FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Enhancing Requirements Change Impact Analysis in Critical Systems Using Natural Language Processing

Luís Neto

Department of Informatics Engineering

Supervisor: Gil Manuel Gonçalves

July 31, 2008

Enhancing Requirements Change Impact Analysis in Critical Systems Using Natural Language Processing

Luís Neto

Department of Informatics Engineering

July 31, 2008

Abstract

Machine learning has significantly impacted various domains by extracting meaningful patterns from large datasets, yet its integration into requirements engineering (RE) for safety-critical systems (SCS) presents unique challenges. This study offers a overview of the current research landscape and identifies promising directions for future work in the application Natural Language Processing (NLP) techniques to enhance RE tasks in the context of SCS.

Starting by framing the interrelated nature of SCS and the normative regulation, the research identifies untapped potential for usage of NLP methods. Particularly in case of SCS, we see efficiency potential in the development of tools for assistance in estimation effort and safety risks involved by introducing a change in SCS.

The goals of this research are categorized into three specific activities:

1. Supporting Change-Impact Analysis (CIA).
2. Tracing or eliciting safety-related aspects from existing norms and standards.
3. Facilitating effective analysis, tracing and reuse of change to prevent safety risks and normative violations.

The goals are addressed by a text mining framework that, by usage of topic modelling, assists a requirements engineer to identify and trace areas of impact that a new set of requirements can impose on a running SCS, helping to more effectively identify the extent of a change and the efforts needed to deliver it.

The study identifies research gaps in advancements in NLP techniques for semantic and discourse analysis, domain-specific word embedding models, human in the loop solutions and syntax-aware statement embedding that can benefit an automated retrieval of meaning for analysis of impact changes.

Laying out the particular demands and constrains in SCS development, to answer the goals, this research addresses the identified constrains by exploring and outlining NLP methods and tools in a specific pipeline to improve aspects of the RE process for CIA.

Keywords: Safety Critical, Natural Language processing, Impact Analysis, Requirements Engineering

Acknowledgments

To my family that has supported me through this path, your steady backing and encouragement have been central throughout this journey.

I would like to express my deepest gratitude to my advisor, Professor Gil Manuel Gonçalves, for his uplifting guidance and invaluable support in the preparation of this dissertation.

I am grateful to Critical Software for the opportunity and to all my colleagues whom I have worked with daily throughout this process. I acknowledge the inspiration and encouragement that has been given by the collective assistance and supportive professional environment. Your contributions have most significantly contributed to the successful completion of this work.

I would also like to extend my sincere thanks to Professor Daniel Albuquerque and Professor Márcio Nascimento, as members of the Polytechnic Institute of Viseu, for their vigilant work and heartening dedication, has their guidance has been constitutive for my career.

Luís Neto

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
BoW	Bag-of-Words
CIA	Change-impact analysis
LDA	Latent Dirichlet Allocation
LSA	Latent Semantic Analysis
ML	Machine Learning
NLP	Natural Language Processing
NLP4RE	Natural Language Processing for Requirements Engineering
RE	Requirements Engineering
SCS	Safety-Critical Systems
SDLC	Systems Development Life Cycle
SOTA	State of the Art

Chapter 1

Introduction

Machine Learning (ML) has revolutionized numerous domains by enabling the extraction of meaningful patterns and insights from large datasets. However, its application in requirements engineering (RE), particularly in Safety-Critical Systems (SCS), presents unique challenges and opportunities. This dissertation aims to address these challenges by leveraging natural language processing (NLP) techniques to improve various aspects of RE.

1.0.1 Context

Safety-critical systems are those whose failure could result in significant harm to people, the environment, or substantial financial loss. Examples include transportation systems, medical devices, space missions, and energy sectors.

The development of SCS remains a **highly specialized and regulated field** with unique demands compared to other software industries. The development of such systems stands apart from other industries primarily due to its focus on safety, heavy regulatory norms, rigorous testing and validation processes, and the need for risk management strategies as assuring the integrity of these systems is of absolute importance given the potential of catastrophic consequences of a failure.

Unlike non-critical systems, which may prioritize features, usability, or time-to-market, the development efforts of SCS are geared towards ensuring that the system functions correctly under all conditions and performs robustly in the face of unexpected events or failures; safety and reliability are priorities above all else.

These domains are subject to much more stringent standards and regulations around development and certification compared to many other industries. The development cycle of SCS is characterized by a strong emphasis on risk management and mitigation strategies and proactive techniques like **Reliability, Availability, Maintainability, and Safety (RAMS)** analysis to prevent defects from being introduced in the first place.

Rigorous testing and validation are essential to mitigate risks and ensure that the system operates as intended in real-world environments. Critical systems undergo extensive and independent verification and validation, processes that aim to confirm that the system meets all specified requirements, adheres to safety standards, and performs reliably. Risk analysis is integrated into

every phase of development to preemptively address issues that could impact safety or reliability, to identifying potential hazards, assessing their likelihood and consequences, and implementing safeguards to minimize risks.

Critical systems development often requires interdisciplinary collaboration among engineers, domain experts, regulatory authorities, and stakeholders. This collaboration is indispensable to ensure that all aspects are thoroughly considered. Compliance with regulatory requirements adds complexity and rigor to the development process, influencing every stage from design to deployment. The extensive V&V, analysis, and certification requirements for safety-critical systems result **in significantly higher development costs and longer timelines** compared to less regulated industries. SCS development requires specialized expertise and skills that are not as prevalent as in other software development domains.

These factors collectively contribute to a distinct approach to development that prioritizes safety above all other considerations. As SCS become more prevalent and complex, advancements in these areas are essential for maintaining high levels of dependability and safety.

1.0.2 Motivation

As safety-critical systems become increasingly complex, there is a growing need for advanced tools to manage this complexity. This research aims to contribute by providing insights into the application of NLP in SCS, correlating between existing NLP models and practical applications in a highly regulated and safety-conscious industry.

SCS involve cross-functional teams, including domain experts, safety engineers, and software developers. NLP can help bridge communication gaps by enabling automated analysis and summarizing of technical documents. By linking requirements to other artifacts and test cases, NLP can enhance traceability and ensure that safety concerns are addressed throughout the development lifecycle. NLP can be used to accelerate the analysis of safety requirements, hazard logs, and failure modes, enabling faster and more comprehensive safety assessments and enhanced collaboration between stakeholders.

Traditional methods of managing requirements and their changes can be labor-intensive. Automating parts of RE process through NLP can lead to significant time and cost savings. It reduces the manual effort required, speeds up the analysis process, and helps in early detection of potential issues, thereby preventing costly rework and delays.

Safety-critical systems generate large amounts of, often, unstructured data, such as design documents, incident reports and maintenance logs. NLP techniques can be used to extract valuable insights from this data to inform decision-making and improve system safety.

For the author, writing this thesis provides an opportunity to gain knowledge into an area of personal and professional interest. The practical applicability of the research in real-world industrial settings can have a tangible impact on improving the safety and efficiency of in RE tasks.

1.0.3 Problem

To contextualize the problem, the goals of this research have been categorized into specific areas within RE. Each goal addresses an aspect of the RE process, ensuring a comprehensive approach to enhancing system safety and efficiency.

1. Supporting Change-Impact Analysis (CIA).
2. Tracing or eliciting safety-related aspects from existing norms and standards.
3. Facilitating effective analysis, tracing, and reuse of change to prevent safety risks and normative violations.

This research aims to provide a robust NLP-based framework for RE in safety-critical systems by addressing these goals.

1.0.4 Structure of the document

In addition to the introduction Chapter 1, this dissertation contains 6 more Chapters.

Chapter 2 contains a comprehensive review of the State of the Art in the research area, further investigation on the significance and relevance of this research within the existing body of knowledge setting the stage for a rationale, objectives, and contributions.

Chapter 3 traces an outline focused on contextualizing the research problem in the topic area. There, background information, including the significance and relevance of the problem is described.

The chapter 4 focuses on the results from research. There the overall justification and rationale for selecting what methods and techniques to design solution for the problem under analysis is described.

Chapter 5 describes lower level practical implementation of a solution with detail on the methods, providing technical details, algorithms, tools, and technologies.

The chapter 6 presents evaluation methods, to assess the effectiveness of a solution. It describes the evaluation methodologies and experiments to be conducted.

The chapter 7 summarizes the key findings and contributions of the research, reflecting on the strengths and limitations of the approach. It also contains a reflection for potential areas for future research or improvements.

Chapter 2

SOTA

Requirements engineering sets ground for comprehending a system by providing a structured and documented path for understanding stakeholder needs, managing changes, and ensuring that the final system aligns with the project objectives and the normative context. RE and safety analysis are a key factor for the successful development and certification of SCS.

Managing a SCS development encompasses collecting, linking, and structuring diverse project information, such as requirements, hazards, and components. As a foundation in the software development life cycle, eliciting and documenting the needs and expectations of stakeholders shape the trajectory of the entire project, and is seen as crucial to understand the context and the explanation behind requirements and design decisions, to provide argumentation on the system suitability for use (Almendra and Silva (2020)).

NLP solutions are being integrated into development tools, enabling developers to interact with code and development artifacts in natural language, and in the process simplifying the communication and comprehension of a system to individuals with diverse backgrounds.

The leap in public accessible tools of Artificial Intelligence (AI) has brought significant changes in various aspects of development, influencing how software is designed, implemented, tested, and maintained. The adoption of AI is facilitated in areas where development is flexible, or unrestricted, where developers are encouraged to rapidly prototype and experiment. In industries that are not leashed by heavy regulation and strict compliance measures, the flexibility in regulation allows for easier integration of innovative and adaptive technologies.

SCS such as those used in aviation, automotive, railway, healthcare, and nuclear industries, have rigorous requirements and regulations to ensure the reliability, availability, maintainability, and safety of the systems. The certification of such systems requires compliance, assurance and accountability. As a result, the development of such software is subject to strict regulatory frameworks. Adapting AI to SCS development presents unique challenges and complexities compared to less restrictive development environments.

Integrating AI that can continuously learn and adapt introduces challenges in terms of ensuring that the learning process aligns with safety requirements and standards. Understanding how SCS arrives at decisions is crucial, as such demand a high level of reliability. The methods traditionally

used for validating software may not be directly applicable, and establishing the reliability of AI models under various conditions is a significant challenge.

Considering that advances in AI regulation will still prevent the technology to be directly used as a safety critical application, it can indirectly be used in tools for SCS development, whose result is then subject of human validation.

2.1 Research Question

Well-defined requirement management strategy provides a basis for quality assurance and validation activities. Traceability links requirements to various stages of the development process, ensuring that every aspect of the system is aligned with the original requirements, facilitating navigating complexity in change and validation efforts. The goal here is also to ensure that modifications to the system are carefully assessed and integrated without negatively impacting other aspects.

In the context of safety critical applications, a change implies that there is a clear comprehension of the inner workings and dependencies of the system. These systems usually undertake very long life-cycle and achieve massive dimensions. To help identify potential risks and challenges in the evolution of a system, engineers must comprehend how the impact of the change affects the overall safety. Additionally, any the change should be plainly visible traced in all it's associated artifacts for homologation or audit purposes.

Having identified a gap in support Change-impact analysis (CIA) with focus on safety analysis (Ou et al. (2018)), there is a lack of modern tools to provide the information needed to answer questions such as whether a change of requirements or design may cause safety violations. Hence, this study looks on the state of NLP for RE in the context of critical systems, with focus on addressing the challenges associated with the elicitation and reuse of requirements, mapping the impact of a change on safety.

RQ: Within SCS contexts in software engineering, what tools leveraging NLP are available and effective in supporting CIA? How do these tools employ NLP for requirements engineering? Furthermore, how do these AI-driven approaches trace or elicit safety-related aspects from existing norms and standards, facilitating effective tracing, reuse and analysis of system requirements to prevent safety violations?

2.2 Methodology

The author address the questions by a snowballing approach, as support for evidence-based software engineering referring to the guidelines proposed by Wohlin (2014). A systematic literature review using a snowballing approach is a means of identifying, evaluating, and interpreting available research relevant to a particular research question as well as to support evidence-based developments.

2.2.1 Search Strategy

The "natural language processing" (NLP) was included to focus on research related to the computational understanding and generation of human language, a central aspect in the context of requirements engineering. "Artificial intelligence" is added as correlative term to avoid only capturing papers referring specifically to NLP and missing papers using a different terminology. "Requirements" was chosen to capture the literature related to the various phases of requirement engineering. The term "critical systems" was included, emphasizing the importance of enclosing studies relating requirements or impact analysis in such high-stakes environments. Lastly, the terms "standards" and "Norms" were incorporated to explore requirements engineering and the integration of the aforementioned technologies in the normative context.

2.2.2 Data Sources

From the various data sources available, the search was constrained to the Database of the professional organization ACM Digital Library, the full text collection. The decision to focus the search exclusively on the ACM Digital Library is based in the unique and specialized nature of this repository. The ACM Digital Library stands out in the field of computer science and information technology, hosting a wealth of peer-reviewed articles, conference proceedings, and journals that are highly relevant to the subject of NLP, AI and RE. The ACM Digital Library is known for its extensive coverage of conferences and journals within these domains, ensuring a concentrated and comprehensive exploration of literature that is both seminal and current. Restraining the search to this platform not only streamlines the review process but also guarantees a high relevance to the core themes of the study. An additional benefit of restricting the search to ACM is including studies from reputable journals or conferences sources. This contributes to the overall rigor, credibility, and ethical standing of the research literature being reviewed. This approach ensures that the synthesis of knowledge is built on a foundation of high-quality and reliable research contributions.

2.2.3 Inclusion/Exclusion Criteria

In the first search attempt has resulted in an astounding 711,428 results. To narrow the results, it was decided to focus on articles published since 2018. This pruning brought our results down to a still unmanageable amount of 213,328, around 30% of the initial set. By introducing the keyword ‘Software’ into the search term lead to the final set of 85 artifacts.

Exclusion criteria:

- Studies not written in English.
- Studies that focus on Hardware Engineering or other non-Software-related topics.
- Studies that focus on certifying AI Models for Critical Systems
- Studies related to Autonomous Driving

- Studies related to Machine Learning Systems Validation
- Studies related to Model Driven Engineering
- Books

Inclusion criteria:

- Studies published since 2018
- Conference papers
- Journal articles

2.2.4 Initial set of papers

Out of the initial collection of 85 results, 49 were identified as books and subsequently excluded from further consideration. The remaining 36 candidates underwent a detailed review based on their alignment with the specified search terms. Among these, 3 studies were identified to be directly related to hardware, while another 3 candidates classify as research on Autonomous Driving. Additionally, 6 candidates were focused on the validation of Machine Learning systems, and 4 candidates centered around Modeling or Model-Driven Engineering.

Additional studies were deemed out of scope, including 3 candidates related to security and privacy and 2 studies focusing on theorem proving. Furthermore, individual studies covering a range of topics such as android development, process and methods, software maintenance, software verification and validation, code generation and optimization, and cloud databases were also considered outside the scope. After a thorough analysis of inclusion and exclusion criteria for the results of the search term, a subset of 6 candidates was identified, presented in Table 2.1.

2.2.5 Snowballing iterations

For every paper in the initial set, a snowballing search was conducted, both backward and forward, examining the references within, and citations of each paper. The methodology encompassed a review, analyzing at first the context surrounding citations and title. When in doubt, abstracts and conclusions where consulted. Subsequently, inclusion and exclusion criteria was employed upon text reading to determine relevance, thereby adding papers meeting the criteria to the original set. The analysis of individual papers, also presented in Table 2.2;

- For C1, cited in 46 studies according to ACM, this effort led to the inclusion of 6 relevant journal articles.
- C2, despite having 24 references with only 2 are posterior to 2018, not included, the only result was the addition of 1 pertinent study from 7 citations found in Google Scholar.
- C3, with a substantial 294 references, saw the exclusion of 14 ISO standards, and only 1 study met the inclusion criteria.

Table 2.1: Initial Candidates

ID	Author	Title	Year
C1	Zhao, Liping; Alhoshan, Waad; Ferrari, Alessio; Letsholo, Keletso J.; Ajagbe, Muideen A.; Chioasca, Erol-Valeriu; Batista-Navarro, Riza T.	Natural Language Processing for Requirements Engineering: A Systematic Mapping Study, Zhao et al. (2021)	2021
C2	Singh, Maninder	Using Natural Language Processing and Graph Mining to Explore Inter-Related Requirements in Software Artefacts, Singh (2019)	2019
C3	Perez-Cerrolaza, Jon; Abella, Jaume; Borg, Markus; Donzella, Carlo; Cerquides, Jesús; Cazorla, Francisco J.; Englund, Cristofer; Tauber, Markus; Nikolakopoulos, George; Flores, Jose Luis	Artificial Intelligence for Safety Critical Systems in Industrial and Transportation Domains: A Survey, Perez-Cerrolaza et al. (2023)	2023
C4	Ou, Andrew Yi-Zong; Rahmaniheris, Maryam; Jiang, Yu; Sha, Lui; Fu, Zhicheng; Ren, Shangping	Safetrace: ASafety-Driven Requirement Traceability Framework on Device Interaction Hazards for MD PnP, Ou et al. (2018)	2018
C5	Almendra, Camilo; Silva, Carla	Managing Assurance Information: A Solution Based on Issue Tracking Systems, Almendra and Silva (2020)	2020
C6	Haris, M Syauqi; Kurniawan, Tri Astoto	Automated Requirement Sentences Extraction from Software Requirement Specification Document, Haris and Kurniawan (2021)	2021

- C4, with references are mainly predating 2018, with 5 citations in Google Scholar, contributing to the consideration of 2 studies for inclusion.
- C5, with 24 references, generated 4 candidates for inclusion after excluding 14 pre-2018 references and those not aligned with the scope. The paper was cited twice, once unrelated to the research and once in a book.
- C6, among its 19 references mainly before 2018, yielded 1 study meeting the inclusion criteria. Cited 8 times, according to Google Scholar, contributing with 5 additions to the set.

