CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

An NLP approach to assess information security policies

Application of GPT-3 within a policy domain

Master's thesis in Computer science and engineering

Hampus Lundblad
Pouya Faramarzi

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2022

Master's thesis 2022

An NLP approach to assess information security policies

Application of GPT-3 within a policy domain

Hampus Lundblad
Pouya Faramarzi

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2022

An NLP approach to assess information security policies
Application of GPT-3 within a policy domain
Hampus Lundblad
Pouya Faramarzi

Supervisor: Miroslaw Staron, Interaction Design and Software Engineering
Advisor: Daniel Dalevi, Centiro
Examiner: Lucas Gren, Interaction Design and Software Engineering

Master's Thesis 2022
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2022

An NLP approach to assess information security policies
Application of GPT-3 within a policy domain
Hampus Lundblad
Pouya Faramarzi
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Threats to companies' information security are ever-increasing, and to adequately protect the companies' information assets; a proper information security policy needs to be established. For this purpose, information security standards such as ISO 27001:2013, created by the International Organization of Standardization, exist. However, for a policy to be complete towards ISO 27001:2013, the policy must fulfill up to 114 different requirements, also called controls. Experts within information security policies often do this work, which can be time-consuming and error-prone.

Due to this, this study aimed to use natural language machine learning models to classify if a text extract from a given information security policy is complete towards a specified control or not. Ultimately the study wants to investigate whether language models are a good fit for software engineering topics that are also business-critical.

The study utilized the design science methodology. A framework for determining policy completeness was constructed and different natural language machine learning classifiers were evaluated. The main focus was on the large-scale pre-trained model GPT-3 by OpenAI. Three different datasets were constructed to train the models, each consisting of annotated text extracts from information security policy. These were labeled as either being ISO certified or not, depending on if the company, or the policy itself, mentioned an ISO certification. The models were then evaluated on these three datasets, where the metrics for evaluation were F1-score and accuracy. Lastly, a validation session with a policy expert from a case company that specializes in software solutions and policy compliance was conducted to determine how GPT-3's evaluation of policies compares to the evaluation of an expert.

The results showed that GPT-3 and the pre-trained word embedding model GloVe with SVC as a classifier could perform better in policy classification than other machine learning models. However, when compared to an expert, GPT-3 fails to distinguish between policies that are not complete towards ISO and policies that are partially complete towards ISO. Something which the policy expert was able to do.

We conclude that GPT-3 has the potential to perform well in the domain of information security policy. However, due to a lack of data and expertise in the domain of information security policies, the results from the validation session do not reflect this. Hence, the authors provide a discussion regarding this and recommendations for future work.

Keywords: software engineering, information security policy, ISO, NLP, OpenAI, GPT-3, machine learning

Acknowledgements

We would like to express our gratitude to our supervisor from Chalmers, Miroslaw Staron, who provided valuable and relevant feedback, helped us with the direction of the thesis, and provided support throughout the entire project. We would also like to thank Centiro for offering us the opportunity to pursue our thesis together with them. A special thank you to Daniel Dalevi, Mikael Böörs, Gustaf Stawåsen, and Thomas Herkel from Centiro for their expertise and support. Their feedback and perspective has helped us immensely. Lastly we would like to thank our examiner Lucas Gren, who read and gave us feedback on our thesis and how to complete it.

Hampus Lundblad, Gothenburg, June 2022
Pouya Faramarzi, Gothenburg, June 2022

Contents

List of Figures

List of Tables

1
Introduction

Compliance with International Organization of Standards (ISO) standards regarding information security policies requires an organization to have an information security policy. Still, it is not easy to create a good one. An information security policy alone is far from sufficient in terms of providing adequate safety measures for an organization, and due to the effects of the increased significance of information technology, it has also received considerable attention. Additionally, the protection and security of information assets by organizations have become an increasingly more challenging task as the complexity of security threats has also increased [8]. The fact that large and high status companies such as Twitch (game-streaming platform) have been subjected to a data leak [9] and Kaseya (IT company that provides software to, among others, COOP) have been subjected to a ransomware attack [10] within the last twelve months signalizes that no organization is truly safe. Thus, on behalf of the stakeholders, there now exists a pressure and demand that organizations accept their responsibility in terms of offering adequate information security measurements [11].

Well-known standards, such as the COBIT or the ones defined by ISO [11], provide guidelines and frameworks with various objectives that are used towards implementing a robust information security policy that reflect the needs and risks of an organization [12]. Although, it has been suggested that the guidelines provided by the standards are too generic, and organizations find it challenging to assess the completeness of their information security policy concerning them, it ultimately presents a key issue towards achieving certification [13] [14]. As a result, organizations find themselves in a time-consuming and expensive certification process [15].

A considerable amount of research has been conducted into assessing the completeness of privacy policies, mainly by utilizing machine learning methods [16] [17]. However, little research has been done within the domain of information security policies. Therefore it is not yet clear whether the use of machine learning methods can be modified to be applied within software engineering domains that are also business-critical. In particular, to assess the completeness of information security policies. Hence, additional studies on using machine learning methods within the domain of information security and its policies are needed.

This study aims to alleviate the aforementioned completeness issue by investigating to what extent *Natural Language Processing* (NLP) models can help with determining information security completeness in relation to the ISO 27001:2013 standard

with the intent to generalize the findings by offering a framework for completeness checking. Due to scarce publicly available information security policies, pre-trained language models, such as GPT-3, are used to maximize model learning rates with a small sample size.

Additionally, this thesis aims to provide a framework that can be used without expert analysis to adjust an organization's information security policy and fill the gap between the guidelines and individual characteristics of an organization by using accessible and inexpensive methods. Furthermore, there also exists a gap between defining a policy and applying it in practice. For example, it is possible for any organization to have an adequate information security policy in relation to ISO, but difficult to establish whether the organization actually implements the policy in practice by reviewing the policy alone. Hence, the study deals with "completeness", i.e., the presence of critical elements, rather than "compliance", in relation to ISO. In order to investigate the practical feasibility of the study as a real-world application and gain access to the domain knowledge of experts, a collaboration was established with Centiro. Centiro, which is henceforth referred to as this thesis's *case company*, is an organisation specializing in software solutions and policy compliance and is located in Borås, Sweden.

1.1 Practical scenario

The starting point for any organization to achieve ISO certification is to first define an ISO 27001 scope statement. The scope statement sets the boundaries on what processes, products, or departments an information security management system should cover within the organization. More importantly, the scope also allows for choosing what aspects of the ISO 27001 standard are needed to be implemented in order to be granted an ISO certification. These aspects are more commonly referred to as *controls* and the ISO 27001:2013 is made up of 114 of these controls. Hence, an organization is only needed to comply with controls that are deemed necessary based on their set scope. In other words, not all 114 controls are required in an information security policy for an organization to achieve ISO certification [18]. Thus, the first step of completeness checking becomes to determine the controls to check against and map what part of the policy corresponds to which control. The second step in the process is to assess the completeness of the controls by confirming the presence of key elements crucial to the controls. Experts in the field commonly perform the second step, and the key elements are defined by the characteristics of a control. In an ISO 27001 completing document, referred to as ISO 27002, a guidance for control implementation is provided for each control and can be used as grounds to identify the key elements [19]. Observe the activity diagrams in Figures 1.1 and 1.2 for illustrations of the process overview and an example of mapping and completeness checking of a control.

In the diagram given by Figure 1.1, a circle symbolizes the start and endpoint, a rectangle represents a process, and a rectangle (or rectangles) with a wavy bottom edge represents a document to serve as an input or output used by the processes.

Meanwhile, in the activity diagrams given by Figure 1.2, a circle symbolizes the start and endpoint, a diamond coupled with a question represents a decision, a diamond without question represents a merge, a green rectangle indicates a resulting successful action, and a red rectangle indicates a resulting unsuccessful action.

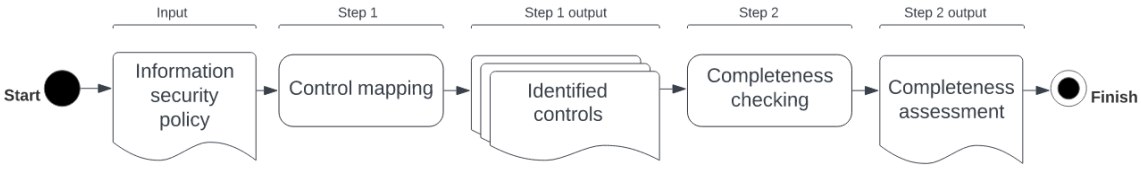


Figure 1.1: Activity diagram of a practical scenario.

