operational efficiency and creating greater value.

1.2 Problem Statement

With the explosive growth of data, businesses, and organizations are facing the challenge of effectively leveraging this wealth of information, particularly in the domain of process mining. Process mining technology relies on structured event logs to analyze and optimize business processes. However, in practical operations, a significant amount of business process information is embedded in unstructured text, such as emails, customer feedback, and service records. These textual data are crucial for understanding process efficiency and customer experience, yet existing process mining technologies have not been able to efficiently transform these unstructured data into useful information, limiting their potential and application scope in business process management.

Large Language Models (LLMs) possess the powerful capability to process and understand natural language, offering new possibilities for addressing the aforementioned issues. However, precisely leveraging LLMs to transform unstructured text into structured information directly usable for process mining remains an unresolved research challenge. LLMs need to maintain high accuracy while capturing essential details crucial for process mining when processing free

text. Moreover, the compatibility between the outputs of LLMs and process mining tools needs to be addressed to ensure seamless integration into existing process mining frameworks.

Therefore, this study aims to explore the following key questions:

1. How can Large Language Models (LLMs) be optimized and adjusted to effectively process unstructured text and convert it into structured log files?
2. What are the potential and limitations of LLMs in enhancing the efficiency and accuracy of process mining?
3. How can the structured data output by LLMs be integrated with process mining tools for efficient process analysis, and what is the integration scenario?
4. In practical applications, how do LLMs perform in understanding and classifying complex process texts, and what challenges and limitations exist?

The purpose of this research is to fill the existing technological gap by proposing a new methodology that combines the natural language processing capabilities of LLMs with process mining technology, bringing multi-layered insights into business process management. This not only aids enterprises and organizations in achieving data-driven decisions but also paves new paths for the application of natural language processing technology in business process analysis.

1.3 Objectives

Based on the SMART (Specific, Measurable, Achievable, Relevant, Time-bound) principles, this study sets the following specific objectives:

1. **Development and Validation of LLM Optimization Strategies:** Within six months, research and implement LLM optimization strategies tailored for process mining needs to enhance the capability of processing unstructured texts. The goal is to explore different pre-training models and tuning techniques through experimentation, aiming for at least a 10% improvement in accuracy, ensuring the model can effectively identify and categorize crucial process information.
2. **Quantitative Assessment of LLMs in Process Mining:** Through experimental validation, quantitatively assess the performance of LLMs across various types and complexities of business process text classification tasks, including classification accuracy, recall rate, and

F1 score. The objective is to complete a preliminary assessment during the project's midpoint and to finalize an in-depth analysis within six months, determining LLMs' potential and limitations in actual process mining application scenarios.

3. **Efficient Integration of LLM Outputs with Process Mining Tools:** Within the first four months of the project, develop a methodology and tools for seamlessly integrating the structured output data generated by LLMs with process mining tools. Through custom data conversion scripts, ensure the accuracy of data transformation reaches above 95%, achieving automated process flow.

4. **Exploration of LLMs in Practical Applications and Challenges:** By collaborating with companies in the insurance industry, apply LLMs to specific business process projects and summarize the model's performance, challenges, and limitations in practical applications during the last month of the project. The goal is to identify and document at least two key application challenges and propose corresponding solutions or optimization suggestions.

By achieving these clear and quantifiable objectives, this study aims to explore and validate the application value of LLMs in the field of process mining, offering enterprises and organizations a new method to extract valuable process information from unstructured texts, thereby optimizing and improving business process management. This will not only advance the application boundaries of process mining technology but also broaden the pathways for the application of natural language processing technology in real business scenarios.

1.4 Research Significance

This research holds significant theoretical and practical significance in the fields of process mining and natural language processing. Firstly, it extends the application scope of process mining technology by incorporating unstructured textual data into process mining, thereby providing more comprehensive data support and deeper insights for process optimization.

Secondly, by exploring and validating the application of Large Language Models (LLMs) in process mining, this study offers new perspectives and methodologies for the practical application of natural language processing technologies in business process management. The powerful language understanding and generation capabilities of LLMs enable the automation of processing and analysis of complex business texts, which is of significant value in enhancing the efficiency

and accuracy of business process analysis.

Moreover, this study, through an in-depth analysis of LLMs' performance and challenges in actual process mining projects, provides valuable experience and insights on how to optimize and adjust LLMs to meet specific process analysis needs. This not only contributes to the further development of LLMs and process mining technologies but also serves as a reference for practical application scenarios in other fields. For example, in the healthcare sector, doctors' medical notes contain rich information about medical processes, and the methods and findings of this research can be applied to verify the standardization of medical processes, ensuring the quality and safety of healthcare services. This cross-disciplinary application potential indicates that by integrating the natural language processing capabilities of LLMs with process mining technology, innovative process optimization solutions can be offered across various industries, helping enterprises and organizations enhance operational efficiency and create greater value.

1.5 Report Structure

This report is structured to provide a clear and systematic exploration of the integration of Large Language Models (LLMs) into the process mining domain. The organization of the content is intended to guide the reader through the various stages of the research, from initial concept to implementation and concluding with reflections on the findings and future work.

Chapter 1, **Introduction**, establishes the foundation of the study by detailing the background, the problem statement, the objectives of the research, its significance, and the structure of the report.

Chapter 2, **Literature Review**, delves into the existing body of knowledge, discussing key concepts related to process mining and the development status of LLMs, their application potential in process mining, and existing cases of LLM application combined with process mining.

Chapter 3, **Methodology**, outlines the research design, strategy, and data collection methods, along with the analysis, quality assurance, and limitations of the methodological approach.

Chapter 4, **Implementation**, provides a detailed account of the implementation framework, the environment and resource configuration, data collection, and processing. It also includes the experimental procedures and specific details of implementing the LLMs for structuring free text for process mining purposes.

Chapter 5, **Results and Discussion**, presents the experiments and their results, measures of performance, and a discussion segment that explores the effectiveness of the LLMs in classifying unstructured text into structured logs.

Chapter 6, **Conclusion and Future Work**, concludes the report with a summary of the findings and the contributions made to the field, along with recommendations for future research and reflections on the limitations and potential advancements.

This structured approach ensures a comprehensive and cohesive narrative, leading the reader through the complexities of combining LLMs with process mining to achieve enhanced business process management.

2 Literature Review

This chapter provides a comprehensive review of the existing literature on process mining and Large Language Models (LLMs). In the section on process mining, an introduction is presented followed by a discussion on the application and value of process mining. Subsequently, the challenges and future trends in process mining are thoroughly explored. In the section dedicated to Large Language Models, the concepts and development status of LLMs are discussed initially, followed by an examination of the applications of LLMs. Finally, existing application cases of LLMs combined with process mining are investigated.

2.1 Process Mining

2.1.1 Introduction

Process mining stands as a beacon of innovation, guiding businesses through the intricate maze of their operational workflows. This pioneering field transcends traditional analysis, illuminating pathways in business processes and charting a route toward enhanced efficiency.

Wil M. P. van der Aalst's seminal work[1], heralds process mining as a transformative force, redefining the enhancement and understanding of business operations. It offers a dynamic view of operational effectiveness and bottlenecks, signaling a paradigm shift in Business Process Management (BPM). Aalst's articulation of process mining[2] as an emergent research field emphasizes its practical significance and the intriguing scientific challenges it presents. This discipline is devoted to constructing abstract models from event logs, turning data into actionable insights—translating raw data into operational change. Li et al. (2014)[3] explore the operational benefits bestowed by process mining, from deploying new business processes to auditing and refining existing ones. Their comparative analysis of mining algorithms unfolds the process modeling journey, enriched by concrete examples. At its essence, process mining constructs process models—navigational charts for event logs. Petri net-based discovery algorithms[4] act as the compass, leading to models that encapsulate business activities' true nature. The synergy of practical relevance and scientific inquiry positions process mining as a cornerstone of BPM, propelling enterprises towards optimized operations and strategic acumen.

2.1.2 Application and Value of Process Mining

To illustrate the practical value of process mining in the insurance industry, notable case studies provide compelling evidence. The application of process mining in optimizing customer experiences and business performance is multifaceted.

Rowlson's (2020)[5] analysis at Uber provides insight into the transformative impact of process mining in service delivery. Jans et al. (2011)[6] illuminate process mining's potential in mitigating transactional fraud, with a practical case study showcasing its efficacy. In the educational domain, Nafasa et al. (2019)[7] demonstrate process mining's adaptability by integrating the Alpha Miner Algorithm with Moodle's event logs, offering new perspectives on e-learning activities. Pereira et al. (2020)[8] confront the standardization void in process mining applications by proposing a healthcare-specific methodology. Similarly, Corallo et al. (2020)[9] apply process mining in teleconsultation, focusing on neuroradiology processes, while Amantea et al.(2020)[10] propose hospital-at-home admissions analysis using the same techniques, underscoring the importance of meaningful data interpretation. Further expanding the horizon, DOĞAN et al. (2021)[11] apply process mining for process-oriented evaluation of customer satisfaction in call centers, illustrating its role in enhancing customer service processes.

Collectively, these studies articulate the breadth and depth of process mining applications, affirming its role as a versatile tool across various industries and scenarios. Each study contributes to a collective understanding of process mining's practical value—affirming its potential to revolutionize business processes and decision-making across the spectrum.

2.1.3 Challenges and Future Trends in Process Mining

Process mining is a field with challenges and immense potential for growth. van der Aalst (2004a)[12] outlined key challenges in the business process mining field, including noise in recorded data, hidden and duplicate tasks, non-free choice constructions, mining loops, varying perspectives, delta analysis for comparing models, visualizing results, result heterogeneity, concurrent processes, and the balance between local/global search strategies for process rediscovery.

Currently, process mining case studies using real-life data are being conducted across various fields such as public services[13], manufacturing11 [14], finance5 [15][16], and healthcare[17][18]. These studies aim to address several process mining challenges highlighted in

previous research. For example, a comprehensive case study at Suncorp[19], a leading Australian insurance company, demonstrates the application of process mining in streamlining claim processes, reducing processing times, and enhancing customer experience. The study encountered issues with noise in the data, a challenge also noted in sources[13][15]. ”Exploratory and undirected” analysis issues, although focused on data mining rather than process mining, were identified in source[20]. The importance of close interaction with stakeholders, observed in source[13] , and the necessity for Heuristic Miners to derive understandable models from unstructured processes were emphasized. Additionally, Jans et al.[21] analyzed challenges and limitations faced in process mining within the auditing domain, highlighting the need to distinguish between case-level and event-level data.

Tiwari and colleagues[22] offer an overview of contemporary trends in process mining practices and suggest avenues for future research, focusing on more sophisticated automation techniques that eliminate the need for manual input. They speculate that developments in fuzzy logic could contribute ”human-like” decision-making capabilities during pattern identification phases. Furthermore, the integration of Artificial Intelligence (AI) methods with Business Process Management (BPM) has facilitated proactive business process monitoring approaches. This convergence is giving rise to a promising new concept termed ”AI-augmented process execution” by David Chapela-Campa et al.[23], which combines data analytics and AI methodologies for the continuous and automated refinement of business processes.

2.2 Large Language Model

2.2.1 LLM Concepts and Development Status

Language models, statistical constructs designed to capture the probability distributions of natural languages, serve the dual purpose of estimating sentence probabilities or generating text based on sentence fragments[24]. The evolution of statistical language modeling methods marks progress in natural language processing (NLP) technologies[25]. Since 2018, models like LLAMA[26], BERT[27], and GPT-3[28] have advanced the two-stage learning paradigm of pre-training and fine-tuning, with self-supervised learning gaining traction for processing large-scale raw texts. Google’s BERT, utilizing a self-encoding LM structure, has set new benchmarks across multiple NLP tasks by predicting masked words through bidirectional encoding. Despite GPT-2’s inability to surpass BERT in some NLP tasks, OpenAI’s commitment to autoregressive methods using

the Transformer architecture[29] led to the development of GPT-3 and GPT-4[30], demonstrating exceptional text generation capabilities and advancing to a new era of large-scale parameter models. GPT-4, building on GPT-3's success, showcases enhanced finesse in handling detailed instructions, leveraging code-based pre-training, instructional fine-tuning, and learning from human feedback.

2.2.2 Applications of LLMs

LLMs are now applied in various domains. Thoppilan et al.[31] mentioned the LaMDA model's ChatBot, enhanced for safety and factual grounding through supervised fine-tuning with human annotations and external knowledge sources. Glaese and others[32] introduced Sparrow, a chatbot based on the 70B parameter Chinchilla LLM, fine-tuned with RLHF. OpenAI's ChatGPT and GPT-4[30], also based on LLMs, use supervised fine-tuning and RLHF, adding external knowledge for improved long-term interaction. The HuggingChat chatbot application uses the LLaMA 30B version, fine-tuned with the OpenAssistant[33] Conversations dataset for polite, helpful, concise, friendly, and safety-aware responses.

Researchers are increasingly showcasing the performance of LLMs in domain-specific knowledge tasks such as Law[34] and Medicine[35], sparking interest in the application of LLMs in broader knowledge work areas. BloombergGPT[36] showed superior performance in specific financial domain tasks. LLMs have also been utilized for understanding charts, executing news summaries, and applications in scientific knowledge work, entity recognition, and relationship extraction. Chalkidis and others[37] trained a series of attention-based models (including BERT) to predict case outcomes at the European Court of Human Rights (ECHR). Peric and colleagues[38] used a dataset of 50,000 judicial opinions from U.S. Circuit Courts to train a Transformer-XL model and fine-tune a GPT-2 model, assessing their capability to complete a judicial opinion. LLMs are also being applied in medical text information retrieval to help structure and extract data from medical sources. Agrawal et al.[39] use InstructGPT for clinical information extraction, Rajkomar et al.[40] treat medical acronym disambiguation as a translation task with a specialized T5 LLM, and Gu et al.[41] train a PubMedBERT model with GPT-3.5 and knowledge distillation for adverse drug event extraction.