Table 2.2: Snowball candidates

ID	Author	Title	Year
C1R1	Binkhonain M, Zhao L	A review of machine learning algorithms for identification and classification of non-functional requirements, Binkhonain and Zhao (2019)	2019
C1R2	Horkoff J, Aydemir FB, Cardoso E, Li T, Maté A, Paja E, Salnitri M, Piras L, Mylopoulos J, Giorgini P	Goal-oriented requirements engineering: an extended systematic mapping study, Horkoff et al. (2019)	2019
C1R3	Dalpiaz F, Ferrari A, Franch X, Palomares C	Natural Language Processing for Requirements Engineering: The Best Is Yet to Come, Dalpiaz et al. (2018)	2018
C1R4	Young T, Hazarika D, Poria S, Cambria E	Recent Trends in Deep Learning Based Natural Language Processing, Young et al. (2018)	2018
C1R5	Irshad M, Petersen K, Poulding S	A systematic literature review of software requirements reuse approaches, Irshad et al. (2018)	2018
C1C1	Moharil A, Sharma A	TABASCO: A transformer based contextualization toolkit, Moharil and Sharma (2023)	2023
C1C2	Gartner AE, Fay TA, Gohlich D	Fundamental Research on Detecting Contradictions in Requirements: Taxonomy and Semi-Automated Approach, Gärtner et al. (2022)	2022
C1C3	Sonbol R, Rebdawi G, Ghneim N	Learning software requirements syntax: An unsupervised approach to recognize templates, Sonbol et al. (2022a)	2022

Continued on next page

Table 2.2 – continued from previous page

ID	Author	Title	Year
C1C4	Levy Y, Stern R, Sturm A, Mordoch A, Bitan Y	An impact-driven approach to predict user stories instability, Levy et al. (2022)	2022
C1C5	Sonbol R, Rebdawi G, Ghneim N	The Use of NLP-Based Text Representation Techniques to Support Requirement Engineering Tasks: A Systematic Mapping Review, Sonbol et al. (2022b)	2022
C1C6	Li G, Zheng C, Li M, Wang H	Automatic Requirements Classification Based on Graph Attention Network, Li et al. (2022)	2022
C2C1	Pauzi Z, Capiluppi A	Applications of natural language processing in software traceability: A systematic mapping study, Pauzi and Capiluppi (2023)	2023
C3R1	Bencomo N, Guo JL, Harrison R, Heyn HM, Menzies T	The Secret to Better AI and Better Software (Is Requirements Engineering), Bencomo et al. (2022)	2022
C4C1	Horn D, Ali N, Hong JE	Towards Enhancement of Fault Traceability Among Multiple Hazard Analyses in Cyber-Physical Systems, Horn et al. (2019)	2019
C4C2	Yi-Zong OU	Safety-driven software design and engineering in medical cyber-physical-human systems, YI-ZONG (2019)	2019
C5R1	Agrawal A, Khoshmanesh S, Vierhauser M, Rahimi M, Cleland-Huang J, Lutz R	Leveraging Artifact Trees to Evolve and Reuse Safety Cases, Agrawal et al. (2019)	2019
C5R2	Heeager LT, Nielsen PA	A conceptual model of agile software development in a safety-critical context: A systematic literature review, Heeager and Nielsen (2018)	2018
C5R3	Steghofer JP, Knauss E, Horkoff J, Wohlrab R	Challenges of Scaled Agile for Safety-Critical Systems, Steghöfer et al. (2019)	2019

Continued on next page

Table 2.2 – continued from previous page

ID	Author	Title	Year
C5R4	Vilela J, Castro J, Martins LE, Gorschek T	Safe-RE: a safety requirements metamodel based on industry safety standards, Vilela et al. (2018)	2018
C6R1	Osman MH, Zaharin MF	Ambiguous software requirement specification detection: an automated approach, Osman and Zaharin (2018)	2018
C6C1	Delima R, Khabib Mustofa, Anny Kartika Sari	Automatic Requirements Engineering: Activities, Methods, Tools, and Domains–A Systematic Literature Review, Delima et al. (2023)	2023
C6C2	Grasler I, Preus D, Brandt L, Mohr M	Efficient Extraction of Technical Requirements Applying Data Augmentation, Grasler et al. (2022)	2022
C6C3	Antoniou C, Bassiliades N	A tool for requirements engineering using ontologies and boilerplates, Antniou and Bassiliades (2024)	2024
C6C4	Banerjee S, Dutta A, Layek S, Sahoo A, Joyce SC, Hazra R	Redefining Developer Assistance: Through Large Language Models in Software Ecosystem, Banerjee et al. (2023)	2023
C6C5	Antoniou C, Kravari K, Bassiliades N	Semantic Requirements Construction Using Ontologies and Boilerplates, Antoniou et al. (2023)	2023

2.3 Fundamental Concepts

Safety-critical systems differ from conventional systems in that their failures can have serious consequences. The highest criticality industrial systems must ensure an extremely low failure probability of up to 10^{-9} per hour of operation (approximately one failure in 114.155 years), IEC 61508 (-1/7). This imposes strict requirements to achieve independence of execution among integrated functions and reduce the probability of failure due to, e.g. random hardware errors and systematic errors (e.g. human, process and tool errors)(Perez-Cerrolaza et al. (2023)). Safety-critical systems are developed through explicit safety engineering process that includes specification of properties, verification and validation processes, and is based on evidence of defined and dependable processes, meaning also high development costs where, as a rule of thumb, “the higher the safety integrity level, the higher the cost of safety certification”.

Significant amount of research has analysed the impact of requirement changes on source code, but unfortunately, the impacts of requirement changes on safety analysis has yet to be addressed Ou et al. (2018).

In critical systems contexts, any modification or updates made, whether in requirements, design, or code, can bear significant implications for the overall functionality, safety, reliability, and adherence to regulatory standards. Change-Impact Analysis involves a evaluation of the potential consequences from modifications introduced to components within SCS. Through CIA, risks associated with the proposed changes are evaluated, especially regarding safety, reliability, and adherence to regulatory guidelines. CIA acts as a preemptive strategy, aiming to predict any adverse outcomes from modifications at any phase of the software development life cycle. This process typically encompasses several stages. Initially, it involves documenting the proposed changes, subsequently, the analysis phase assesses how these alterations might affect interconnected components, functionalities, safety constraints, or compliance with established standards.

2.4 Review of Existing Tools and Methodologies

In the first edition of natural language processing for requirements engineering workshop (NLP4RE), 2018, Dalpiaz et al. (2018) exposed diverse paths for innovation, including the **identification of quality defects and ambiguity, classification and clustering of extensive requirement collections, extraction of key abstractions, generation of models, and establishing traceability**. Despite the scarcity of public requirements data, the authors assert the necessity of NLP4RE tools to help organizations make sense, cross-reference and reuse large amounts of artifacts. The authors stress the role of **"Human-in-the-Loop"** asserting that NLP technologies are intended to empower experts rather than replace them.

In a systematic mapping study Zhao et al. (2021) identify a diverse array of 231 NLP technologies, encompassing 140 techniques, 66 tools, and 25 resources. The authors highlight that syntactic techniques (understand the relationships between words and phrases) dominate NLP4RE research, with POS tagging being the most frequently used technique, followed by tokenization, parsing, stop-words removal, term extraction, and stemming. The study further elucidates that Stanford CoreNLP is the most frequently utilized NLP tool, with WordNet being the most employed NLP resource.

The limitations of CIA and the manual nature of this process in industry are brought up by Zhao et al. (2021), citing the work of Arora et al. that uses NLP approach on software requirements for CIA. Additionally, Zhao highlights the a gap for research focusing on the development of an automated approach to detect change-impacted regions of software requirements.

The paper by Singh (2019) reviews the limitations of existing CIA processes in the context of software requirements and emphasizes the need for an automated approach to detect change-impacted regions of software requirements, and mentions that some researchers have tried to expand CIA to requirements phase using NLP-based approaches.

Key findings from Binkhonain and Zhao (2019) systematic review, focused on the identification and classification of non-functional requirements, indicate the overall success of ML-based approaches, achieving over 70% accuracy in detecting and classifying non-functional requirements. Notably, supervised learning algorithms outperform, with support vector machines and Naive Bayes demonstrating the best performance among supervised algorithms. The study identifies challenges, emphasizing the need for a clear articulation of pre-processing steps in requirements document analysis. Furthermore, an open challenge highlighted is the scarcity of shared training datasets, hindering the application of supervised learning algorithms due to the labor-intensive nature of dataset creation (classification labelling by humans), requiring the collection, classification, and validation of real-life non-functional requirements.

Ou et al. (2018) propose an automated trigger for a change-impact analysis through the SafeTrace framework, which provides a approach for managing traceability among safety requirements, design, and safety analysis. The impact-analysis algorithms aims to identify the effects of requirement and design changes on safety analysis by leveraging traceability graph and algorithms to detect how changes in artifacts might impact downstream the existing Fault-Tree Analysis.

Gärtner et al. (2022) explores identifying contradictions in requirement documents by Aristotelian Logic law on Non-contradiction, formalizing subtypes of contradictions using this theory. By demonstrating the method's simplicity and applicability, it offers a taxonomy for automated contradiction detection in requirements.

Traceability solutions need to be transparent, especially when traceability is a factor in homologation and audit. Recent works have resulted in an increasing number of black-box NLP services and tools. Despite huge successes in large language models, their blackbox nature hinders key goals of NLP, particularly in explainability. In cases where traceability plays an important role (such as adherence to regulations), the black-box nature of these advanced solutions proves as a drawback, as validation of the results becomes difficult. There is still a considerable amount of dependency on tacit knowledge that is integral to traceability solutions with NLP. This dependency is hampering efforts in automated effective traceability due to the limitations of models in every domain, also related to how the artifacts are represented. Moreover, there is a need to incorporate efforts in explainable AI and model reasoning to reduce bias and fill in the gap of dependencies on tacit knowledge from human experts.

2.4.1 Ontology / Domain specification

On another paper presenting a systematic mapping study focusing on NLP and its applications, in the context of software traceability, Pauzi and Capiluppi (2023). The study identifies key issues that are direct results of using NLP in the proposed traceability solutions. Traceability between artifacts stems from identifying components that are linked to one another. To achieve this, the declaration of concepts needs to be synchronized in terms of syntax and semantic similarities. The natural language present in artifacts needs to be represented uniformly in various parts of the SDLC, and achieving similarity in each of those representations is an open challenge that NLP continues to play a major part in solving.

In the context of domain specificity, there is a highlight in the need for NLP tools to adapt to the specific jargon, business rules, and process practices inherent in different domains (Dalpiaz et al. (2018)). Addressing the domain specificity requires the adoption of domain-specific and even company-specific ontology's, especially considering the challenge of eliciting tacit knowledge concealed in the minds of experts.

Semantic ambiguities in artifacts written in natural language pose a challenge in tracing explicit links with other artifacts, due to polysemy (a linguistic phenomenon where a single word or phrase has multiple related meanings), non-uniform identifiers, ambiguity in content and vocabulary mismatch (Pauzi and Capiluppi (2023)).

2.4.2 AI-Driven Approaches for Tracing and Reuse of Requirements

The research presented in the paper by Maninder Singh(Singh et al. (2018)) addresses a critical gap in the literature by combining semantic analysis and graph mining to enhance the understanding of inter related requirements and their implications for CIA. The authors propose a novel approach that leverages semantic analysis and graph mining. The primary focus is on exploring inter-relations in Software Requirements Specification documents. The paper emphasizes the importance of understanding CIA by considering the interrelated nature of requirements. The authors phase trough four crucial areas on their proposed solution: Semantic Analysis, NLP, Graph Mining for Software Requirements and CIA.

2.5 Discussion and Gaps in Research

The NLP4RE workshop (Dalpiaz et al. (2018)) evidences the gap in research related to the integration of NLP methods with RE tasks. The number of researches that use NLP-based representation is still limited for some RE tasks (requirements prioritization, effort estimation, semantic-based quality assessment) (Sonbol et al. (2022b)).

Zhao et al. (2021) discusses the research gaps in the field of NLP4RE. The article emphasizes the need for industrial case studies and open-science practices, such as sharing datasets and tools. Additionally, it highlights the lack of advanced NLP techniques for semantics and discourse analysis, indicating that NLP research has not yet been fully exploited.

Despite several solutions proposed for a wide range of RE phases and tasks, there is still a relevant gap in terms of industrial case studies. Only a small percentage of the studies were evaluated in an industrial setting, highlighting a general lack of industrial evaluation of NLP4RE research results. Also finds little evidence of industrial adoption of NLP4RE tools and a lack of shared RE-specific language resources. This suggests a gap in the practical adoption of NLP4RE solutions in industrial settings. Also, most of the NLP4RE solutions do not consider advanced NLP techniques for semantics and discourse analysis, indicating that NLP research has not yet been exploited to its full potential.

Building a domain specific word embedding models is notoriously a challenging task, especially with the little size of data available in the RE domain. Having more domain-specific

language models to represent words is considered one of the important research directions in future works. Most of the proposed embedding-based techniques represent requirements statements without giving attention to the special structure of a software requirement that consists of clear components (actor, action, data object, etc). Using syntax-aware statements embedding could be one of the possible solutions to have more domain specific representations that reflect the semantics of requirements in a better way (compared to traditional aggregation approaches) (Sonbol et al. (2022b)).

Singh et al. (2018) paper outlines the experiment design, with the initial results obtained using the Latent Semantic Analysis (LSA) algorithm. Future work is suggested involving the evaluation of the proposed approach using various semantic algorithms such as LSA, Latent Dirichlet Allocation (LDA), and Random Projections (RP). Additionally, the paper proposes a study to evaluate the ripple effect of CIA using N-hop path algorithms by applying interactive change fixations in the requirements.

2.6 State of the art conclusions

In this survey of the current landscape surrounding CIA activity in RE for SCS and their alignment with norms and standards, the review looked on the intersection of SCS, RE, and the current state of NLP tools and techniques. The synthesis of existing methodologies and tools, unveiled a spectrum of approaches and current interest in the matter. Yet, it concurrently revealed distinct gaps in industrial case studies and the adoption of NLP4RE tools in real-world scenarios. This emphasizes the need to bridge the breach between theoretical advancements and practical implementation, urging for empirical validation within industrial contexts.

Throughout this investigation, the potential of AI-driven approaches is clear. However, challenges such as the scarcity of shared datasets and the black-box nature of advanced NLP solutions pose significant obstacles, calling for collaborative efforts to enhance transparency and auditability for usage in SCS development.

A future trajectory involves a effort in refining domain-specific word embedding models, harnessing advanced NLP techniques, and fostering industrial adoption of NLP4RE tools. This path forward needs not only technological innovations but also the nurturing open-science practices, enabling a robust foundation for AI-driven solutions within SCS.

Chapter 3

Contextualizing Requirements Engineering in Safety-Critical Systems

Only with a thoughtful insight on the the problem, constrains, and its importance can lead to a considerate selection of the method and tools to be employed. Taking in the reflections of the SOTA, it is evident that there are ample number of studies that map techniques, tools and resources available in the NLP domain, al well as several identified gaps in research. This chapter aims to contextualize the relevance of the presented problem, supported with background information gather in the SOTA.

3.1 Requirements and sensitive information

The nature of systems operating in sectors such as healthcare, transportation and finance, are characterized for safety, security, and economic stability demands. SCS are built on, and operate with, highly sensitive information that performs functions vital for the safety of human life's, economic stability and ecological well being.

For instance, healthcare systems handle patient data and treatment protocols, aerospace systems control aircraft navigation and communication, and financial systems oversee transactions and risk assessments. Given the nature of these applications, **any changes to the system must be rigorously analyzed** to prevent potential malfunctions, operational failures or data breaches which could lead to severe consequences such as significant financial losses, security threats or loss of human life(s).

3.1.1 Vulnerabilities and exploit risks

The construction and operation of these systems is reflected in requirements databases. Unauthorized access to these documents could expose vulnerabilities, allowing malicious actors to exploit the system. A malevolent actor with access to the information on the systems architecture, design and functional requirements, can recreate the solution for industrial competition purposes or influence the control of a running system for leading to inadvertent outcomes. This is conclusive as

why SCS industries, such as healthcare and finance, have strict regulations around the handling of requirements.

The security of this highly **sensitive information** is ensured through a combination of data protection measures, such as strict access controls and secure storage. Access is given as necessary to allowed personnel only. To maintain the confidentiality and integrity, while keeping the availability required for development and assessment is a demanding balance.

Operating highly sensitive information over the internet to third parties poses significant risks as it exposes proprietary data, increases the attack surface and the number of potential entry points for unauthorized access. Also at stake is the liability of third party systems, if they are compromised, the data could be accessed.

Due to the security, control, and compliance needed to keep the confidentiality of requirements, and by the fact that SCS must adhere to rigorous data protection regulations (e.g. ISO/IEC 27001[1], ISO/IEC 27701[2], GDPR[3], HIPAA[4], CCPA[5]), organizations like to maintain control over their data. This explains why some industrial sectors seem predisposed to developing and operating their own machine learning solutions instead of using third-party services.

By developing their own AI solutions, organizations can ensure compliance with the regulations, avoiding potential legal penalties and safeguarding their intellectual property.

3.2 Limitation to usage of AI in safety critical systems

Currently there is no accepted framework for homologation of AI-based systems that can meet the necessary safety-critical certifications for compliance with the safety standards that regulate SCS development activities (specification, design, testing, verification, and validation).

The stochastic nature of outputs from models can make it challenging to interpret their reasoning. While they perform well in practice, understanding how they arrive at certain decisions is a challenge. The lack of predictability due to the "black box" nature of many AI models makes it difficult to fully understand and analyze their decision-making processes. This lack of transparency and comprehensibility is a major barrier to using AI in safety-critical applications where the construction behind the outcomes must be clearly understood, verified and validated.

Because the models process texts based on statistical features, they sometimes produce topics that merely reflect linguistic features in the text, topics may not be of any value, Chang et al. (2009).

The regulatory and standards landscape for the use of AI in safety-critical systems is still emerging, with new standards like SOTIF ISO 21448:2022 (E) with limited industry consolidation. The exception of regulatory advancements on autonomous driving, incorporating AI/ML in SCS is currently not a reality, as policy-making and standardization lagging behind the research.

[1] Information Security Management
[2] Privacy Information Management
[3] General Data Protection Regulation
[4] Health Insurance Portability and Accountability Act
[5] California Consumer Privacy Act

3.3 Enhancing requirements engineering with NLP tools

As per Sonbol et al. (2022b) the number of researches that use NLP-based representation is still limited for some RE tasks (requirements prioritization, effort estimation, semantic-based quality assessment).

The NLP4RE workshop, Dalpiaz et al. (2018), evidences the gap in research related to the integration of NLP tools in RE tasks. In the first edition of the workshop (NLP4RE) the authors emphasize the necessity of NLP4RE tools to assist organizations in understanding, cross-referencing, and reusing large volumes of artifacts as well as highlighting various paths for innovation, such as the classification and clustering of extensive requirement collections and establishment of traceability. They also mark the importance of the "Human-in-the-Loop" approach, arguing that NLP technologies are designed to augment the capabilities of experts rather than replace them.

3.3.1 Potential for AI/ML

Barriers to AI in SCS do not apply to the use in RE activities, as it does not involve direct control over critical systems outcomes without a ruling from human actors. While the route for integration of AI in safety-critical systems is paved with challenges, there is a clear potential for AI, specifically using NLP, to significantly improve RE activities. Analysis and automation with usage of NLP techniques can improve efficiency in RE activities, making it easier to analyse and manage large sets of requirements.

There is a fit for tools that support the human-in-the-loop approach, augmenting human performance and expertise rather than replacing it. By supporting analysts with automated insights, these tools can assist RE professionals to make more informed decisions, prioritizing work, and estimating development efforts more accurately.

The ability to think abstractly, consider ethical implications, and foresee potential long-term impacts are indisputable reasons why human insight is essential in navigating the uncertainties and complexities inherent in systems development. While AI can process vast amounts of data and identify patterns, the subtleties of human judgment are indispensable for interpreting these patterns in context, making informed decisions, and forming innovative solutions.

The application of AI in RE can enhance analysis and traceability as NLP tools can track the relationships between norms, requirements, design documents, test cases, and code, ensuring that changes in one area are consistently reflected across all related artifacts. This promises effective improvements in maintaining the integrity and coherence of the system throughout the development lifecycle.

While the direct application of AI in the operational aspects of safety-critical systems is far to see on the horizon, leveraging AI, particularly NLP, in the RE tasks is a promising path.

3.4 The Role of Change Impact Analysis

Safety-critical systems development is based on evidence of defined and dependable processes and outcomes, framed to regulated engineering processes that include specification, verification and validation processes, all associated to high development costs.

One of the most commonly used methods for SCS development is the Systems Development Life Cycle (SDLC), that provides a structured approach, governing the entire life-cycle from initial planning until maintenance. The V-Model, a graphical representation of SDLC, shows the complementary approach to validation and verification in each phase of development.

3.4.1 Mapping SCS through the V-Model

The V shaped model illustrates the relationship between each phase of development (left side) and its corresponding phase of testing (right side), each phase of development on the left side (such as requirements specification, system design, and implementation) having a corresponding testing phase on the right side (such as unit testing, integration testing, and system testing).

Abiding to the V-model systematically ensures that each development stage is verified and validated against its specifications, enhancing the reliability and safety of the system.

To forecast and meticulously plan each phase of development is also one of the goals by which organizations aim to achieve by structuring development through the V-Model, as one of the benefits of is to provide a clear structured method to systems development, reducing the probability of rework, helping in early detection and correction of defects, saving time and cost, with great emphases thorough documentation at each phase for future maintainability.

This also means that any modification or updates made, whether in requirements, design, or code, bear significant implications in the overall cycle.

3.4.2 Propagation of change through V-Cycle

A Change-Impact Analysis is a pre-evaluation of the consequences that will be introduced by a change in the system. Through CIA, effort and risks associated with the proposed changes can be evaluated. CIA acts as a precautionary protective effort, aiming to predict costs and any adverse outcomes from modifications, especially regarding safety, reliability, and adherence to regulatory guidelines.

Due to the multifaceted nature of these environments, efficiently performing CIA on complex systems requires knowledgeable, highly skilled and experienced expert judgment: understanding the intricate dependencies and interactions within the system, evaluating the potential ripple effects of proposed changes, and identifying areas that automated tools may overlook. Only experts in the field can bring a depth of experience, critical thinking, and domain-specific knowledge that allows them to foresee unintended consequences, assess risks accurately and make informed decisions that balance technical requirements with operational and normative constraints.

