University Of Ottawa

Faculty of Engineering

Computer Science

Course:

NLP

Professor:

Prof. Dr. Inkpen

Final Project Report

Romina Etezadi (300335563)

Nima Meghdadi (300368777)

Amin Abbasi Shahkoo (300336919)

1. Abstract

Regulatory compliance in software engineering is essential for ensuring the integrity, security, and ethical use of software systems in various industries. In this project, we investigate the role of ***three*** different Natural Language Processing (NLP) approaches to enhance the regulatory compliance practices within the realm of software engineering, especially on Requirement documents. Requirement documents, also known as requirements specifications, are formal documents that outline the specifications, features, functionalities, and constraints of a software system or product all written in natural language. NLP techniques offer innovative solutions for automating compliance tasks, extracting relevant information from regulatory documents, and analyzing requirement texts to ensure adherence to regulations. By leveraging NLP, software engineers can develop advanced compliance monitoring systems, streamline regulatory reporting processes, and proactively identify and mitigate compliance risks.

The report has the following structure:

- ***Related Works:*** We will briefly introduce the related works that have been done in this domain.
- ***Dataset and Statistics:*** We discuss the dataset we're utilizing along with its statistical characteristics.
- ***Methodology:*** We will discuss the approaches we have used to solve this problem:
    - Multi-Class Multi-Label Classification
    - Semantic Textual Similarity
    - Prompt Engineering
- ***Conclusion:*** We'll wrap up our project and delve into the effectiveness of every method we've employed.

2. Related Works

Several studies have explored the application of Natural Language Processing (NLP) techniques in the domain of regulatory compliance, aiming to automate and streamline compliance processes, enhance regulatory monitoring, and improve decision-making. Below are some key works in this area:

In the paper "Automated Compliance Checking of Legal Documents: A Natural Language Processing Approach" [1], the authors present a methodology for automatically detecting violations of regulations in legal texts using NLP techniques such as text classification and information extraction using semantic frames.

In "Regulatory Compliance Monitoring Using Natural Language Processing" [2], the authors discuss the use of NLP for extracting compliance-related information from regulatory texts, enabling organizations to stay updated with regulatory changes and ensure compliance with relevant laws and standards.

Researchers have explored the automation of regulatory reporting processes using NLP to extract and analyze relevant information from textual data. For example, in "Automating Regulatory Reporting Using Natural Language Processing" [3], the authors propose a framework for automating regulatory reporting tasks by leveraging NLP techniques to process regulatory documents and generate compliance reports.

Some studies have focused on semantic annotation of regulatory documents to enhance their interpretability and usability. In "Ontology-Based Semantic Annotation of Regulatory Documents for Compliance Checking" [4], the authors present an ontology-based approach for annotating regulatory documents with semantic metadata, enabling automated compliance checking and decision support.

Given the existing research, investigations into language models and prompt engineering techniques have been lacking. Hence, our project aims to assess various language models and prompt engineering methodologies to understand their effectiveness in addressing this industrial challenge.

3. Dataset and Statistics

For this project, we have used the dataset called HIPAA (Health Insurance Portability and Accountability Act) [5] which is publicly available. This dataset comprises anonymized healthcare data that adheres to the stringent privacy and security regulations outlined in the HIPAA law. Developers use the HIPAA dataset to validate and verify their software solutions. They can test whether their systems adequately protect patient privacy, maintain data integrity, and adhere to access control policies as mandated by HIPAA.
In order to begin we first describe the terms we are going to use in this project:

- ***Requirement:*** A requirement is a document which represents a specific feature, function, quality, or characteristic that the system or product must possess to fulfill its intended purpose. An example of a requirement in HIPAA is "*The system shall allow authorized personnel to access records via secure connectivity from anywhere, at any time.*"
- ***Regulation / Regulatory Code:*** Regulatory codes / Regulations refer to a set of rules, standards, laws, or guidelines established by a regulatory authority to govern and oversee specific industries, activities, or practices. These codes are designed to ensure compliance with legal and ethical standards, promote safety, protect consumers, and maintain fairness within the regulated domain. An example of a regulation in HIPAA is *"Access Control. Implement technical policies and procedures for electronic information systems that maintain electronic protected health information to allow access only to those persons or software programs that have been granted access rights as specified in? 164.308(a)(4)."*.

- ***Trace Link / Link:*** A trace link between a regulation and a requirement establishes a clear connection between a specific regulatory code and the corresponding system or product requirement that addresses that regulation. For example, the two examples from the above are linked.