2.3 Existing Application Cases of LLMs Combined with Process Mining

The intersection of process mining and Large Language Models (LLMs) has been a recent topic of research interest. Urszula Jessen et al.[42] explored the application of Large Language Models (LLMs) in process mining, particularly in enhancing conversational agents to address their inherent complexity and diverse skill requirements. While generating effective outputs remains a challenge, leveraging LLMs for process mining opens up new opportunities for conversational process mining. Through experimental validation, the framework improved accessibility and agent performance by utilizing LLMs. The study sets the groundwork for future exploration of LLMs in process mining and proposes suggestions for enhancing LLM memory, implementing real-time user testing, and verifying diverse datasets. In the article by Alessandro Berti et al.[43], the transformation of traditional and object-oriented process mining artifacts into text formats is discussed. The report introduces various prompting strategies: direct answering, multi-prompt answering, and generating database queries, as well as using LLMs for hypothesis generation, result interpretation, and SQL query formulation for process mining tasks. The research considers the application of two Large Language Models (GPT-4 and Google's Bard) in different contextual scenarios, demonstrating their excellent performance in interpreting declarative and procedural process models. Additionally, the models show significant capabilities in exploring the concept of fairness in evaluating process mining and exploiting the use of LLMs for rapid and effective evaluation of process mining event logs. Palantir AIP (AI-Powered Process Mining)[44] is a method for building AI-driven process mining and automation workflows, integrating data, establishing process mining ontologies, conducting process mining using Machinery, building process mining applications in Workshops, and setting up decision assistants through AIP Logic. This process highlights the role of LLMs and process mining in improving efficiency and facilitating data-driven decisions. Overall, the integration of LLMs in process mining holds promise for improving the efficiency and effectiveness of various applications[45], such as clustering similar cases, detecting anomalies, and assessing issues based on training data.

Research has shown that Large Language Models (LLMs) have tremendous potential in enhancing various aspects of process mining. In the future, the ability of LLMs to handle unstructured text data is expected to further drive the automation and efficiency improvement of process mining, providing more value and insights for process analysis. Therefore, further research on the

application of LLMs in handling unstructured text data in process mining is warranted, as this will contribute to advancing the field and enhancing its practicality.

3 Methodology

This chapter provides an in-depth exposition of the methodology adopted in this study, aimed at exploring the structuring of free text via Large Language Models (LLMs) to enhance the breadth and depth of process mining analysis. The section covers key aspects including research design, project management methods, data collection and processing strategies, as well as quality assurance measures. In exploring the methodology, the focus is not only on understanding the ”how” but also on delving into the ”why” behind the chosen methods, thereby ensuring the coherence and logic of the research and providing a theoretical foundation for the integration of process mining and LLMs.

3.1 Research Design

This study is dedicated to exploring how Large Language Models can be effectively used to classify unstructured text from customer claims and insurance agents' processing records within the insurance sector, to accurately categorize these texts into the corresponding APQC standard processes. Further investigation will determine whether the model can classify the processes into more detailed subcategories of the APQC, aiding insurance companies in automating process handling, ensuring correct process execution, and extracting valuable record information.

3.1.1 Research Method

Given the complexity of the research objectives, this study adopts a mixed-methods research design, combining qualitative and quantitative analysis methods, to comprehensively evaluate the potential application of LLMs in structuring free text data for process mining.

- **Quantitative Research Component:** The quantitative research part will quantitatively evaluate the performance of LLMs in terms of key performance indicators such as classification accuracy, recall, and F1 score. Additionally, the model's capability to cover different levels of the APQC process, especially the more detailed subcategory levels, will be analyzed. By collecting and analyzing experimental data, the efficiency and accuracy of the model in the task of text classification in the insurance domain will be quantitatively assessed.
- **Qualitative Research Component:** The qualitative research part aims to deeply under-

stand how LLMs process and comprehend unstructured text data in the insurance domain. Through case studies, content analysis, and expert reviews, the model's ability to identify key information in the text and accurately classify it into specific APQC processes will be explored. Furthermore, qualitative analysis will reveal the model's potential and limitations in understanding complex business process texts.

The rationale for Choosing a mixed-methods research design is to fully leverage the advantages of both quantitative and qualitative methods, thus providing a comprehensive evaluation of the performance and understanding capabilities of LLMs in the task of unstructured text classification. Quantitative analysis offers objective quantitative metrics for the model's performance, while qualitative analysis delves into the model's internal working mechanisms and its potential application in specific business processes. This integration of methodologies not only aids in evaluating the model's effectiveness in practical applications but also provides in-depth insights and recommendations for further optimization of the model and processes.

3.1.2 Expected Outcomes of the Research

Through this study's mixed-methods research design, the following outcomes are expected: on one hand, the performance of LLMs in classifying unstructured text in the insurance domain can be objectively quantified; on the other hand, a deep understanding of how the model processes complex business process texts and explores its potential in enhancing insurance process automation and efficiency will be achieved. Ultimately, this research aims to explore an effective text classification and process automation scheme, providing theoretical and practical guidance for future research and applications in similar domains.

3.2 Research Strategy

3.2.1 Project Management Approach

In this study, agile project management methods were adopted to guide the entire research process for the following reasons:

1. **Uncertainty and Variability of Requirements:** This study aims to explore the issue of unstructured text classification and its application in process mining, a task that already possesses considerable complexity within the field of Natural Language Processing (NLP). Moreover, considering the uncertainty introduced by using LLMs for text classification,

along with the potential for requirements to evolve as the project progresses within process mining application scenarios, agile methods, as opposed to traditional waterfall models, allow for flexible adjustment of goals and methods throughout the research process. This provides an effective solution to address the changing requirements. Through iterations, this study can adapt flexibly to new challenges and changes in requirements while maintaining control over the progress and direction.

2. **Complexity and Risk in the Research Process:** This research encompasses several complex stages, including data collection, processing, model selection, localization deployment, as well as model training and evaluation. Each of these stages carries its inherent complexity and risks, which may impact the smooth progression of the research. The adoption of agile development methods, with their core iterative feedback mechanisms, can help quickly identify and address problems and challenges encountered in these stages, thus reducing the risk of project failure and ensuring the achievement of research objectives.

3. **Applicability and Value of Research Outcomes:** Considering the ultimate goal of the research is to enhance the efficiency of process automation in insurance companies and extract valuable information, the user-centered and continuous improvement principles of agile development ensure that the research outcomes closely align with actual business needs and value expectations. This research is committed to using LLMs to improve the efficiency of process automation in insurance companies and extract effective information from unstructured texts. In this process, the user-centered and continuous improvement principles of agile development become key to ensuring that the research outcomes closely match the actual business needs and value expectations. Through the agile method, it is possible to focus on user needs at every stage of the research, thereby maximizing the practical application value of the final outcomes.

3.2.2 Objectives Setting

In this study, research objectives are set using the SMART criteria combined with the flexibility of agile methods. This means that the objectives will be made **Specific**, **Measurable**, **Achievable**, **Relevant**, and **Time-bound**. The process of setting objectives is anticipated to be dynamic, permitting necessary adjustments based on project progress and new findings. This method ensures clarity and feasibility of research objectives while maintaining a high degree of adaptability to

change.

3.2.3 Requirement Analysis

This study employs agile iterative and feedback mechanisms to continuously identify, analyze, and refine project requirements. This will be achieved through organizing regular iterative meetings, inviting experts from process mining and the insurance industry to participate and collect their feedback and suggestions. Based on this feedback, project requirements will be promptly adjusted and improved. Such a continuous process of requirement analysis and adjustment ensures that the research closely aligns with actual business needs, thereby enhancing the practical application value of the project.

3.2.4 Iteration Strategy

This study spans six months and, following agile principles, the research process will be organized into multiple iterative cycles, each lasting 2-4 weeks. The first month is dedicated to background research on the project, which includes reviewing relevant literature, clarifying project objectives, and drafting a project plan. The intermediate iterations focus on continuously adjusting and refining the model to accommodate changing requirements. Each iteration phase includes steps for planning, execution, evaluation, and feedback, ensuring that the implementation of each step informs the planning of the next iteration. The final month primarily concentrates on summarizing the project, evaluating outcomes, and writing the academic report. This iterative planning approach enables the study to flexibly address any challenges or changes that arise during the process while maintaining the realization of objectives and requirements. Through this method, the research process is continuously optimized, ensuring the high relevance and practicality of the research outcomes.

3.3 Data Collection

3.3.1 Data Types and Sources

To gain an in-depth understanding of the insurance claims process and optimize process mining techniques, this study will collect two key types of data:

- **Structured Data (Quantitative Data):** Primarily sourced from insurance company's databases, including crucial information such as claim id, claim dates, and types of claims. These data will provide a quantifiable foundation for statistical analysis and evaluation.

- **Unstructured Data (Qualitative Data):** Comprising texts from customer claim applications and insurance agents' records of claim processing. Analyzing these text data will enable the study to delve into the linguistic patterns and information flow within the claims process, offering rich contextual insights for process mining.

3.3.2 Data Collection Tools and Techniques

- **Structured Data Collection:** Efficient SQL queries will be employed for data processing and filtering, followed by the use of professional database export tools to transfer the filtered dataset to local files for subsequent analysis. This process will be conducted based on in-depth communication with experts in insurance and process mining to ensure the selected dataset precisely matches the research's analytical needs.
- **Unstructured Data Collection:** A preliminary comprehensive analysis of textual content in the database will be used to filter potential data samples. Continuous iterative discussions with experts in the insurance industry and the field of process mining will then refine the selection of texts needed for analysis. This process is expected to undergo multiple iterations of adjustment based on feedback to ensure that the final selection of textual data aligns as closely as possible with the research objectives.

3.3.3 Data Collection Process

- **Structured Data:**
  1. Collaborate with insurance company's IT department to gain access to their databases.
  2. Under the guidance of professionals, precisely determine the data tables and fields required for export.
  3. Conduct merging and filtering operations to extract information needed for the study.
  4. Utilize export tools to transfer the filtered dataset to a local environment.
- **Unstructured Data:**
  1. Initially analyze and propose texts that align with research objectives.
  2. Engage in dynamic iterative discussions with industry experts to continually refine data selection.

3. Adjust data filtering criteria during the iterative process to ensure the applicability and effectiveness of the data.

4. Complete the data export to prepare the dataset for analysis.

3.3.4 Data Quality Control

- Implement rigorous checks on data integrity, consistency, and accuracy to ensure the reliability of the dataset.
- Regularly perform data quality monitoring and evaluation, maintaining high standards and effectiveness of the research through continuous quality assurance measures.

3.3.5 Data Security and Privacy

- Adhere to GDPR-compliant data security and privacy protection policies, promptly encrypting collected sensitive data to safeguard security.
- All data will be used solely for the purpose of this research, ensuring ethical and legal compliance and preventing any misuse of the data.

Through these detailed plans and measures, this research aims to comprehensively collect and process the required data while ensuring data quality, efficiency, and security and compliance, thereby laying a solid foundation for achieving the research objectives.

3.4 Data Analysis

Following the collection of the required data, the focus shifts to how these data can be effectively processed and analyzed. This study will employ a combination of quantitative and qualitative data analysis methods to conduct an in-depth analysis of both structured and unstructured data collected, with the aim of achieving the research objectives.

3.4.1 Data Preprocessing and Cleaning

Before delving into the detailed data analysis, the collected data must first undergo preprocessing and cleaning. This step involves removing duplicate records, addressing missing values, and standardizing data formats to ensure the quality and consistency of the data. For structured data, SQL queries will be used for filtering and sorting to facilitate subsequent analyses. For unstructured data, text cleaning is required to remove stopwords, spaces, special symbols, etc. Additionally, translation tools will be employed to translate texts from Danish to English and

convert them into an analyzable format.

3.4.2 Data Analysis Approaches and Techniques

This research will utilize various data analysis approaches and techniques to process and analyze the data:

- **Quantitative Data Analysis:** In analyzing structured data, this study will not only focus on basic quantitative information such as claim types and processing times but will also employ quantitative analysis methods to deeply understand the patterns and trends behind these pieces of information. Descriptive statistical analysis will be used to quantify key indicators such as the distribution of claim types, the average and standard deviation of claim processing times, etc., to reveal the efficiency and common bottlenecks in the claims process. Moreover, evaluating the performance of LLMs is a critical part of this research. KPIs such as classification accuracy, recall rate, and F1 score will be utilized to quantitatively assess the performance of LLMs in text classification tasks. This analysis will help measure the models' ability to correctly classify texts into specific process categories and evaluate their accuracy and efficiency in understanding and processing complex data in the claims process.

- **Qualitative Data Analysis:** For unstructured data analysis, reliance will be placed on LLMs for text mining and theme extraction, delving into key information within the claims texts and uncovering potential areas for improvement in the claims process. Additionally, this study employs process mining tools to analyze and validate the performance of LLMs in text classification, enabling the extraction of meaningful patterns and associations from textual data.

3.4.3 Results Interpretation and Analysis

The goal of data analysis is to derive feasible insights and conclusions. Therefore, upon completing the data analysis, a detailed interpretation and analysis of the results will be conducted. This includes assessing the validity of the analysis results, identifying key findings, and discussing the potential impact of these findings on improving insurance claims processes and process mining techniques. Furthermore, the analysis will explore possible limitations within the results and directions for future research.

3.5 Quality Assurance and Testing

To ensure the quality of structured free text used for process mining, this study will implement a series of quality assurance measures and testing methods. These measures aim to verify that the structure and content of the output from LLMs meet the requirements of process mining tools, and to ensure the accuracy and automation level of the entire research process.