3.4.3 The gaps in change impact analysis automation

Both papers, Singh (2019) and Zhao et al. (2021), highlight the a gap for research focusing on the development of an automated approach to detect change-impacted regions of software requirements. Both reviews point to the limitations of existing CIA processes mention and the manual nature of this CIA in industry. Additionally, it as been noted by Ou et al. (2018) that the impacts of requirement changes on safety analysis has yet to be addressed.

The current manual approaches to CIA are resource-consuming, highlighting the need for research into automated solutions that can effectively bolster accuracy and efficiency. Further developments on this can not only improve the effectiveness of CIA but also ensure that safety-critical systems continue to operate safely and reliably.

3.4.4 Domain specific challenges

Certain terms in domain-specific texts may have multiple meanings or may exhibit biases inherent to that domain, which can affect the performance and fairness of data mining models. As the texts can have limited coverage, disambiguation these terms and biases correctly requires contextual understanding, careful consideration and domain expertise, which may not be readily available.

This lack of interpretability can hinder the effectiveness of data mining techniques and make it challenging to extract meaningful insights. The specialized vocabulary, nuances, and unique context affects the performance of data mining algorithms, particularly those that rely on statistical patterns or co-occurrence statistics.

Addressing the domain specificity requires the adoption of domain-specific and even company-specific ontology's, especially considering the challenge of eliciting tacit knowledge concealed in the minds of experts.

As stated by Dalpiaz et al. (2018), there is a gap in NLP resources to adapt to the specific jargon, business rules, and process practices inherent in different domains.

Overall, while transformer-based models (Devlin et al. (2019), Vaswani et al. (2017), Radford et al. (2019) Liu et al. (2019), Brown et al. (2020)) have pushed the boundaries of what is possible in NLP, there is still room for improvement. Bridging the gap between statistical performance and deep semantic understanding remains a challenge.

For maintaining SCS there is a considerable amount of dependency on tacit knowledge, integral not only to the system, as well as the area of domain. This dependency diminishes results in automation and effective traceability due to the limitations of models in specific domains, related also to how the artifacts are kept and represented.

Additionally, the reliance on tacit knowledge, which is deeply embedded in human expertise and experience, poses another backlash to fully automate traceability and impact analysis with current models to due to their probabilistic 'understanding' of text.

Related, another the significant challenges of NLP4RE is the need for domain-specific understanding and efficiency when handling industry-specific terminologies and jargon.

The research by Zhao et al. (2021) points out the limitations in current NLP models to fully exploit semantics (study of linguistic meaning) and discourse analysis (understanding language use beyond individual sentences). While models like Bidirectional Encoder Representations from Transformers (BERT, Devlin et al. (2019)) have achieved impressive results in various NLP tasks, including some aspects of discourse analysis through techniques such as next sentence prediction and contextual embedding, they often fall short in capturing the deeper, nuanced meanings.

Current benchmark solutions are trained on large, diverse corpora that may not fully capture the specialized language and intricate discourse patterns prevalent in a particular SCS as general-purpose measurements for the relatedness of text strings (embeddings) might not accurately reflect the semantics of requirements in specialized contexts. Fine-tuning models for specific domains is one path, but often requires substantial amounts of domain-specific training data, always with some risk of biased outcomes.

Due to limited data available, contextual nuances, complex terminology and Jargon, building a domain specific word embedding models is challenging task, especially with the little size of industrial data available. Domain-specific embeddings require high-quality, domain-relevant annotations. Creating these annotations often necessitates the expertise of subject matter experts, making the process time-consuming and expensive. Having more domain-specific language models to represent words is considered by Sonbol et al. (2022b) as research direction for the future.

3.4.4.1 Traceability challenges

The study by Pauzi and Capiluppi (2023) identifies key issues that are direct results of using NLP in the proposed traceability solutions. Traceability between artifacts stems from identifying components that are linked to one another. To achieve this, the declaration of concepts needs to be synchronized in terms of syntax and semantic similarities. The natural language present in artifacts needs to be represented uniformly in various parts of the SDLC, and achieving similarity in each of those representations is an challenge in NLP.

Semantic ambiguities in artifacts written in natural language also poses a challenge in tracing explicit links with other artifacts, due to polysemy (a linguistic phenomenon where a single word or phrase has multiple related meanings), non-uniform identifiers, ambiguity in content and vocabulary mismatch.

3.4.5 Syntax-aware embedding for rule-Based requirements

Combining syntax-aware statements embedding and rule based analysis could be one of the possible solutions to have more domain specific representations that reflect context specific semantics (compared to traditional aggregation approaches) as most of the proposed embedding-based techniques represent requirements statements without giving attention to the structure of a software requirement components (actor, action, data object, etc).

Purely relying on stochastic models might not be sufficient for all applications. Combining statistical models like BERT with rule-based approaches, like controlled natural language, boilerplate's or requirement templates could enhance the effectiveness in discourse analysis. Although, these rules are context, corporation or industry industry specific.

3.5 Industrial Relevance

Mapping existing methodologies and tools reveals a spectrum of approaches and current interest, yet it concurrently uncovers gaps in industrial case studies and the adoption of NLP tools in real-world scenarios. This emphasizes the pressing need to bridge the gap between theoretical advancements and practical implementation. While academic research has made significant strides in NLP4RE, the empirical validation of these techniques within industrial contexts remains limited.

The scarcity of shared RE-specific language resources poses a obstacle to development for industrial adoption. Despite the promising advancements of benchmark models, their application in industrial settings is constrained by the need for substantial domain-specific training data and fine-tuning to capture nuanced discourse patterns, as well as conflict with normative context and threats to intellectual property and system safety.

Studies also indicates that NLP research has not yet been fully exploited to its potential for NLP4RE. Advancing in techniques, such as syntax-aware embeddings and rule-based analysis, offer promising paths to improve the semantic understanding and contextual accuracy of requirements engineering tools.

The high development costs associated with SCS makes a compelling argument for the need for efficient CIA tools. By integrating these techniques, industrial applications can benefit from more precise and meaningful analysis, facilitating better estimation, decision-making and risk management during CIA tasks, ultimately contributing to competitive development of safer and more reliable systems.

Chapter 4

Latch Research Results to Methodological Framework

The following chapter contributes with a reflexion on researched NLP tools and techniques and how they can serve to address the described goals, keeping in mind the contextual conditioning for SCS development. Let's start by recalling the goals:

1. Supporting Change-Impact Analysis (CIA).
2. Tracing or eliciting safety-related aspects from existing norms and standards.
3. Facilitating effective analysis, tracing and reuse of change to prevent safety risks and normative violations.

Approaching a problem with a blueprint, a structured framework or reference helps logical progression through the different phases and helps maintain focus on the objectives. It also establishes an organization for information and enhances the clarity and coherence of the final results.

Requirements engineering is a systemic process. For large systems, it is common that the final requirement baseline does not end at the elicitation phase. Thinking on a automation tool for systematic analysis, it would start by considering all text as data, then retrieve meaning from the text in different business contexts, presenting this meaning to the user, and allowing the user to gather information to adapt their CIA. This interaction with the user can allows for further optimization and parameterization of the tool for better results. From the previous chapter we can elicit additional conditions and constrains:

- Requirements are mainly written in natural language. For the purpose of this dissertation we are focusing solely on natural language requirements. Language is the data to be analyzed for extraction of meaning.
- The language varies with context and industry. Benchmark models are not adapted to the nuances of context specific wording.

- Due to nuances of language and the normative ruling for SCS, there must be human validation of the outcome. A tool would serve to promote human efficiency, and, at the same time, human interaction can help to optimize the same tool.
- A complete CIA traces all impacted artifacts throughout the V-Model / SDLC.
- There is a lack of labeled datasets that can be used for the goal at hand. Manual labelling is laborious and time consuming. This means that supervised learning algorithms are not applicable to this problem.

4.1 CRISP-DM as a Structure

CRISP-DM (Cross-Industry Standard Process for Data Mining) was introduced by Wirth and Hipp (2000), it has been widely adopted in data mining projects due to its flexibility and structure. The model is designed to be applicable across different industries and technologies. It provides a generic framework for planning, communication, and documentation, and can be adapted to specific project needs. The generic model provides an excellent foundation for developing a specialized process model which prescribes the steps to be taken. Its makes the process repeatable, manageable and measurable.

Given this treatment of language tokens as data, we prospect how CRISP-DM can provides a structured approach to integrate NLP and CIA processes in the RE domain. By aligning with this method, the research ensures a systematic approach to developing and verifying a solution, facilitating a logical transition from theoretical exploration to practical implementation.

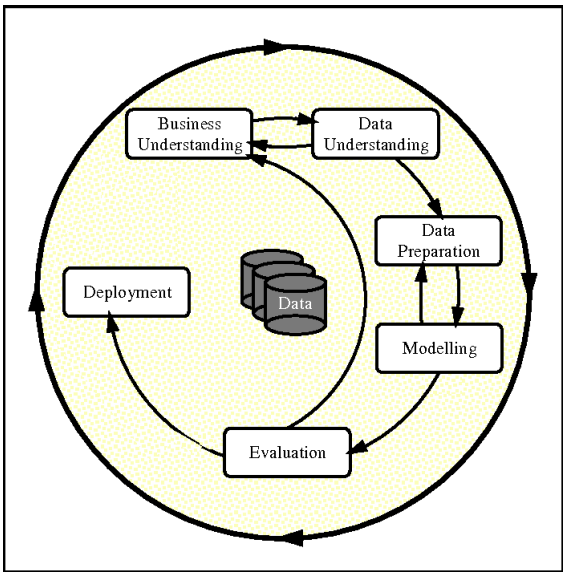

Figure 4.1: CRISP-DM scheme, Wirth and Hipp (2000)

Following, we will go through each phase of the process and align to the exposed problem.

4.2 Business Understanding

The primary goal of RE is to establish a clear and comprehensive understanding of what the business requires from a system; to define the necessary scope, functions, and constraints to achieve the system objectives.

As requirements evolve, due to stakeholder feedback or normative factors, change management process is central to assess CIA, predict costs and follow through to update required artifacts.

A change fragments into different artifacts and dependencies. A change analysis starts by considering the change against the requirements in place, then spreading to related objects, recurring to the traceability between the different levels of requirements or other artifacts, such as design documents, test cases, source code, norms, standards and certification documentation. RE methods and activities can be well defined, and if well kept, can be a solid basis to support a system change.

Identifying the various contexts and specific terminologies used within different organizations is crucial. Although, under a Business Understanding phase we are looking at the context of the system, which involves not only the system under development as well as organizational factors such as process and tools used. The surroundings establish the structure, process and way of working, domain language, acronyms and references used, as well as normative regulation for documentation that needs to be kept.

It is essential for the CIA tool to handle the specific needs and language of different business environments. A tool should be feed, recurrently, with relevant information to the field, and, it's output should fit in the organizational scheme of the system under development.

4.3 Data Understanding

Natural language techniques can extract a variety of valuable data from a set of documents, generating new attributes that can be useful for analysis and understanding.

Understanding and processing human language involves multiple layers of analysis. Sonbol et al. (2022b) study presents two dimensions of NLP tools, calling them the **syntactic** level, focused on the grammatical structure of sentences, and the **semantic** level, the meaning of the text. In addition, we identify a first level, the **morphological** (or lexical) level, that concerns with examining and processing the structure of individual words, focusing on their roots, prefixes, suffixes, and inflections.

For developing a effective and accurate solution, it's important to have a clear understanding of these levels of text analysis, the techniques and methodologies that can be employed to extract morphological, syntactic and semantic meaning from text. We identify key techniques listed in the three levels and briefly contextualized.

- **Morphological** (or lexical) level, involves analyzing and processing the structure of individual words, including their roots, prefixes, suffixes, and inflections.

  - Tokenizing: Splitting text into individual words or phrases, which serves as the foundational step for most NLP tasks.
  - Stemming: Reducing words to their root form by removing suffixes, which helps in grouping similar terms together.
  - Lemmatizing: Transforming words to their base or dictionary form, ensuring proper contextual meaning and reducing inflected forms.
  - Stopwords removal: Filtering out common words that carry little meaning, like "and," "the," or "is," to focus on more significant terms.
  - Regular expressions: Utilizing patterns to search, match, or manipulate text based on specific criteria, aiding in text preprocessing and data extraction.

- **Syntactic** level, focuses on analyzing the grammatical structure of sentences.
  - Parts-of-Speech (POS) tagging, the process of tagging each token in a sentence with its corresponding part of a speech tag (such as noun, verb, adjective, etc.) based on its syntactical context.
  - Chunking, the process of detecting syntactic constituents such as Noun Phrases and Verb Phrases in a sentence.
  - Dependency Parsing, the process of analyzing the syntactic structure of the sentence, by finding out the grammatically related words, as well as the type of the relationship between them.
  - Named-Entity Recognition, It seeks to locate and classify named entities mentioned in the sentence into predefined categories such as person names, organizations, locations, etc.
  - Key phrase extraction, involves automatically identifying and extracting the most relevant and significant phrases from a text

- The **Semantic** level focuses on understanding the meaning of the text. The main goal of semantic processing is to automatically map a natural language sentence into a formal representation of its meaning.
  - Ontology-Based Representation, a data model that represents a set of concepts within a domain and the relationships between those concepts. WordNet, a large lexical database of english nouns, verbs, adjectives and adverbs grouped into sets of cognitive synonyms, each expressing a distinct concept, is one of the widely used lexical ontologies. Used to assign words to a predefined set of concepts and to measure semantic similarity between them using different ontology-based measures.
  - Vector Space Model (VSM), basic representation model that represents text as a term-by-document matrix. The Bag-of-Words (BOW) model is a special case for VSM where words frequencies are used as weights, and words are used as features. Several weighting factors can be used in VSM, such as IDF and TF-IDF.

- Topic Modeling-Based Representation, statistical modeling approach used to discover the latent or abstract topics that occur in a set of texts. This approach helps in finding a approximation to the term-document matrix by retaining the semantic relations between words. Latent Dirichlet Allocation (LDA) and Latent Semantic Analysis (LSA) are two common examples of this approach.
- Advanced Embedding Techniques, an efficient method for learning high-quality vector representations of words from large amounts of unstructured text data. Word embedding can capture the context of a word within a document, which allows words with similar meanings to have similar vector representations. Several pretrained word embeddings are available to the public, such as Word2Vec, GloVe, BERT, etc.

4.4 Data Preparation and Collection

Data collection and preparation are primary steps that ensure the quality and integrity of the input data, setting the stage for successful project execution.

Data collection phase involves gathering raw data from the relevant sources for analysis. Data preparation, on the other hand, involves transforming raw data into a clean, consistent, and usable format. The goal is the acquisition of data and identifying attributes to ensure comprehensive coverage of the attributes and features required for the subsequent phases.

Data preparation also comprise transformation, converting the data into a suitable format, including normalization, encoding categorical variables, and, if needed, re scaling.

4.4.1 Data Integration

Often, data comes from multiple sources and may need to be integrated into a single dataset for analysis. Integration involves combining data from different sources while resolving schema conflicts and processing for consistency.

4.4.2 Data Transformation

Data transformation involves converting the data into a suitable format for mathematical analysis. This includes feature engineering, scaling, normalization, encoding categorical variables, and creating new derived features that enhance the predictive power of the dataset.

In some cases, not all variables in the dataset are relevant for modeling. Feature creation selection techniques can be applied to identify new variables that contribute significantly to the prediction task, sometimes reducing dimensionality.

4.5 Modeling

Modeling in text mining applications involves the use of algorithms and statistical methods to identify patterns, relationships, and structures within large collections of text data. The principles

of modeling in text mining for clustering techniques aim to group similar items based on certain criteria. Clustering is defined as the problem of finding groups of similar objects based on a similarity measure in a given dataset without possessing any previous information on the group memberships of these objects.

Assuming single document covers a set of concise topics based on the words used within the document, Topic modeling is one particular area of application of text mining techniques, that enables to identify these topics and assign them to documents (Rüdiger et al. (2022)). A topic links a group of similar documents, and the term "topic" corresponds to the term "cluster" in data mining.

4.5.1 Topic modeling

Topic modeling is a type of statistical modeling used in NLP to identify abstract topics that occur in a collection of documents. It is an unsupervised learning method that analyzes the words in the documents and clusters them into topics based on their co-occurrence patterns. **Each topic is represented by a distribution over words, and each document is represented by a distribution over topics.**

Topic models techniques enable users to categorize and summarize text collections at a scale that would be impossible using only human annotation. It extracts theme-level relations, enabling a succinct overview of themes in a document collection as the topics (or clusters) revealed through topic modeling should correlate well with human understanding of the texts Blei (2012).

Topic modeling algorithms usually output lists of the most representative words strongly associated with each topic, called topic descriptors. However, the term “topic” does not necessarily correspond to an intuitive understanding of the term, as the procedures captures texts based on statistical features, they sometimes produce topics that merely reflect linguistic features in the text. Thus, the topics themselves must be identified by manual pruning and labeling after applying the algorithm. For this reason, manual validation of the results is imperative.

Manual validation remains crucial to ensure the topics generated align with human understanding and provide meaningful insights. By doing so, there is a closed feedback on the results of the algorithm, similar to manual labelling, since topics can be used as features of the data.

Having sufficient number of validated topics, or labels, or a high degree of confidence in the topic modelling results, can open path to the usage of supervised algorithms.

Given the similarity with data mining and clustering, it is no surprise that the use of topic models comes with challenges typical for unsupervised learning methods. One major challenge is that most topic modeling algorithms requires to specify the number of clusters to be discovered. However, this number is not easily determined in advance.

Many scholars have proposed methods to extract topic structures from document collections, with Latent Dirichlet Allocation (LDA) being the most successfully cited.

4.5.1.1 Latent Semantic Analysis

Latent Semantic Analysis (LSA), as described by Landauer and Dumais Landauer et al. (1998), is a method for extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of text. The underlying idea is that the aggregate of all the word contexts in which a given word does and does not appear, provides a set of mutual constraints that largely determines the similarity of meaning of words and sets of words to each other

It operates as an unsupervised approach, particularly adept at revealing synonymous terms and capturing semantic nuances across a corpus by employing dimensionality reduction techniques, LSA uncovers latent relationships and similarities among documents and the terms they comprise.

4.5.1.2 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a generative probabilistic model for topic modeling, introduced by Blei, Ng, and Jordan in 2003 Blei et al. (2003). Unlike traditional methods that require predefined topic numbers, LDA assumes that documents are mixtures of topics and that each word's presence is attributable to one of the document's topics. The underlying premise is that documents exhibit multiple topics, and each topic exhibits a distribution of words.

LDA employs a probabilistic framework based on the Dirichlet distribution. This allows LDA to model uncertainty and variability in topic assignments. For each document in the corpus, LDA outputs a distribution of topics and for each topic, a distribution of words. This dual-level distribution provides insights into which topics are prevalent in which documents and which words are characteristic of each topic.

Sensitivity to parameters such as the number of topics and the Dirichlet priors can influence the quality of topic modeling results. Another downsiede is that LDA may produce topics that are not coherent or interpretable if the document corpus is noisy or if there are overlapping themes.

LDA has been extensively studied and applied across various domains, providing insights into the thematic structures of large text corpora. Despite its limitations, the ongoing development of advanced models and inference techniques continues to enhance its utility and applicability.

4.6 Evaluation

Metrics are indispensable as they provide a quantitative basis for assessing the quality, validity, and efficiency of the models. The goal of the evaluation is to assess the performance of the solution across different parameters and scales, measure the quality of the topics generated, ensure the model's robustness, and explore the Intelligibility and distinctiveness of the topics. To achieve this, a study and selection of evaluation metrics has to be considered.

A register of the configuration, parameters, and resulting metrics is needed in each run of model creation. This is essential for evaluating the quality of the models throughout their evolution. Ensuring that results on data are organized and accessible allows for the models to be

recreated. This, in turn, facilitates easy sharing and further exploration of model outcomes. Additionally, it enables comprehensive analysis and interpretation of results.

4.6.1 Unsupervised learning metrics

In general, evaluation metrics for assessing a topic model quality aim to determine whether the identified topics are coherent, relevant, and distinct, to ensure that the model is producing meaningful and understandable topics that align with human understanding. Evaluating topic modeling solutions is necessary to determine how well the model captures the topics in a corpus. There are several metrics and methods used for this evaluation, the challenge is the choice of suitable metric.

Unsupervised learning tasks, such as clustering, have different goals compared to supervised tasks. In supervised learning, performance metrics like accuracy, precision, and recall are calculated based on the comparison between predicted labels and true labels.