The dataset has three files:

1. Requirements: contains **1891** requirements,
2. Regulations: contains **10** regulatory codes, and
3. Trace links which include **243** trace links between requirements and regulations (Figure 1). Among these 243 links, there are **230** unique requirements, which means one requirement can have more than one link (i.e. a requirement can be linked to multiple regulations) while other requirements have no links to any regulation (we call this class Else). Figure 1 displays the distribution of requirements linked to the 10 regulatory codes. “Else” represents a requirement without having any link to any of the regulations.

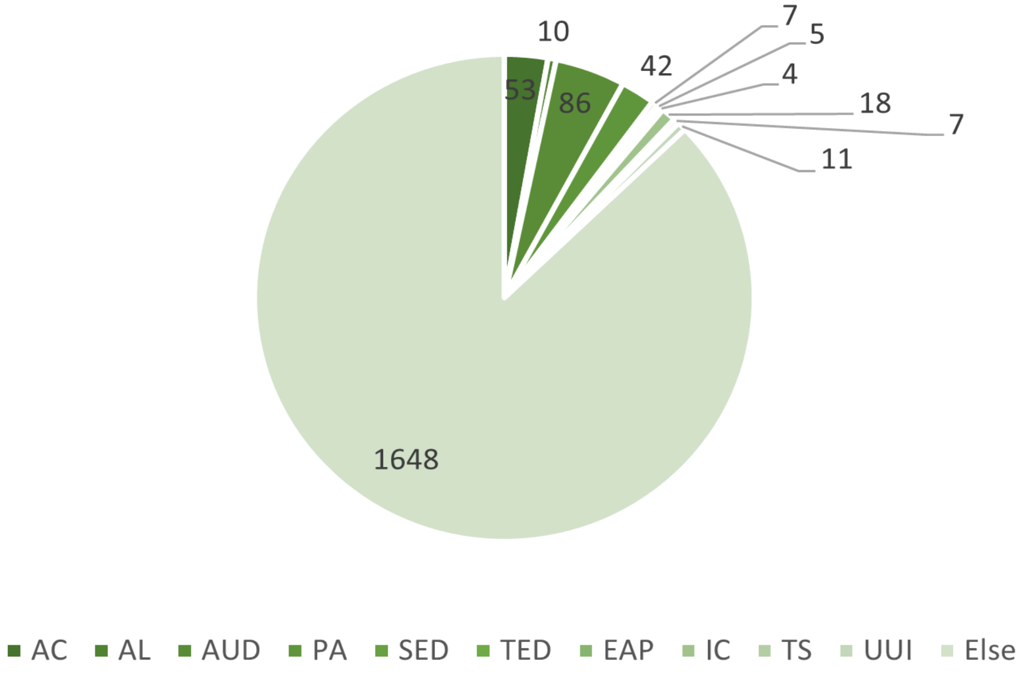


Figure 1. Distribution of the number of linked requirements

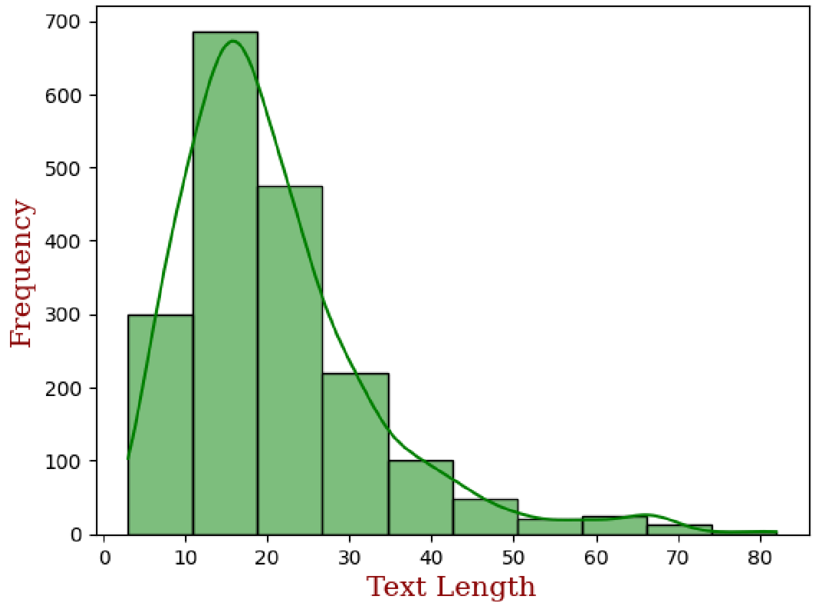


Figure 2. Distribution of text length

Figure 2 shows the histogram of the requirement's length:

- The distribution seems roughly unimodal, with a single peak around the text length of 10-20.
- There is a long right tail, indicating that there are fewer texts as the length increases but it stretches out to a length of about 80.
- Most texts are between 10 and 30 characters long, with this range containing the highest frequency of texts.
- The frequency sharply decreases after this range, with fewer texts having a length greater than 30 characters.