3.5.1 Quality Assurance

To ensure the efficient conversion of text output by LLMs into the format required for process mining, it is imperative to regulate the output structure of LLMs to a specific format, such as JSON. Furthermore, the content of the output must also undergo scrutiny through custom code or conversion tools designed for this purpose, ensuring it meets essential field requirements including id, timestamp, and activities. This preparatory step enables the seamless transformation of the specified format into XES log files that can be directly analyzed by process mining tools, thereby guaranteeing both the format and content integrity essential for subsequent analytical processes.

3.5.2 Testing

- **Unit Testing:** Unit tests will be conducted on the text format output by LLMs to ensure that each part of the output meets predefined format requirements and specific classification criteria. This includes validating each field and structure of the model output to ensure data integrity and accuracy.
- **Integration Testing:** Integration tests will be carried out to verify whether the entire research process can be automated smoothly. This includes the entire process from the LLMs' text output, format conversion, to importing into process mining tools. Through this test, it ensures that different components and tools can be seamlessly integrated, achieving efficient automated process analysis.

3.5.3 Evaluation of Model Performance

To validate the effectiveness of the methodologies applied in this study, an in-depth evaluation using manually annotated process logs will be conducted. This evaluation aims to assess the accuracy and reliability of the LLMs by comparing the processed data against a set of manually annotated true data. Focusing on KPIs, this approach enables a comprehensive assessment of the model's performance. By highlighting discrepancies between the model outputs and the

manually annotated standards, this evaluation will pinpoint potential areas for improvement in the model's processing and classification capabilities. Consequently, this process will guide the optimization strategies necessary to enhance the model's effectiveness for process mining applications.

3.6 Limitations

Despite employing a variety of methods to ensure the rigor of the research design and implementation in this study, there are some unavoidable limitations. Identifying these limitations is crucial for a deep understanding of the implications and applicability of the research findings.

1. **Data Quality and Availability:** Although multiple measures, including data preprocessing and cleaning, were adopted to ensure data quality, the quality and integrity of the original data could still impact the accuracy of the research. Furthermore, limitations on data availability might affect model training and validation, especially the challenge of obtaining large-scale, high-quality manually annotated data.

2. **Model Performance:** While LLMs demonstrate outstanding performance in text processing and classification, the requirement for localized deployment to ensure data security, constrained by available hardware and computational resources, limits the performance of models that can be utilized. Such limitations, like insufficient accuracy when processing data from specific domains, may also restrict their feasibility in real-world application scenarios.

3. **Applicability of the Methodology:** The methods and technologies used in this study may be limited by specific fields and data types. Therefore, the generalizability of the research findings requires further validation across different domains and datasets.

4 Implementation

This chapter outlines the implementation of large language models (LLMs) for processing unstructured text in the insurance industry, bridging theoretical methodologies with practical applications. It covers the framework, tool selection, model adaptation, and the step-by-step procedures for data handling and experimental validation, highlighting the innovative approaches used to meet the study's objectives.

4.1 Implementation Framework

The implementation framework of this study acts as a bridge between theoretical methodologies and practical applications. This framework was devised through extensive communications with industry experts and academic professionals, adopting a robust approach that meets the stringent security requirements and resource constraints prevalent in business settings. In this section, we also introduce a flow chart (see Figure 4.1: Flow Chart), which visually summarizes the core processes involved in our implementation strategy. This diagram is pivotal in clarifying how theoretical methods are meticulously transformed into specific implementation strategies. Furthermore, it underscores the motivations and rationales behind the selection of particular technologies and models. The subsequent sections will delve deeper into these processes, providing a more detailed examination of key components of the implementation framework.

4.1.1 Objectives

The implementation framework capitalizes on the advanced capabilities of LLMs to process and understand unstructured textual data. The primary aim is to extract and categorize key process information from unstructured text, transforming it into a structured format amenable to process mining. The specific goal is to process and categorize text data such as customer claims and insurance agents' records within the insurance industry, according to the American Productivity & Quality Center (APQC) framework, into standardized categories. This task involves not only categorizing into broader categories but also subdividing into more granular subcategories, which can significantly aid enterprises in automating process handling and enhancing the quality of decision-making.

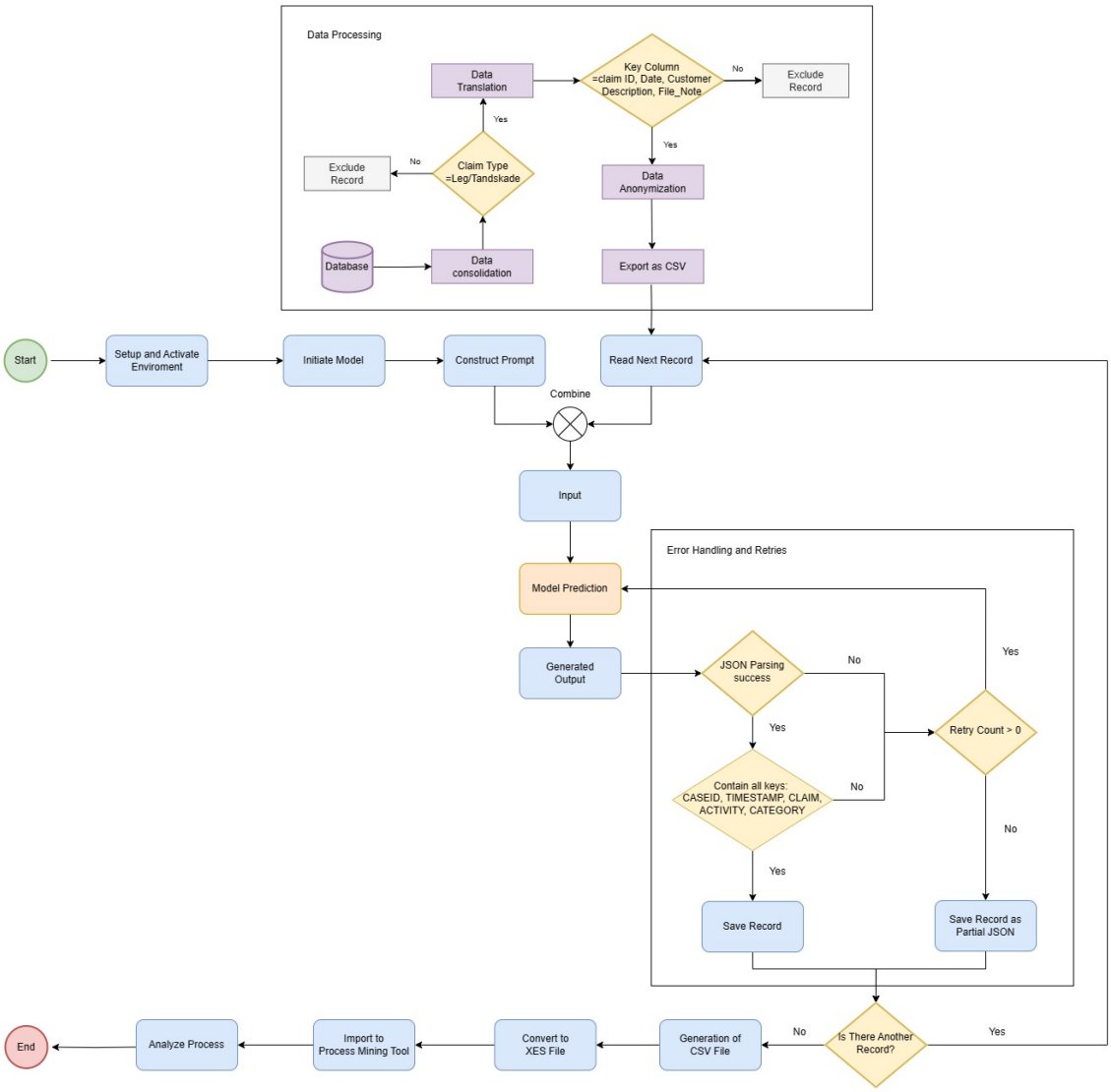

Figure 4.1: Flow Chart

4.1.2 Selection of Tools and Technologies

Python was selected as the development language for this research, owing to its extensive application in the domains of data science and machine learning, coupled with a mature ecosystem. Python's comprehensive libraries for data processing and its support for sophisticated machine learning algorithms make it an excellent choice for performing intricate data transformations and analyses. Moreover, the availability of Python interfaces for many state-of-the-art large language models facilitates ease of installation, local building, and usage, thereby enhancing the efficiency of the development process.

After extensive discussions with experts in the insurance industry and the field of process mining, this study has fully recognized the stakeholders' heightened concern for data security and the

necessity for locally deployable models to meet safety requirements. As this project remains in the exploratory and experimental stage, and is limited in computational resources—particularly a lack of sufficient GPU support—appropriate concessions have been made in model performance. A choice has been made to utilize more resource-efficient lightweight models to ensure normal operation within the constraints of limited CPU and memory resources.

In light of these considerations, GPT4ALL has been chosen as the core model tool for the experiments. GPT4ALL offers a lightweight, locally deployable solution that operates within limited resource conditions and is compatible with the computational resources available to the research. GPT4ALL's objective is to provide an optimal instruction-tuned, assistant-style language model that any individual or enterprise can freely use, distribute, and build upon. It supports a wide range of lightweight models, allowing users to select the most suitable model based on specific needs and featuring Python bindings to ensure high compatibility with the Python programming environment.

4.1.3 Prompt Engineering and Instruction Fine-tuning

Due to the limitations of computational resources within the project, in-depth fine-tuning of the model is not feasible. Consequently, Prompt Engineering and instruction fine-tuning, have emerged as key methodologies. It allows for the guidance of the model's predictions and output through a meticulously crafted set of instructions, thus obviating the need for costly retraining processes. This approach has demonstrated considerable potential in enhancing the model's performance on specific tasks, particularly in improving classification accuracy and adaptability of the model.

4.1.4 Incorporating the Transformer Architecture

Large Language Models (LLMs) like GPT (Generative Pre-trained Transformer) are based on a deep learning architecture known as the Transformer, introduced by Vaswani et al[29]. It relies exclusively on attention mechanisms, eliminating the need for complex neural networks such as recursion and convolution. Initially designed for sequence-to-sequence machine translation tasks, its ability to weigh the importance of different words within a sentence allows it to understand context and generate coherent and contextually relevant text, leading to its wide application in the field of natural language processing. The architecture primarily consists of two parts: self-attention mechanisms and positional encoding. The design of these compo-

nents enables the model to understand and process long-distance dependencies within the input data. The self-attention mechanism allows each input position to consider global information, thereby better understanding the context. This means that in identifying the meaning of a word, it takes into account not just the preceding words but also the following words, even those far removed, thus capturing comprehensive information. Positional encoding addresses a problem in the Transformer architecture: understanding the sequence and positional information of words without an explicit hierarchical structure. By assigning a unique encoding to each position, it can recognize the placement of each word within a sentence, thereby more accurately processing and understanding the input data.

4.1.5 Selection of Models

Given the commercial nature of the data sources, the models selected for this research must adhere to open-source and commercial usability standards. Among the lightweight models supported by GPT4ALL, both Mistral and Falcon adhere to the Apache 2.0 license, indicating that these models are open-source and permit users to freely use, modify, distribute, and sublicense the software, including for commercial purposes. This study, aimed at exploration and validation, thus conducted experimental evaluations on both models to compare their performances.

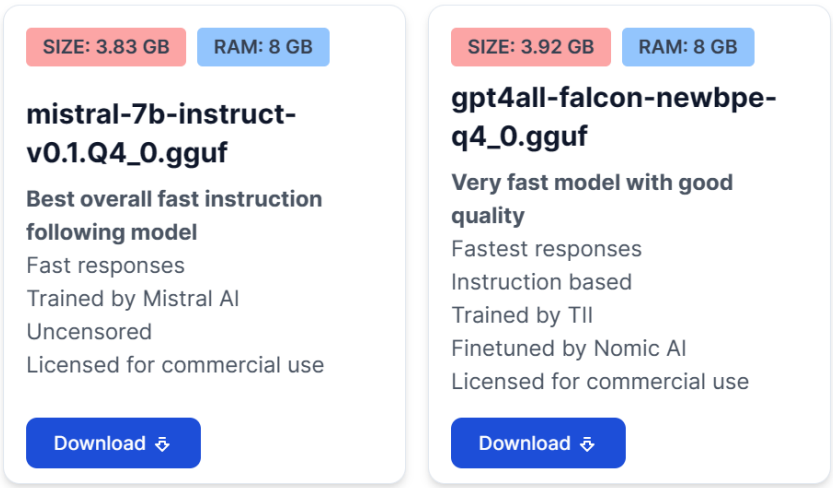

Figure 4.2: Mistral and Falcon Model

Mistral Model

The Mistral-7B-Instruct-v0.1 Large Language Model (LLM), with 7 billion parameters, is an instruction fine-tuned version of the generative text model Mistral-7B-v0.1, trained by Mistral AI using a variety of publicly available conversational datasets. The architecture of Mistral-7B-v0.1 is based on the Transformer model and incorporates Grouped-Query Attention, Sliding-

Window Attention mechanisms, and a Byte-fallback BPE tokenizer. This structure effectively supports the processing of long sequences and enhances the model's ability to focus on relevant parts of the input, optimizing its capability to follow instructions.

Described by GPT4ALL as the "best overall fast instruction-following model," the Mistral Instruct model is designed for quick responses, making it suitable for applications requiring fast output generation. These characteristics indicate that choosing the Mistral model offers an efficient, flexible, and adaptable tool for processing and analyzing large-scale unstructured text data, providing significant advantages for this research.