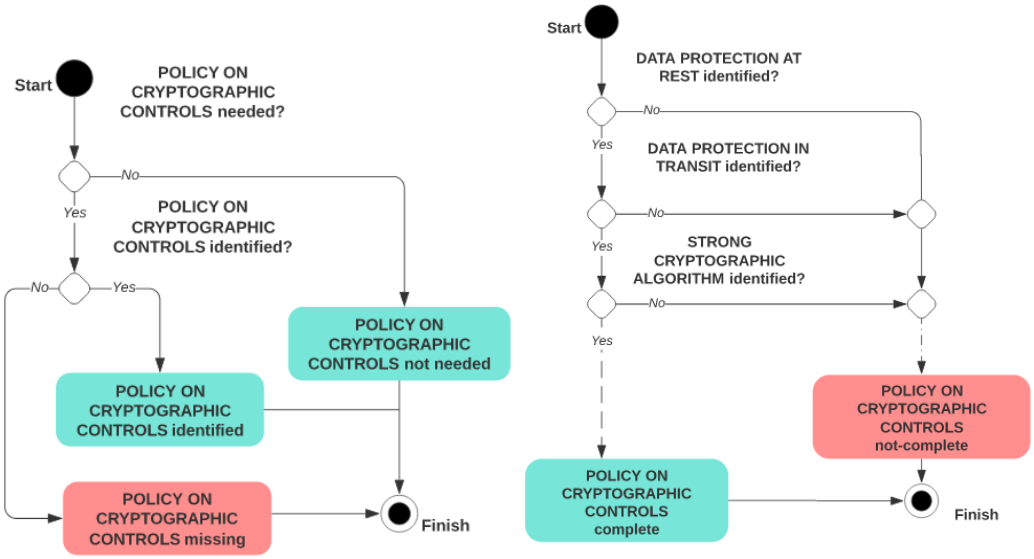


Figure 1.2: Activity diagrams of a control mapping (left) and a completeness check (right). As an example, the control of "Policy on the use of cryptographic controls" from ISO 27001:2013 was used and its implementation guidance provided in ISO 27002:2013 was referenced when determining examples of characteristics for completeness.

In Figure 1.1, the practical scenario is given from start to finish in an activity diagram on a holistic level. It is divided into one input, an information security policy, leading to two processes and two outputs, one for each step. The output from the control mapping step is a collection of extracts representing the result of annotating the information security policy by controls. The left activity diagram in Figure 1.2 is an example of such mapping, where a scope statement defines the need for the control [18].

After the analysis has been established, the second step in the process is to check the extracted control's completeness by confirming the presence of key factors. An example of this is given in the right activity diagram in Figure 1.2. The output from the completeness checking is then a completeness assessment where each control is

deemed either complete or incomplete. Of course, for an expert to manually annotate an information security policy that may have 114 control present and check for completeness of each one is not only time-consuming and labor-intensive, but also error-prone. Furthermore, a non-expert who lacks the knowledge of what is required to be complete in relation to ISO for each control, for example, what a strong cryptographic algorithm is, would struggle to define the lowest levels of the information security policy. Hence, supplying both experts and organizations with an automated solution is desirable.

An automated solution could also be of interest to software engineers, not only because it can dictate what software tools and methods of working are available to them, but also since it promotes the practice of information security aware development. Therefore, product owners who might lack legal expertise and legal experts who may lack knowledge of the software domain, should work in close collaboration to define a consensus on various elements for products and processes which could in its turn stem requirements for, for example, reliability and security.

1.2 Research questions

The following research questions are divided into two major themes where one is made towards establishing a quality framework for evaluating information security policies in order to identify missing ISO compliance factors (**RQ1**), while the second one focuses on data analysis, model analysis, model evaluation, and comparison (**RQ2**). Ultimately, the latter research question aims to support the first with a language model.

RQ1: *What characterizes a good quality machine learning framework based on factors such as the amount of data and manual labor needed for information security policy?*

RQ2: *To which degree does a GPT-3 language model determine the ISO completeness of various organizations' information security documents?*

RQ2.1: *Which machine learning model features are beneficial for determining document coverage and alignment in relation to ISO?*

RQ2.2: *To what degree can the features of GPT-3 enhance the document classification process, and how does it perform versus other algorithms?*

RQ2.3: *How does the GPT-3 model trained and evaluated on information security policy documents compare to performances of GPT-3 models applied in other domains?*

Addressing **RQ1** leads to the design of a framework, which lays the foundation for the document classification in **RQ2**. **RQ2**, a broader question, is further broken down into three sub-research questions. **RQ2.1** allows for experimentation and analysis of different machine learning algorithms (and their various properties) to establish an optimal benchmark that is needed to compare future results with. Answering **RQ2.2** leads to comparisons with the GPT-3 model and the previously established benchmark while also leading to an investigation into what aspects of

GPT-3 aids (or hinders) in its classification ability versus standard algorithms. Finally, addressing **RQ2.3** leads to comparisons of achieved performance results of GPT-3 with applications within other domains to deem whether information security policy completeness is a reasonable domain of application of GPT-3.

1.3 Delimitation

This thesis uses existing information security policies and already implemented natural language processing models and does not define nor create any new ones. Furthermore, the time restrictions of this thesis limit the size of the dataset of gathered information security policy documents; with more time, a more defined and versatile dataset could be created. Furthermore, this thesis does not deal with compliance but rather completeness, as the former requires more attention and involvement of ISO-compliance experts. Finally, this thesis does not attempt a real-world application and evaluation of models but focuses on designing a possible solution to the problem context.

1.4 Report structure

The remainder of this paper is divided into six chapters, where Chapter 2 covers the existing research and Chapter 3 provides the theory behind the study. Chapter 4 explains the execution of the study, while Chapter 5 presents the results. Afterwards, Chapter 6 provides discussion in relation to the results, threats to validity, and suggestions for future work. Finally, Chapter 7 provides explicit answers to the research questions together with final remarks.

2
Related Work

The area of text classification can roughly be divided into two categories [20], Rule-based methods and Machine Learning (ML) based methods. Rule-based methods require the researchers to have deep domain knowledge and use pre-defined rules to classify texts. In contrast, ML methods require models and pre-labeled data to learn the relations between the texts and their corresponding labels.

In this section, the area of machine learning methods in natural language processing is further studied to understand what methods researchers are currently using to achieve state-of-the-art performance in terms of preprocessing steps, models, resampling procedures, and evaluation metrics. Furthermore, a closer look at similar studies involving the classification of policies are provided, to better estimate what the field looks like and what approaches different researchers have taken to tackle the problem of policy classification. Lastly, it is vital to understand how large-scale models such as GPT-3 and BERT are used in current research, their limitations, and the steps needed to achieve good performance.

2.1 Text classification in Natural Language Processing

To better understand which ML models and preprocessing steps are commonly used in research concerning natural language processing, two main studies have been examined. First, Rahman et al. [21] used standard classification models, such as Support-Vector Machine (SVM) and Random forest, to classify sentiments of tweets in two different datasets. As a pre-sampling procedure, the researchers used K-fold cross-validation, where $k = 4$, to divide the data into training and validation sets. The best performing model (MaxEnt) achieved an average F1-score of 76%. Whereas an F1-score is a measurement of how well a model performs with the addition of taking into account any misclassification. [22] used ML models to classify emails as either phishing emails or not. The models used were SVM, Naïve Bayes, Decision Tree, Long Short Term Memory (LSTM), and Convolutional Neural Networks. The model with the best average accuracy was leveraging Convolutional Neural Networks. Furthermore, similar studies have been done by Miao. et al [23]. who used ML models to classify Chinese newspapers. The researchers used several different models, but the conclusion was that a Support Vector Machine (SVM) with a TF-IDF vectorizer yielded the best F1-score of 95.7%. Dadgar et al. [24] conducted a similar study; however, they analyzed English newspapers instead.

The evaluation was performed on two datasets, one from BBC, which contained five categories, and one from 20NewsGroup, which contained 20 categories. Their best-performing model was SVM with a TF-IDF vectorizer. The model was used on a dataset from the BBC and 20NewsGroup, where it achieved an F1-score of 95.67%.

Furthermore, a similar study was made by Tzimourtas et al. [25] where SVM, Random Forest, and Naive Bayes were compared on the 20NewsGroup dataset. The best scoring model was SVM, with an accuracy of 95%. Even though SVM is a well-performing classifier, other models can also be used for text classification. Kim [26] investigated if simple Convolutional Neural Networks (CNN) with a small number of hyperparameters could perform well on text classification tasks. The researcher tested four models on seven different datasets and managed to achieve a better performance when compared to other studies that were made at the time.

Sharma and Moh [27] conducted a study that used classifiers such as SVM, Naive Bayes, and a dictionary-based classifier to predict the outcome of the Indian election by determining the sentiment of tweets related to the election. The result was that the best performing model was SVM with an accuracy of 78.4%.

2.2 Policy classification

In policy classification, based on the studies included in this section, the most popular policy to classify were privacy policies. However, these still give a good overview of what different approaches are common when classifying policies.

Story et. al [28]. conducted a study where privacy policies of mobile applications were analyzed to determine if the privacy policy covered the kind of data that the application was accessing and potentially sharing. They divided the privacy policy into categories, such as *Email address*, which indicated whether the app collected the user's email address or not. For example the text "We collect your email address" would be classified as *True* by the *Email Address* classifier, but false by the *GPS Location* classifier. Furthermore, each classifier was split into classifying if a first-party or third party accessed the data. Hence both *GPS Location 1st Party classifier* and *GPS Location 3rd Party classifier* needed to be trained. The reason for dividing the classifiers in this manner was that data could be accessed by a first-party but not a third party in privacy policies. To train their models, Story et al. used an annotated set of documents, also known as an annotated *corpus*. Preprocessing steps such as stop word removal, vectorization, and normalizing of sentences were conducted in order to improve the models' performances. Along with vectorization, the authors also used a manually crafted vector of boolean values, which indicated the absence or presence of characteristic words. The model used in the research was SVC, and the result was a mean F1-score of 71% over the 26 objectives that were classified. The conclusions that the authors drew from the results were that compliance issues in mobile privacy policies were common and that their proposed model, along with a mobile application analysis, can improve privacy transparency.