As for example, Accuracy, a common metric for evaluating machine learning models, particularly used in classification tasks, refers to how well the algorithm has grouped similar data points together (and separated dissimilar ones) by measuring the proportion of correctly predicted instances out of the total. Unsupervised learning lacks a ground truth, making it necessary to evaluate the model based on other criteria.

Topic modeling falls under the category of unsupervised learning because it does not require labeled data. For a broader research in metrics that can be used for evaluation, there is a need to fit topic modeling in more general machine learning tiers, hence we have correlated with eh term cluster, used in data mining (Rüdiger et al. (2022)), since a "topic" links groups of similar documents.

Clustering is a machine learning technique used to group a set of objects in the same group (or cluster). It is primarily associated with unsupervised learning because it does not require labeled data, instead, clustering algorithms identify inherent structures in the data based on similarity or distance measures. Clustering algorithms do not provide direct associations between true cluster labels and predicted cluster labels, as opposed to the accuracy metric, which relies on matching true and predicted labels, which could only be applicable if there is a manual effort to match documents to topics.

While clustering is fundamentally an unsupervised learning technique, it can sometimes be used in conjunction with supervised learning algorithms. For instance, clustering can be used as a preprocessing step to identify groups in the data, which can then be labeled and used to train a supervised learning model. This hybrid approach leverages the strengths of both unsupervised and supervised methods.

Evaluation metrics can provide benchmarks during the evolution of the tool. Metrics provide insights into how well the unsupervised learning method has captured the underlying structure of the data. Without proper metrics, the development of the tool can be guided to sub-optimal results. For best result we shall look at different metrics, and how they can highlight the strengths and weaknesses of the tool.

Metrics can be categorized into **internal** and **external** metrics based on whether they rely on information intrinsic to the model and data or on external references and reasoning. Internal metrics are useful for evaluating the technical performance of the model, while external metrics are more concerned with the practical relevance and usefulness of the topics generated.

4.6.2 External validation metrics

External metrics, are used to assess the model's ability to align with human understanding and to perform well in downstream applications. They depend on external references for evaluation, as they include methods like human labelling and comparison to external ground truth, sometimes referring to a manually-obtained labelling of clusters. Obviously, such a manual labelling is based on human perceptions and depends on the expertise of the raters. This option is known as external cluster validity. In an exploratory scenario, it is usually not available, as no groundtruth is present (Rüdiger et al. (2022)). Examples of external metrics include:

- **Semantic Coherence**: Evaluates how well the topics match human judgments of semantic relatedness.
- **Held-out Likelihood**: Measures the likelihood of held-out documents, with higher likelihood indicating better generalization.
- **Downstream Task Performance**: Evaluates how well the topic model performs on extrinsic tasks like text classification or retrieval.
- **Human Evaluation**: Directly assesses the comprehension and usefulness of the topics through human ratings.

4.6.3 Internal validation metrics

Internal validation metrics consider structural aspects and do not rely on any additional information related to the input data (Rüdiger et al. (2022)), meaning that they do not require external information as they use data and model internal properties.

These metrics are typically used to assess the model's ability to capture the underlying structure of the data by means such as measuring cluster cohesion and separation, on the statistical analysis of the proximity matrix (Palacio-Niño and Berzal (2019)).

Examples of internal metrics include:

- **Perplexity**: Measures how well a probabilistic model predicts a new sample of data, with lower perplexity indicating better model fit.
- **Topic Distance**: Measures the cosine distance between topics vectors, with higher distance indicating more distinct topics.

- **Symmetric KL-Divergence**: The Kullback-Leibler divergence between topics, measures how one probability distribution diverges from a second reference probability distribution, with lower divergence indicating more distinct topics.
- **C_UCI, C_UMASS, C_NPMI**: Word-based topic coherence metrics that measure the semantic coherence of the top words in each topic
- **Topic Divergence**: Word-based topic coherence metric, related to the dissimilarity or divergence between topics, specifically focuses on the differences between the topics inferred from a corpus of text.
- **C_V, C_W2V**: More advanced word-based topic semantic coherence metrics that incorporate word embedding to evaluate the quality of the topics.

While automated metrics like perplexity and coherence are useful for quick assessments, human judgment remains crucial for evaluating understanding and meaningfulness of the topics.

The following subsections detail the internal metrics considered in this study.

4.6.3.1 Perplexity

Perplexity measures how well a probability distribution or model predicts a sample. In topic modeling, it is measured on a held-out set of documents, which are not used during the training of the model. This provides an unbiased estimate of the model's performance on unseen data. It is calculated as the negative log-likelihood of the model given the sample. A lower perplexity score indicates that the model is more accurate in predicting the held-out documents.

Perplexity may not always correlate well with human judgment of topic quality.

4.6.3.2 Topic Distance

Topic distance is a measurement of the cosine distance between topics, with higher distance indicating more distinct topics. Topic distance is calculated as the complement of the cosine similarity with respect to 1.

The cosine similarity, measure used to assess the similarity between documents, texts, or any other data that can be represented as vectors, measures the cosine of the angle between two non-zero vectors in an n-dimensional space. The cosine similarity score ranges from -1 to 1, where a score of 1 indicates that the vectors are identical, a score of 0 indicates that the vectors are orthogonal (i.e., they have no similarity), and a score of -1 indicates that the vectors are diametrically opposed. It is particularly useful because it is independent of the magnitude of the vectors, focusing solely on their orientation.

To know the topic distance, first we need to represent each document as a vector in the topic space. This means that for each document, an empty vector with a length equal to the number of topics in the LDA model is initialized where it shall be assigned, to the corresponding positions in the dense vector based on the topic index, a value indicating the calculated value of the document

attributed to each topic. This vector represents the probability distribution of topics for that document. Once all dense vector, representing a document's topic distribution, have been processed, we can treat it as an array to facilitate numerical manipulations.

Then the cosine similarity between all pairs of document vectors are computed using the cosine similarity function from the sklearn module. To calculate cosine similarity, the function first computes the scalar product of the two vectors, which is a measure of their combined length. Then, it divides this scalar product by the product of the vectors' magnitudes (i.e., their Euclidean norms). This normalization step ensures that the resulting similarity score is bounded between -1 and 1. The resulting cosine similarity values are stored in a matrix, where each entry represents the cosine similarity between the topic distributions of a pair of documents.

To know the topic distance, the similarity matrix is converted to a distance matrix by subtracting the distribution value from one. Topic distance is calculated as the complement of the cosine similarity with respect to 1: distance matrix = 1 - similarity matrix. This operation is used to convert similarity measures to distance measures, as it inversely scales the values.

Topic distance can be used to evaluate an LDA model by assessing how distinct and meaningful the topics generated by the model are. By calculating the distance between document-topic distributions, one can evaluate how well the LDA model differentiates between documents. If similar documents have similar topic distributions and dissimilar documents have distinct topic distributions, the model is likely performing well.

There are other ways how topic distance can be used to evaluate an LDA model such as creating inter-topic distance maps. Inter-topic distance maps help to visualize the distance between topics. This can be done using tools like t-SNE (t-distributed Stochastic Neighbor Embedding) or multidimensional scaling to project high-dimensional topic vectors into a 2D or 3D space. A well-separated set of topics indicates that the LDA model has generated distinct topics.

4.6.3.3 Average Topic Coherence

Topic coherence measures the semantic similarity between the high-probability words in a topic. By computing the pairwise distances (e.g., cosine distance) between the word vectors in a topic, you can assess how coherent each topic is.

Topics with high coherence scores indicate that the words within the topic are closely related in meaning, which suggests a well-performing LDA model.

For a set of topics generated by an algorithm, Average Topic Coherence is computed by averaging the coherence scores of all individual topics. A higher average coherence score across topics leads to reason that the topics are well-defined and coherent.

Average Topic Coherence can be used to compare different topic models or tuning parameters within the same model. It helps selecting the optimal number of topics and evaluating the overall results, thereby improving the utility and reliability of the generated topics in practical applications.

4.6.3.4 Intra-topic Distance

Intra-topic distance refers to the measure of similarity or cohesion within a topic. It assesses how closely related the documents within a single topic are to each other. A lower intra-topic distance indicates that the documents or words assigned to a particular topic are more similar to each other in terms of their content, meaning, or thematic focus. This suggests that the topic is well-defined and represents a coherent theme or concept.

To quantify intra-topic distance, can be done by examining Word Distribution, the distribution of words or terms within a topic can reveal how concentrated or spread out the terms are. A more concentrated distribution indicates lower intra-topic distance, as the terms are more closely related.

Another way to quantify intra-topic distance is by Semantic Similarity, Analyzing the semantic similarity between words or terms within a topic using techniques such as cosine similarity or other distance metrics.

In practical terms, understanding intra-topic distance helps in assessing the quality of topics generated by topic modeling algorithms. Topics with lower intra-topic distance are, in principle, more meaningful. Researchers and practitioners use intra-topic distance as a criterion for selecting the optimal number of topics and improving the overall quality and coherence of topic modeling outputs.

4.6.4 Other Methods

This section outlines additional methods that could be explored to improve the evaluation and quality of the topic modeling solution. Considering these metrics and approaches, can give a more comprehensive understanding of the model's performance.

4.6.4.1 Symmetric KL-Divergence

In the context of topic modeling, Symmetric KL-Divergence measures how distinct the topics are by comparing their probability distributions. Symmetric KL-Divergence is an extension of the Kullback-Leibler divergence that addresses its asymmetry by averaging the divergence between two distributions in both directions. This metric helps in evaluating the uniqueness of each topic, ensuring that the topics generated by the model do not overlap significantly. Lower values of Symmetric KL-Divergence indicate that the topics are well-separated and distinct, which is desirable for a coherent topic model.

4.6.4.2 Word-Based Topic Coherence Metrics

Word-based topic coherence metrics assess the quality of the topics by examining the co-occurrence patterns of the top words within each topic. These metrics are crucial for ensuring that the topics are understandable and meaningful from a human perspective. Some notable word-based coherence metrics include:

- **C_UCI**: This metric uses pointwise mutual information (PMI) to evaluate the coherence of word pairs within a topic. It focuses on the statistical co-occurrence of words, ensuring that words within a topic frequently appear together in the corpus.
- **C_UMASS**: Based on document co-occurrence counts, C_UMASS measures coherence by considering the conditional probabilities of word pairs. It relies on the raw co-occurrence frequencies of words in the corpus, providing a straightforward evaluation of topic coherence.
- **C_NPMI**: Normalized Pointwise Mutual Information (NPMI) adjusts PMI to account for the frequency of words and their co-occurrences. This metric normalizes the co-occurrence scores, making it more robust and reliable for evaluating topic coherence across different datasets.

4.6.4.3 Advanced Topic Coherence Metrics with Embeddings

Recent advances in natural language processing have introduced word embeddings as a powerful tool for capturing semantic relationships between words. Advanced topic coherence metrics leverage word embeddings to provide a deeper evaluation of topic quality:

- **C_V**: Combines features of NPMI and cosine similarity, using word embeddings to measure the semantic similarity between top words in a topic. This metric offers a robust evaluation by considering both statistical co-occurrence and semantic relationships.
- **C_W2V**: Utilizes word2vec embeddings to assess the coherence of topics. By capturing the contextual similarities between words, C_W2V provides a more nuanced understanding of how coherent and semantically related the words within a topic are.

These advanced metrics can be useful for ensuring that the topics generated by the model are not only statistically coherent but also semantically meaningful.

4.7 Deployment

Implementation details may vary depending on the use case, the complexity of the model, and the deployment environment. Several packages provide built-in methods for saving and loading models, meaning that a trained model can be saved to disk for later use in inference.

The system will be developed and evaluated locally. Since there is no direct industrial application, **deployment will not be necessary, and this phase is disregarded**.

When determining the deployment environment, such as a web application, a standalone script, or an API, ensure the necessary dependencies are installed. It is crucial to monitor the performance of the deployed solution and update or retrain the model as necessary to maintain its effectiveness over time.

Chapter 5

Technical Architecture and Workflow

The proposed solution in this study is implemented in Python, utilizing Visual Studio Code as the integrated development environment. The entire codebase, along with supporting documentation is stored in a dedicated GitHub repository [1]. This repository not only serves as a version control system but also ensures that the project is accessible, shareable, and collaboratively maintainable.

Resource selection

The choice of the "best" algorithm depend on the specific use case and dataset. Taking the considerations from the previous chapter, the thoughts on the solution are centred on topic modeling. For this purpose **gensim** library was chosen.

Gensim, Context and Description

Gensim is a python based NLP open-source library for unsupervised topic modeling using modern statistical ML techniques.

Gensim is widely used for extracting topics from large text databases, applicable in document classification, information retrieval, and content recommendation. The library supports several topic modeling techniques, including Latent Dirichlet Allocation, Latent Semantic Indexing, and Hierarchical Dirichlet Process.

From on the systemic reviews done by Zhao et al. (2021) and Binkhonain and Zhao (2019), there is no definitive evidence that Gensim's LDA implementation performs better than other LDA algorithms. However, the Gensim LDA implementation is highly scalable, robust, and well-optimized for performance. It processes data one text at a time, allowing it to handle datasets larger than the available memory. Gensim also supports distributed computing, leveraging multiple cores or cluster environments for parallel processing. The recent addition of the 'LdaMulticore' class allows it to leverage multiple CPU cores to speed up training, which can significantly reduce the time required for large corpora.

[1] https://github.com/lfpneto/NLP2RE_Sandbox

There are no clear, comprehensive studies cited in the search results that definitively prove Gensim LDA outperforms other LDA algorithms across the board. The performance is likely to be dataset and use-case dependent.

The competitive factor seems to be more on optimizing the Gensim LDA implementation itself rather than direct benchmarking against alternatives, highlight the importance of factors like data preprocessing, hyperparameter tuning, and evaluation metrics when assessing the performance of topic modeling algorithms like LDA.

An understanding of the following components is relavant to describe and effectively utilize Gensim.

Document: one unit of text. In Gensim, a "document" refers to a single piece of text data that is part of a larger collection (or corpus). This could be anything from a sentence, a paragraph, an article, a research paper, or any unit of text you want to analyze.

Each document is treated as an individual entity for processing and analysis. Typically parsed as a string of text, it needs to be tokenized and processed previous to further analysis.

Dictionary: token to numeric ID. A Dictionary in Gensim is a mapping between words (or tokens) and their unique integer IDs. Each unique token (string) in the corpus (collection of documents) is assigned an ID, for efficient processing and storage.

It is essentially a lookup table that converts each word in the corpus to a unique identifier.The dictionary is built from the entire corpus and helps creating vector representations of documents.

A Dictionary is not fixed. At any point, for a given solution, more tokens can be added, or removed. This means that as a solution scales, with input from more documents, the dictionary adapts by attributing new id's to previously unknown words.

The dictionary object can also filter out tokens based on frequency, either removing extremely rare words or very common words that do not carry meaningful information.

Bag-of-Words: vector space representation.

The Bag-of-Words (BoW) model represents each document as a vector of word counts. It ignores grammar and word order, focusing instead on the frequency of each word within a document.

BoW simplifies the text data to numerical vectors, making it suitable for machine learning algorithms. Each document is represented as a sparse vector where each element corresponds to a word's ID (from the dictionary) and its frequency in the document. It serves as a foundational representation for more advanced models like LDA, TF-IDF, and Word2Vec.

5.1 Jupyter Notebooks

Throughout the investigation of methods and techniques, their applications, and the comparison of different methods available, Jupyter Notebooks was used for experimentation and analysis.

Jupyter Notebooks is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and markdown text. It is widely used for data cleaning and transformation, statistical modeling, data visualization and machine learning.

The notebooks created during this study document the step-by-step processes and findings, providing a transparent and reproducible workflow. Each notebook contains the code, results, and commentary that illustrate the techniques applied and the conclusions drawn from various experiments. This approach facilitates understanding, documentation and further development.

All Jupyter Notebooks used in this investigation are stored in the GitHub repository.

5.2 Dataset

To develop a practical solution a dataset is needed. The raw data was gathered from PURE (PUblic REquirements dataset), a dataset of 79 publicly available natural language requirements documents collected from the Web Ferrari et al. (2017).

The dataset gathers 34,268 sentences to be used for natural language processing tasks that are typical in requirements engineering. It also presents the requirements collection formatted in the common XML format, with the goal of facilitating replication of NLP experiments. All XML files share a common namespace, used for providing uniquely named elements and attributes in an XML document, simplifying the development of a function to parse the XML files.

For development, the author focused on 3 requirement documents from the railway domain: - UIC Project EIRENE System Requirements Specification, Version 1, 2006 - UIC Project EIRENE Functional Requirements Specification, version 7, 2006 - ERTMS/ETCS Functional Requirements Specification, version 5, 2007-06-21

Focusing on one domain ensures that problems related to tracing, ambiguities, incompleteness or overlapping of requirements can be unveiled.

Some steps had to be taken to prepare the raw dataset for usage. The document "2006-eirene_sys_15.xml", containing the UIC Project EIRENE System Requirements Specification, was not correctly formatted for the namespace in the set: the requirements tag was not placed in the requirements text. The document was corrected by usage of regular expressions.

5.2.1 Data Integration

Data collection phase involves gathering the raw data from sources. Considering one requirements document (one .XML file) as an artifact. An artifacts class points to where all artifacts are kept. It keeps a collection of all XML, instantiating for each a artifact objetc.

To consolidate the different sources into a cohesive dataset, when instantiated, a artifact parses the text to a pandas dataframe, where each information, paragraph or requirement (marked by its associated tag), is processed as a unique document (in the gensim paradigm). Each document will be treated as a row, meaning that each individual tag will be treated as a distinct document.

The parsing function is further developed to include, in the dataframe, attributes for the relevant XML tags, indicating whether the text corresponds to a requirement or information, the XML tree of the entry (or path) for debbuging purposes and the associated requirement ID.

Future processed information regarding the text attribute are stored as a new attribute, as, for example, list of tokens retrieved from the text.

Table 5.1: Attributes in Parsed XML Dataframe

Attribute	Description
Text	The actual content extracted from the XML document, categorized as either a requirement or general information.
Tag	Indicates whether the text corresponds to a requirement or general information.
Path	The XML tree path or entry from which the text was extracted. This provides the hierarchical location within the XML structure.
Requirement ID	Unique identifier associated with each requirement entry, facilitating traceability and organization.

5.3 Data Preparation

Proper text data preparation is a prerequisite for applying advanced techniques like sentiment analysis, topic modeling, and text classification.

Raw text data is not in a format that is processable. The preparation step aims structure of the text data, making it "machine-readable". This involves cleaning the text data by removing noise such as stop words, punctuation, and other terms that do not carry much meaning.

5.3.0.1 Tokenization

Splitting text into smaller units called tokens, Tokenization, is a fundamental step in the data preparation process for NLP tasks as it identifies the linguistic units to be processed as data.

Each level of tokenization (Word-Level, Subword-Level, Character-Level Tokenization) has its own use cases and benefits, depending on the complexity and nature of the NLP task at hand.

While character-level tokens can be useful for tasks like language modeling, it is not particularly useful for topic modeling operations because it breaks text into individual characters rather than meaningful units of language.

Topic models like Latent Dirichlet Allocation (LDA) rely on the co-occurrence of words to discover latent themes in a corpus. However, when using character-level tokens, the model loses the semantic meaning and relationships between words. For example, the words "dog", "dogs", and "doggy" would be chopped and treated as completely unrelated data, even though they are clearly related and likely to appear in similar topics.

5.3.0.2 N-grams

N-grams are grouped sequences of n items in a given sample of text, where the "n" denotes the number of items in the sequence. For instance, in the context of word analysis, a unigram (1-gram) is a single word, a bigram (2-gram) is a sequence of two consecutive words, and a trigram (3-gram) consists of three consecutive words.

By considering sequences of tokens rather than individual words, n-grams capture more contextual information, hence its usage in various natural language processing tasks such as text prediction, machine translation, and sentiment analysis.

When dealing with domain-specific terminologies or technical language, processing tokens as n-grams can foster better results, as they capture the context and co-occurrence patterns of words. In technical texts, many terms are composed of multiple words that together convey a specific meaning (e.g., "machine learning", "neural network"). Treating these multi-word expressions as n-grams can provide the model with a way to compute them as cohesive units, improving the accuracy of semantic understanding.