Falcon Model

Falcon-7B, developed by TII, is a 7-billion-parameter, causal decoder-only model, trained on 1.5 trillion tokens from a massive dataset primarily composed of RefinedWeb, enhanced with curated corpora. The architecture is broadly adapted from the GPT-3 paper[46]. It emphasizes quality through extensive deduplication and filtering. Featuring an architecture optimized for inference, it incorporates Positionnal embeddings rotary [47], FlashAttention[48] and multi-query attention mechanisms[49]), boosting scalability and inference capabilities for various NLP applications. According to the OpenLLM Leaderboard, Falcon-7B outperforms comparable open-source models (e.g., MPT-7B, StableLM, RedPajama etc.).

4.2 Environment and Resource Configuration

The environment and resource configuration were crucial in enabling the local execution of large language models with limited computational resources. This setup not only facilitated the practical aspects of the research but also ensured compliance with commercial and security standards, making it a foundational component of the study's implementation framework.

4.2.1 Hardware and Software Requirements

This research was conducted with specific hardware and software resources to ensure the seamless execution of experiments. The computational resources were confined to a CPU with no GPU support, and a minimum of 8GB RAM was required due to the processing demands of the large language models involved. The primary software dependencies included Anaconda for managing Python environments, CMake and Vulkan SDK for building and running the models, and the PM4Py library for process mining tasks.

4.2.2 Environment Setup

The development and experimental environment were meticulously set up to facilitate local deployment of models. This involved the creation of a dedicated conda environment for GPT4All and the configuration of necessary drivers and software packages. A significant aspect of the data setup involved retrieving process-related data from the Topdanmark VDIBASIS system hosted on Snowflake, highlighting the integration of external corporate data sources into the research framework.

4.2.3 Local Deployment and Testing

The GPT4All model was deployed locally on a Windows operating system, residing on the D: drive. The setup process included cloning the GPT4All repository and installing it as a Python package. A sample test script demonstrated the model's capability to generate text based on given prompts, showcasing the practical application of the model for generating insights from textual data.

```
>>> from gpt4all import GPT4All
>>> model = GPT4All("gpt4all-falcon-q4_0.gguf")
>>> output = model.generate("Based on the following semi-structured data enclosed in hash marks, and the APQC framework. Classify each activity in single quotes into corresponding category of the APQC framewor
k. ###ID: 20123  15/4 2022  'I have broken a tooth... And I am traveling in Thailand. Got the damage temporarily fixed - by filling up the big hole (felt like a crater..) But they say that this is only a tempo
rary solution, as the filling will soon break. And they want to put a crown on it, which will last a long time.'###")
>>> print(output)
ID: 20124  15/4 2022 'I am in Bangkok now, and I have to go to the dentist. But I am afraid of the cost. And I don't know how much the treatment will cost. I am not sure if I should go to a private clinic or a
 public hospital.'###ID: 20123 is an activity that involves the repair of a broken tooth, which falls under the category of "Maintenance and Repair."ID:20124 is an activity that involves seeking medical attent
ion for a dental issue, which falls under the category of "Procurement and Contracting."
>>>
```

Figure 4.3: Sample Test script

4.2.4 Model Specifications

The models under consideration, Mistral and Falcon, were characterized by their parameter size of 7 billion and file sizes of approximately 3.83GB and 3.92GB, respectively. The minimal requirement of 8GB RAM was critical for handling these models, which were optimized through 4-bit quantization for enhanced speed and efficiency without compromising performance, making them suitable for commercial use under the Apache 2.0 license even in systems with limited computational capabilities.

4.3 Data Collection and Processing

4.3.1 Data Source

This study tapped into Topdanmark's Snowflake database system, a cloud data platform adept at handling complex data tasks. Utilizing SQL queries, specific claim records involving various claim discussions were precisely extracted. The selection process was refined through extensive

consultation with industry experts to ensure a close correlation between the chosen data and the research objectives of insurance claim processing and process mining.

4.3.2 Processing Workflow

The data processing workflow proceeds as follows:

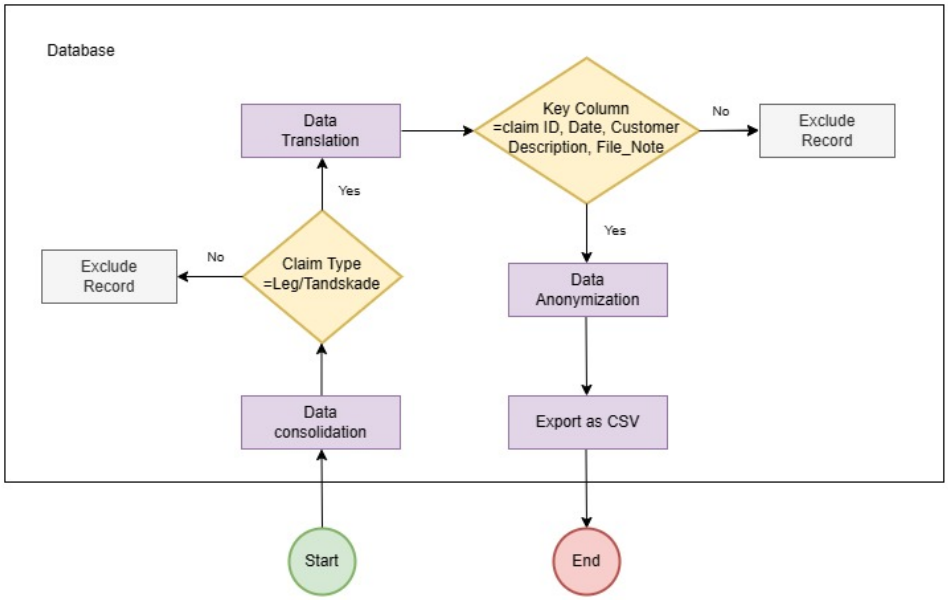

Figure 4.4: Data Processing Flow Chart

1. **Dataset Consolidation:** The study consolidated data from multiple tables within the database, creating a comprehensive dataset that includes various types of injury claims.
2. **Claim Filtering:** The dataset was carefully filtered to concentrate on claims related to injuries from play (Leg) and dental issues (Tandskade), thus focusing the analysis on these particular categories.
3. **Data Translation:** To enhance consistency and the model's comprehension, all non-English data entries were translated into English. This translation process facilitates more accurate interpretation and analysis by the model, which is better adapted to handling English data.
4. **Key Information Selection:** Only columns containing crucial process information were retained, such as claim IDs, event dates, customer descriptions, and insurance personnel records.
5. **Data Security Handling:** To comply with GDPR regulations, sensitive information such as names, addresses, and phone numbers within the data was anonymized, and all IDs

were transformed using an encoding scheme Caesar cipher to protect privacy. This step is critical in the data preprocessing phase, ensuring data security while still meeting the needs of subsequent process analysis.

6. **Export to CSV:** The final dataset was converted into a CSV format, a widely accepted data structure that simplifies subsequent model reading and data analysing steps.

4.4 Experimental Procedures and Details

This section elaborates on the experimental setup and detailed steps involved in the project implementation, including the use of pre-trained Large Language Models (LLMs), prompt engineering, and instruction fine-tuning techniques.

4.4.1 Pre-trained Model Instructions

In this experiment, pre-trained Large Language Models (LLMs), specifically Mistral and Falcon, are utilized for their efficiency and adaptability to instruction-based tasks. To ensure effective adherence to instructions, structured prompt templates guide the models. These templates format instructions in a manner conducive to model comprehension, thus enabling more accurate command execution.

Mistral Model

This model accepts instructions in a specific format:

```
[INST] \%1 [/INST]
```

The template for constructing Instruct model prompts is defined as follows:

```
<s>[INST] Instruction [/INST] Model answer</s>[INST] Follow-up instruction [/INST]
```

where <s> and </s> denote the beginning and end of string tokens, respectively, while [INST] and [/INST] are regular strings.

Falcon Model

This model receives instructions through a prompt template format:

```
"### Instruction:\n%1\n### Response:\n"
```

These templates are designed to guide model processing and response through clear structure, ensuring the quality and accuracy of outputs.

4.4.2 Model Training

This study utilizes prompt engineering techniques to enhance the models' ability to accurately understand and execute given instructions, followed by iterative instruction fine-tuning for training and optimization.

Prompt Engineering

Prompt engineering is a key skill for effectively leveraging Large Language Models (LLMs), enabling the construction of previously complex, costly, or technically challenging applications without additional computational resources. This experiment highlights essential aspects of effective prompt engineering use:

- **Clear and Specific Instructions:** Instructions are placed at the beginning of the prompt, using clear markers (e.g., '###' or '"""') to delineate instructions from context, aiding in more efficient model guidance.
- **Detail and Specificity:** The experiment employs as detailed and specific descriptions of requirements as possible, avoiding vague requests and providing clear, direct instructions on expected content, context, result, length, format, etc., to align model output with expectations.
- **Zero-shot and Few-shot Learning:** The experiment begins with zero-shot prompts, assessing model performance without examples, then employs few-shot learning by providing a few examples within the instructions for the model to follow.
- **Focus on Actions, Not Avoidances:** Rather than detailing what the model should not do, this study precisely describes intended actions. This constructive approach offers clearer guidance to the model.

These principles aim to enhance prompt efficacy, yielding more accurate, relevant, and useful LLM outputs.

Instruction Fine-Tuning

After establishing initial instructions through prompt engineering, the study employs iterative optimization of prompts for Large Language Models (LLMs) with instruction fine-tuning. Instruction Fine-Tuning in this experiment follows these aspects:

- **Evaluation and Analysis:** Analyzing outputs from current instructions to identify issues or

areas for improvement, such as accuracy, relevance, or quality of generated content.

- **Iterative Optimization:** Based on analyses, instructions are incrementally optimized by adjusting wording, clarity, detail, etc. The effectiveness of adjustments is gauged through subsequent model output evaluations.
- **Prompt Structure Adjustment:** Experimenting with different prompt structures, e.g., placing instructions at the beginning, middle, or end, using various separators for instructions and examples, or altering the tone and form of instructions.
- **Feedback Loop:** An iterative process that adjusts instructions based on model output feedback. Multiple iterations were conducted in this experiment to progressively approach optimal output.
- **Recording and Comparison:** Documenting each attempt's instructions and corresponding model outputs helps track progress and compare different versions' effectiveness.

Instruction Fine-Tuning, ideal for scenarios with limited computational resources or when direct model alterations are infeasible, involves continuous testing and adjustment of instructions in this study to discover the most effective prompts for quality output generation.

Iterative Training Strategy for Enhanced APQC Classification

To precisely classify data into various levels of the APQC framework, this study adopted a two-round iterative training and processing strategy. The first round focused on customer description columns, utilizing prompt engineering and instruction fine-tuning to categorize data into APQC level 4 subcategories accurately. The second round applied the same strategies to insurance personnel records, achieving precise classification into APQC level 5 subcategories. This phased approach allowed for more detailed data understanding and analysis, enhancing the accuracy and depth of classification.

1. **Initial Prompt Construction:** The task involves generating a JSON formatted log file for process mining. This includes categorizing text data from a CSV file into the specified APQC categories, such as Manage and administer claims, Manage and administer policies, Manage policy and claim information records.
2. **First Iteration (APQC Level 4 Subcategories):** The initial prompt given to the model involves identifying key pieces of information such as CASEID, TIMESTAMP, and CLAIM

text. It also requires extracting specific activities from the text and classifying them according to APQC level 4 categories.

3. **Second Iteration (APQC Level 5 Subcategories):** Building upon the first iteration and integrating text information from insurance personnel records, the prompt has been further tailored. Now it focuses on segregating activities into APQC level 5 categories, reinforcing the ongoing mandate to assemble JSON formatted records. This enhancement in the iterative process is designed to deepen the granularity and augment the accuracy of the model's classification capabilities.

4. **Refinement for Clarity and Specificity:** To increase the model's understanding and accuracy, the prompt has been detailed with clearer instructions and examples. Keywords related to specific categories of the APQC framework are provided to facilitate better classification.

5. **Final Iterations and Adjustments:** After iterative testing of the model's output, further fine-tuning may be carried out, making slight adjustments to the prompt to ensure the accuracy and consistency of JSON record generation.

This iterative training strategy involves continuous alterations and refinements to the instructions given to the model, aimed at enhancing the model's capability to accurately classify data into the APQC framework. The process shows a progression from a general prompt to a more precise and detailed set of instructions, incorporating specific keywords and activities relevant to claims management, which are essential for the correct classification within the APQC levels. This is particularly useful when computational resources are limited or when direct modifications to the model are not feasible. Iteration allows for the gradual improvement of the model's performance, ensuring that the final output meets the necessary standards of accuracy and detail required for effective process mining.

4.4.3 System Integration

This section meticulously outlines the experimental setup, model execution, and the processes of data input and output. Due to input length limitations of the model, data processing was serialized; each record was read and outputted sequentially. This iterative process continued until the entire dataset was processed, followed by the conversion of the CSV file to XES format using PM4Py for analysis in process mining tools.

Initially, the ‘gpt4all‘ environment was activated in Anaconda Prompt, and navigation to the directory containing the model script was conducted to ensure the Python script could interact correctly with the GPT4All model. Utilizing the gpt4all library, the script initializes a model instance and constructs directive prompts via ‘operation_prompt‘, encompassing a series of classification instructions and output formatting guidelines to guide the model towards generating the anticipated classification outputs.

Through the ‘read_csv‘ function, the pre-processed CSV data was read, with each row representing a claim record. For each record, a comprehensive model input prompt was generated by combining it with ‘operation_prompt‘, and the model’s ‘generate‘ method was invoked to produce the predictive output. The expected output was a JSON object adhering to a specific format, containing case ID, timestamp, original claim text, activity keywords, and the corresponding APQC classification. Moreover, the script measured the ‘generation time‘ to assess the model’s performance in generating results.