Furthermore, [17] conducted a study where the researchers analyzed privacy policies and their completeness towards GDPR compliance. The analysis identified the absence or presence of metadata types in a text. For example, one metadata type was *processing purposes*, which concerned the "purposes of the processing for which personal data is being collected." The researchers used three different approaches to classify which metadata types were present in a policy. These approaches were a machine learning approach using Support Vector Machines (SVM), a similarity-based classification using cosine similarity of sentence embeddings, and a keyword-based classification method that compared sentences to keywords related to a specific metadata type. The result was that their completeness checking model, which used machine learning, had an F1-score of 91.47%. Compared to a keyword-based approach, this method improved the F1-score by 32.35%. Narksenee and Sripanidkulchai [29] have conducted similar studies using machine learning models to determine if an application's behavior complied with the application's privacy policy. Thotawaththa et al. [30] investigated how machine learning models such as BERT could be used to classify privacy policies. Additionally, Alabduljabbarr et al. [31] conducted a study with the goal of reducing the read-time on privacy policies from a user perspective. The researchers utilized machine learning and deep learning models to classify the content of the policies and reduce the number of paragraphs that the users needed to read. The models utilized preprocessing steps such as stop-word removal, lemmatization, stemming, TF-IDF, Doc2Vec, Universal Sentence Encoder, and WordPiece. An ensemble of six machine learning and deep learning models was used to classify the privacy policies. The result was an F1-score of 91% on their validation dataset, and after an user study, they concluded that the read-time was reduced by 39.14%.

Liang and Ye [32] conducted a study that aimed to create a classification process using three-way decisions [33] for inclusive policies. Their proposed model used a two-stage process. Firstly, an ensemble is trained and output a category with a confidence value (probability score). The value is then passed through a threshold, and if the probability score is lower than the threshold, the same data is then passed to a traditional machine learning model. With this setup, the researchers managed to achieve greater performance with their three-way decision model compared to ten other baseline models. The best-performing model used AdaCNN for the first stage and SVM for the second stage.

These studies give insight into how previous work on policies have been conducted. All the studies divided the policies into categories and then created a classifier for each category. However, the researchers utilized different approaches. For example, Story et al. [28] used a dataset that was annotated by domain experts and used the categories in the dataset. Thotawaththa et al. [30] used a combination of domain expert insight and user perspective to choose categories.

2.3 GPT-3 & BERT

In this section, studies related to GPT-3 and BERT is presented; these two models are also further explained in Chapter 3.

GPT-3, which is a pre-trained deep learning model by OpenAI, has been used for text classification purposes, such as classifying emails [34], detecting hate speech, and classifying racist or sexist texts [35]. However, GPT-3 has its limitations. One study by Moradi et al. [6] investigated if GPT-3 could perform well on text classification tasks in the biomedical domain. The conclusion was that the model could not achieve state-of-the-art performance on the chosen NLP tasks when trained on just a few examples.

BERT, which stands for Bidirectional Encoder Representation from Transformers, is another pre-trained machine learning model that has also been used in several studies. To understand how BERT can improve performance in binary classification tasks, Zhang and Zhang [36] conducted a study where BERT was used as an embedding layer for a downstream ML model. The researcher evaluated this model against benchmarks on the IMDB dataset. The result was that the model that used BERT as an embedding layer had an F1-score of 93.11%, which was an improvement of 2.01% compared to the best performing baseline model.

3
Background

This section introduces the domain background, i.e., an overview of the information security policy and the ISO 27001:2013 standard. Afterward, the necessary background pertaining to the technical approach is summarized.

3.1 Domain background

Information security policies represent an organization's ability to safeguard information assets proactively. In other words, they are meant to exist as documentation of an organization's approach to managing information security. They have, therefore, also become acknowledged as an organization's most crucial information security mechanism [37]. Information security alone is about providing "... protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in order to provide confidentiality, integrity, and availability" [38]. However, researchers argue that technical implementations alone are no longer adequate for protecting an organization's information assets and need to include more factors, such as the management and employees [8]. Thus, the information security policy, which provides "... directives, regulations, rules, and practices that prescribe how an organization manages, protects and distributes information" [39], has also become recognized as a crucial business document of any organization [8].

3.2 ISO 27001:2013

Implementing an information security policy alone is not sufficient to safeguard the organization's information assets. Although no perfect security and protection plan exists to this date, proper framework and technique implementation in the shape of various security standards help in minimizing the risk from harmful exploitation and establishes the best practices for information security management within an organization [12][8]. ISO 27001 is an example of such a security standard for information management systems and it provides rules and guidelines for organizations to follow in order to decrease the risk of information and information systems being exposed [11]. Increasing an organization's compliance to such standards, therefore, assists with establishing a robust Information Security Management System (ISMS) [8].

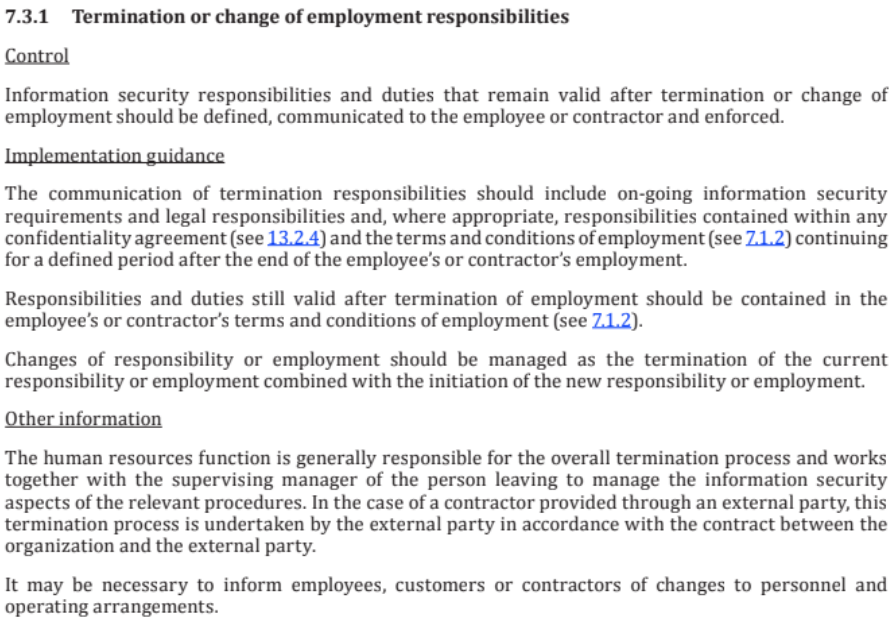
The standard, in its entirety, specifies 114 controls divided into 35 control objectives which are further divided into 14 domains. Where a control is a type of safeguard, a control objective is a statement that defines the result of implementing said control/controls, and a domain is a grouping of control objectives that belong to a specific theme [11] [40]. The need for each control to be implemented is defined by an ISO 27001 scope set by an organization prior to applying for an ISO certification. Hence, only a subset of controls are required to be ISO compliant, but most commonly, all. Observe the extract from ISO 27001 in Figure 3.1 for an example of an entire domain.

A.7 Human resource security		
A.7.1 Prior to employment		
Objective: To ensure that employees and contractors understand their responsibilities and are suitable for the roles for which they are considered.		
A.7.1.1	Screening	*Control* Background verification checks on all candidates for employment shall be carried out in accordance with relevant laws, regulations and ethics and shall be proportional to the business requirements, the classification of the information to be accessed and the perceived risks.
A.7.1.2	Terms and conditions of employment	*Control* The contractual agreements with employees and contractors shall state their and the organization's responsibilities for information security.
A.7.2 During employment		
Objective: To ensure that employees and contractors are aware of and fulfil their information security responsibilities.		
A.7.2.1	Management responsibilities	*Control* Management shall require all employees and contractors to apply information security in accordance with the established policies and procedures of the organization.
A.7.2.2	Information security awareness, education and training	*Control* All employees of the organization and, where relevant, contractors shall receive appropriate awareness education and training and regular updates in organizational policies and procedures, as relevant for their job function.
A.7.2.3	Disciplinary process	*Control* There shall be a formal and communicated disciplinary process in place to take action against employees who have committed an information security breach.
A.7.3 Termination and change of employment		
Objective: To protect the organization's interests as part of the process of changing or terminating employment.		
A.7.3.1	Termination or change of employment responsibilities	*Control* Information security responsibilities and duties that remain valid after termination or change of employment shall be defined, communicated to the employee or contractor and enforced.

Figure 3.1: ISO 27001:2013 extract of the domain Human resource security.
The table is an extract from table A.1 in Annex A of SS-EN ISO/IEC 27001:2017 and is reproduced with due permission from SIS, the Swedish Institute for Standards, who holds the copyright and also sells the complete standard www.sis.se.

In Figure 3.1, the domain is defined as Human resource security (A7) and consists of three control objectives: A.7.1, A.7.2, and A.7.3. The control objectives pertain to the different possible statuses of employment. These control objectives then have six controls: A.7.1.1, A.7.1.2, A.7.2.1, A.7.2.2, A.7.2.3, A.7.3.1. The implementation of these controls is then what is required to fulfill the control objectives and ultimately

also be compliant with the domain [18]. Furthermore, each control is supplied with implementation guidance in an ISO 27001 completing document known as ISO 27002:2013. For an example of such guidance, observe the extract from ISO 27002 provided in Figure 3.2.



7.3.1 Termination or change of employment responsibilities

Control

Information security responsibilities and duties that remain valid after termination or change of employment should be defined, communicated to the employee or contractor and enforced.

Implementation guidance

The communication of termination responsibilities should include on-going information security requirements and legal responsibilities and, where appropriate, responsibilities contained within any confidentiality agreement (see 13.2.4) and the terms and conditions of employment (see 7.1.2) continuing for a defined period after the end of the employee's or contractor's employment.

Responsibilities and duties still valid after termination of employment should be contained in the employee's or contractor's terms and conditions of employment (see 7.1.2).

Changes of responsibility or employment should be managed as the termination of the current responsibility or employment combined with the initiation of the new responsibility or employment.