When used as features in machine learning models, n-grams can capture more detailed patterns in the data. For instance, bigrams (2-grams) and trigrams (3-grams) can encode dependencies between adjacent and nearby words respectively. Technical jargon often includes terms that have multiple meanings depending on context. N-grams help in disambiguation such terms by incorporating surrounding words into the representation providing richer linguistic context.

However, it's important to consider that using n-grams can significantly increase the number of features in the dataset, which may require more computational resources and usage of dimensionality reduction techniques. Rare n-grams may not generalize well or contribute meaningfully to the model's performance. Techniques like TF-IDF (Term Frequency-Inverse Document Frequency) weighting can help mitigate this issue by emphasizing important n-grams.

While Gensim's LDA model does not natively support n-grams, it is possible to preprocess the text to generate n-grams and then use those as input, adding them to te dictionary as unique tokens.

5.3.1 Preprocessing function

Text prepossessing means to refine raw text from the artifact dataframe at input, performing operations to clean and standardize the text, then stored as a different attribute in the dataframe.

These operations include removing common words that don't carry much meaning (stopwords), and reducing words to their base form (stemming or lemmatization). Overall, the function serves as a initial step in preparing textual data for deeper analysis and interpretation.

5.3.1.1 Stopword removal

A Stopword is a word with low discrimination power. And it is only used for connecting high discriminating power words while building sentences. For example, in English language 'a', 'an', 'the', 'and', etc. They have no meaning or any predictive capability. Stopwords can be broadly categorized as either: a **Generic stopwords**, that cover frequent words which are language-specific, or **Domain-Specific stopwords**, taking as example the Education domain: 'pen', 'table', 'student', 'book, 'classroom', 'board', etc.

They have a high frequency of occurrence in a text, can account for 40-50% of a corpus. Removing stopwords reduces the corpus size, thus helping in reduction of time and space complexity of the overall application.

The techniques for stopword Identification can be grouped as **static or dynamic** based. In a **static** approach, a predefined stopword list is created for the particular language. The list itself

would not need to be refreshed or changed automatically. In a **dynamic** approach, the stopword is identified on the go. A stopword list will be generated as per rules or statistics and it is not fixed apriorly. The list of stopword is decided based on given text source.

Removing stopwords is done in majority of NLP Pre-processing applications, including Information Retrieval and Text Classification, although, it is not an good idea for performing tasks such as a translation, language modelling, text summarization and question-answering problems and sentiment analysis. For example, in sentiment analysis if we consider “not” as a stopword, then removing it might lead to misinterpretations(Ladani and Desai (2020)).

Given the nature of requirements, words like "system", "user" or "shall" have a significant frequency, we consider that they can be omitted in text classification tasks. These words can be considered stopwords, yet they will not be fitered in a static stopword scan.

Schofield et al. (2017) argues that the improvement from removing stopwords is superficial. Beyond very frequent terms, stopword removal does not improve a model’s ability to learn topics over the other terms. As topic interpretability is typically judged by the most frequent terms in the topic, post-hoc stopword removal from a model can substantially increase understanding without modifying the model. The author argues that beyond high probability terms, the effects of stoplists on training are limited, and that removing unwanted terms after training should be sufficient.

The result from simple Tokenization can be words, numbers, or punctuation marks. **Caution is necessary when removing the stop words and tokenizing in the context of norms and standards.** Taking as e.g., the particular standards in the european aerospace sector for safety approach, ECSS-Q-ST-40 (Safety). Improper usage of tokenization and stopword removal might return a set such as: ["ECSS","Q","ST","40","30"]. Individually, these tokens do not have any meaning, while using them as in context will add semantic value to the model.

Generating a corpus-specific stoplist to remove rarer contentless words provides relatively little utility to training a model. This justifies why we consider in our solution to evaluate both implementing and not any pre-model training stopword removal technique.

5.3.1.2 Lemmatization and Stemming

Lemmatization and stemming are both natural language processing techniques used to reduce words to their base or root form. The main difference between a stem and a lemma lies in their definitions and applications in linguistic processing.

Lemmatization is a NLP task that involves a string-to-string transduction from an inflected word form to its citation form, known as the lemma (Malaviya et al. (2019)). Lemmatization examines the language and morphology of the words, while **stemming** targets to remove inflectional forms only and to return the root.

A **stem** is the root form of a word obtained by removing prefixes or suffixes. It may not necessarily be a valid word. One example, for the word "running," the stem is "run."

A **lemma** is the base or dictionary form of a word, typically a valid word. It represents the canonical form from which inflected forms can be generated. For example, the word "better," the lemma is "good."

For example, words like gone, goes, going will map to the stem "go". The word "went" will not map to the same stem, but the lemmatizing process will convert the word "went" too to the lemma "go". For instance, talk may also appear as talks, talked or talking, depending on the context.

Lemmatization is a more intensive technique as it takes into account the context and meaning of words. There are a lot of similarity between stemming algorithms and if one algorithm scores better in one area, the other does better in some other area, as per stated by Jivani (2011) and Singh and Pateriya (2015). In fact, none of them is 100% satisfactory useful to the text mining, NLP or information retrieval applications.

The main variance lies in using either a rule-based approach or a linguistic one. A rule based tactic may not always give correct output and the stems produced may not at all times be correct words. **As the linguistic approach is concerned, since these approaches are based on a lexicon, words external to the lexicon are not stemmed properly**. It is of highest relevance that the lexicon being used is robust, which is concern of language study.

A statistical stemmer may be language independent it does not give a trustworthy and stem every time. According to Jivani (2011), Porters Stemmer Produces the best output as compared to other stemmers. As a drawback, the stems produced are not always real words.

There are two types of errors in stemming. When words belonging to different roots are stemmed to same root is called **Over-stemming**, ex: design, designate->design. When words belonging to same root are converted to different roots is called **Understemming**, ex: foot->foot, feet ->feet. The problem of over stemming and under stemming can be reduced if the composition as well as the POS of the sentence is taken into consideration. Combined with a dictionary look-up can support in decreasing the errors and converting stems to words.

It does not seems apparent that lemmitization is more suitable for the goal at hand. For purposes of evaluation of different results, both techniques can be considered.

5.3.2 Dictionary

For NLP tasks numeric representation are achieved by converting text tokens, each associated with a unique numeric ID. This is what is called creating a dictionary.

In the NLP context, a dictionary encapsulates the mapping between tokens and their integer IDs. This dictionary is constructed from all the documents contained in the artifacts under consideration and is managed in the artifacts class, one dictionary for the collection of artifacts.

This dictionary is then used to vectorize the artifacts and any future inputs for analysis, ensuring consistency in the numeric representation of the textual data across the entire dataset.

5.3.3 BOW

The Bag-of-Words (BoW) is a list of tuples, each containing a token ID and its corresponding token count. Once the vocabulary is established, a sparse matrix is created based on the frequency of the vocabulary tokens in the artifacts. In this sparse matrix, each row represents a document vector, with the length (number of columns) equal to the size of the vocabulary.

The BoW representation captures the frequency of tokens within a document or a collection of documents. This representation is then stored in the respective artifact instance for further processing and analysis.

5.4 Modeling

As stated, due to the lack of labeled data, one constrain for the solution is to use unsupervised learning models. These are designed to learn from datasets where the input data does not have associated labels. The models aim to discover patterns, relationships, or groupings within the data without any prior knowledge.

Examples include clustering algorithms and dimensionality reduction techniques. Clustering is the process of partitioning a dataset into distinct groups, or clusters, where data points within the same cluster are more similar to each other than to those in other clusters.

For uncovering hidden patterns and structures in data, considering similarity between related requirements texts, we consider topic modeling as a sub type of clustering model. A Topic modeling can be described as a probabilistic model that contains information about topics in a text.

A Topic, as the name implies, is a underlying idea or theme represented in a text. Topics can be created by a probabilistic distribution of words. That's how, using topic models, one can describe documents as a probabilistic distributions of topics.

Topic modeling is used for summarising text. It can be used to organise documents, for example, to group requirements together into interconnected sections, such as organising all the requirements that impact one system or functionality.

There are several different methods for topic modeling, each pros and cons. A evaluation of the models results based on coherence scores or other metrics can be done to accurately determine the model with best performance for the identified problem. For demonstration purposes, LSA and LDA are considered.

Different Methods of Topic Modeling

- Latent Dirichlet Allocation (LDA)
- Non Negative Matrix Factorization (NMF)
- Latent Semantic Analysis (LSA)
- Parallel Latent Dirichlet Allocation (PLDA)
- Pachinko Allocation Model (PAM)
- Hierarchical Dirichlet Process (HDP)

5.4.0.1 Latent Semantic Analysis

Latent Semantic Analysis (LSA), as described by Landauer and Dumais Landauer et al. (1998), is a method for extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of text.

LSA operates as an unsupervised approach, particularly adept at revealing synonymous terms and capturing semantic nuances across a corpus by employing dimensional reduction techniques. The Gensim LSA model has several hyperparameters that can be tuned to optimize its performance for a specific use case, such as number of topics (dimensions), minimum document frequency (filtering out terms that appear in too few documents), maximum document frequency (removing terms that appear in too many documents) and normalization (applying techniques, such as TF-IDF or L2 normalization). Careful experimentation and evaluation of these hyperparameters are necessary to achieve optimal results for a specific use case.

5.4.0.2 Latent Dirichlet Allocation

In LDA, words that frequently co-occur in documents form these clusters. LDA can be thought of as a clustering technique, but it is important to understand the specific nuances that differentiate it from traditional clustering methods.

Traditional clustering techniques, such as k-means or hierarchical clustering, group data points into clusters based on similarity. Each data point belongs to exactly one cluster, and the goal is to maximize intra-cluster similarity and minimize inter-cluster similarity. Unlike **hard clustering** methods where each document belongs to exactly one cluster, LDA performs **soft clustering**: Each document can be associated with multiple topics, with each topic having a certain probability. This allows for more nuanced representations, where a document can partially belong to several topics.

LDA represents each document as a distribution over these topics. This is different from traditional clustering, where the focus is on grouping documents directly. Instead, LDA finds clusters of words (topics) and then represents documents based on these word clusters. Considering, for example, a collection of news articles. Traditional clustering might group these articles into clusters such as "sports," "politics," "technology," etc., based on their content. LDA, however, would identify topics like "sports" and "politics" as distributions of words. An article about a football match might be represented as 70% "sports" topic and 30% "politics" topic if it also discusses political issues related to sports.

While LDA is inherently an unsupervised learning technique, it can be combined with supervised learning methods in some workflows. For example, topics discovered by LDA can be used as features in a supervised machine learning model.

5.5 Process flow

With a better idea on the different methods and techniques, abstracting them as components, we proceed to lay down a simple pipeline style architecture for a tool. But before we dive in the

drawing board, it's essential to keep the project goals in mind. Remembering these goals will bound the understanding and assist to create an effective blueprint for our tool.

The goals of the research:

1. Supporting Change-Impact Analysis (CIA).
2. Tracing or eliciting safety-related aspects from existing norms and standards.
3. Facilitating effective analysis, tracing and reuse of change to prevent safety risks and normative violations.

Let's establish some use cases:

- On a first run, the user applies the tool to existing requirements database. The tool identifies topics, associating each requirement in the database to the found topics (automatic labelling / feature creation).
- Given a new requirement by the user, the tool uses topic modeling to compare on existing requirements topics. The new requirement is then associated with existing, enabling a user to identify areas of impact in the requirements and traced artifacts.
- User uploads norms and stardarts to the tool. The tool will perform topic modelling on the documents and cross associate with the requirements used to create the model.
- On a first run, the user applies the tool norms and stardarts. The tool identifies topics, associating each norms to a found topics (automatic labelling). Given a new requirement by the user, the tool uses topic modeling to trace to the related norms.

5.5.1 Architecture design

This blueprint separates the process into distinct activities, organizing components into a linear sequence, where the output of one component serves as the input to the next.

Initially, each artifact in the collection is parsed. Next, we tokenize the text, allowing for customizable n-grams, where multiple n-gram configurations can be applied. Following the tokens undergo optional static stopword removal. Subsequently, there's an option to apply either lemmatization or stemming.

The processed results from each individual artifact are consolidated into a single dictionary representing the entire artifact collection, with individual bag-of-words created for each artifact. Moving to the modeling phase, we have the flexibility to choose between LSA or LDA, with respective parameters.

The resulting topics are then presented to the user for evaluation. At this stage metrics are retrieved and the user has the ability to add stopwords to a dynamic stopword list. Post-user evaluation, we re-initiate the process, now incorporating dynamic stopwords added.

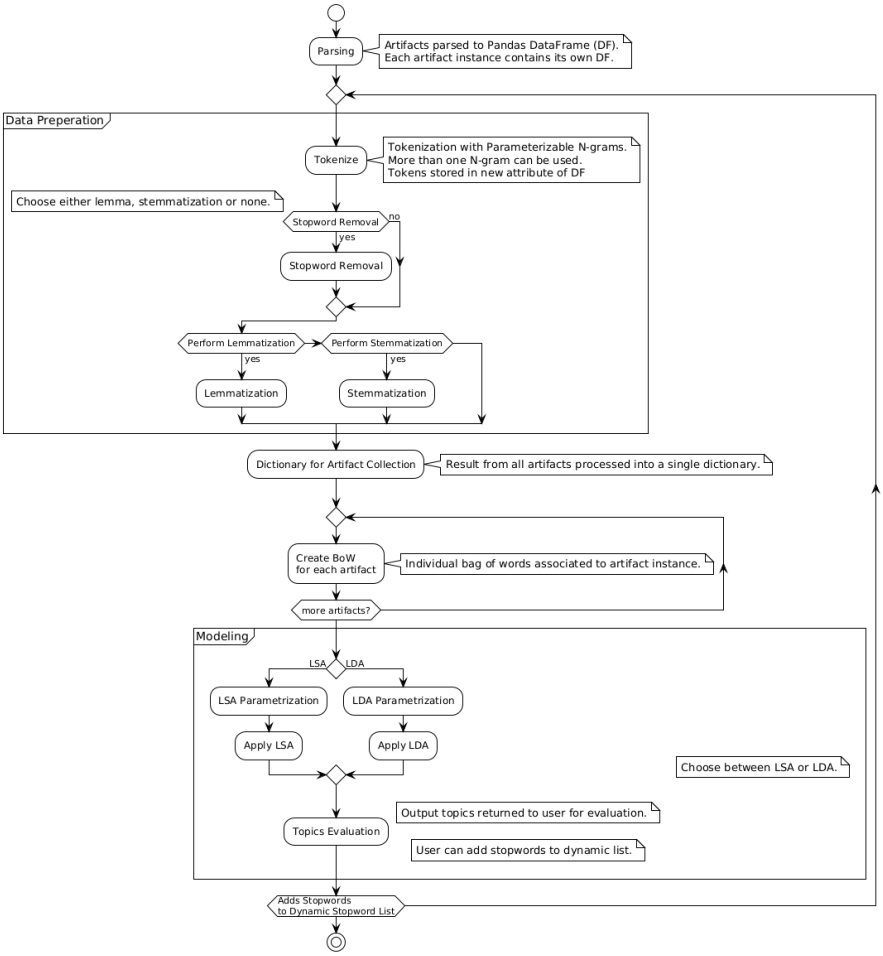

Figure 5.1: Activity diagram

5.6 Use Cases

This section clarifies how a requirements manager can benefit from the framework developed in this study by providing detailed use cases that demonstrate its practical applications.

1. **Initial Model Creation and Setup**

   (a) **Data Preparation** - The first step involves preparing a dataset of existing requirements. These requirements are collected and prepared to ensure that the data is in a suitable format to feed the tool.

   (b) **Model Training** - The requirements engineer creates a first a topic model, with the suggested process flow and the prepared dataset. This model identifies and categorizes topics within the requirements, associating each requirement with one or more topics based on its content.

   (c) **Model Evaluation and Optimization** - The requirements engineer reviews the quality of the generated topics, making adjustments to parameters and reprocessing the steps as needed to improve the relevance and coherence . This iterative process continues

until the user is satisfied with the quality and accuracy of the topics. The optimized model can then be stored, associated to the specific dataset of requirements it was trained on.

2. **Handling Change Requests**

   (a) **Change Request Preparation** - When a change request is received, the requirements engineer analyses the change and prepares draft requirements in the context of the SCS. Either the new requirements, or a general description of the change will be used as inputs to the model previously created.

   (b) **Topic Association** - The trained model analyses the new input and associates it with one or more existing topics. This association helps to identify the most relevant existing requirements that are likely to be impacted by the change.

   (c) **Change Impact Analysis** - The requirements engineer retrieves the existing requirements and traced artifacts associated with the identified topics. Using human judgment and domain expertise, the engineer evaluates whether these areas are affected by the change. This assessment helps in understanding the scope and implications of the change across the project.

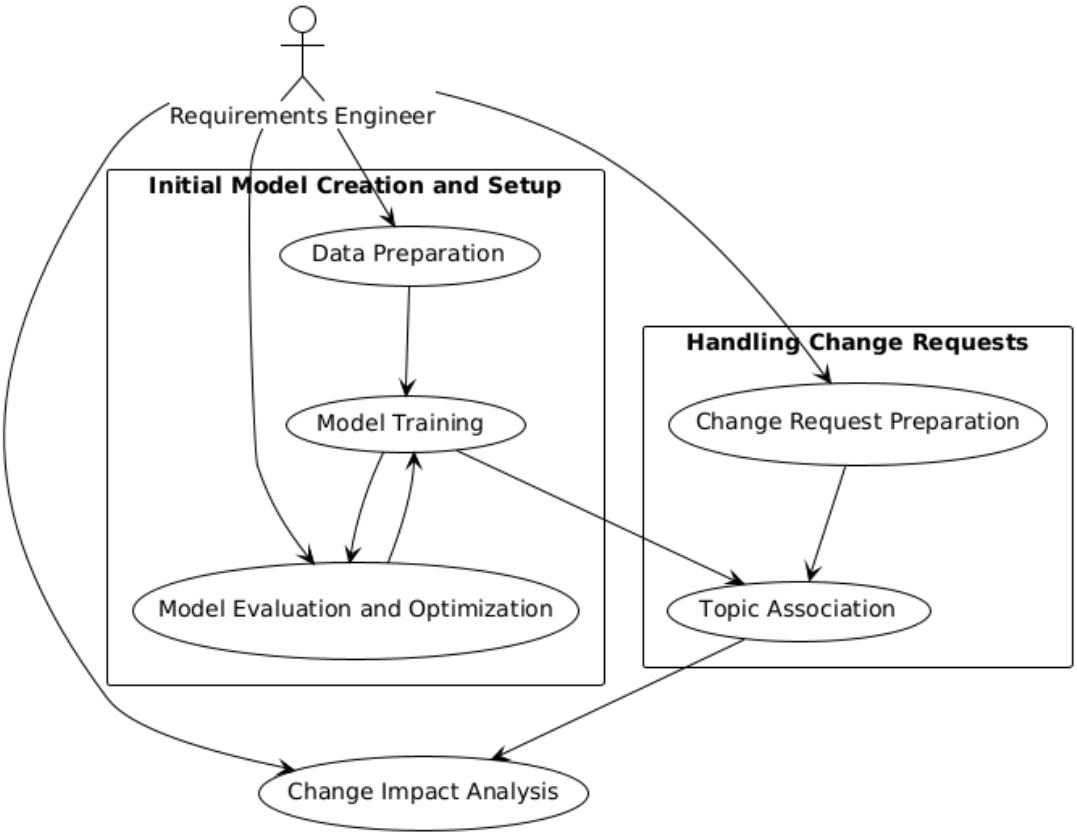

Figure 5.2: Use cases diagram

Depending on the existing system's configuration management and development environment, the tool can enable tracing the impact of changes across different levels or phases of the SDLC. For instance, a change in a high-level requirement can be traced down to its impact on detailed design, implementation, and testing phases, ensuring comprehensive impact analysis. This is dependent on how the tracing is managed in development environment of the SCS in analysis and should be adapted accordingly.

Based on topic associations, the framework facilitates efficient retrieval of requirements for analysis. This feature is particularly useful in large projects with extensive requirements databases, enabling quick access to relevant information.