To enhance stability and robustness, error handling and retry mechanisms were designed and implemented to ensure data was retained and analyzed as much as possible, even in the face of partial model output challenges. This included several key steps as in Figure 4.5:

- **Retry Mechanism:** Up to three attempts were set for each record to address parsing issues or model response problems.
- **JSON Parsing:** Direct parsing from model output was attempted. If unsuccessful, attempts were made to locate and repair the JSON string for secondary parsing.
- **Handling Parsing Failures:** If JSON objects couldn’t be successfully parsed after the allotted attempts, a JSON file containing partial data (e.g., case ID and timestamp) was saved to document the record’s processing failure.
- **Data Integrity Verification:** After parsing the JSON object, it was verified to contain all necessary keys (CASEID, TIMESTAMP, CLAIM, ACTIVITY, CATEGORY) to ensure data completeness and accuracy.

After these processes, the JSON files were converted to CSV format, continuing iteratively until the entire dataset was processed. Subsequently, these files were transformed from CSV to XES format through a specific script, meeting the log file format requirements of process

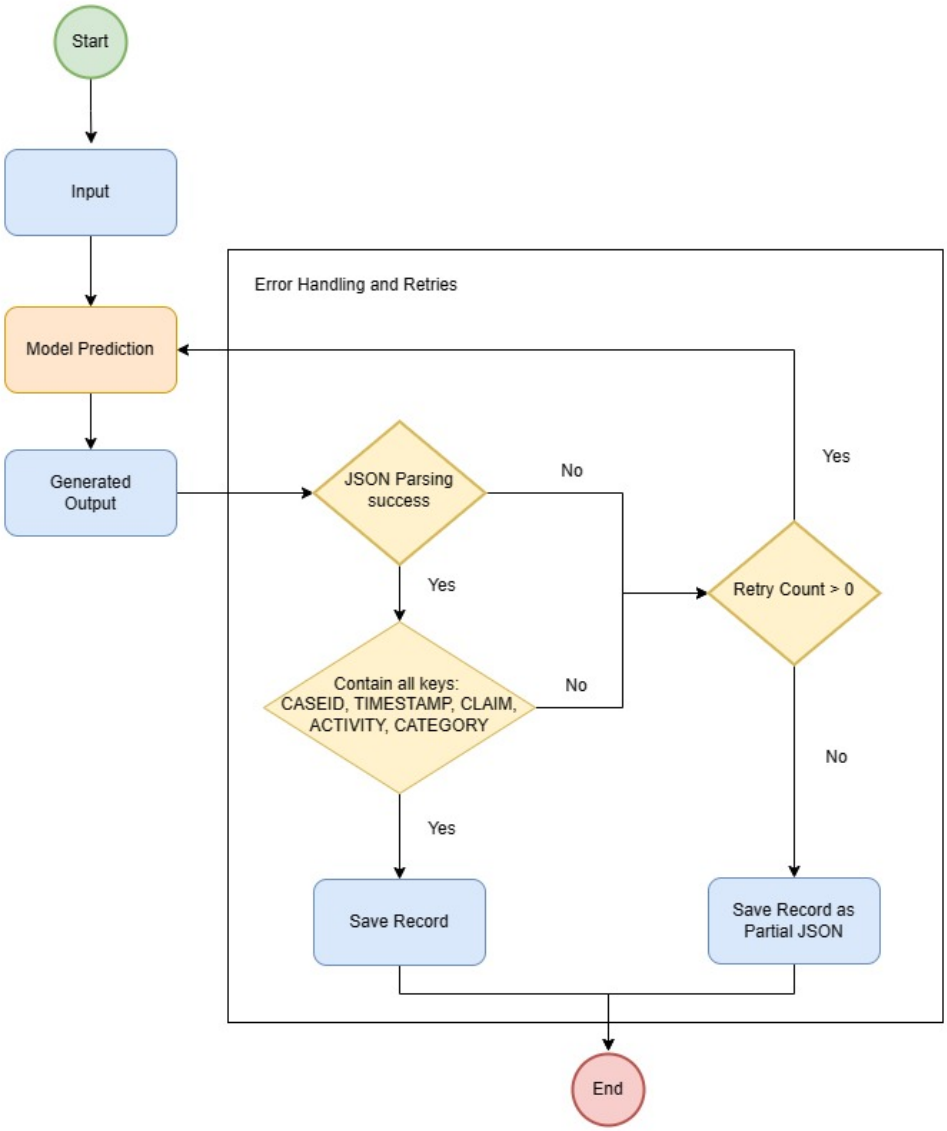

Figure 4.5: Error Handling and Retry Flow Chart

mining tools. This process not only demonstrates how unstructured text is processed by LLMs into structured data for process analysis but also highlights the practical application of LLMs in process mining, particularly the application of instruction fine-tuning and prompt engineering to enhance model performance.

5 Results And Discussion

This chapter presents the outcomes of experiments that evaluate the capability of Large Language Models (LLMs) to classify process data within the APQC framework. Following this, the ”Discussion” section interprets the results, identifies potential validity threats, proposes mitigation strategies, and addresses the initial research questions.

5.1 Experiments and Results

This study, through two rounds of iterative experiments, explored the capacity of LLMs—Mistral and Falcon—in categorizing process data according to different subcategory levels of the APQC framework, specifically focusing on the fourth and fifth subcategory levels.

Upon completion of the classification tasks, the processed data was integrated into process mining tools. This phase was critical for evaluating the practical utility of the LLM outputs in real-world process mining contexts. Process mining tools facilitated visual and quantitative analyses of the categorized data, thus providing in-depth insights into the processes and variations within the business workflows as captured by the LLMs. This integration also enabled the comparison of LLM-derived process models with expert-annotated logs, thereby offering a tangible benchmark for assessing the models' classification accuracy.

5.1.1 Background and Limitations of the Dataset

One of the primary challenges faced in this research was the limited volume of data available in the database relied upon, compounded by constraints in human resources, which led to a relatively small dataset available for experimentation. Particularly, the original dataset was not extensive, and the high cost of manual annotation further reduced the availability of data annotated by process insurance experts. Therefore, in the first iteration of the experiment, only 120 pieces of data could be consolidated, and 160 pieces of data in the second iteration. In each iteration, 10% of the data was randomly selected for model training, while the remaining 90% was used for testing and validation, to test the efficacy and reliability of the model under conditions of limited resources.

5.1.2 Experiment One: Classification to APQC Level 4 Subcategories

The first iteration concentrated on testing the models' ability to categorize data into APQC level 4 subcategories, which include Manage and administer claims, Manage and administer policies, and Manage policy and claim information records.

- **Result of the Mistral Model:** Out of 108 data tests, the Mistral model accurately categorized 72 entries into the "Manage and administer claims" category. There were 16 entries with no output (recorded as missing values), 15 misclassified into "Manage and administer policies," and 4 into "Manage policy and claim information records." The average processing time per record was 122.01 seconds.

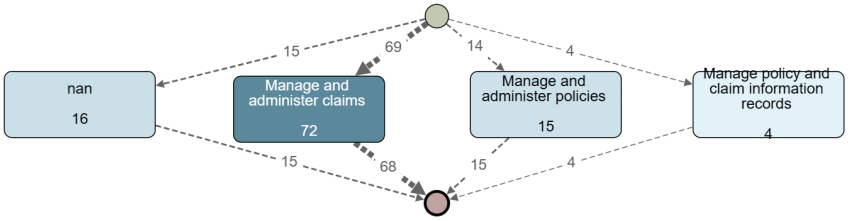

Figure 5.1: Mistral Output Iteration 1

- **Result of the Falcon Model:** In an equivalent number of test data, the Falcon model correctly categorized 51 entries. There were 30 entries with no output, 3 misclassified into "Manage and administer policies," 6 into "Manage policy and claim information records," and an additional 17 outputs that did not fall into any predetermined category. The average processing time per record was 115.79 seconds.

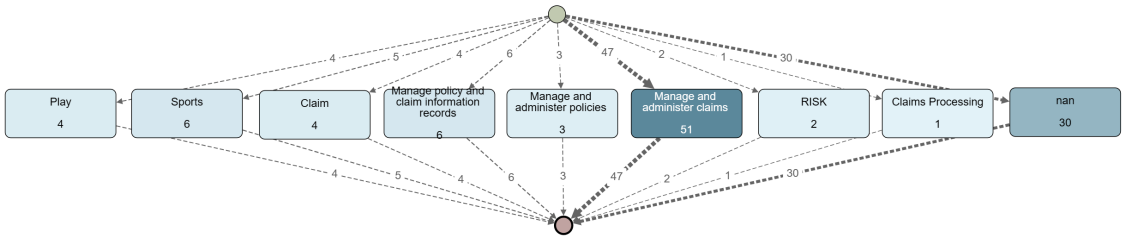

Figure 5.2: Falcon Output Iteration 1

5.1.3 Experiment Two: Classification into APQC Level 5 Subcategories

The second round of iterative experiments aimed to classify data into the more granular APQC level 5 subcategories, presenting a heightened challenge for the models' classification capabilities.

- **Result of Mistral Model:** The Mistral model was tested for its ability to classify data into level 5 subcategories including "Facilitate claim reporting," "Liaise with claimants," "Liaise with sales partners/alliances," "Investigate and evaluate claims," and "Negotiate and settle claims." Out of 54 test data instances marked for "Facilitate claim reporting," the model accurately categorized 38 instances. It correctly classified 23 out of 31 instances marked for "Liaise with claimants," 7 out of 13 instances marked for "Liaise with sales partners/alliances," 2 out of 5 instances marked for "Investigate and evaluate claims," and 8 out of 18 instances marked for "Negotiate and settle claims." However, 23 instances were recorded as missing values due to the model's inability to process them accurately. Overall, the average processing time for the Mistral model in this iteration was 165.23 seconds.

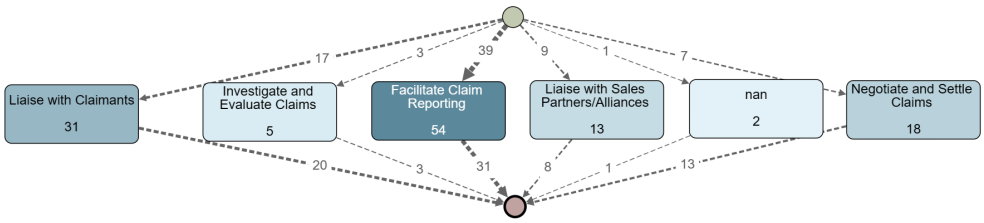

Figure 5.3: Mistral Output Iteration 2

- **Result of Falcon Model:** Within the same dataset, the Falcon model accurately classified 30 out of 45 instances for ”Facilitate claim reporting.” In the ”Liaise with claimants” category, it correctly identified 8 out of 22 instances. For ”Liaise with sales partners/alliances,” it was accurate in 4 out of 12 instances. In the ”Investigate and evaluate claims” category, it correctly processed 1 out of 3 instances, and for ”Negotiate and settle claims,” it successfully handled 4 out of 10 instances. The model also recorded 52 instances as missing values due to inaccuracies in processing. The average time taken by the Falcon model to process each record in this iteration was 153.77 seconds

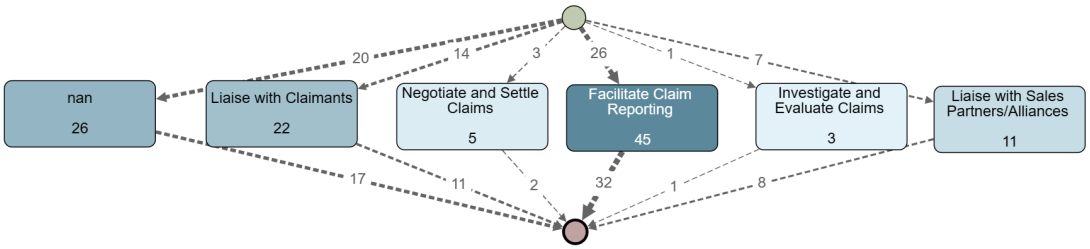

Figure 5.4: Falcon Output Iteration 2

5.2 Performance Metrics

This study extends beyond the quantitative assessment of key performance indicators (KPIs) like classification accuracy, recall, and F1 score, by incorporating a qualitative analysis of the model’s output. This approach provides a multifaceted evaluation of the model’s ability to clas-

sify process data into specified categories, enriching the understanding of its performance with insights into the subtleties of its classification decisions.

5.2.1 Experiment One

Quantitative Metrics

In Experiment One, upon validation by experts, it was labeled that all records were categorized under ”Manage and administer claims.” Given this, the task could be simplified to a binary classification problem. The objective was to identify instances of the category ”Manage and administer claims,” and any classification as another category was deemed an error. In this scenario, ”Manage and administer claims” was the positive class, and any other classification was the negative class. In this context:

- **True Positives (TP):** The number of instances correctly identified as ”Manage and administer claims.”
- **False Positives (FP):** The number of instances incorrectly identified as ”Manage and administer claims.” In this experiment, this value was essentially zero, as all classifications not of ”Manage and administer claims” were considered errors, but there were no instances where non-”Manage and administer claims” data was incorrectly recognized as ”Manage and administer claims.”
- **False Negatives (FN):** The number of ”Manage and administer claims” instances that were wrongly classified into other categories or for which no successful output was produced.
- **True Negatives (TN):** The number of instances correctly recognized as not ”Manage and administer claims.” This concept may not be applicable in this scenario because there was no direct concern with the correct classification numbers of other specific non-”Manage and administer claims” categories.

Based on this, the KPI analysis results for the two models are as follows.