4. Methodology

This project aims to define and map regulatory compliance in three situations. We use multiple methods to carefully assess and choose the best performance. Additionally, we will determine which strategy is most adaptable and effective in instances with little training data that is not balanced. This detailed examination will help us determine the best compliance strategy and understand its efficacy in data-constrained contexts. This project will provide insights for optimizing regulatory compliance operations without significant training datasets.

We divide the data into two sets: train and test. For training, we assign 80% of the requirements associated with each category (regulatory code) to the train set and 20% to the test set. These samples are treated as positive examples. To incorporate negative samples (label Else), we randomly choose requirements without any trace link and include them in both the train and test sets. It's important to mention that we undersample the dataset to ensure a balance between the number of positive and negative examples.

4.1. Multi-Class Multi-Label Classification

In this setup, we consider each regulation as a class or label. As a result, each requirement is associated with multiple labels, meaning it's connected to multiple regulations. The number of classes in this problem would be 11 (10 regulation codes + 1 which is the Else class). In this approach, the model will only receive the requirements text as input. Figure 3 illustrates the method we have used in this scenario:

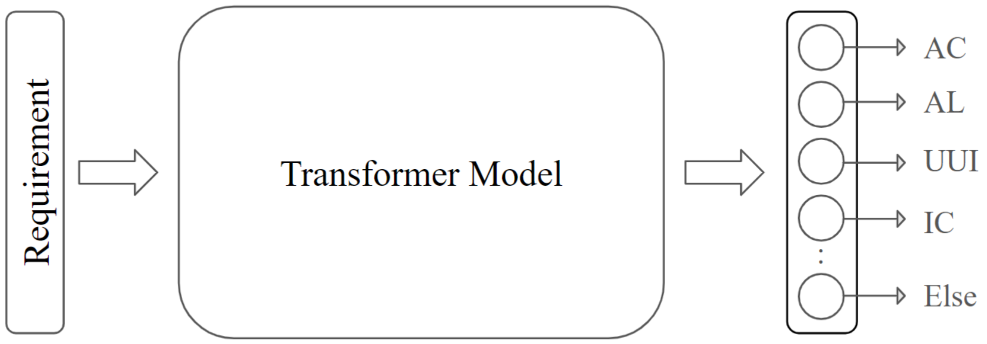


Figure 3. Illustration of the multi-class multi-label classifier. Input comprises requirement text, while the output encompasses the classes to which the requirement belongs.

This capability facilitates parallel processing, leading to a significant enhancement in efficiency compared to earlier sequential models. At the word level, the self-attention mechanism allows the model to assign varying degrees of importance to different words in the input sequence, capturing complex relationships and dependencies between them. We utilized the **bert-base-cased** language model and fine-tuned it using our train set. We attempted to employ a simpler transformer model due to the limited instances available for fine-tuning.
To evaluate its performance, we considered recall, precision, f1 score, and accuracy. Employing this approach, we obtained the following outcomes:

Table 1. Results using bert-base-cased using multi-class multi-label classification

Recall	Precision	F1-Score	Accuracy
42.30	22.0	28.94	23.94

4.2. Semantic Textual Similarity (STS)

In this setup, we formulate the problem as semantic textual similarity (STS). We operate under the assumption that the similarity between requirements and regulations sentence embedding with a trace link is greater than a specified threshold. In this approach, unlike the multi-class multi-label classification, the model also takes the regulation text as one of its inputs. Moreover, sentence transformers are capable of having a better representation of sentences rather than word-level LMs [6]. The workflow of this approach can be seen in Figure 4.