Other information

The human resources function is generally responsible for the overall termination process and works together with the supervising manager of the person leaving to manage the information security aspects of the relevant procedures. In the case of a contractor provided through an external party, this termination process is undertaken by the external party in accordance with the contract between the organization and the external party.

It may be necessary to inform employees, customers or contractors of changes to personnel and operating arrangements.

Figure 3.2: ISO 27002:2013 extract of the implementation guidance for the control A.7.3.1 defined in ISO 27001.
The text is taken from SS-EN ISO/IEC 27002:2017 and is reproduced with due permission from SIS, the Swedish Institute for Standards, who holds the copyright and also sells the complete standard www.sis.se.

In the Figure 3.2, the implementation guidance provides more information regarding what the implementation of the control related to termination or change of employment responsibilities should cover. However, the guidance is not tailored to the control requirement of organizations, and its implementation may also not be sufficient to pass a certification [19].

3.3 Natural language processing

Natural Language Processing (NLP) is an area within AI that is concerned with using computational science to process natural language data to enable machine learning model construction. More specifically, by utilizing various NLP tools, any human language set of documents can be processed and represented in numerical forms that can be used in conventional data analysis or machine learning techniques [41]. The relevant NLP tools and techniques used in this study are mainly related to text normalization and text vectorization. Where text normalization defines a standardized word format, and text vectorization maps the elements of a text to a numeric format.

3.3.1 Text normalization

Text normalization consists of a set of tasks pertaining to converting natural language text to a simplified and standard format that enables comparison with other normalized texts by eliminating various redundancy and anomalies through generalization. Among these tasks, a few are commonly found in any normalization processes and are also relevant to this study. These are mainly word tokenization and word format normalizing [42].

The task of word tokenization is the simplification part of text normalizing and consists of segmenting (or *tokenizing*) words from a running text and adding these to a comprehensive set (or *vocabulary*). This is also known as a text parsing operation and acts as an entry point toward word format normalizing. Meanwhile, word format normalizing comprises tasks that change the segmented words into a standard format that is defined by a chosen pipeline of tasks [42]. Case folding, stop-word removal, and lemmatization are a few examples of what could be included in that pipeline. Refer to the list below for a description of each task.

Case-folding: Maps all letters to lower case such that, for example, *Policy* and *policy* are represented the same. Case-folding, due to its simplicity, has been recognized as a common practice among practitioners, and thus its usage also enables popular NLP libraries, packages, and word lists [43] [44]. However, a disadvantage of using case-folding is the inherent ambiguity [42]. For example, words such as *GloVe* a method for learning word embeddings and *glove* a clothing item would be considered the same.

Stop-word removal: Removes a class of words known as *stop-words*. Stop-words are words that are frequently present in any text, such as *has* and *a*. Therefore, their presence is trivial in most use-cases. The removal can be done by removing stop-words by using a predefined list or by removing a top percentile of words in any vocabulary set [42].

Lemmatization: Maps all variation of a word to its corresponding root (or *lemma*) [42]. For example, *has*, *had*, *have*, and *having* are mapped to their shared lemma *have* and *recognized* and *recognize* to its lemma *recognize*.

For a visual representation of each task, observe the example given in Figure 3.3. In the Figure, the first row of the first column demonstrates the tokenization process. In contrast, the second column represents the output of each previously mentioned task with the tokenized text as input. Finally, the second row of the first column provides an example output of how a result of the text-normalization process could look like if all the tasks were to be used in a pipeline.

3.3.2 Text vectorization

Text vectorization consists of a set of tasks in NLP pertaining to mapping words or sentences from a text to a vector within a predefined vector space, also known as a *vector space representation*. This process is also more commonly known as a *word embedding* or *embedding* technique. The embedding may take on different repre-

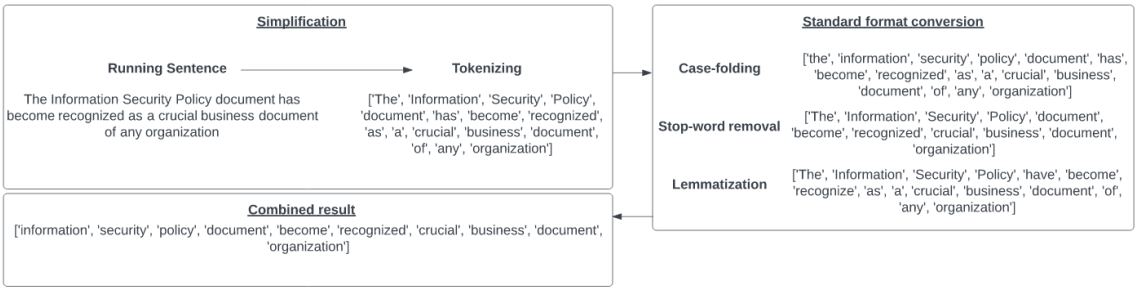


Figure 3.3: Text normalization example of processing a sentence with each of the following NLP tasks: tokenization, case-folding, stop-word removal, and lemmetization.

sentations depending on the vector space utilized and the corresponding embedding technique used to map to it [42]. Moreover, the embedding techniques can also be context-insensitive and context-sensitive [45].

Context-insensitive embedding techniques such as *Frequency* and *Sequence* based representations primarily deal with mapping single words to single vectors [45]. Whereas a frequency-based representation has a word frequency-based mapping while a sequence-based representation has a mapping with a focus on sequencing sets of words [42] [46].

Context-sensitive embedding techniques such as *contextual* based representations, on the other hand, maps multiple contexts for the same word to multiple vectors [45].

Although all embeddings use multidimensional vectors and can be coupled with machine learning algorithms, their success in various practical applications and insights gained from these applications may differ [46]. For a simple example of how a basic frequency based embedding could look like, observe Figure 3.4.

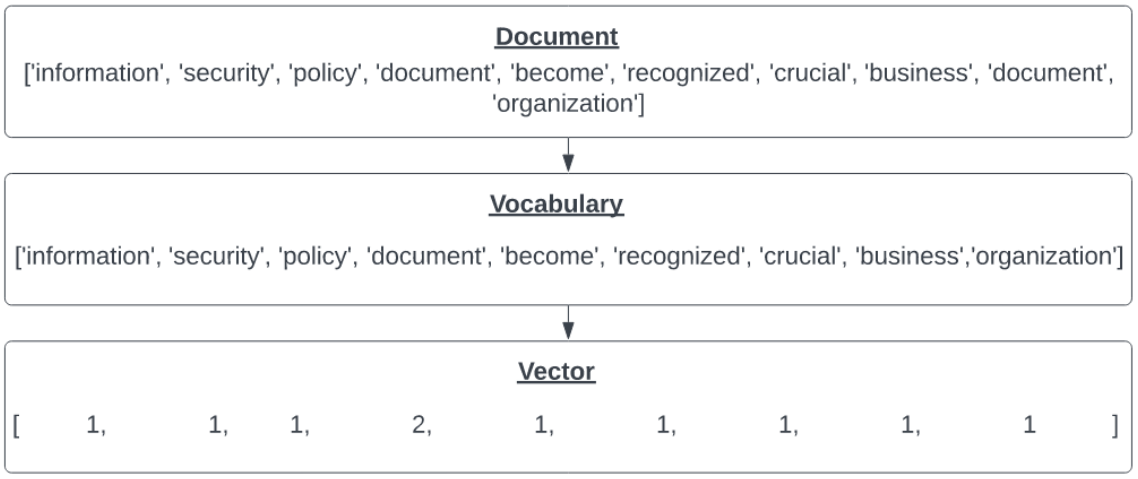


Figure 3.4: Text vectorization example using a frequency based approach.

In Figure 3.4, the box on the top represents the input into the vectorizer. The middle box is the vocabulary that stems from the input. Finally, the box in the bottom is the resulting vector from processing the input in relation to the vocabulary. This approach simply counts all the occurrences in the document and is therefore considered a frequency based representation.

3.3.2.1 Frequency based representations

Frequency-based word embedding is the most commonly used vector space representation and uses an approach of counting word occurrences to construct sparse multidimensional numeric vectors. More specifically, a set of words is mapped to a matrix where each word corresponds to a column in the matrix, and its frequency is contained in the rows [46].

The resulting frequency-based matrix is characterized as being high-dimensional and sparse. This is due to large vocabulary sizes that directly correspond to a large set of columns (i.e., words) where most of the rows (i.e., word frequencies) are zero since each document only contains a small subset of the comprehensive vocabulary [46]. Furthermore, an important observation is that a frequency approach considers the order and position of words as irrelevant. Hence, this approach is also known as a *bag-of-words* approach [47]. An example of word embedding techniques that utilize the bag-of-words approach is the *Term Frequency-Inverse Document Frequency* (TF-IDF) vectorizer.

The TF-IDF vectorizer uses the product of two terms. The first term is the term frequency, i.e., the frequency of a word in a text, and the second term is the inverse of the document frequency, i.e., the presence of a word across all documents. Hence, the values are either zero or a positive real value. While the first term alone is sufficient in certain applications, the second term provides a normalization factor such that words that appear in few documents are given a higher weight [42]. However, using both the terms may also give higher weight to errors and misspellings that were not captured during the preprocessing step. Therefore, the choice of whether to use a pure term frequency or TF-IDF can be application, and corpus specific [46].

3.3.2.2 Sequence based representations

Sequence-based word embedding is built on the notion of a distributional structure suggested by Harris [48] which states that similar words tend to occur in similar contexts. Hence, this embedding class uses an approach of capturing useful syntactic and semantic properties in a given text in order to construct its vectors [42].