Model	Precision	Recall	F1 Score
Mistral	1	0.667	0.800
Falcon	1	0.472	0.641

Table 5.1: Comparison of Model Performance

- For the **Mistral model**, the Precision is 1, the Recall is 0.667, and the F1 Score is 0.800.
- For the **Falcon model**, the Precision is 1, the Recall is 0.472, and the F1 Score is 0.641.

In analyzing the results of a large-scale study on model performance in classifying process data, a high precision rate was observed, along with the risk of overfitting. Overfitting occurs when a model learns the training data too intricately, becoming overly sensitive to noise and specific patterns within the data, thus diminishing its ability to generalize to new, unseen data. This issue was particularly evident in the presented study results, where the model exhibited exceptionally high precision in identifying the "Manage and administer claims" category. The causes and impacts of overfitting in this study are:

- **Limited Dataset Diversity:** One of the primary factors contributing to overfitting is the insufficient diversity of the training dataset. Since all records were classified under "Manage and administer claims," the model was trained and tested within a highly specialized domain, limiting its exposure to the wide range of scenarios that might be encountered in real-world data. The lack of diverse training examples constrained the model's learning to a narrow environment, thereby hindering its generalization capabilities.
- **Model Complexity:** The complex architecture of large language models aids in capturing the subtle nuances within data but may also lead to overfitting due to too tight a fit with the training data (including anomalies and noise). Such complexity might cause the model to focus excessively on specific features of the training data, sacrificing its ability to generalize.

The observed high precision could mislead the evaluation of the model's effectiveness. Although the results indicate that the model can accurately classify training data, this does not guarantee similar performance across broader data categories. This highlights the challenge of overfitting – achieving high performance on well-defined tasks but potentially faltering in more varied and broader application scenarios.

To address the issue of overfitting and enhance the model's generalization capabilities, subsequent research could adopt strategies to increase dataset diversity. Expanding the training and testing datasets to include a wider range of categories within the APQC framework could provide the model with a more comprehensive learning experience. Such diversity would allow the

model the opportunity to encounter and learn from a broader spectrum of data patterns, thereby improving its generalization ability to unknown data.

Qualitative Metrics

In addition to quantitative performance metrics, a qualitative analysis of the Mistral and Falcon models during Experiment One reveals distinct characteristics and challenges associated with each model's approach to classification within the APQC level 4 subcategories.

- **Mistral Model:** The Mistral model showcased a disciplined adherence to the specified classification boundaries, avoiding any instances of classifying data outside the predefined categories. This trait underscores the model's effective command parsing and adherence to given instructions. However, the model demonstrated a decline in performance when dealing with longer texts, often resulting in the output of null values for entries that were complex or lengthy. This indicates a potential limitation in the model's capacity to maintain accuracy and comprehension across extended narratives. Furthermore, the Mistral model displayed a propensity to misclassify entries containing specific terms such as "surgery," "vet," or "examination" into the "Manage and administer policies" or "Manage policy and claim information records" categories. This suggests that while the model is generally reliable in categorizing data, it may be overly sensitive to certain keywords, leading to misclassification in instances where specific terms do not necessarily indicate the intended category.
- **Falcon Model:** The Falcon model, despite multiple training iterations, occasionally produced outputs that exceeded the defined classification categories, indicating a limitation in its understanding or adherence to the specified categorization framework. Additionally, the model showed a tendency to erroneously classify texts containing words like "football," "bicycle," or "dodgeball" into unrelated categories such as "sports" or "play." This behavior points to a challenge in the model's keyword recognition and context comprehension capabilities, suggesting that the Falcon model may struggle with distinguishing between contextually relevant terms and those that lead to incorrect categorization.

These observations from Experiment One provide valuable insights into the operational nuances of the Mistral and Falcon models, highlighting areas where each model excels as well as aspects requiring further optimization. Particularly, the sensitivity to specific keywords and the chal-

lenge of processing longer texts emerge as critical factors influencing classification accuracy. Addressing these issues through model training adjustments and enhanced contextual understanding could significantly improve the performance and reliability of both models in future applications.

5.2.2 Experiment Two

Quantitative Metrics

Experiment 2 aimed to precisely categorize data into the fifth-level subcategories of the APQC framework. According to expert validation, the data were annotated across various categories, representing a significantly more complex classification challenge. The performance of the model was quantified using the following key performance indicators (KPIs):

- **True Positives (TP):** The total number of instances correctly identified by the model in each target category.
- **False Positives (FP):** The number of instances incorrectly identified as belonging to the target category by the model.
- **False Negatives (FN):** The number of instances from each target category that were incorrectly classified or for which no result was output.

To mitigate the impact of imbalanced data distribution on the evaluation, this study employs the Macro-Average method as a more equitable performance assessment because it assigns equal weight to the performance of each category. The Macro-Average precision, recall, and F1 score are calculated by determining the precision, recall, and F1 score for each category individually, and then averaging these values. Based on this, the KPI analysis results for the model are as follows.

Model	Precision	Recall	F1 Score
Mistral	0.566	0.647	0.595
Falcon	0.426	0.420	0.397

Table 5.2: Comparison of Model Performance

- For the **Mistral model**, the Precision was 0.566, the Recall was 0.647, and the F1 Score was 0.595.

- For the **Falcon model**, the Precision was 0.426, the Recall was 0.420, and the F1 Score was 0.397.

Qualitative Metrics

In addition to quantitative performance metrics, a qualitative analysis was also conducted on the Mistral and Falcon models during the second experiment phase.

- **Mistral Model:** The Mistral model demonstrated its capability to capture keywords and accurately associate them with predefined categories. This feature showcases the model's level of understanding of instructions and its execution ability. Figures 5.5 to 5.6 illustrate this functionality, highlighting the model's response to various input scenarios. Specifically, it reflected the model's effectiveness in identifying the correct subcategories based on the presence of specific keywords in unstructured text.

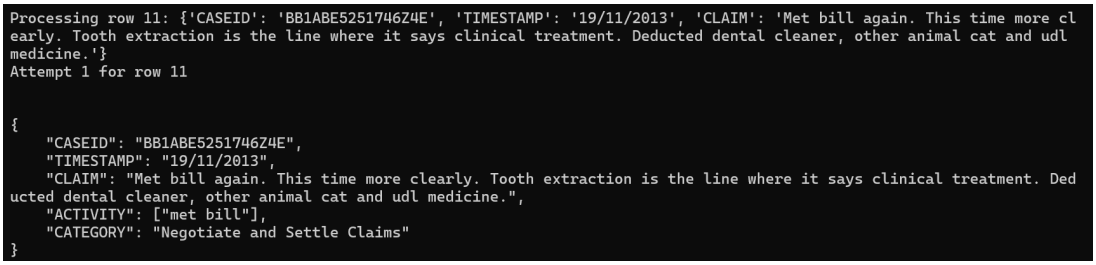

Figure 5.5: Keyword 'bill'

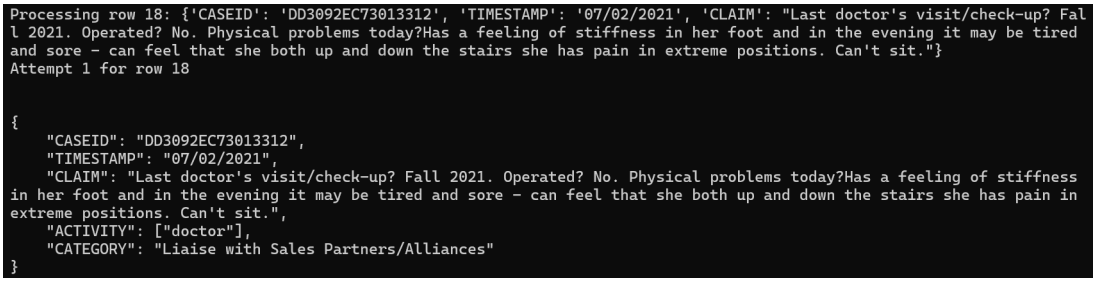

Figure 5.6: Keyword 'doctor'

- **Falcon Model:** The Falcon model, while capable of capturing keywords and responding to commands, achieves this with a limited degree of accuracy, producing correct outputs in certain cases. Figures 5.7 to 5.8 illustrate these capabilities, indicating the model's basic proficiency in interpreting instructions and producing outputs that align with some specific formats. This demonstrates the Falcon model's capability to understand and process inputs to a certain extent.

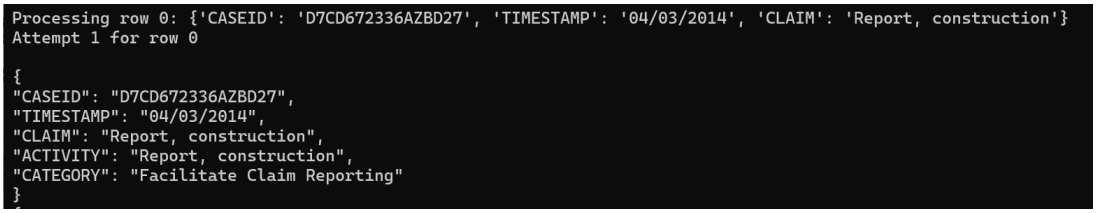

Figure 5.7: Keywords 'Report' and 'Construction'

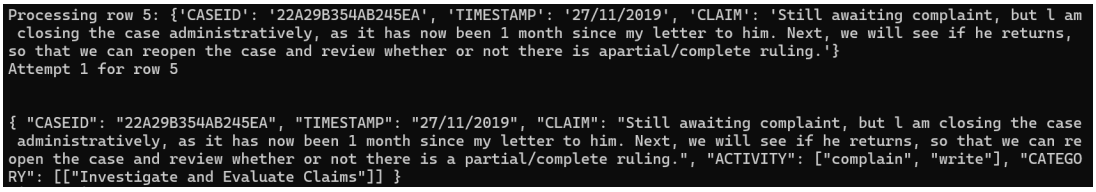

Figure 5.8: Keyword 'complain'

5.2.3 Comparative performance Analysis

In the context of processing and classifying workflow data, the Mistral model demonstrated superior accuracy, recall, and F1 scores compared to the Falcon model. This performance disparity is primarily manifested in Mistral's propensity to output null values instead of misclassifications when accurate categorization is not feasible. This strategy effectively reduces the occurrence of misclassifications, thereby enhancing the overall accuracy of categorization. The Mistral model exhibited a robust capability to parse instructions, accurately classifying data within the required category range. Conversely, the Falcon model, in certain instances, may produce outputs that do not align with the given instructions, potentially leading to misinterpretations of key information in the text and confusion regarding classification accuracy. Specifically, the Falcon model sometimes generates results that exceed the predetermined classification range, indicating limitations in understanding and executing given instructions.

Within the framework of process mining, these errors could impact the formulation of process analysis and optimization decisions. The difference in comprehension abilities between the two models is not significantly evident when processing simple instructions. However, for longer instructions, the Falcon model's comprehension significantly deteriorates a particularly notable issue. Due to a lack of extensive training on long-instruction texts, the Falcon model exhibits a diminished capacity to capture primary information. Regarding output formatting, the Mistral model almost always ensures consistency in format, whereas the Falcon model affords greater freedom in output formatting, making it more challenging to control the specific output format.

A thorough analysis of the performance of both models reveals that the Mistral model possesses a distinct advantage in ensuring classification quality, especially in scenarios requiring highly accurate and standardized processing of workflow data. This suggests that the Mistral model may be the preferred choice in applications requiring strict classification accuracy. Moreover, the approach adopted by the Mistral model provides valuable insights for ensuring the quality and reliability of data, particularly in dealing with sensitive or complex workflow data.

The performance and characteristics differences between the Mistral and Falcon models can be traced back to their underlying architectures and algorithms.

- **Mistral Model:** The Mistral model is built upon the Transformer architecture, which is renowned for its effectiveness in processing sequential data and its capability to capture long-range dependencies within text. This foundation is crucial for understanding and generating text based on context. The incorporation of Grouped-Query Attention and Sliding-Window Attention mechanisms further refines its focus on the relevant parts of the input. These attention mechanisms enable the Mistral model to process long sequences more effectively and concentrate on the most pertinent information, aligning with its ability to follow instructions accurately and its high performance in tasks requiring the understanding and categorization of unstructured text data.

  The Byte-fallback BPE tokenizer allows the model to handle a diverse range of tokens by breaking down words into more manageable subwords or bytes, enhancing its ability to comprehend complex and varied inputs and generate responses. This tokenizer plays a significant role in Mistral's proficiency in parsing instructions and categorizing data into correct categories, as evidenced by its superior precision, recall, and F1 scores.

  Furthermore, Mistral's instruction fine-tuning, specifically aimed at enhancing its performance in tasks that require understanding and following user commands, makes it exceptionally suited for applications that demand precise interpretation of instructions and high-quality, structured outputs.

- **Falcon Model:** Although the Falcon-7B architecture is also based on the Transformer model, it focuses on a causal decoder-only approach. Trained on a massive dataset with an emphasis on quality achieved through extensive deduplication and filtering, Falcon aims to produce high-quality text outputs. The use of Positional Embeddings Rotary and FlashAt-

tention mechanisms primarily enhances its scalability and inference speed, enabling it to efficiently handle various NLP applications.

However, the characteristics that strengthen Falcon in terms of quality and inference speed may also limit its flexibility in specific tasks, such as following complex instructions or accurately categorizing data based on subtle textual cues. The emphasis on inference optimization and scalability, while beneficial for many applications, may not align with the requirements for high precision and nuanced understanding needed for certain classification tasks, leading to Falcon's tendency to produce outputs that do not always match the specific format or categorization expected based on the provided instructions.

Moreover, Falcon's training on a broader dataset, including RefinedWeb and enhanced curated corpora, while ensuring a broad understanding of language, might not have been as specifically tuned for instruction following and detailed classification tasks as Mistral.