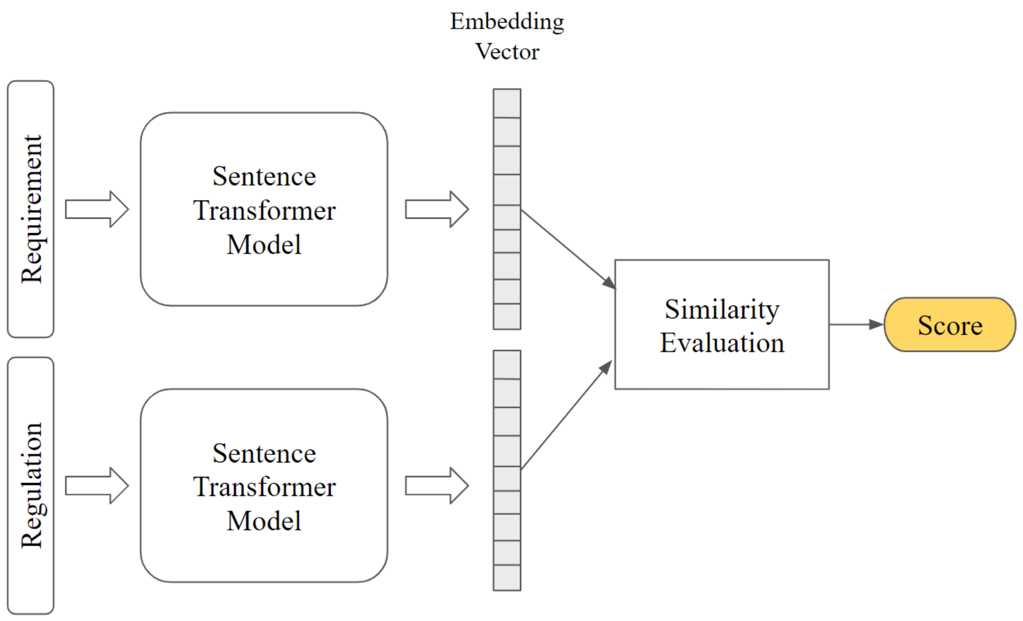


Figure 4. Illustration of the STS technique. Input comprises requirement and regulation texts, while the output encompasses the cosine similarity score between their texts.

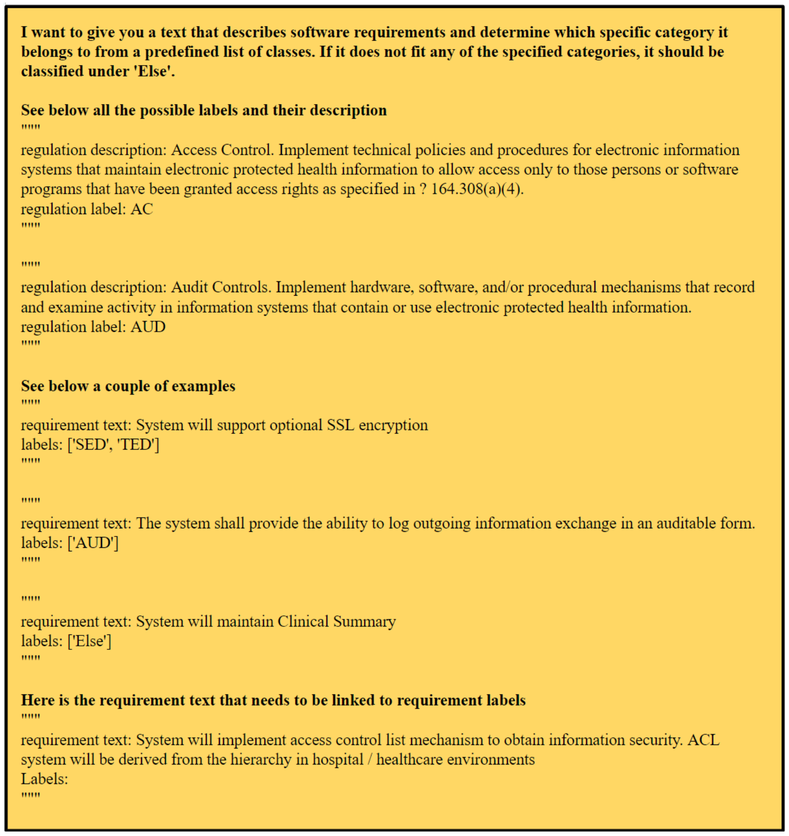
We employed the **paraphrase-mpnet-base-v2** model and fine-tuned it with our train set, using the cosine similarity metric to evaluate the similarity between two sentence embeddings. In positive examples (pairs of requirements and regulations that are linked), the similarity score is set to 1, while in negative examples (pairs of requirements and regulations not linked), the similarity score is 0. After fine-tuning the model, we employ a threshold of 0.5. This value is chosen based on the heuristic that the similarity should exceed the median value. To evaluate its performance, we considered recall, precision, f1 score, and accuracy.
Employing this approach, we obtained the following outcomes:

Table 2. Results using sentence transformers technologies as the STS problem

Recall	**Precision**	**F1-Score**	**Accuracy**
97.67	100	98.82	98.91

4.3. Prompt Engineering

Prompt engineering is an essential technique used to communicate effectively with generative AI models, such as Google's Gemini or OpenAI's GPT. It's akin to formulating the right question to get the best possible answer. The quality and structure of the input prompt significantly influence the model's output. In the context of this project, the prompts were designed to instruct the model to predict the regulatory codes for a given requirement text. The input and output of this approach are tied to the multi-class classification technique, with the input comprising solely the requirement text. In this setup, we employed a **few-shot prompting** which involves incorporating a small number of examples (shots) directly into the prompt itself (we've utilized both the entirety of the training instances and a subset comprising 20% of the training instances). This is powerful because it leverages the model's existing knowledge from pre-training while providing just enough specific guidance for the new task at hand.
Figure 5 shows an example of the completed prompt. The prompt template, used in out project, includes the following placeholders:
{examples}: A list of training examples, where each example consisted of a requirement text and a list of corresponding regulation codes.
{labels}: A list of all possible regulation codes.
{text}: The requirement text for which the model was to predict the regulatory codes.



I want to give you a text that describes software requirements and determine which specific category it belongs to from a predefined list of classes. If it does not fit any of the specified categories, it should be classified under 'Else'.

See below all the possible labels and their description
""""
regulation description: Access Control. Implement technical policies and procedures for electronic information systems that maintain electronic protected health information to allow access only to those persons or software programs that have been granted access rights as specified in ? 164.308(a)(4).
regulation label: AC
""""

""""
regulation description: Audit Controls. Implement hardware, software, and/or procedural mechanisms that record and examine activity in information systems that contain or use electronic protected health information.
regulation label: AUD
""""

See below a couple of examples
""""
requirement text: System will support optional SSL encryption
labels: ['SED', 'TED']
""""

""""
requirement text: The system shall provide the ability to log outgoing information exchange in an auditable form.
labels: ['AUD']
""""

""""
requirement text: System will maintain Clinical Summary
labels: ['Else']
""""

Here is the requirement text that needs to be linked to requirement labels
""""
requirement text: System will implement access control list mechanism to obtain information security. ACL system will be derived from the hierarchy in hospital / healthcare environments
Labels:
""""

Figure 5. An example of the utilized prompt

In this experiment, we used the latest version of Gemini (gemini-1.5-pro-latest) which is a cutting-edge Large Language Model from Google AI, recently entering public preview in April 2024 [7]. To explore the effectiveness of prompts, we employed varying portions of few-shot examples provided to the model. We assessed the LLM's effectiveness using a combination of metrics: recall, precision, F1-score, and accuracy. The results of this approach are presented in the following table:

Table 3. Results using prompt on Gemini LLM

#few_shots	Recall	Precision	F1-Score	Accuracy
100% of train	83.72	85.71	84.70	96.88
20% of train	90.69	86.66	88.63	97.58

5. Conclusion

In conclusion, we have explored the integration of Natural Language Processing (NLP) techniques to address the challenges of regulatory compliance in software engineering. Through the application of Multi-Class Multi-Label Classification, Semantic Textual Similarity, and Prompt Engineering methods on the HIPAA dataset, we have provided a comprehensive analysis of the potential of these NLP strategies to automate and enhance compliance operations. The study has shown promising results in terms of precision and recall, particularly with the use of sentence transformers and prompt engineering with large language models. Moreover, the report highlights the importance of balanced datasets and the adaptability of models to scenarios with limited data availability. This indicates that with a low number of training data instances, prompt engineering outperforms, especially when combined with sentence transformers, emphasizing their ability to provide informative sentence representations. Future research should aim to extend these methodologies to other domains and larger datasets to further validate their efficacy and robustness. Ultimately, the findings underscore the significant benefits of leveraging advanced NLP techniques in improving the management and implementation of regulatory requirements, thereby ensuring more reliable and secure software systems in regulated industries.

References

[1] Malinowski, T., et al. (2016). Automated Compliance Checking of Legal Documents: A Natural Language Processing Approach.
[2] Swamy, V., et al. (2019). Regulatory Compliance Monitoring Using Natural Language Processing.
[3] Jain, S., et al. (2020). Automating Regulatory Reporting Using Natural Language Processing.
[4] Rodriguez, J. A., et al. (2017). Ontology-Based Semantic Annotation of Regulatory Documents for Compliance Checking.
[5] Guo, Jin, Marek Gibiec, and Jane Cleland-Huang. "Tackling the term-mismatch problem in automated trace retrieval." Empirical Software Engineering 22 (2017): 1103-1142.
[6] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.
[7] Team, Gemini, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut et al. "Gemini: a family of highly capable multimodal models." *arXiv preprint arXiv:2312.11805* (2023).