In contrast to the frequency-based representation, a sequence-based word embedding investigates the likelihood of a set of words ending up near one another by training a machine learning prediction model. The learned weights are then transferred to the word embedding matrix. As a result, unlike a frequency-based word embedding, which utilizes the entire vocabulary, the sequence-based one registers fewer properties. Therefore, the resulting matrix from sequence-based word embedding is

characterized as dense, short, and can contain any real-value [42].

Two examples of prominent sequential word embedding techniques are *Word2Vec* and *Global Vectors* (GloVe). Although both use local context to capture various word semantics, i.e., semantics between words within a defined set size, GloVe takes it one step further to include global context. More specifically, GloVe also attempts to find semantic relationships between the words on a corpus level by utilizing global corpus statistics such as word co-occurrence probability ratios [45]. However, the techniques also have a shortcoming of poor prediction on words that have previously not been seen, i.e., out-of-sample predictions.

3.3.2.3 Contextual based representations

Contextual-based representations provide representations of words in context. Unlike frequency and sequence-based representations that utilize a single vector embedding per word, contextual-based representation yields an entirely new vector every time a word is encountered in a new context. The vector is then a representation of a specific word type in a specific context. This embedding can be used to compare differences between two words in a context and determine their similarities [42].

3.3.3 Pre-trained models

In recent years, researchers have found out that pre-trained NLP models often outperform models that have been built from scratch by using task-specific corpora. More specifically, these are pre-trained models that have learned relevant information from large sets of corpora prior to being applied to a new practical application [49]. This discovery has not only led to an increase of publicly available pre-trained word embedding models [50], but has, together with *transformer* models and contextual embeddings, also enabled a new functionality referred to as *few-shot learning*.

Few-shot learning models require only a few samples to achieve good performances. Its emergence is built on predecessors that, with the years, have become more advanced and inherits increasingly more parameters and improvements in terms of successful NLP task accomplishment. They are then trained on enormous corpora to require little fine-tuning and still be able to achieve state-of-the-art performance. Observe Figure 3.5, for an overview of the size difference between a few pre-trained language models. In the figure, the two models GPT-3 by OpenAI [1], and BERT by Google [5] are presented along with their sizes and are of interest to this study.

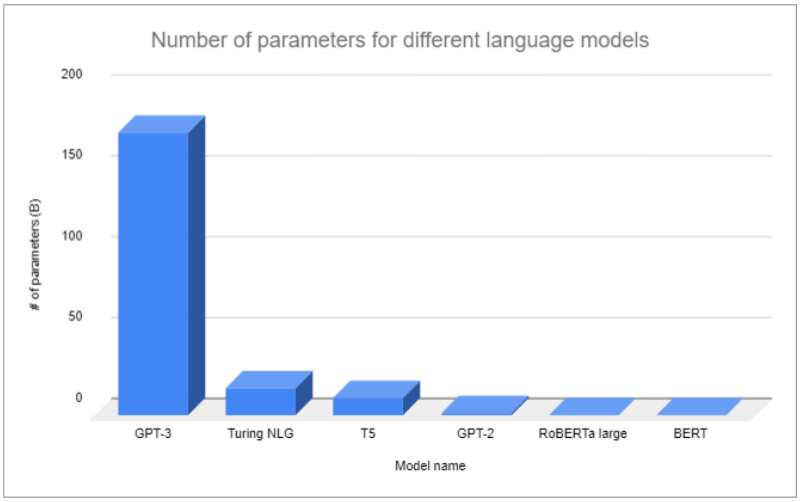


Figure 3.5: A graph displaying different NLP models along with the number of parameters for each model [1], [2], [3], [4], [5].

3.3.3.1 Transformers

One of the main enablers of large pre-trained models is the neural network architecture *Transformer*, as described by Vaswani et al. [51]. The purpose of the Transformer, as described by the authors, was to create an architecture that was less complex and more efficient, compared to models utilizing recurrence and convolutions. Therefore the Transformer utilizes the Attention mechanism, which has the advantage of only requiring O(1) sequential operations, while a Recurrent Neural Network requires O(n) operations, where n is the sequence length. Furthermore, it uses a multi-headed self-attention operation over the input context tokens followed by position-wise feed-forward layers to produce an output distribution over the target tokens. This has been shown to outperform other machine learning models on various tasks such as machine translation and language modeling.

The self-attention layer in the *Transformer* architecture builds on the attention mechanism proposed by Bahdanau et al. [52]. The self-attention layer allows the model to simultaneously attend to different parts of the input sequence. It has been used in several works [53][54][55] but then often in combinations with RNN. Vaswani et al. proposed that there is no need for using RNNs and only the attention mechanism is enough.

In the Transformer model, Vaswani et al. used Scaled Dot-Product Attention which is defined in 3.1. In the equation, Q, K, and V represent a Query, a Key, and a Value, each of these values are words from the input sentence. d_k represents the keys of dimension. Scaled dot-product attention is almost identical to normal dot-product attention apart from the use of the factor $\frac{1}{\sqrt{d_k}}$. Vaswani et al. motivated this factor by indicating that for large values of d_k, the dot-product itself will grow very large. The Attention score in 3.1 is calculated for each word in the input.

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{3.1}$$

A simplified overview of the Transformers architecture can be seen in Figure 3.6. The use of shifting the output, along with the masked multi-head attention layer, ensures that output prediction only relies on inputs which are previous to the output. [51].

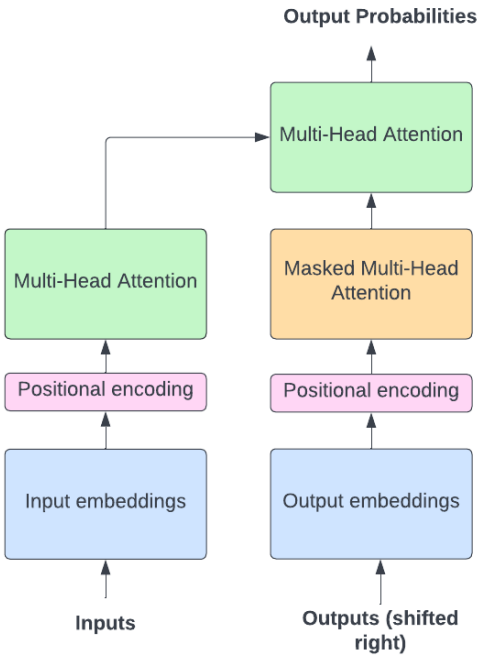


Figure 3.6: A simplified overview of the transformer's architecture.

3.3.3.2 GPT-3 by OpenAI

GPT, which stands for Generative Pre-trained Transformer, was first introduced by Radford et al. [56] in 2018. This research aimed to create a model that could achieve strong natural language understanding without the need for large changes when applying the model to different tasks such as entailment tasks, similarity tasks, question answering tasks, and commonsense reasoning tasks. GPT uses a multi-layer *Transformer decoder* model [51] due to its excellent transfer performance on different tasks.

The training phase of GPT consisted of two stages. Firstly it is trained on a large corpus of unlabeled data. Secondly, the model's parameters are adapted using discriminative fine-tuning. To accomplish the first stage, the researchers use standard language modeling to calculate the likelihood L, which depends on the conditional probability P. The conditional probability is modeled using a neural network where

the parameters are trained using *Stochastic Gradient Descent*. For the second stage, the goal is to maximize the likelihood seen in 3.2

$$L_2(C) = \sum_{(x,y)} P(y|x^1, ..., x^m) \tag{3.2}$$

Where C is a labeled dataset, x is the input tokens, and y is the labels. The model was trained on the BookCorpus dataset, containing 7000 unpublished books. Then, to benchmark the model, it was fine-tuned on another set of data depending on the task. This resulted in GPT achieving state-of-the-art performance on 9 out of 12 datasets.

However, GPT was only the first iteration of this model. In 2019 Radford et al. [3] released a new study where they had conducted further research to create a new model, which was called GPT-2. GPT-2 also uses the *Transformer* architecture, and its foundation is similar to the original GPT, however, with a few changes. Most notably, the researchers' largest version of GPT-2 contains 1.5 billion parameters and 48 layers. Furthermore, the vocabulary was expanded, context size was increased, and a larger batch size was used. The model was trained on the WebText dataset, which the researchers created using web scraping techniques focusing on retrieving only high-quality documents. The resulting dataset consisted of over 8 million documents [3]. GPT-2 achieved state-of-the-art performance on 7 out of 8 studied datasets in its *zero-shot* setting, meaning that it was not fine-tuned on any training data before evaluation.

The most recent iteration of GPT, called GPT-3, was proposed by Brown et al. [1] in 2020. The trend continues to expand the number of parameters for the model. In total, eight models were created, with the number of parameters ranging from 125 million to the largest having 175 billion parameters[1]. The largest model is the one known as *GPT-3*. Compared to GPT-2, a few modifications were made, but it still follows the same architecture. It was trained on a larger dataset than GPT-2 and nearly matched performance to fine-tuned models on benchmark datasets. The authors noted that this is a promising result due to GPT-3 only requiring 10-100 examples in its few-shot setting to achieve good performance. Comparing this to fine-tuned models, which can require training labeled datasets with hundreds of thousands of examples.

3.3.3.3 Using GPT-3

GPT-3 cannot be used in an offline fashion compared to models such as BERT and models from *SciKit-learn*. These models can be downloaded to the computer and trained, evaluated, and validated without an internet connection. GPT-3 is different since its primary method of communication is an API. The API gives the user access to OpenAI's file uploading system, use of their models, fine-tuning of models, and also the creation of embeddings.