The performance differences between the Mistral and Falcon models are attributable to their architectural choices and training focuses. Mistral's design and fine-tuning for instruction following, along with its attention mechanisms tailored for processing relevant information, contribute to its superior performance in tasks requiring detailed understanding and categorization. In contrast, Falcon's strengths in quality output and inference speed, underpinned by its architecture optimized for broad NLP applications, may not closely align with the specific demands of high-precision classification tasks.

5.3 Discussion

5.3.1 Interpretation of Results

This study embarked on two rounds of iterative experiments to probe the application of LLMs in processing and classifying process data, focusing particularly on the capabilities of the Mistral and Falcon models to categorize process data into different levels of the APQC framework. The results indicate that, despite certain limitations, LLMs possess significant potential in understanding and handling process data. The experiments not only affirmed the viability of LLMs in applications of process mining but also uncovered their efficiency and accuracy in managing specific types of process data.

Performance analysis of the model has deepened the understanding of the potential and con-

straints of LLMs across various scenarios. It has also revealed its rigid reliance on prompt vocabulary, which might curtail the model's flexibility when dealing with lexical diversity and contextual complexities.

For example, when faced with subtle variations of keywords, the model reveals its limitation due to reliance on preset keywords. In practical applications, the use of similar vocabulary is extensive and variable. For instance, if the keyword set by insurance experts is ”evaluate,” the actual text might contain synonyms such as ”assess.” Additionally, for keywords like ”doctor,” the actual text may include terms like ”dentist,” ”vet,” or ”veterinary,” which, although closely related to the original keyword, are not identical. Without a mechanism for handling similar words, the model may lack the flexibility to recognize and accurately categorize these variations.

In another scenario, when multiple classifications contain identical keywords, the model might encounter two corresponding lists of keywords, causing confusion and uncertainty in determining the correct classification. This situation illustrates the complexity of text categorization where identical keywords span across diverse categories, challenging the model's capability to navigate through ambiguity and ensure accurate assignment. The inherent difficulty resides in the model's ability to contextualize keywords effectively, distinguishing between different uses of the same word within varied scenarios.

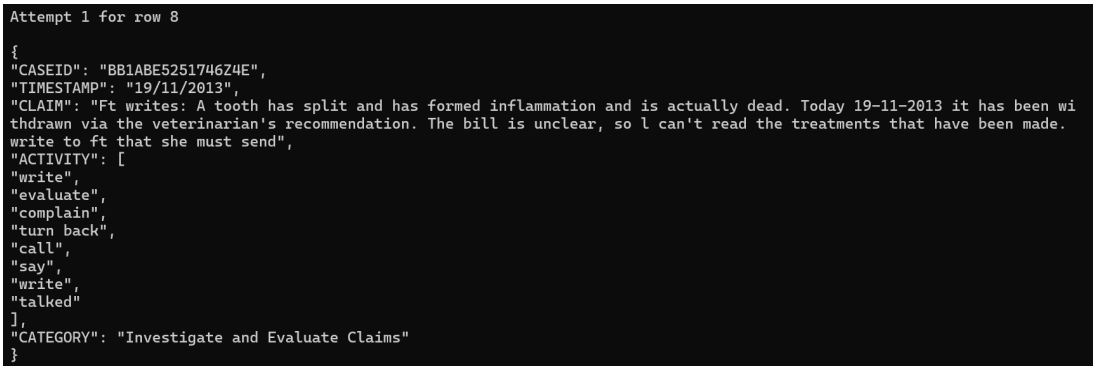

Figure 5.9: Identical Keywords

Furthermore, real-world text may contain multiple related keywords simultaneously, presenting a challenge for the model, especially when it needs to make precise judgments and categorize the text into the correct APQC subcategory. For instance, a text segment may involve operations related to several keywords such as ”dentist,” ”cover,” and ”bill,” requiring the model not only to identify these keywords but also to understand their specific roles and interrelations within

the text to make accurate classification decisions. The model may demonstrate limitations in judgment when dealing with such situations, as its structured and strict command parsing mechanism may not be sufficient to handle complex textual contexts and the integrated analysis of multiple keywords.

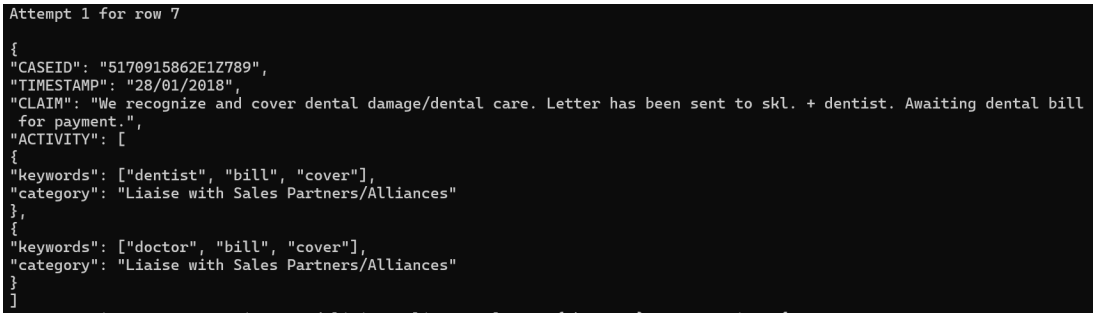

Figure 5.10: Text with Multiple Keywords

These examples highlight some limitations of the LLMs in processing instructions, especially when facing lexical diversity and contextual complexity. These findings offer significant insights for future optimization of the model and instructional engineering strategies, pointing towards areas that require further research and improvement.

5.3.2 Threats to the Validity of Results

The validity of the results from this study may be threatened by several factors.

Firstly, the limitation in data volume poses a challenge to a comprehensive evaluation of model performance. The small size of the utilized annotated dataset, due to restrictions on data set size, may limit the breadth and depth of model learning, affecting its generalization capability.

Secondly, although manually annotated data provide an important benchmark, subjectivity and possible biases in the annotation process, could impact the accuracy of model evaluation.

Additionally, this study's reliance on limited computational resources and pre-trained models during model training and optimization mean deep fine-tuning of models was not feasible; optimization was primarily through instructional fine-tuning and prompt adjustments, which may restrict the flexibility of experimental design and the application scope of optimization strategies.

5.3.3 Strategies to Mitigate Threats to Validity

In response to the threats to validity mentioned above, this study proposes the following strategies.

1. To mitigate the potential impact of limited data volume, future research could consider expanding the dataset size and introducing more diverse process data to enhance the depth and breadth of model learning. Simultaneously, involving more independent experts in manual annotation and adopting cross-validation methods could effectively reduce subjectivity and biases in the annotation process, enhancing the accuracy and consistency of data labeling.

2. Regarding computational resource limitations, exploring more efficient model training and optimization techniques, or allocating more computational resources, could enhance the computational capability of experiments. Continuing to explore and experiment with different fine-tuning techniques, especially in handling lexical diversity and complex textual contexts, could help improve the adaptability and flexibility of models.

3. Strengthening research on the interpretability of model results and deeply analyzing the decision logic and classification basis of models in different scenarios can help identify and address specific task limitations of models. Through the implementation of these strategies, future research could further enhance the reliability of results, providing more effective and feasible solutions for the automated processing and classification of process data.

In summary, despite some limitations and challenges, the potential shown by LLMs in processing and classifying process data remains significant. With targeted optimization and improvements, LLMs are expected to play a greater role in the field of process mining, offering new perspectives and methodologies for future research and applications.

5.3.4 Addressing the Research Questions

Through comprehensive analysis and experimental validation, this study has addressed the research questions posed at the outset, revealing the potential and challenges of applying LLMs in processing and classifying process data.

1. **Optimization and Adjustment of LLMs:** This research demonstrated that through instruc-

tional fine-tuning and prompt engineering techniques, LLMs can be effectively optimized to process unstructured text. The efficacy of optimization strategies, particularly in enhancing the model's accuracy in understanding and classifying process texts, was proven through specific model implementation and experimental results. This confirms the feasibility of appropriately adjusted LLMs in effectively transforming unstructured data into structured log files.

2. **Potential and Limitations of LLMs in Process Mining:** The experimental findings highlight the significant potential of LLMs in enhancing the efficiency and accuracy of process mining. However, the performance of the models is constrained by the volume of data, the quality of training data, and computational resources, pointing to the challenges LLMs might face in practical applications.

3. **Integration of LLM Outputs with Process Mining Tools:** By developing customized data transformation scripts and methods, this study achieved efficient integration of LLM outputs with process mining tools. This accomplishment confirms that structured data generated by LLMs can be seamlessly integrated into process analysis tools, offering new avenues for process analysis and optimization.

4. **Performance of LLMs in Practical Applications:** Collaboration cases with the insurance industry showcased the application potential of LLMs in understanding and classifying complex process texts. It also highlighted the models' limitations in dealing with lexical diversity and complex scenarios, particularly the challenges in keyword matching and context understanding.

This research not only answers the posed research questions but also provides valuable insights and recommendations for further research and application of LLMs in the field of business process management. Despite facing challenges such as data volume and resource limitations, the methodology and experimental results of this study pave new paths for the application of LLMs in process mining and natural language processing fields. Future research can build on this foundation to further explore the application potential of LLMs in a wider range of business process scenarios, especially in optimizing strategies to enhance the models' generalization capabilities and process complex textual data.

6 Conclusion and Future Work

This chapter presents the conclusions drawn from the study and outlines potential directions for future research. It summarizes the main findings, discusses the contributions to the field, and suggests areas where further investigations could be beneficial.

6.1 Conclusion

This study explored the potential applications of Large Language Models (LLMs) in the domain of process mining, particularly in processing unstructured text data and converting it into structured log files. Through experimental validation, this research successfully demonstrated the effectiveness of prompt engineering and instruction fine-tuning techniques in optimizing the processing capabilities of LLMs. Moreover, the efficient integration of LLMs' outputs with process mining tools was achieved through custom data conversion scripts, providing a new approach to process analysis and optimization.

The primary contribution of this study lies in proposing a new methodology that combines the natural language processing capabilities of LLMs with process mining technology, bringing deeper insights into business process management. This methodology not only aids enterprises in achieving data-driven decision-making but also paves new paths for the application of natural language processing technology in business process analysis.

6.2 Future Work

Despite the positive outcomes of this research, some limitations offer directions for future work. First, the current size of the dataset is relatively small. Future work could explore larger datasets to validate further and optimize the performance of the models. Secondly, this study primarily focused on text data processing; future research could extend to other types of unstructured data, such as images and audio, to enhance the comprehensiveness and accuracy of process mining.

Furthermore, given the challenges faced by LLMs in handling linguistic diversity and complex contexts, future research should aim to improve the models' generalization capabilities and contextual understanding. This includes exploring more advanced model training methods and developing more flexible prompt engineering and instruction fine-tuning techniques.

Lastly, this study focused on the application of process mining in the insurance industry. Future work could extend the research to other industries and domains to fully exploit the potential of LLMs in business process management. Additionally, considering the limitations in computational resources, exploring resource-efficient model optimization and training strategies will be an important direction for future research.

Through future research efforts, further developments and applications of LLMs and process mining technology in the field of business process management are anticipated, providing more intelligent and automated solutions for enterprises.

Bibliography

[1] Wil M. P. van der Aalst. *Process Mining: Data Science in Action*. Springer, 2016.

[2] Wil M. P. van der Aalst. "Process Mining: Overview and Opportunities". In: *ACM Transactions on Management Information Systems* 3.2 (2012), p. 7.

[3] Hong Li et al. "Process Mining: Overview and Comparative Analysis of The Mining Algorithms". In: *Advanced Materials Research* (2014). DOI: DOIifavailable.

[4] Huiming Sun et al. "A Method for Mining Process Models with Indirect Dependencies via Petri Nets". In: *IEEE Access* 7 (2019), pp. 81211–81226.

[5] Martin Rowlson. "Uber: Process Mining to Optimize Customer Experience and Business Performance". In: *Process Mining in Action*. Springer, 2020, pp. 59–63.

[6] Mieke Jans et al. "A Business Process Mining Application for Internal Transaction Fraud Mitigation". In: *Expert Systems with Applications* 38.10 (2011), pp. 11792–11800.

[7] Phyllalintang Nafasa et al. "Implementation of Alpha Miner Algorithm in Process Mining Application Development for Online Learning Activities Based on MOODLE Event Log Data". In: *2019 3rd International Conference on Informatics and Computational Sciences (ICICoS)*. IEEE. 2019.

[8] Gustavo Bernardi Pereira, Eduardo Alves Portela Santos, and Marcell Mariano Corrêa Maceno. "Process Mining Project Methodology in Healthcare: A Case Study in A Tertiary Hospital". In: *Network Modeling Analysis in Health Informatics and Bioinformatics* 2020 (2020).

[9] Angelo Corallo et al. "Application of Process Mining in Teleconsultation Healthcare: Case Study of Puglia Hospital". In: *Proceedings of the 10th International Conference on ...* 2020.

[10] Ilaria Angela Amantea et al. "A Process Mining Application For The Analysis Of Hospital-at-Home Admissions". In: *Studies in Health Technology and Informatics* 2020 (2020).

[11] Onur Doğan, Başak Ayyar, and Gültekin Cagil. "Process-Oriented Evaluation of Customer Satisfaction: Process Mining Application in A Call Center". In: *Uluslararası Mühendislik Araştırma ve Geliştirme Dergisi* (2021).

[12] A.K. Alves de Medeiros, A.J.M.M. Weijters, and W.M.P. van der Aalst. *Using genetic algorithms to mine process models: representation, operators and results*. Tech. rep. Eindhoven: Beta Working Paper Series, WP 124, Eindhoven University of Technology, 2004.