The framework can be applied independently to various levels of requirements or other types of textual documentation, such as design documents, test cases, and user manuals. This flexibility allows for a holistic approach to change impact analysis, ensuring that all related documents are considered when evaluating changes. By ensuring that all changes are thoroughly analysed and documented, the tool aids in maintaining compliance with industry regulations and standards. It can also help managers to assess the impact of a change, the needed resources and the time it can take until the changes are in place.

Adhering to this framework can support better collaboration among team members by providing a clear and organized way to manage and communicate requirements changes. Stakeholders such as different engineering areas can easily identified and informed about the changes under discussion, accelerating the discussion on informed decisions.

5.7 Summary

To conclude this chapter presents for leveraging NLP techniques, through the use of gensim, enhancing RE in SCS, although it can be applied to several other context with the same constrains. The developed tool uses topic modeling on existing requirements to identify related topics when a new requirement is introduced. Based on topic similarity, it assists impact analysis by identifying requirements affected by the introduction of a new requirement.

The choice of the Gensim library for topic modeling was driven by its robustness in handling large text databases and its support for techniques such as LSA and LDA. This enables the extraction of meaningful topics from requirements documents, for understanding of complex domain-specific information.

Jupyter Notebooks were instrumental throughout the study, facilitating transparent and reproducible workflows. These notebooks integrated live code with explanations and visualizations, documenting the discovery process and findings in a structured manner.

Development artifacts such as requirements documents are introduced in a cleaning and structuring processes, including tokenization, stopword removal, and optionally stemming or lemmatization. This rigorous preparation ensured standardized text data suitable for subsequent analysis. N-grams were chosen to capture contextual information by grouping sequences of tokens

Given the unsupervised nature of the study, topic modeling served as a ground technique to uncover latent patterns within requirements documents. By applying algorithms such as LDA from Gensim, the study identified and categorized topics, with ultimate goal to provide insights into the underlying structure and relationships among requirements.

The chosen topic modeling model is applied to existing requirements to identify and categorize topics, aiding in understanding the thematic content of requirements. The tool compares topics of new requirements with existing ones to assess impact and ranks existing requirements based on similarity to the new requirement using topic modeling results.

By integrating these techniques and methodologies, the study wishes to structure a comprehensive tool for Requirements Engineering. **This approach not only enhances the understanding and management of CIA but also improves decision-making processes** by providing structured insights into complex domain-specific data, making processes more efficient, transparent, and informed within Safety-Critical Systems. The integration of NLP and topic modeling techniques provides a powerful means to manage and trace requirements changes, ensuring that all potential impacts are identified and addressed promptly.

Chapter 6

Evaluation

In this chapter, we evaluate the framework's ability to identify requirements affected by new changes. We start by briefly explaining the evaluation method, considering that models will be created using 3 requirements datasets in the railway domain:

- **EIRENE Functional Requirements Specification (Version 7.0)**GSM-R Functional Group (2006)
  - **Source:** GSM-R Functional Group
  - **Date:** 17 May 2006
  - **Version:** 7.0
  - **Description:** Defines functional requirements for the EIRENE railway telecommunications system, based on the ETSI GSM standard. It includes mandatory, system, and optional requirements to ensure technical compatibility and interoperability of railway telecommunications systems across Europe.
- **EIRENE System Requirements Specification (Version 15)**GSM-R Operators Group (2006)
  - **Source:** GSM-R Operators Group
  - **Date:** 17 May 2006
  - **Version:** 15
  - **Description:** Outlines requirements for the railway telecommunications system based on the ETSI GSM standard. It ensures interoperability between national railways, specifying network infrastructure and mobile equipment.
- **ERTMS System Requirements Specification (Version 5.00)**European Railway Agency (2007)
  - **Source:** European Railway Agency
  - **Date:** 21 June 2007

  - **Version:** 5.00
  - **Description:** Specifies system requirements for the European Rail Traffic Management System (ERTMS), including technical requirements for safe and efficient railway operation across Europe.

To evaluate the performance, we will refer to the publicly accessible requirements from the document **"European Integrated Railway Radio Enhanced Network Functional Requirements Specification," GSM-R Functional Group (2015) Version 8.0.0, dated 21 December 2015**.

This Version 8.0, as developed by the GSM-R Functional Group, is labeled with the final update date of March 2016 and replaces Version 7.0, published on May 2006. Both documents are structured around the core requirements of the EIRENE system, but Version 8.0 includes updates reflecting advancements in technology, changes in operational requirements, and improvements in interoperability standards

The specification outlines EIRENE's system services, including:

- Voice services (e.g., point-to-point calls, emergency calls, broadcast calls)
- Data services (e.g., text messages, general data applications, train control applications)
- Call-related services (e.g., priority and pre-emption, advanced call handling)
- Railway-specific applications and features

To evaluate our solution we select an updated area in Version 8.0 GSM-R Functional Group (2015) , focusing on the "Mobile Equipment Core Specification" (Chapter 4):

  - **4.1.3:** Mobile equipment must limit external interference impacts. (I)
  - **4.1.3i:** All EIRENE mobiles must operate in the EIRENE frequency band. (MI)
  - **4.1.3ii:** All EIRENE mobiles must operate in public GSM 900 frequency bands. (M)
  - **4.1.3iii:** All EIRENE mobiles shall/should operate in the extended GSM-R frequency band. (I)
    * a) Cab Radio must operate in the extended GSM-R frequency band. (M)
    * b) General Purpose Radio, Operational Radio, and Shunting Radio should operate in the extended GSM-R frequency band. (O)
  - **4.1.3iv:** All EIRENE mobiles should operate in other public GSM frequency bands. (O)
  - **4.1.4:** Equipment operating in 4.1.3i, 4.1.3ii, and 4.1.3iii bands must function at speeds of 0–500 km/h. (MI)

  - **4.1.5:** EIRENE mobiles must store data for network and subscriber identification. (M)

We parse this text to analyze it against our dataset using the trained model, aiming to identify requirements impacted by updates. Below is the same section from Version 7.2 GSM-R Functional Group (2006) for comparison:

- **4.1.2:** Three distinct mobile radio types are required, categorized by role and operational environment: (I)
  - a) Cab Radio – for train drivers and on-train systems (e.g., ERTMS/ETCS).
  - b) General Purpose Radio – for railway personnel use.
  - c) Operational Radio – for railway personnel involved in operations like shunting and maintenance.
  - **Note:** Variants of General Purpose and Operational radios may exist to meet railway needs (e.g., handheld, vehicle-mounted).
- **4.1.3:** All mobiles must operate in 900 MHz frequency bands for railways and public GSM networks. (M)
- **4.1.4:** Equipment must function at speeds of 0–500 km/h. (M)

Having identified the procedure, we will follow to describe the model creation and parameter tuning throughout the framework to identify impact areas and evaluate results. Each model run outputs a '.json' file with parameters, topics, and traced requirements to register the model creation 'run' results.

6.1 Registering results

Recording parameters, configurations, and metrics is crucial for tracking and evaluating model quality. Each model run outputs a JSON file with model parameters and artifact analysis results.

Topic are organized by ID, containing their corresponding keywords with respective weights, IDs of requirements linked to each topic (grouped by artifact) and the records for total number of requirements per topic and artifact.

For each processed requirement, the tool registers ID, text, tokenized text, associated topic ID, and topic keywords, compiling this information into JSON format. This approach ensures organized data, facilitating easy sharing and comprehensive model outcome analysis.

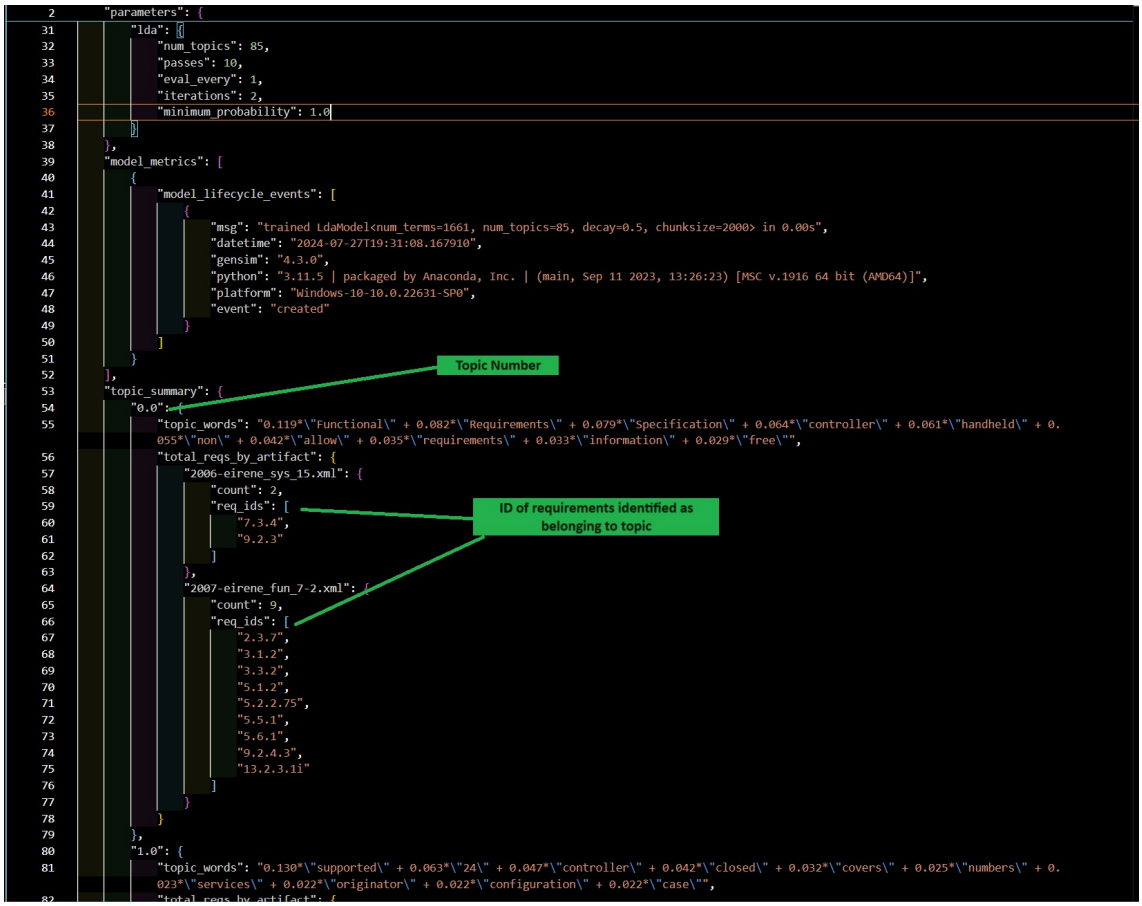

Figure 6.1: Evaluation results: Summary of topics

6.2 Model creation

As shown in Figure 6.3, the process begins with data preparation after collection.

The first model run uses these parameters:

- **N-Grams:** 1 (unigrams)
- **Stopwords:** Not removed
- **Stemming/Lemmatization:** Not applied
- **Algorithm:** LDA with hard clustering (one topic per requirement)

The results, shown in Figure 6.4, include requirements, tokenized text, and identified topics. For requirements "1.4.1.4", "10.2.1", and "10.3.2", the "text_clean" shows the data after cleaning. Not removing stopwords resulted in irrelevant words like "it" appearing in topics, along with isolated numbers.

To clean topics, we could set an LDA parameter to discard words with a probability below a threshold. However, as Figure 6.4 shows, there's insufficient separation to define an accurate

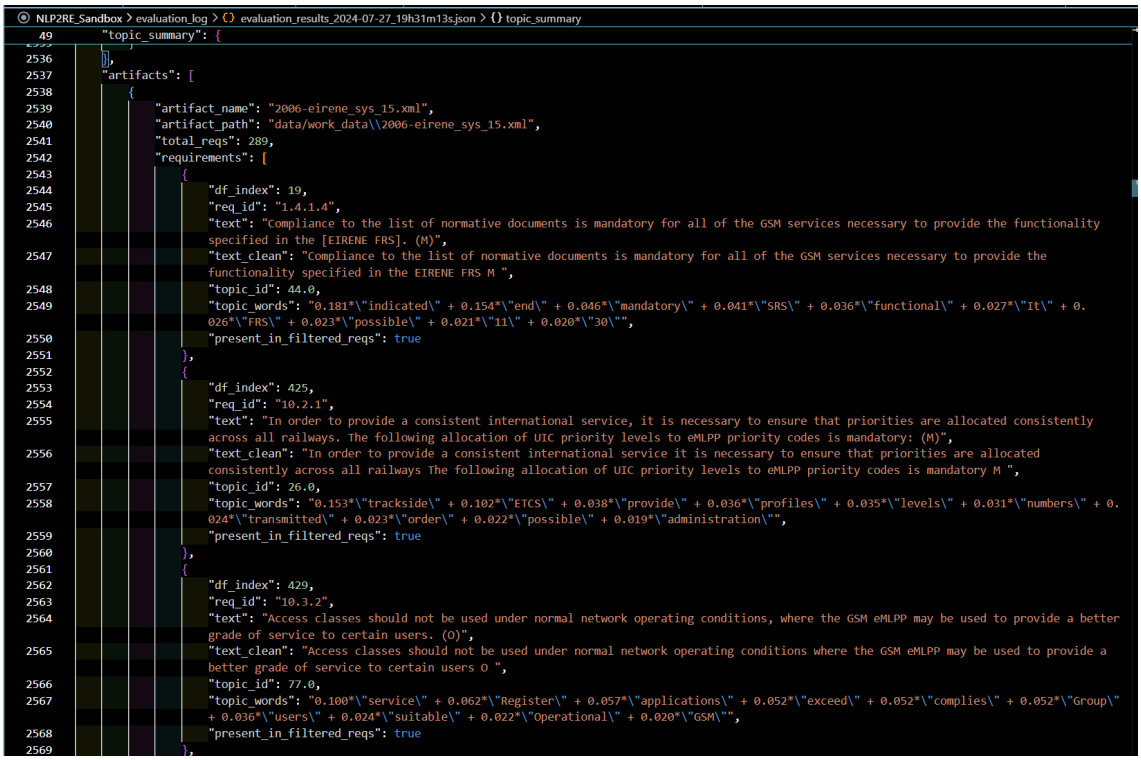

Figure 6.2: Evaluation results: Requirement analysis

threshold. Discarding words like "it" with a weight of 0.027 could also remove meaningful words like "transmitted," "order," and "possible" from topic 26.

In this first run, we also evaluate results using internal metrics, such as cosine similarity. Topics with high proximity should be reviewed for similarity, as they should have a similar semantic reading. Figure 6.5 shows topic with a inverse distance (closeness) higher than 30% , a relatively low threshold.

In the example of pair 1, we can see that the tokens "emergency," "shunting," and "railway" are repeated. Having unique topics is a good metric of a model. To end this first run evaluation we refer to human evaluation, by directly assessing the comprehension and usefulness of the topics through human ratings. The results in the topics do not help to assess the content of what is in the requirements identified by the same topic, meaning that topic creation has not been successful.

Following, we perform a second run removing static stopwords, to remove words such as "it" from the dataset used in topic creation. After removing the stopwords and looking at the topics, we can see that the word **"shall"** is present in almost all topics (Figure 6.6).

The word **"shall"** is commonly used in requirements documents, especially in the context of software, system, and engineering specifications. It is widely recognized in the field of requirements engineering as a precise and unambiguous way to specify mandatory requirements. The use of **"shall"** reduces ambiguity, It means that the requirement is not optional and must be implemented or fulfilled without exception. This contrasts with words like **"should"** or **"may"**, which indicate recommended or optional requirements, respectively. As commonly used in requirements