GPT-3 has a playground mode which will not be used in this study since it is ineffective when classifying larger quantities of data. The playground mode presents

a prompt to the user where it is possible to input tasks to GPT.3. GPT-3 can be asked to complete a sentence, classify an animal, or translate something from one language to another. A example of this can be seen in Figure 3.7. The "=>" sign, which is seen in the Figure, is called the separator, and it tells GPT-3 where the task ends. This sign can be chosen arbitrarily as long as it is not present anywhere else in the prompt. The playground mode will not be used largely in this study. However, it can be good for demonstration purposes and to explore how GPT-3 behaves when certain tasks are fed.

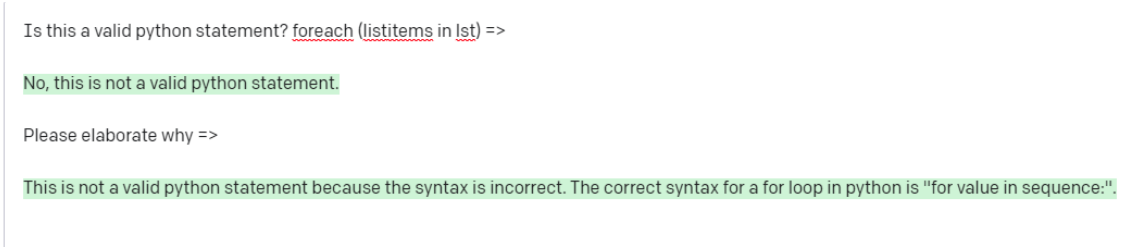


Figure 3.7: The GPT-3 playground where the user can input tasks. In this case GPT-3 was asked to validate if a given code line is valid in the programming language Python. The green text is GPT-3's response.

Furthermore, OpenAI provides an API for communicating with GPT-3. The API can be accessed by OpenAI's Javascript library, Python library, and CURL commands. These methods make it possible to input classification tasks using code rather than the playground prompt. This study will mainly focus on using the Python library rather than the playground prompt, as it enables multiple requests without pasting the tasks into the playground prompt.

3.3.3.4 Using GPT-3 as a classifier

GPT-3 can be used as a classifier for simpler tasks in its text completion mode. In Figure 3.8, GPT-3 is used to classify if a language is either an object-oriented language, or a functional language. This classification works well, and if the use case were to use GPT-3 as a language classifier, it would work. However, when tasking GPT-3 with more complicated domains, such as information security policies, its text completion mode is insufficient. This is displayed in Figure 3.9 where it determines that *Encryption* is not part of ISO 27001:2013, even though it is. For this reason, this study will use GPT-3 in its classifier setting instead.

Example of using GPT-3 as a classifier [1] GPT-3, in its few-shot setting, works a bit differently than its text completion mode. The whole procedure can be seen in Figure 3.10. The main differences to text completion are that examples of the data need to be uploaded to OpenAI. GPT-3 then uses the most relevant example data to classify the input.

Using GPT-3 as a classifier can be divided into the following steps,

1. Format the data in JSONL format

[1] https://beta.openai.com/docs/guides/classifications

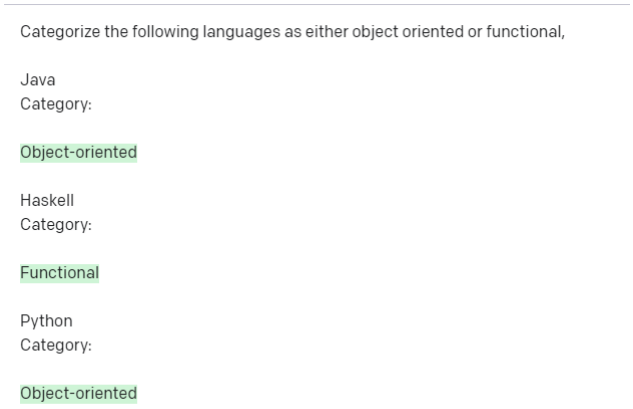


Figure 3.8: The playground prompt where GPT-3 is used as a classifier in its text completion mode. The green text is GPT-3's response.

Figure 3.9: The playground prompt where GPT-3 is used as a classifier in its text completion mode. The green text is GPT-3's response. Here GPT-3 fails to classify encryption as a part of the ISO 27001:2013 standard.

2. Upload the data to OpenAI, the API will respond with a file id corresponding to the uploaded data. This needs to be saved for later use.
3. Use either CURL commands, the Python library, or the Javascript library to send a classification request to GPT-3.

In order to improve the classification shown in Figure 3.9, the steps mentioned can be used as follows.

1. First, some example data needs to be provided, which will be created manually for this example. The input to GPT-3 will be of the format "Is [X] part of ISO 27001:2013?". The sample data can be seen in table 3.1, which will need to be formatted into JSONL[2] before submitting it to OpenAI
2. Using Python with the openai library will submit the data to OpenAi.
```
openai.File.create(file=open("example_data.jsonl"),
purpose="classifications").
```
3. Then the following code will submit the query to GPT-3 for classification.
```
model = openai.Classification.create(
file=fileid,
```

[2]https://jsonlines.org/

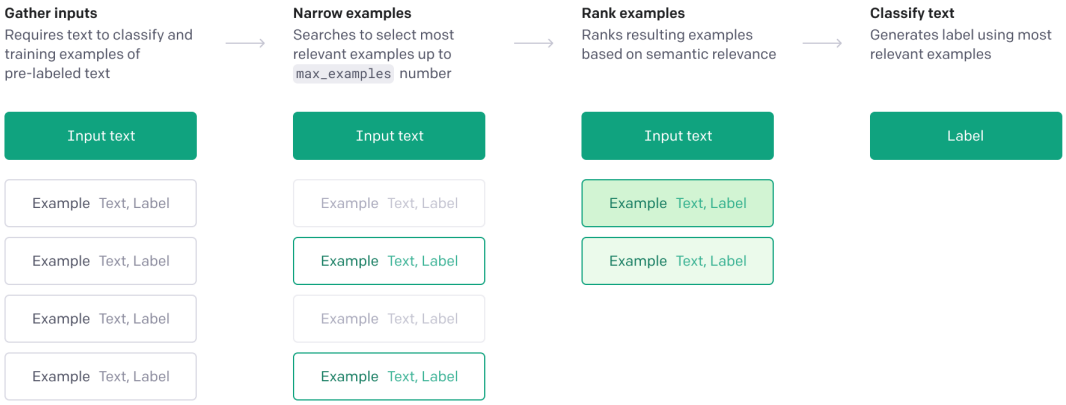


Figure 3.10: The four step procedure taken by GPT-3 when used in its few-shot setting. Source of the image is https://beta.openai.com/docs/guides/classifications

```
query="Is Cryptography part of ISO 27001:2013?",
search_model="ada",
model="curie",
max_examples=5
)
```

4. The response from GPT-3 can be seen in Figure 3.11. GPT-3 can now predict that Cryptography is part of ISO 27001:2013. The score indicates how useful GPT-3 deems the example to be. All examples seem to have a rather high score, indicating that GPT-3 is somewhat unsure of how to use these examples. However, this small example still gives insight into how GPT -3's classification works.

.

Text	Label
Is User Access Management part of ISO 270001:2013?###	Yes
Is Information Classification part of ISO 270001:2013?###	Yes
Is geography part of ISO 270001:2013?###	No
Is handling of squirrels part of ISO 270001:2013?###	No

Table 3.1: The data used in the example.

3.3.3.5 BERT by Google

BERT, which stands for Bidirectional Encoder Representation from Transformers, is another pre-trained model similar to GPT-3 but uses a different architecture. BERT was originally proposed by Devlin et al.[5]. The model is designed to pre-train deep bidirectional representations from unlabeled texts by jointly conditioning on both left and right contexts in all layers. BERT was built on top of the Transformer, an attention-based model that learns contextual relations between words in a text. The

Figure 3.11: Response from GPT-3 when sending the query "Is Cryptography part of ISO 27001:2013?"; done using the python library OpenAI. GPT-3's answer can be seen at line 3 where it labels the input as *Yes*. In the *selected_examples* column returns how useful the uploaded examples were in classifying the input. Higher score means more useful.

Transformer was initially designed for machine translation, but BERT is adapted for natural language understanding tasks such as sentence classification, question answering, and next sentence prediction. BERT represents words in a text as vectors, or "embeddings". These embeddings are learned jointly with a two-layer bidirectional Transformer encoder. The Transformer encoder reads the text input sequentially and learns to predict words that are masked (replaced with [MASK]) or randomly shuffled. The training process of BERT is unsupervised, meaning that it does not require labeled data. This makes it more efficient and scalable than previous models trained on labeled data. BERT is effective because it can capture the context of a word in a sentence, rather than just the word itself. This is due to the bidirectional nature of the Transformer encoder.

3.4 Support Vector Machine (SVM)

A support vector machine (SVM) is a kernel linear classifier [57]. SVM works in all dimensions [58], but for simplicity's sake, this section only considers the linear case.

SVMs are based on the concept of dividing the data into classes by using the best fitted decision boundary [57], and this can be observed in Figure 3.12. In higher dimensions, the decision boundary is referred to as a hyperplane. For example, if the data were in three dimensions, the decision boundary would be a plane.