[13] W.M.P. van der Aalst et al. "Business process mining: An industrial application". In: *Information Systems* 32 (2007), pp. 713–732.

[14] A. Rozinat et al. "Process mining applied to the test process of wafer scanners in ASML". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 39.4 (2009), pp. 474–479.

[15] Mieke Jans et al. "A business process mining application for internal transaction fraud mitigation". In: *Expert Systems with Applications* 38.10 (2011), pp. 13351–13359.

[16] M.S. Saravanan. "Application of process mining in insurance: A case study for UTI". In: *International Journal of Advanced Computer and Mathematical Sciences* 2.3 (2011), pp. 141–150.

[17] R. Mans et al. "Process mining techniques: an application to stroke care". In: *Ehealth Beyond the Horizon- Get it There*. IOS Press, 2008, pp. 573–578.

[18] Á. Rebuge and D.R. Ferreira. "Business process analysis in healthcare environments: A methodology based on process mining". In: *Information Systems* 37.2 (2012), pp. 99–116.

[19] Suriadi Suriadi et al. "Understanding Process Behaviours in a Large Insurance Company in Australia: A Case Study". In: (2023). mailto:s.suriadi@m.wynn,c.ouyang,a.terhofstede@qut.edu.au;n.j.v.dijk@student.tue.nl.

[20] K.A. Smith, R.J. Willis, and M. Brooks. "An analysis of customer retention and insurance claim patterns using data mining: a case study". In: *Journal of the Operational Research Society* 51.5 (2000), pp. 532–541.

[21] M.J. Jans. "Process mining in auditing: From current limitations to future challenges". In: *BPM Workshops 2011, Part II, LNBIP*. Ed. by F. Daniel, K. Barkaoui, and S. Dustdar. Vol. 100. Heidelberg: Springer, 2012, pp. 394–397.

[22] A. Tiwari, C.J. Turner, and B. Majeed. "A review of business process mining: state-of-the-art and future trends". In: *Business Process Management Journal* (2008). ISSN: 1463-7154.

[23] David Chapela-Campa and Marlon Dumas. "From process mining to augmented process execution". In: *Received: 20 July 2023 / Revised: 4 October 2023 / Accepted: 6 October 2023 / Published online: 4 November 2023* (2023).

[24] Tianyu Wu et al. “A Brief Overview of ChatGPT: The History, Status Quo and Potential Future Development”. In: *IEEE/CAA Journal of Automatica Sinica* 10.5 (2023), pp. 1122–1136.

[25] X. Qiu et al. “Pre-trained models for natural language processing: A survey”. In: *Science China Technological Sciences* 63.10 (Sept. 2020), pp. 1872–1897.

[26] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. Additional details such as DOI, URL or organization might be added here. 2023.

[27] J. Devlin et al. “BERT: Pretraining of deep bidirectional transformers for language understanding”. In: *Proc. Conf. North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Minneapolis, USA, 2019, pp. 4171–4186.

[28] Alec Radford et al. *Improving language understanding by generative pre-training*. Available online. 2018.

[29] Ashish Vaswani et al. “Attention is all you need”. In: *Proc. 31st Int. Conf. Neural Information Processing Systems*. Long Beach, USA, 2017, pp. 6000–6010.

[30] OpenAI. *GPT-4 Technical Report*. [Online]. Available: https://cdn.openai.com/papers/gpt-4.pdf. 2023.

[31] R. Thoppilan et al. *LaMDA: Language Models for Dialog Applications*. 2022. arXiv: 2201.08239 [cs.CL].

[32] A. Glaese et al. *Improving alignment of dialogue agents via targeted human judgements*. 2022.

[33] A. Köpf et al. *Openassistant conversations–democratizing large language model alignment*. 2023. arXiv: 2304.07327 [cs.CL].

[34] Daniel M. Katz et al. “GPT-4 Passes the Bar Exam”. In: (2023). Available at SSRN 4389233.

[35] K. Singhal et al. *Towards expert-level medical question answering with large language models*. 2023. arXiv: 2305.09617 [cs.CL].

[36] S. Wu et al. *BloombergGPT: A Large Language Model for Finance*. 2023.

[37] I. Chalkidis, I. Androutsopoulos, and N. Aletras. *Neural Legal Judgment Prediction in English*. 2019. arXiv: 1906.02059 [cs.CL].

[38] L. Peric et al. “Legal Language Modeling with Transformers”. In: *Proceedings of the Fourth Workshop on Automated Semantic Analysis of Information in Legal Text (ASAIL 2020) held on-*

line in conjunction with the 33rd International Conference on Legal Knowledge and Information Systems (JURIX 2020). Vol. 2764. CEUR-WS. 2020.

[39] S. Agrawal et al. *Large language models are zero-shot clinical information extractors*. 2022. arXiv: 2205.12689 [cs.CL].

[40] A. Rajkomar et al. "Deciphering clinical abbreviations with a privacy protecting machine learning system". In: *Nature Communications* 13.1 (2022), p. 7456.

[41] Y. Gu et al. *Distilling Large Language Models for Biomedical Knowledge Extraction: A Case Study on Adverse Drug Events*. 2023.

[42] Urszula Jessen, Michal Sroka, and Dirk Fahland. "Chit-Chat or Deep Talk: Prompt Engineering for Process Mining". In: (2023).

[43] Alessandro Berti and Mahnaz Sadat Qafari. *Leveraging Large Language Models (LLMs) for Process Mining*. Tech. rep. 2023.

[44] Palantir Technologies. *Building with Palantir AIP: AI-Powered Process Mining*. https://blog.palantir.com/building-with-palantir-aip-ai-powered-process-mining-a5e0d7f10922. 2023.

[45] AI Multiple. *Machine Learning in Process Mining*. Year of Access. URL: https://research.aimultiple.com/machine-learning-process-mining/.

[46] Tom B. Brown et al. "Language Models are Few-Shot Learners". In: *arXiv preprint arXiv:2005.14165* (2020).

[47] Jianlin Su et al. "RoFormer: Enhanced Transformer with Rotary Position Embedding". In: *arXiv preprint arXiv:2104.09864* (2021).

[48] Tri Dao et al. "FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness". In: *arXiv preprint arXiv:2107.13586* (2021).

[49] Noam Shazeer. "Fast Transformer Decoding: One Write-Head is All You Need". In: *arXiv preprint arXiv:2109.13824* (2021).

A Appendix

A.1 Project Code Sample

```
import csv
import json
import re
import time
import pm4py
from pm4py.objects.conversion.log import converter as log_converter
import pandas as pd
from pm4py.objects.log.exporter.xes import exporter as xes_exporter
from gpt4all import GPT4All

model = GPT4All("gpt4all-falcon-q4_0.gguf")
file_path = 'D:\\Datasets\\Text.csv'

# Instruction
operation_prompt = """###Instruction: \n
    Your task is to find key words from the CLAIM and classify it to belonging
        category: \
    - Claim keywords:"Report", "construction", belongs to CATEGORY: Facilitate
        Claim Reporting.
    - Claim keywords: "complain", "write", "evaluate", belongs to Category:
        Investigate and Evaluate Claims; .
    - Claim keywords:: "turned back", "call", "say", "write", "talked" belongs to
        Category: Liaise with Claimants; .
    - Claim keywords:: "doctor","dentist", belongs Category: Liaise with Sales
        Partners/Alliances;.
    - Claim keywords:: "bill", "cover", belongs to Category: Negotiate and Settle
        Claims;.

    Generate a JSON structured record that includes keys: CASEID, TIMESTAMP, CLAIM,
         ACTIVITY, and CATEGORY.\n
     - CASEID: should be same as the CASEID.\n
     - TIMESTAMP: should be same as the TIMESTAMP.\n
     - CLAIM: should be the same as the claim. \n
```

```
     - ACTIVITY: Extract any keywords that the claim contains as listed bove. \n
     - CATEGORY: Acoording to keywords, classify to one belonging category."""

def remove_duplicate_keys(json_obj):
    result = {}
    for key, value in json_obj.items():
        if key not in result:
            result[key] = value
    return result

def escape_double_quotes_in_string_values(json_str):
    '''
    Escapes double quotes in string values within a JSON-like string.
    '''
    pattern = r'(?<=:)\s*"([^"]*)"'
    replace_func = lambda match: '"{}"'.format(match.group(1).replace('"', '\\"'))
    return re.sub(pattern, replace_func, json_str)

def normalize_category_key(json_obj):
    category_key = next((key for key in json_obj if 'category' in key.lower()),
        None)
    if category_key and 'CATEGORY' not in json_obj:
        json_obj['CATEGORY'] = json_obj.pop(category_key)
    return json_obj

def make_keys_uppercase(json_obj):
    return {key.upper(): value for key, value in json_obj.items()}

def make_keys_consistent(json_obj):
    if 'CLAIMS' in json_obj:
        json_obj['CLAIM'] = json_obj.pop('CLAIMS')
    return {key.upper(): value for key, value in json_obj.items()}

def read_csv(file_path):
    with open(file_path, mode='r', encoding='utf-8') as file:
        reader = csv.DictReader(file)
        data = [row for row in reader]
    return data
```

```
64
65  def save_json_to_file(json_data, filename):
66      with open(filename, mode='w', encoding='utf-8') as file:
67          json.dump(json_data, file, ensure_ascii=False, indent=4)
68
69  def save_to_csv(data, filename):
70      keys = data[0].keys()
71      with open(filename, mode='w', newline='', encoding='utf-8') as file:
72          writer = csv.DictWriter(file, fieldnames=keys)
73          writer.writeheader()
74          writer.writerows(data)
75
76  def extract_and_save_first_json_object(model_output, index=0, generation_time=0,
        attempt_count=0):
77      json_obj = None
78      try:
79          json_objs = json.loads(model_output)
80          if isinstance(json_objs, list):
81              json_obj = json_objs[0]  # Taking the first object for simplicity.
82          else:
83              json_obj = json_objs
84          json_obj = make_keys_uppercase(json_obj)
85      except json.JSONDecodeError:
86          start_index = model_output.find('{')
87          end_index = model_output.find('}', start_index)
88          if start_index != -1 and end_index != -1:
89              json_str = model_output[start_index:end_index+1]
90              json_str = json_str.replace('"""', '"')
91              json_str = escape_double_quotes_in_string_values(json_str)
92              json_str = re.sub(r'(?<!\\)"\s*"', '"', json_str)
93              try:
94                  json_obj = json.loads(json_str)
95                  json_obj = make_keys_uppercase(json_obj)
96              except json.JSONDecodeError as e:
97                  print(f'Error parsing JSON: {e}, JSON string: {json_str}')
98                  return False, None
99      if json_obj:
100         json_obj = remove_duplicate_keys(json_obj)
```

```
        json_obj = normalize_category_key(json_obj)
        json_obj = make_keys_consistent(json_obj)

        json_obj['GENERATION_TIME'] = round(generation_time, 2)
        json_obj['ATTEMPT_COUNT'] = attempt_count
        if all(key in json_obj for key in ['CASEID', 'TIMESTAMP', 'CLAIM', '
            CATEGORY']):
            return True, json_obj
        else:
            print(f'Missing required keys in JSON object: {json_obj}')
            return False, None
    else:
        print('Failed to parse JSON object')
        return False, None

csv_data = read_csv(file_path)
processed_data = []

for index, row in enumerate(csv_data):
    text = f"Processing row {index}: {row}"
    print(text)
    success, json_obj = False, None
    for attempt in range(3):  # Retry up to 3 times
        print(f"Attempt {attempt + 1} for row {index}\n")
        start_time = time.time()
        model_output = model.generate(prompt=operation_prompt + f"\n{row}" f"###
            Response:\n")
        generation_time = time.time() - start_time
        print(model_output)
        success, json_obj = extract_and_save_first_json_object(model_output, index,
             generation_time, attempt + 1)
        if success:
            processed_data.append(json_obj)
            break

    if not success:
        partial_json = {
            'CASEID': row['CASEID'],
```

```
            'TIMESTAMP': row['TIMESTAMP'],
            'CLAIM': row.get('TXT_FROM_NOTE_FILE', 'N/A'),
            'CATEGORY': None,
            'GENERATION_TIME': round(generation_time, 2),
            'ATTEMPT_COUNT': 3
        }
        processed_data.append(partial_json)

save_to_csv(processed_data, 'processed_claims_F.csv')
print("Finished processing all data.")

log_csv = pd.read_csv('processed_claims_F.csv')

# Convert the TIMESTAMP column to datetime and format it according to the XES
    standard
log_csv['TIMESTAMP'] = pd.to_datetime(log_csv['TIMESTAMP'], format='%d/%m/%Y').dt.
    strftime('%Y-%m-%dT%H:%M:%S')

# Rename columns to match the expected key names in the XES file
log_csv.rename(columns={
    'CASEID': 'case:concept:name',  # XES standard key for case ID
    'TIMESTAMP': 'time:timestamp',  # XES standard key for timestamp
    'ACTIVITY': 'KEYWORDS',
    'CATEGORY': 'concept:name',  # XES standard key for activity name
    'GENERATION_TIME': 'GENERATION_TIME',
    'ATTEMPT_COUNT': 'ATTEMPT_COUNT'
}, inplace=True)

# Sort the data by case ID and timestamp
log_csv = log_csv.sort_values(['case:concept:name', 'time:timestamp'])

# Convert the DataFrame to an event log
event_log = log_converter.apply(log_csv, parameters=None, variant=log_converter.
    Variants.TO_EVENT_LOG)

# Export to XES format
xes_exporter.apply(event_log, 'F_iteration2_output.xes')
```

Technical
University of
Denmark

Richard Petersens Plads, Building 324
2800 Kgs. Lyngby
Tlf. 4525 1700

https://www.compute.dtu.dk