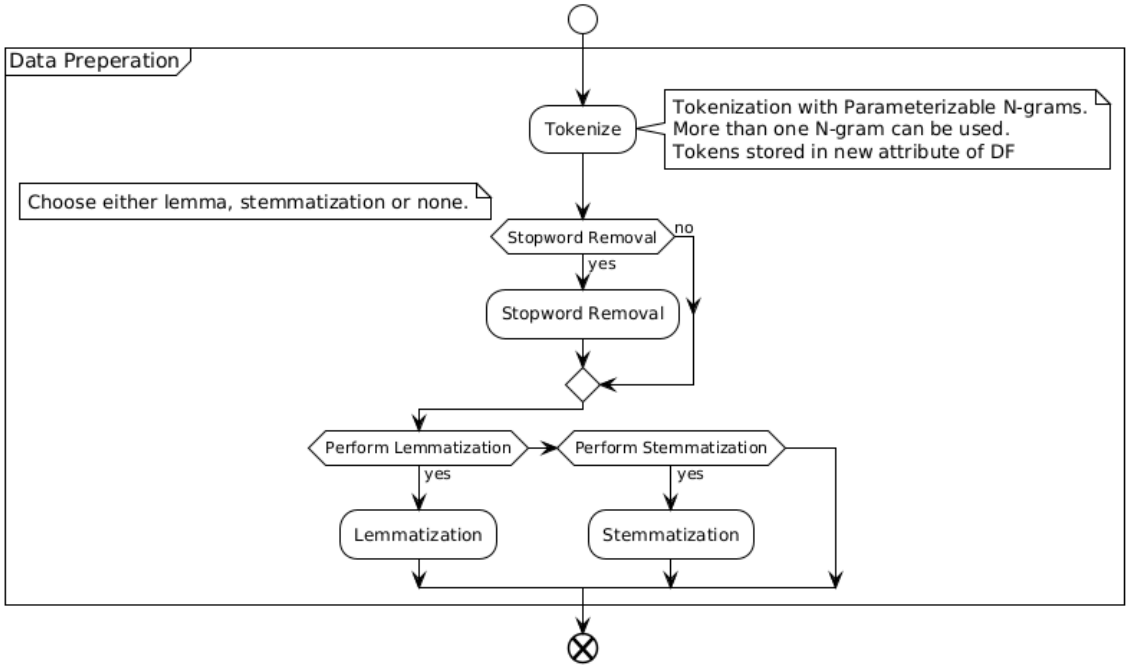

Figure 6.3: Activity Diagram, data preparation

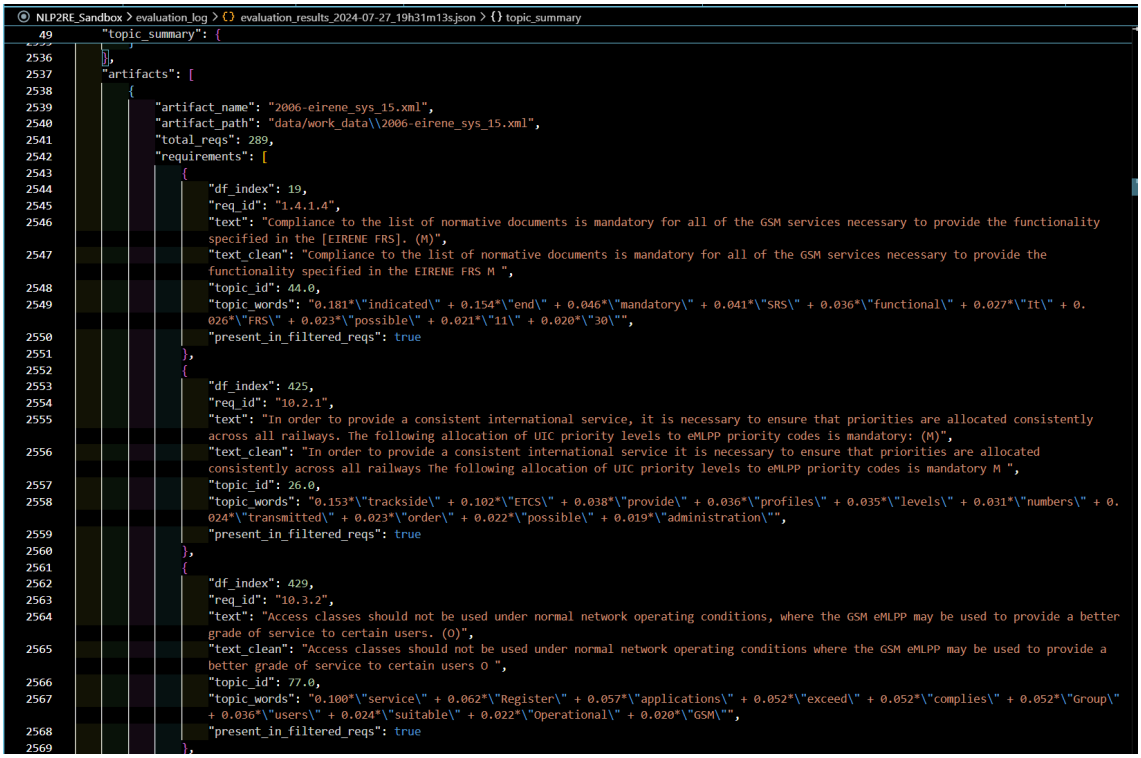

Figure 6.4: First run, requirements and identified topics

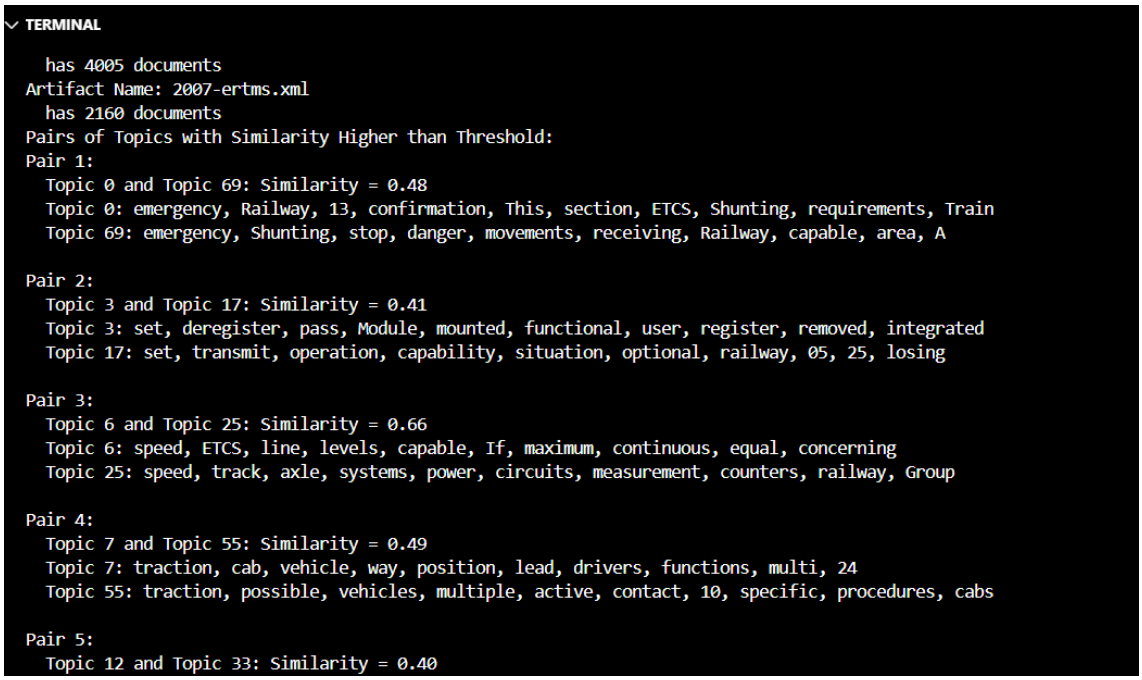

Figure 6.5: First run, cosine similarity between topics

of different contexts, the word does not add any value to a topic. Such words can be removed, as they have a high weighting in the topic but do not add anything to topic syntax comprehension. So,on a third run, we will remove dynamic stopword, words selected by a user for removal. Additional words have also been considered for removal (Figure 6.7). The selection was done by retrieving the most frequent tokens from the created dictionary and seeing which ones did not add meaning to a topic, due to their ambiguity.

From the result of the third we observe that there are tokens that have the same semantic meaning but are present in the cleaned text and in the topics in different forms. As seen in Figure 6.8, "call" is present two times in requirements 11.5.2, 11.5.3, and 11.5.4, as "calling" and "called." Let's reduce the size of our set of tokens and avoid these errors by applying stemming. Stemming was chosen as to avoid the error of changing any semantic meaning by changing the token to the canonical form of the word.

6.3 Model testing with new requirements

In the previous steps, there was a initial analysis of requirements against the topics generated by the model. This section explores how the model behaves when given a new input from EIRENE FRS Version 8.0 GSM-R Functional Group (2015) (section from Chapter 4 "Mobile Equipment Core Specification"). The input are handed to the tool as a string without the ID of the requirements.

As seen in Figure 6.9, the framework export a report for each input for evaluation of areas of potential impact.

Figure 6.6: First run, cosine similarity distributions

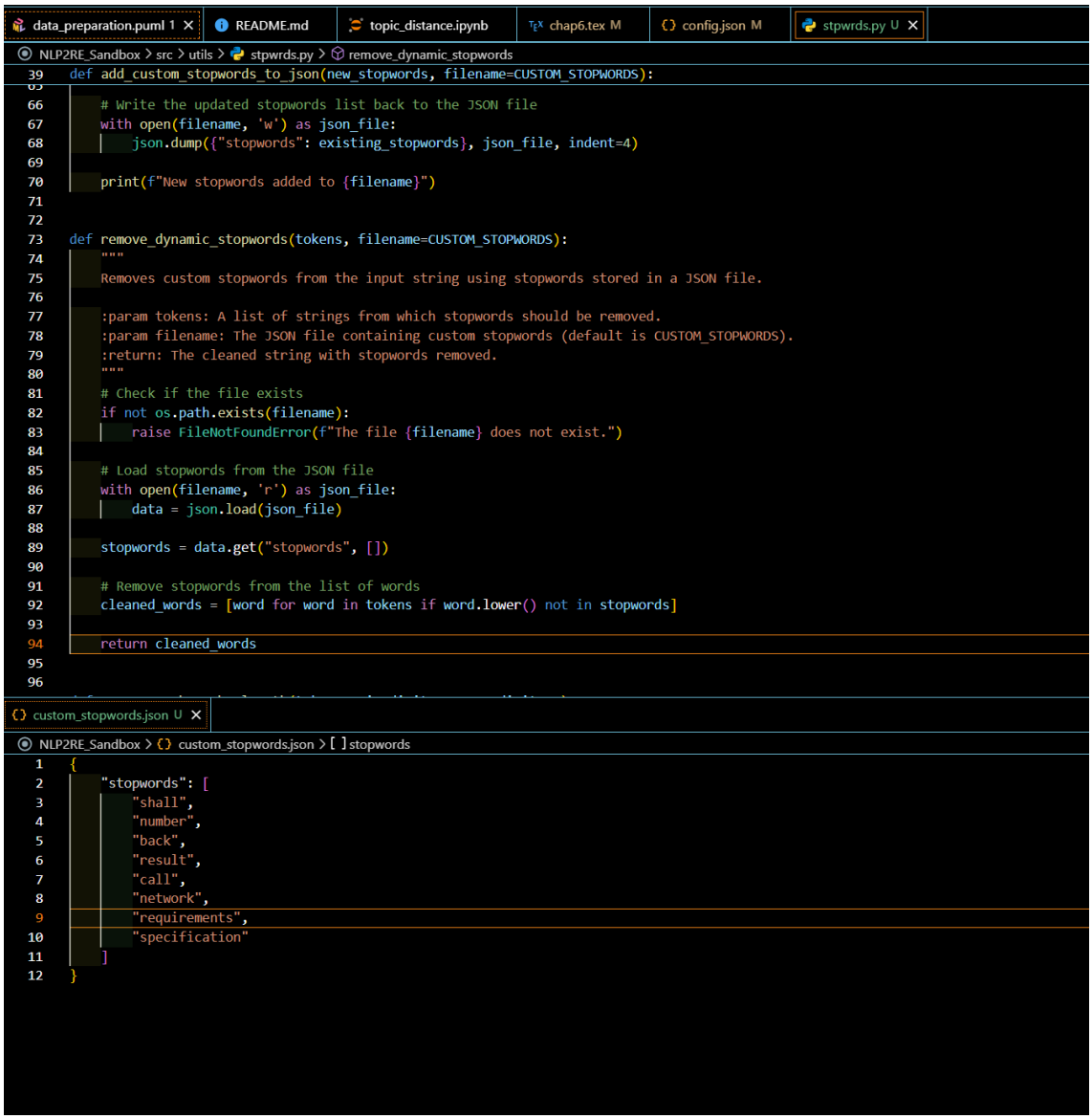

Figure 6.7: Third run, dynamic stopwords removal function and list

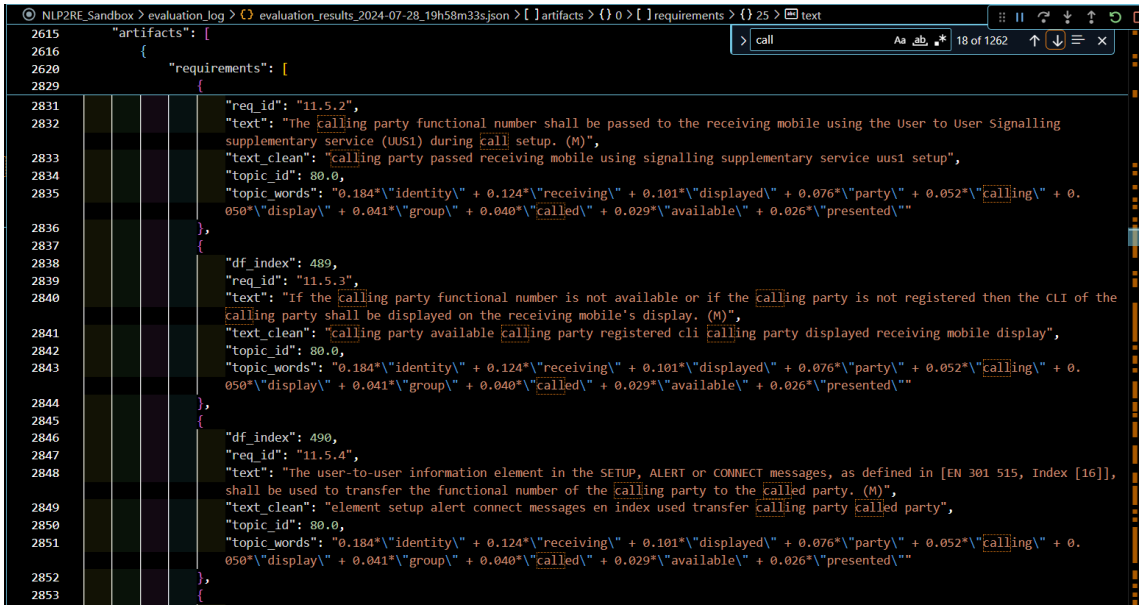

Figure 6.8: Third run, requirements and identified topics

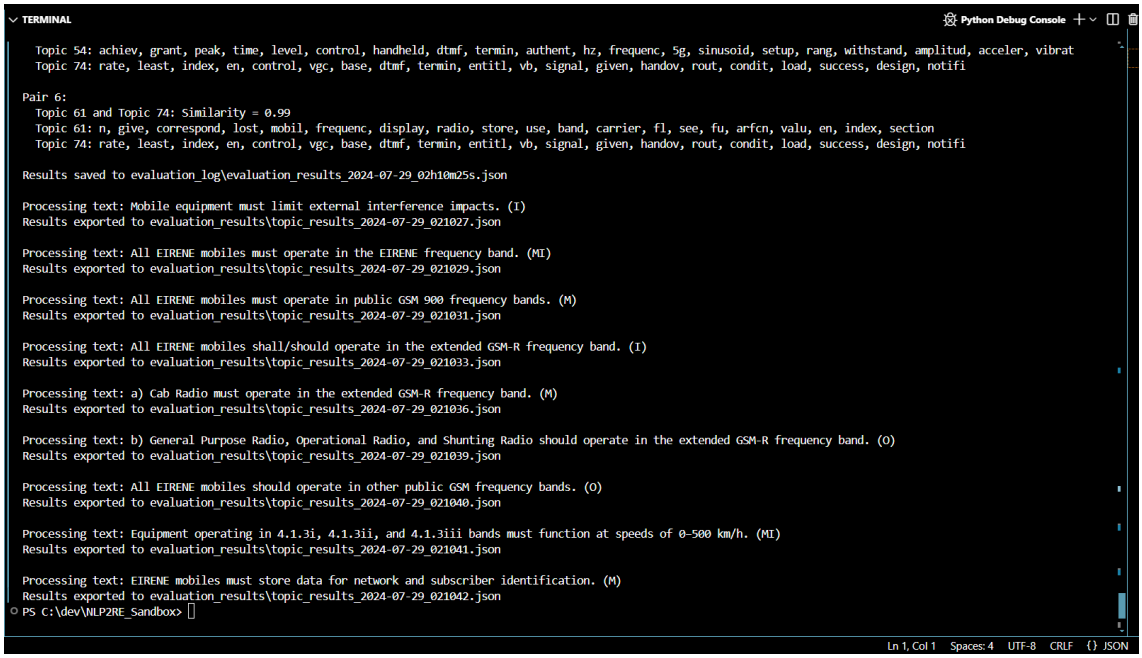

Figure 6.9: Tool interface: New requirement analysis export

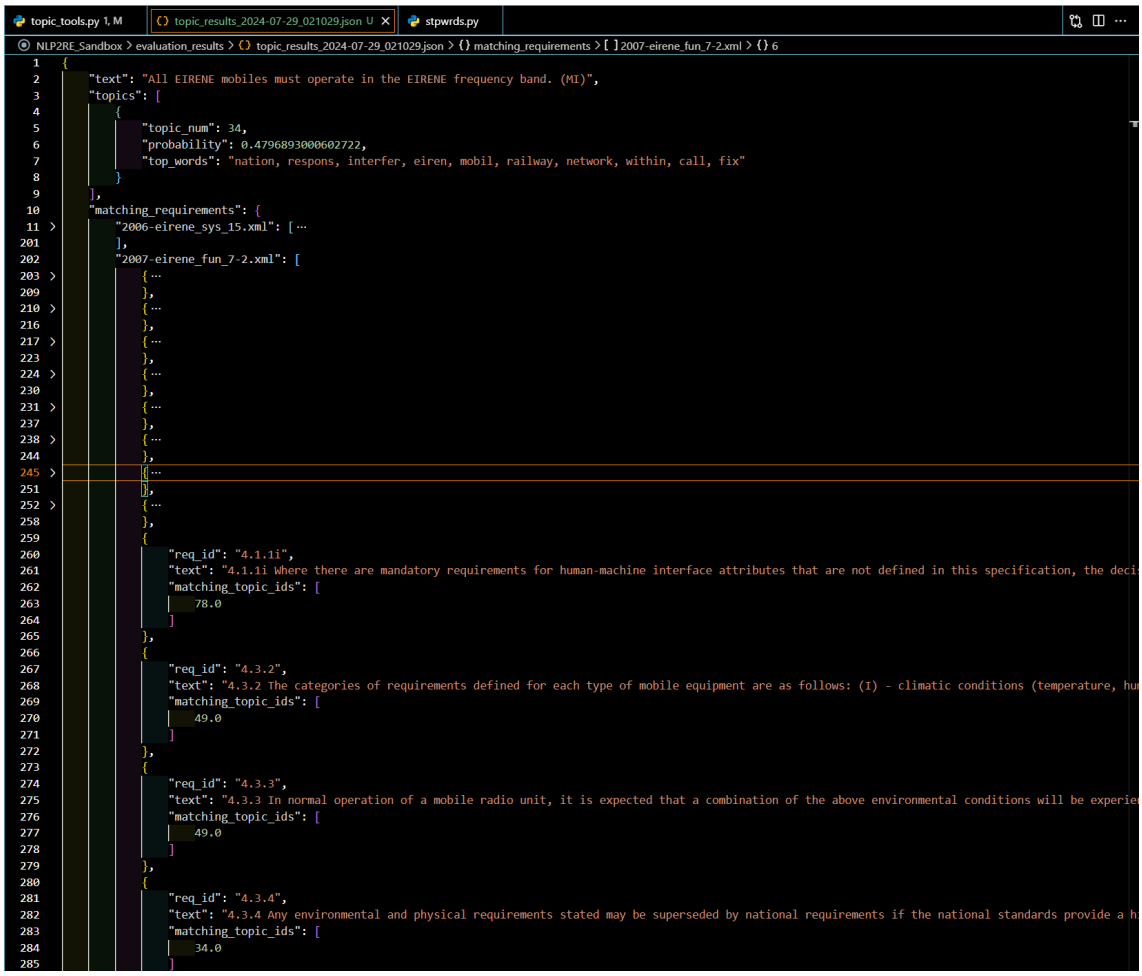

Figure 6.10: Tool interface: Requirement analysis export

When requirements align with specific topics, identifying impacted requirements becomes feasible. The framework can facilitate comprehensive analysis, allowing for cross-reference of new requirements. Although the results for the model created in this evaluation where not satisfactory.

Figure 6.10 illustrates exported result of a input text, related to specific topics, and the insights offered by the tool. A human evaluation of the identified topic does not reveal any semantic reasoning that can be related to the new requirement. Too many requirements are identified as related, and the area of change, chapter 4.1.3, was not identified. There seems also to be an error in correlating the identified topic with the original requirements, as the matching topic ID's do not relate to the topic identified for the new text.

These visualizations aid analysts in pinpointing the areas that require further attention, to ensure comprehensive analysis and traceability assessment.

In conclusion, this chapter demonstrates the framework's handling of large requirement sets, providing insights into flaws for analysis.

6.4 Conclusion

The primary goal of the evaluation is to asses the performance of the proposed solution, as the evaluation is registered for different parameters as the model is further scaled, to measure the quality of the topics generated, ensure the model's robustness, and explore the interpretability and distinctiveness of the topics.

Metrics are indispensable for evaluating the results of machine learning techniques because they provide a quantitative basis for assessing the quality, validity, and efficiency of the models. The model was mainly evaluated by external metrics, mostly human judgment and comparisons to expected outcome, that do not offer qualitative insights into the model's interpretability and practical relevance. We find that not enough internal metric are assessed to evaluate the models performance.

Finally, the outcome of the framework's evaluation showcases the weakness of the created model, although, the improvements throughout the runs indicate that further data prepossessing, coupled with domain-specific insights and algorithmic adjustments, significantly improves the relevance of identified requirement topics, disclosing that the framework can identify and trace requirements.

Chapter 7

Conclusion

The lack of transparency in AI models is a barrier to their use in safety-critical systems (SCS). In these systems, the decision-making process must be clearly understood, verified, and validated. While the direct application of AI in operational SCS is restricted, leveraging Natural Language Processing (NLP) tools in Requirements Engineering (RE) tasks shows great promise for improving efficiency and accuracy in RE activities.

Due to security, control, and compliance requirements, organizations in SCS industries prefer to develop and operate their own machine learning solutions instead of relying on third-party services due to risk of data breaches or unauthorized access to requirements databases, which could expose SCS vulnerabilities.

7.1 Importance of CIA in SCS Development

SCS development follows a structured approach that ensures each stage is verified and validated against its specifications. This process enhances the system's reliability and safety, reduces the probability of rework, and assists in the early detection of defects. Maintaining traceability between different development artifacts is crucial for understanding dependencies and potential ripple effects of changes. Detailed documentation at each phase is a must for future maintainability and system homologation. Any modifications in requirements, design, or code can have significant implications throughout the entire development cycle.

Change Impact Analysis (CIA) aims to identify the potential consequences of a change, allowing for predictions about the scope and impact before implementation. In the context of SCS, CIA is particularly important due to the safety-critical nature of these systems and the need to ensure changes do not compromise reliability, safety, and regulatory compliance.

7.2 NLP in Requirements Engineering

There is significant untapped potential for using NLP solutions in RE activities. NLP can assist human analysis of change impacts on SCS by allowing automated processing of large volumes

of text data, which is essential for extensive requirements documents, standards, and norms. The proposal of this thesis lies in retrieving significant meaning from these databases to assist in CIA tasks, thus reducing the time and effort required.

NLP techniques can compare new requirements with existing ones to help assess the potential impact of changes. They can also align requirements with relevant norms and standards by automatically extracting and associating relevant topics. This ensures regulatory compliance and reduces safety risks.

7.3 Domain-Specific Challenges

Certain domain-specific challenges, such as specialized vocabulary and unique context, affect the performance of models. Adopting domain-specific ontologies is demanding. Purely relying on stochastic models might not be sufficient. Researchs suggest combining syntax-aware embeddings and rule-based approaches as this could enhance the effectiveness of NLP tools in discourse analysis. This approach was not explored, but it represents a path forward in research.