The equation for calculating the decision boundary is given by equation 3.3. In the equation, **w** are the weights, **b** is the bias, and x is the decision point. +1 and -1

are used to label the two classes.

$$\mathbf{w}^{\mathbf{T}}\mathbf{x} + b = \begin{cases} \geq 0 & \text{class } +1 \\ < 0 & \text{class -1} \end{cases} \tag{3.3}$$

However, using this form of classification can lead to misclassification if the points are close to the boundary line with small changes to $\mathbf{x}$ [57]. Therefore to make the model more robust, a parameter ϵ can be added, which denotes how far the closest data point can be separated from the decision boundary. This is also referred to as a margin and the SVM model will attempt to fit a hyperplane that can maximize it. Reasoning behind this is that the larger a margin is, the larger the distance is between the classes, and therefore the easier it is to differentiate between them. The decision boundary can then be written as equation 3.4 and can also be observed in Figure 3.12.

$$\mathbf{w}^{\mathbf{T}}\mathbf{x} + b = \begin{cases} \geq \epsilon^2 & \text{class } +1 \\ < -\epsilon^2 & \text{class -1} \end{cases} \tag{3.4}$$

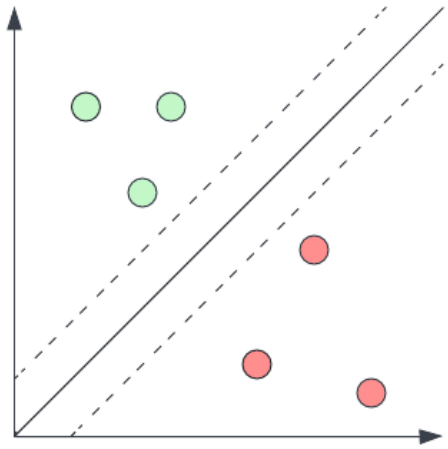


Figure 3.12: The SVM classifier using a boundary line to separate and classify the datapoints.

4
Research Design

This study followed the research method *Design Science Methodology* (DSM) as described by Wieringa [59]. The purpose of DSM is to identify a problem, suggest an artifact that may improve the problem context, and validate whether the artifact operates as intended. This process is defined as a *Design Cycle* and is to be performed as an iterative process. The design cycle is part of a larger problem-solving cycle defined as an *Engineering cycle*. Wieringa defines the engineering cycle with the following five tasks:

1. Problem investigation
2. Treatment design
3. Treatment validation
4. Treatment implementation
5. Implementation evaluation

The design cycle is task one through three of the engineering cycle and may be performed numerous times before attempting a real-world attempt of the treatment, i.e., task four and five. For a visual representation of the engineering and design cycle, refer to Figure 4.1.

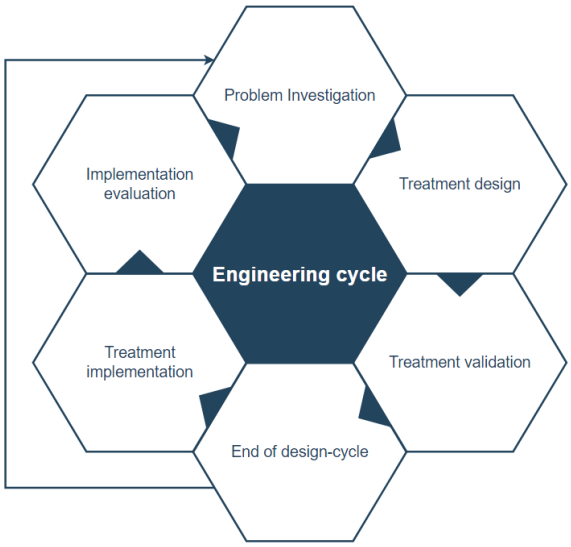


Figure 4.1: The engineering and design cycle as defined by Wieringa.

The methodology behind this study had a design cycle focus, and any real-world treatment implementation and evaluation are left as possibilities for future work.

Hence, this section also follows the design cycle structure by outlining the problem investigation and the methodology utilized to design and validate an artifact. Although the design cycle is an iterative process, this section discusses the tasks in a linear fashion.

In the problem investigation section, the problem context is introduced as provided by the case company. The problem context is then further investigated for the purpose of identifying and defining the problem. After the investigation, an artifact that interacts with the defined problem is suggested in the treatment design stage. Finally, in treatment validation the validation methods for the suggested artifact that were used are presented. Furthermore, a discussion of how the research questions are intended to be answered is provided in this section.

4.1 Problem Investigation

The case company, which specializes in software solutions and policy compliance, defined their problem context as information security policy content review in relation to ISO and expressed a desire for improvement. Hence, the problem context was investigated together with the experts at the case company by conducting continuous knowledge transfer sessions in the form of meetings, where frequent discussions regarding their difficulties and work processes were held. These sessions were performed in order to add more background to the problem context and the current solution. For more details on these sessions, refer to Section 4.4.

Among the identified difficulties was the time-consuming, labor-intensive, and error-prone process of manually checking for completeness of each ISO control (as described in section 1.1). Furthermore, the accepted level of information security that is defined by ISO is rather vague and may be formulated differently depending on the characteristics of an organization, despite using the same standard. Automating the completeness checking process by building machine learning models that are trained on the different variations can save a lot of time for the expert performing the information security policy review in relation to ISO. The saved time can instead be allocated towards controls that either fail the completeness check or are more difficult to classify.

Hence, the problem investigated in this study was to determine the completeness of information security policies in relation to the ISO 270001:2013 standard utilizing different classical and modern NLP and machine learning methods. The main stakeholders affected by the study were any organization concerned with information security, policy experts, and researchers from, but not limited to, the disciplines of software engineering and applied artificial intelligence. The project has the possibility of providing organizations with an affordable method to achieve an acceptable level of information security and/or detect shortcomings in an information security policy in relation to ISO. To establish a solution for this problem context, information security policies needed to be gathered in order to evaluate a machine learning model and framework that ultimately became the design artifact of this study.

4.2 Treatment Design

This section provides details on the approach used by the study for designing an artifact. More specifically, descriptions of understanding and preparations of data, the creation of baseline and benchmark models, and GPT-3 models are provided here. Furthermore, a discussion of what a good quality machine learning framework could be, i.e., **RQ1**, is also provided.

4.2.1 Data understanding

The ISO 27001:2013 standard consists of 114 controls belonging to 35 control objectives which in its turn belong to 14 domains. In other words, a domain is at the highest level, control objective is at the middle level, and a control is at the lowest level. Hence, the possible levels of data granularity to use for model learning are those three, and a large portion of all the domains should be present in any ISO certified information security policy, as established in 3.2. However, the structure of information security policies, regardless of ISO certification, can look quite different from one policy to another.

The differences are not only due to a normalized structure not being used, but also organizations' unwillingness to share their policy at the lowest level as information security research has been defined as "one of the most intrusive types of organization research" [60]. Hence, the lowest level of an organization's information security policy may often be hidden, and its higher levels may be too different from other policies, and/or the information may be too spread out across the policy to define where it is covered. Which ultimately lead to a major realization and decision for the course of this study.

The realization is the fact that it may be impossible to measure the completeness of an information security policy in relation to all the 14 domains, 35 control objectives, or 114 controls. Mainly due to the reasons mentioned above, but also due to time limitations. Therefore, for this study, the focus shifted to establishing a proof-of-concept of measuring completeness in relation to a few controls, control objectives, or domains, rather than the entirety of the ISO standard. The collection of controls were carefully selected together with an expert that had an insight into what content was most often present in any information security policy without having access to the lowest levels.

4.2.2 Data collection and overview

The data collection was done in iterations. It started out with finding information security policies that were publicly available on organization websites and later also moved to contacting organizations through customer support or directly through phone or email. When searching for policies on the internet, search strings such as *information security*, *information security policy*, *isp*, *public information security*

policy, and *security policy* were used, see Appendix A.1 for the full list.

When contacting customer support through email, a pre-written email was sent, which disclosed who the authors were, what the purpose of the data collection was, and that anonymity would be guaranteed if they decided to share their policy, see Appendix A.2 for the template. When choosing companies to contact, a randomly selected subset of 30 was used from the list of the 100 largest companies in Sweden by turnover[1]. The full list of contacted companies can be observed in Figure 4.2.

Company name	Response
[Company]	Information security policy received
Volvo	Redirected to telephone exchange, no success
Skanska	
Vattenfall	Redirected to HR, no success
Electrolux	Redirected to telephone exchange, no success
Securitas	
Webhallen	
Atlas Copco Power Technique Nordic	
Atlas Copco Kompressorteknik	
Industrial Technique Tools Nordics	
Atlas Copco Rental	
Assa Abloy	
Scania CV AB	
Scania AB	
Telia Compan	
H & M Hennes & Mauritz AB	
Nasdaq AB	
SSAB AB	
Lundin Energy AB	
Cytiva Sweden AB	
Indutrade AB	
Pfizer Health AB	Will not hand out
CDON	
Elgiganten	
NetonNet	
Spotify AB	
[Company]	Two information security policies received
Circle K Sverige AB	
Systembolaget AB	

Figure 4.2: The full list of contacted companies. In the **Name** column, the contacted company's name is listed. In the **Response** column, the results from the exchanges are listed. The cells were marked green if a response was received where they disclosed that they would, or would not, share their policy, or if they would redirect us. From two companies information security policies were received, therefore their name was redacted to [Company].

The industries and geographical location of the organizations were diversified, where some had more exposure than others. For example, the leading industry from the gathered dataset was Academics, and the leading country was the UK.

As for the structure of the documents, it varied greatly from policy to policy. The assumptions made in the previous section were quickly found to be true as the policy

[1]https://www.largestcompanies.com/toplists/sweden/largest-companies-by-turnover

content could, for example, be in bullet points, clearly marked out sections, or in plain text. Few policies also referred to internal documents to which access was only allowed to authorized personnel.

The biggest difficulty faced with data gathering was the sparse publicly available data that could be collected, but also confirming ISO certifications. This was made through third-party websites that have a record of ISO 27001:2013 certified companies and extensive searching on company websites.