7.4 Techniques and Parametrization

Throughout the study of available NLP techniques, several parametrizations have been identified that can lead to better results in the context of the problem at hand. For example, careful analysis of stopword removal to prevent the loss of meaning in technical wording, such as names of norms or other domain-specific terms. The use of n-gram tokens can increase semantic relationships when creating topics. Alternating between stemming and lemmatization can help discover the most effective technique, though these cannot be used on tokens longer than one word (e.g., bigrams, trigrams).

Soft clustering by usage of Latent Dirichlet Allocation broadens the spectrum of results. Removing custom stopwords through human input increases topic precision. Filtering tokens in the dictionary or model according to their frequency can reduce size, leading to better results. Optimization within the process should consider direct feedback from the tool and the metrics retrieved.

Human review of topics is important not only for tool optimization but also as a step in CIA. It benefits human understanding by providing direct feedback from a vast database of requirements, where analysis can be difficult.

7.5 Evaluation and Challenges in Unsupervised Learning

Traditional supervised learning metrics, such as accuracy, are not applicable to use unsupervised settings due to the lack of ground truth labels. Different types of evaluation metrics, including internal and external metrics, and human evaluation, can directly assess the usefulness of the topics in practical applications.

A comprehensive evaluation approach should consider both internal and external metrics. By registering the results of various evaluation metrics and configurations, it's possible to gain a deeper understanding of the model's performance and make informed decisions about model selection and tuning throughout it's evolution. Additional metrics such as topic distance and symmetric KL-divergence can be used when evaluating topic models.

7.6 Framework for Developing Topic Modeling Solutions

We propose a pipeline of techniques for developing topic modeling solutions in context-specific applications, taylored to the particular challenges of CIA in RE for SCS. For our evaluation, we executed three iterative runs of the tool to assess its performance in generating accurate and semantically meaningful topics from the requirements datasets. The results indicate that achieving this objective poses considerable challenges. While the tool did not fully meet the initial performance expectations, each successive run demonstrated incremental improvements.

Our findings suggest that further experimentation and parameter adjustments are necessary to enhance the tool's capabilities. In particular, more sophisticated parameterization and fine-tuning of the model are essential to better capture the semantic nuances of the requirements.

Additionally, incorporating advanced data visualization techniques can provide deeper insights into the metrics, facilitating a more precise adjustment of the parameters. These visual aids will be instrumental in identifying patterns and correlations that may not be immediately evident through raw data analysis alone.

In summary, while the current version of the tool shows promise, its full potential can only be realized through ongoing refinement and enhancement. Future work can focus on optimizing the parameters, leveraging from domain-specific knowledge, and utilizing visualization tools to improve the accuracy and relevance of the generated topics. This continuous improvement approach will ultimately bolster the tool's capacity to identify and trace requirements impacted by changes, thereby supporting the maintenance and evolution of complex systems.

7.7 Limitations and Opportunities for Advancements

- **Research on syntax-aware embeddings**: Combining syntax-aware embeddings and rule-based approaches could enhance the effectiveness of NLP tools in discourse analysis and requirements traceability. Purely relying on stochastic models might not be sufficient.
- **Exploring metrics**: There is an opportunity to explore more metrics and how they can assist in creating a more autonomous process.
- **Scalability**: While the current pipeline does not foresee scalability, it is possible and should be considered in future work.

- **Feature extraction**: Eventually, having a evolved and mature topic modelling solution leads to feature (or label creation). The results can be used to train supervised models, enhancing further a context specific solution.

The framework developed in this study is useful for enhancing RE activities in safety-critical systems. It is designed to be applied in an industrial context, capable of processing larger datasets. By integrating advanced techniques and methodologies, this framework could improve the understanding and management of requirement change and subsequent impact in development, also helping in decision-making management, making them more efficient, transparent, and well-informed within safety-critical systems.

References

Ankit Agrawal, Seyedehzahra Khoshmanesh, Michael Vierhauser, Mona Rahimi, Jane Cleland-Huang, and Robyn Lutz. Leveraging Artifact Trees to Evolve and Reuse Safety Cases. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, pages 1222–1233, Montreal, QC, Canada, May 2019. IEEE. ISBN 978-1-72810-869-8. doi: 10.1109/ICSE.2019.00124. URL https://ieeexplore.ieee.org/document/8812137/.

Camilo Almendra and Carla Silva. Managing Assurance Information: A Solution Based on Issue Tracking Systems. In *Proceedings of the XXXIV Brazilian Symposium on Software Engineering*, SBES '20, pages 580–585, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 978-1-4503-8753-8. doi: 10.1145/3422392.3422454. URL https://doi.org/10.1145/3422392.3422454. event-place: Natal, Brazil.

Christina Antoniou, Kalliopi Kravari, and Nick Bassiliades. Semantic Requirements Construction Using Ontologies and Boilerplates. preprint, SSRN, 2023. URL https://www.ssrn.com/abstract=4549644.

Christina Antniou and Nick Bassiliades. tool for requirements engineering using ontologies and boilerplates. *Automated Software Engineering*, 31(1):5, May 2024. ISSN 0928-8910, 1573-7535. doi: 10.1007/s10515-023-00403-y. URL https://link.springer.com/10.1007/s10515-023-00403-y.

Somnath Banerjee, Avik Dutta, Sayan Layek, Amruit Sahoo, Sam Conrad Joyce, and Rima Hazra. Redefining Developer Assistance: Through Large Language Models in Software Ecosystem. 2023. doi: 10.48550/ARXIV.2312.05626. URL https://arxiv.org/abs/2312.05626. Publisher: arXiv Version Number: 1.

Nelly Bencomo, Jin L.C. Guo, Rachel Harrison, Hans-Martin Heyn, and Tim Menzies. The Secret to Better AI and Better Software (Is Requirements Engineering). *IEEE Software*, 39(1):105–110, January 2022. ISSN 0740-7459, 1937-4194. doi: 10.1109/MS.2021.3118099. URL https://ieeexplore.ieee.org/document/9662414/.

Manal Binkhonain and Liping Zhao. A review of machine learning algorithms for identification and classification of non-functional requirements. *Expert Systems with Applications: X*, 1:100001, April 2019. ISSN 25901885. doi: 10.1016/j.eswax.2019.100001. URL https://linkinghub.elsevier.com/retrieve/pii/S2590188519300010.

David M. Blei. Probabilistic topic models. *Commun. ACM*, 55(4):77–84, apr 2012. ISSN 0001-0782. doi: 10.1145/2133806.2133826. URL https://doi.org/10.1145/2133806.2133826.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, pages 1877–1901, 2020.

Jonathan D. Chang, Jordan L. Boyd-Graber, Sean Gerrish, Chong Wang, and David M. Blei. Reading tea leaves: How humans interpret topic models. In *Neural Information Processing Systems*, 2009. URL https://api.semanticscholar.org/CorpusID:215812433.

Fabiano Dalpiaz, Alessio Ferrari, Xavier Franch, and Cristina Palomares. Natural Language Processing for Requirements Engineering: The Best Is Yet to Come. *IEEE Software*, 35(5):115–119, September 2018. ISSN 0740-7459, 1937-4194. doi: 10.1109/MS.2018.3571242. URL https://ieeexplore.ieee.org/document/8474507/.

Rosa Delima, Khabib Mustofa, and Anny Kartika Sari. Automatic Requirements Engineering: Activities, Methods, Tools, and Domains – A Systematic Literature Review. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, 7(3):564–578, June 2023. ISSN 2580-0760. doi: 10.29207/resti.v7i3.4924. URL http://jurnal.iaii.or.id/index.php/RESTI/article/view/4924.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

European Railway Agency. Ertms/etcs system requirements specification. Technical Specification ERA/ERTMS/003204, European Railway Agency, June 2007. PURE database: ertms.xml.

Alessio Ferrari, Giorgio Oronzo Spagnolo, and Stefania Gnesi. PURE: A Dataset of Public Requirements Documents. In *2017 IEEE 25th International Requirements Engineering Conference (RE)*, pages 502–505, Lisbon, Portugal, September 2017. IEEE. ISBN 978-1-5386-3191-1. doi: 10.1109/RE.2017.29. URL http://ieeexplore.ieee.org/document/8049173/.

Iris Grasler, Daniel Preus, Lukas Brandt, and Michael Mohr. Efficient Extraction of Technical Requirements Applying Data Augmentation. In *2022 IEEE International Symposium on Systems Engineering (ISSE)*, pages 1–8, Vienna, Austria, October 2022. IEEE. ISBN 978-1-66548-182-3. doi: 10.1109/ISSE54508.2022.10005452. URL https://ieeexplore.ieee.org/document/10005452/.

GSM-R Functional Group. Eirene functional requirements specification. Technical Specification PSA167D005, International Union of Railways, May 2006. PURE database: eirene_fun_7-2.xml.

GSM-R Functional Group. European integrated railway radio enhanced network functional requirements specification version 8.0.0. Technical Report UIC CODE 950, International Union of Railways, 16, rue Jean Rey 75015 Paris, France, December 2015. Reference: UIC CODE 950.

GSM-R Operators Group. Eirene system requirements specification. Technical Specification PSA167D006, International Union of Railways, May 2006. PURE database: eirene_sys_15.xml.

Alexander Elenga Gärtner, Tu-Anh Fay, and Dietmar Göhlich. Fundamental Research on Detecting Contradictions in Requirements: Taxonomy and Semi-Automated Approach. *Applied Sciences*, 12(15):7628, July 2022. ISSN 2076-3417. doi: 10.3390/app12157628. URL https://www.mdpi.com/2076-3417/12/15/7628.

M Syauqi Haris and Tri Astoto Kurniawan. Automated Requirement Sentences Extraction from Software Requirement Specification Document. In *Proceedings of the 5th International Conference on Sustainable Information Engineering and Technology*, SIET '20, pages 142–147, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 978-1-4503-7605-1. doi: 10.1145/3427423.3427450. URL https://doi.org/10.1145/3427423.3427450. event-place: Malang, Indonesia.

Lise Tordrup Heeager and Peter Axel Nielsen. A conceptual model of agile software development in a safety-critical context: A systematic literature review. *Information and Software Technology*, 103:22–39, November 2018. ISSN 09505849. doi: 10.1016/j.infsof.2018.06.004. URL https://linkinghub.elsevier.com/retrieve/pii/S0950584918301125.

Jennifer Horkoff, Fatma Başak Aydemir, Evellin Cardoso, Tong Li, Alejandro Maté, Elda Paja, Mattia Salnitri, Luca Piras, John Mylopoulos, and Paolo Giorgini. Goal-oriented requirements engineering: an extended systematic mapping study. *Requirements Engineering*, 24(2):133–160, June 2019. ISSN 0947-3602, 1432-010X. doi: 10.1007/s00766-017-0280-z. URL http://link.springer.com/10.1007/s00766-017-0280-z.

Daneth Horn, Nazakat Ali, and Jang Eui Hong. Towards Enhancement of Fault Traceability Among Multiple Hazard Analyses in Cyber-Physical Systems. In *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, pages 458–464, Milwaukee, WI, USA, July 2019. IEEE. ISBN 978-1-72812-607-4. doi: 10.1109/COMPSAC.2019.10249. URL https://ieeexplore.ieee.org/document/8754309/.

IEC IEC 61508(-1/7):2010. Functional safety of electrical/electronic/programmable electronic safety-related systems., 2010.

Mohsin Irshad, Kai Petersen, and Simon Poulding. A systematic literature review of software requirements reuse approaches. *Information and Software Technology*, 93:223–245, January 2018. ISSN 09505849. doi: 10.1016/j.infsof.2017.09.009. URL https://linkinghub.elsevier.com/retrieve/pii/S0950584916303615.

ISO 21448:2022(E). Road vehicles — Safety of the intended functionality. Standard, International Organization for Standardization, Geneva, CH, June 2022.

Anjali Ganesh Jivani. A Comparative Study of Stemming Algorithms. 2, 2011.

Dhara J. Ladani and Nikita P. Desai. Stopword Identification and Removal Techniques on TC and IR applications: A Survey. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 466–472, Coimbatore, India, March 2020. IEEE. ISBN 978-1-72815-196-0 978-1-72815-197-7. doi: 10.1109/ICACCS48705.2020.9074166. URL https://ieeexplore.ieee.org/document/9074166/.

Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

Yarden Levy, Roni Stern, Arnon Sturm, Argaman Mordoch, and Yuval Bitan. An impact-driven approach to predict user stories instability. *Requirements Engineering*, 27(2):231–248, June 2022. ISSN 0947-3602, 1432-010X. doi: 10.1007/s00766-022-00372-w. URL https://link.springer.com/10.1007/s00766-022-00372-w.

Gang Li, Chengpeng Zheng, Min Li, and Haosen Wang. Automatic Requirements Classification Based on Graph Attention Network. *IEEE Access*, 10:30080–30090, 2022. ISSN 2169-3536. doi: 10.1109/ACCESS.2022.3159238. URL https://ieeexplore.ieee.org/document/9733904/.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. In *arXiv preprint arXiv:1907.11692*, 2019.

Chaitanya Malaviya, Shijie Wu, and Ryan Cotterell. A Simple Joint Model for Improved Contextual Neural Lemmatization. In *Proceedings of the 2019 Conference of the North*, pages 1517–1528, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1155. URL http://aclweb.org/anthology/N19-1155.

Ambarish Moharil and Arpit Sharma. TABASCO: A transformer based contextualization toolkit. *Science of Computer Programming*, 230:102994, August 2023. ISSN 01676423. doi: 10.1016/j.scico.2023.102994. URL https://linkinghub.elsevier.com/retrieve/pii/S016764232300076X.

Mohd Hafeez Osman and Mohd Firdaus Zaharin. Ambiguous software requirement specification detection: an automated approach. In *Proceedings of the 5th International Workshop on Requirements Engineering and Testing*, pages 33–40, Gothenburg Sweden, June 2018. ACM. ISBN 978-1-4503-5749-4. doi: 10.1145/3195538.3195545. URL https://dl.acm.org/doi/10.1145/3195538.3195545.

Andrew Yi-Zong Ou, Maryam Rahmaniheris, Yu Jiang, Lui Sha, Zhicheng Fu, and Shangping Ren. Safetrace: A Safety-Driven Requirement Traceability Framework on Device Interaction Hazards for MD PnP. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, SAC '18, pages 1282–1291, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 978-1-4503-5191-1. doi: 10.1145/3167132.3167270. URL https://doi.org/10.1145/3167132.3167270. event-place: Pau, France.

Julio-Omar Palacio-Niño and Fernando Berzal. Evaluation metrics for unsupervised learning algorithms, 2019.

Zaki Pauzi and Andrea Capiluppi. Applications of natural language processing in software traceability: A systematic mapping study. *Journal of Systems and Software*, 198:111616, April 2023. ISSN 01641212. doi: 10.1016/j.jss.2023.111616. URL https://linkinghub.elsevier.com/retrieve/pii/S0164121223000110.

Jon Perez-Cerrolaza, Jaume Abella, Markus Borg, Carlo Donzella, Jesús Cerquides, Francisco J. Cazorla, Cristofer Englund, Markus Tauber, George Nikolakopoulos, and Jose Luis Flores. Artificial Intelligence for Safety-Critical Systems in Industrial and Transportation Domains: A Survey. *ACM Comput. Surv.*, October 2023. ISSN 0360-0300. doi: 10.1145/3626314. URL

https://doi.org/10.1145/3626314. Place: New York, NY, USA Publisher: Association for Computing Machinery.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. In *OpenAI Blog*, 2019.

Matthias Rüdiger, David Antons, Amol M. Joshi, and Torsten-Oliver Salge. Topic modeling revisited: New evidence on algorithm performance and quality metrics. *PLOS ONE*, 17(4):e0266325, April 2022. ISSN 1932-6203. doi: 10.1371/journal.pone.0266325. URL https://dx.plos.org/10.1371/journal.pone.0266325.

Alexandra Schofield, Måns Magnusson, and David Mimno. Pulling Out the Stops: Rethinking Stopword Removal for Topic Models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 432–436, Valencia, Spain, 2017. Association for Computational Linguistics. doi: 10.18653/v1/E17-2069. URL http://aclweb.org/anthology/E17-2069.

Maninder Singh. Using Natural Language Processing and Graph Mining to Explore Inter- Related Requirements in Software Artefacts. *SIGSOFT Softw. Eng. Notes*, 44(1):37–42, March 2019. ISSN 0163-5948. doi: 10.1145/3310013.3310018. URL https://doi.org/10.1145/3310013.3310018. Place: New York, NY, USA Publisher: Association for Computing Machinery.

Maninder Singh, Vaibhav Anu, Gursimran S. Walia, and Anurag Goswami. Validating Requirements Reviews by Introducing Fault-Type Level Granularity: A Machine Learning Approach. In *Proceedings of the 11th Innovations in Software Engineering Conference*, pages 1–11, Hyderabad India, February 2018. ACM. ISBN 978-1-4503-6398-3. doi: 10.1145/3172871.3172880. URL https://dl.acm.org/doi/10.1145/3172871.3172880.

Sundar Singh and R K Pateriya. A Survey on various Stemming Algorithms. *INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING IN RESEARCH TRENDS*, 2(5), 2015.

Riad Sonbol, Ghaida Rebdawi, and Nada Ghneim. Learning software requirements syntax: An unsupervised approach to recognize templates. *Knowledge-Based Systems*, 248:108933, July 2022a. ISSN 09507051. doi: 10.1016/j.knosys.2022.108933. URL https://linkinghub.elsevier.com/retrieve/pii/S0950705122004518.

Riad Sonbol, Ghaida Rebdawi, and Nada Ghneim. The Use of NLP-Based Text Representation Techniques to Support Requirement Engineering Tasks: A Systematic Mapping Review. *IEEE Access*, 10:62811–62830, 2022b. ISSN 2169-3536. doi: 10.1109/ACCESS.2022.3182372. URL https://ieeexplore.ieee.org/document/9794783/.

Jan-Philipp Steghöfer, Eric Knauss, Jennifer Horkoff, and Rebekka Wohlrab. Challenges of Scaled Agile for Safety-Critical Systems. In Xavier Franch, Tomi Männistö, and Silverio Martínez-Fernández, editors, *Product-Focused Software Process Improvement*, volume 11915, pages 350–366. Springer International Publishing, Cham, 2019. ISBN 978-3-030-35332-2 978-3-030-35333-9. doi: 10.1007/978-3-030-35333-9_26. URL http://link.springer.com/10.1007/978-3-030-35333-9_26. Series Title: Lecture Notes in Computer Science.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

Jéssyka Vilela, Jaelson Castro, Luiz Eduardo G. Martins, and Tony Gorschek. Safe-RE: a safety requirements metamodel based on industry safety standards. In *Proceedings of the XXXII Brazilian Symposium on Software Engineering*, pages 196–201, Sao Carlos Brazil, September 2018. ACM. ISBN 978-1-4503-6503-1. doi: 10.1145/3266237.3266242. URL https://dl.acm.org/doi/10.1145/3266237.3266242.

Rüdiger Wirth and Jochen Hipp. Crisp-dm: Towards a standard process model for data mining. 2000. URL https://api.semanticscholar.org/CorpusID:1211505.

Claes Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, pages 1–10, London England United Kingdom, May 2014. ACM. ISBN 978-1-4503-2476-2. doi: 10.1145/2601248.2601268. URL https://dl.acm.org/doi/10.1145/2601248.2601268.

OU YI-ZONG. *Safety-driven software design and engineering in medical cyber-physical-human systems*. DISSERTATION, University of Illinois, Urbana-Champaign, 2019, 2019. URL https://hdl.handle.net/2142/105794.

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent Trends in Deep Learning Based Natural Language Processing [Review Article]. *IEEE Computational Intelligence Magazine*, 13(3):55–75, August 2018. ISSN 1556-603X, 1556-6048. doi: 10.1109/MCI.2018.2840738. URL https://ieeexplore.ieee.org/document/8416973/.

Liping Zhao, Waad Alhoshan, Alessio Ferrari, Keletso J. Letsholo, Muideen A. Ajagbe, Erol-Valeriu Chioasca, and Riza T. Batista-Navarro. Natural Language Processing for Requirements Engineering: A Systematic Mapping Study. *ACM Comput. Surv.*, 54(3), April 2021. ISSN 0360-0300. doi: 10.1145/3444689. URL https://doi.org/10.1145/3444689. Place: New York, NY, USA Publisher: Association for Computing Machinery.