In total, the dataset that was later annotated contained 49 information security policies. The distribution of the dataset with respect to labels was fairly even and can be observed in Figure 4.3.

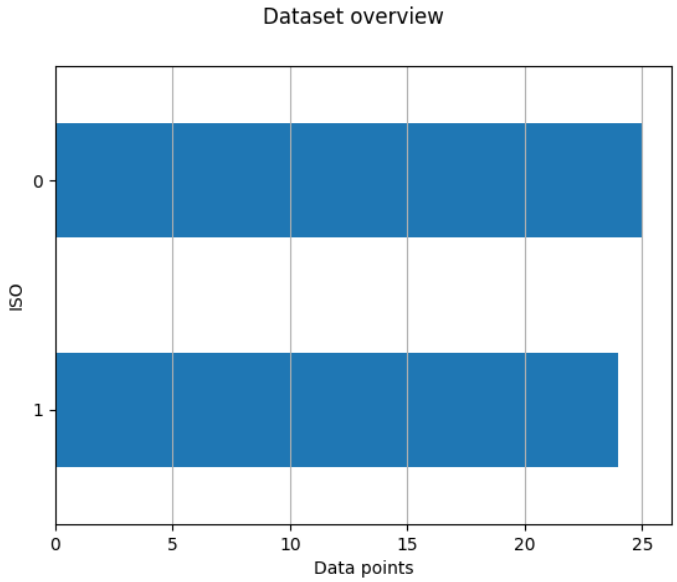


Figure 4.3: A bar-plot of the dataset containing ISO and non-ISO data points.

4.2.2.1 Data annotation

After a data gathering iteration had been performed, the data was extracted and annotated by the authors according to a set of controls and also labeled if it was ISO certified or not. The set of controls was selected together with a policy expert such that the annotations could be carried out by non-experts since it is a time-consuming task. Hence, the data was annotated by two non-experts and was annotated twice per selected set of controls in order to avoid conflicting perspectives. Observe Figure 4.4 to see the template data-sheet that was used for data annotations in this study.

file_name	ISO	Found_control	Annotation_1	Annotation_2	prompt
organization1.pdf	0	0	0	0	[Insert extract here]

Figure 4.4: The template used for data annotations including an example for the sake of illustration.

In Figure 4.4, the first column corresponds to the file name in the dataset and exists to not confuse with un-annotated files. The first column was also immediately dropped before any data processing begun. The second column corresponds to the ISO label of the information security policy. The third column represents whether the control was existent in the policy or not. If it was not found and the policy was labeled as a non-ISO, a random number generator was used for the selection of random text somewhere in the policy. The reasoning behind this was to fill out with data that the model could still learn from. Conversely, if the control was not found and the policy is labeled as an ISO, that extract would be left empty to avoid biasing the model. The fourth and fifth column were used as a method for reducing bias in the annotation process. The first annotator extracted the text and labeled the data point, then put a "1" in the *Annotation_1* column to indicate that the data point had been annotated once. The second annotator then reviewed this annotation, if the second annotator agreed, then the data point was seen as completed. However, if the annotators disagreed, then the data point was discussed until a consensus was reached or be brought up with a policy expert from the case company (see section 4.4). Finally, the sixth column is the extract from the policy that was later used by the models.

4.2.3 Modeling

In order to answer **RQ1**, a machine learning framework needed to be established and the decision was to be made based on the factors of data and manual labor needed at each data granularity level. The data granularity levels are defined by the level of detail in the texts [61]. For example, a domain level is a very high-level statement since it contains the least amount of detail.

The need for data to be sufficient in training a successful model is determined by an estimation of the number of controls covered and their inherent complexity. To illustrate, a domain that contains the most controls, each with their own complexity, is estimated to have a higher need for data than a singular control. In other words, the less detailed level of data granularity chosen, the more data is needed to cover the inclusion of additional controls. This argument is founded in that the more controls are included, the more variance and variables have to be taken into account and thus would have a higher data demand. On the other hand, the more detailed level of data granularity chosen, the tougher it is to identify and annotate without the help of experts.

After many discussions, a consensus was reached together with experts that the finest level of data granularity, i.e., a control level, was the most preferred level of granularity. This choice was made on the basis that a control-level model brought the most benefit for the experts as that was where the completeness checking was most often made. Additionally, this put less pressure on the already scarce dataset. To avoid difficulties with annotations, the controls were selected such that non-experts could also identify them with a few guidelines from the experts.

The final architecture of the framework is presented in Figure 4.5 and remained to be evaluated to deem whether the estimated factors were sensible and if it performed better than baseline models.

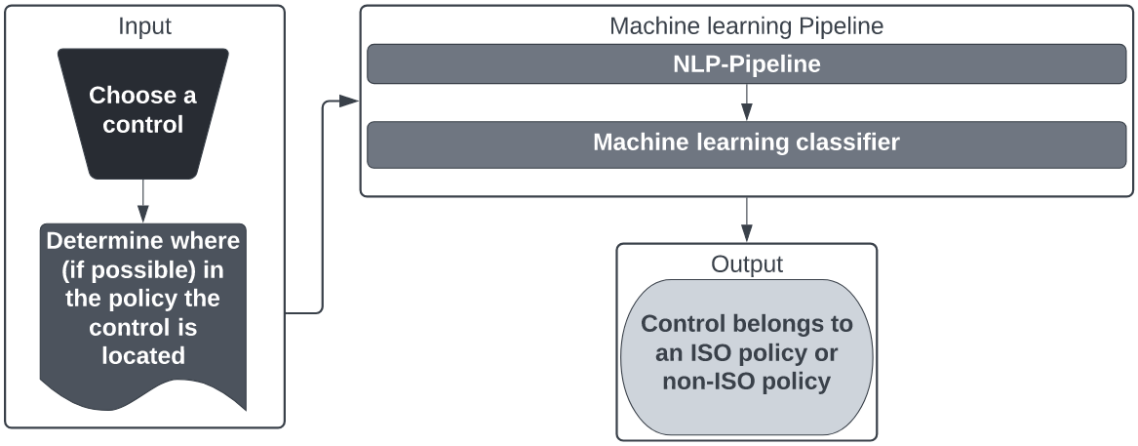


Figure 4.5: Framework architecture for information security policy control classification in relation to ISO.

In Figure 4.5, the input to the framework is defined in the box to the left. It consists of an user inputted selection of control and an extract corresponding to the control from the information security policy in question. The box in the top right represents the machine learning pipeline and consists of an NLP pipeline that is combined with a binary machine learning classifier. Finally, the output in the box to the bottom right is the result from the classifier and outputs either a one for the control having ISO level content and zero if it does not.

4.2.3.1 Control selection

The selected controls were selected together with a policy expert with an additional assumption that the control was present in most public information security policies. Furthermore, the selected controls also varied in factors that needed to be completeness checked against. In other words, this meant that the length, wordiness, and different ways to formulate the extracts varied between the controls. The purpose behind this was to measure the model's success with even the more difficult controls to check completeness against. The selected controls are listed in Table 4.1.

4.2.3.2 Data preparation and models

The data was prepared before being processed by machine learning models by first applying it through an NLP pipeline. Whilst there were many different algorithms and packages to utilized for each task in the pipeline, the overall process is as depicted in Figure 4.6.
In Figure 4.6, documents (or control extracts from information security policies) are inputted into a machine learning pipeline. The pipeline itself consists of two sections. The first is the NLP pipeline, and the second is to combine it with a machine learning classifier. In combination, an output of a classification label should be

Control id	Control name	Control definition
A.5.1.2	Review of the policies for information security	The policies for information security shall be reviewed at planned intervals or if significant changes occur to ensure their continuing suitability, adequacy, and effectiveness.
A.7.2.2	Information security awareness, education and training	All employees of the organization and, where relevant, contractors shall receive appropriate awareness education and training and regular updates in organizational policies and procedures, as relevant for their job function.
A.9.2.3	Management of privileged access rights	The allocation and use of privileged access rights shall be restricted and controlled.

Table 4.1: Table of selected controls.
The table is an extract from table A.1 in Annex A of SS-EN ISO/IEC 27001:2017 and is reproduced with due permission from SIS, the Swedish Institute for Standards, who holds the copyright and also sells the complete standard www.sis.se.

expected. The NLP pipeline comprises of three tasks: text parsing, text normalization, and text vectorization. Meanwhile, the machine learning classifier consists of a single binary classifier. For a more detailed view of each task within the machine learning pipeline, observe Figure 4.7.

In Figure 4.7, text parsing is defined by tokenizing the text and acts as an input to the text normalization task, which has the three sub-tasks: case-folding, stop-word removal, and lemmatization. For case-folding, the python standard library string operations were used. For the stop-word removal and lemmatization, a pre-defined list of stop-words and a lemmatizer algorithm known as the WordNetLemmatizer, of which both came from the NLTK-library, were used. The following task, text vectorization, consisted of the three different types of word embedding models defined in 3.3.2. The TF-IDF vectorizer from the scikit-learn library was selected to represent the frequency-based model. For the sequence-based models, Word2Vec and GloVe models from the Gensim library were selected. Finally, GPT-3 from OpenAI and BERT from the TensorFlow library were selected for the contextual-based models. The output from the NLP pipeline was then inserted as input into the binary machine learning classifier, which was chosen to be the LinearSVC model from the scikit-learn library. The LinearSVC model was essentially a SVM that used a linear separator as described in 3.4. GPT-3 and BERT, on the other hand, did not need a stand-alone binary classifier as the models handled the classification internally.

The main reasoning behind the selected combination of sub-tasks used for text normalization was to mimic the preprocessing of other pre-trained word embedding models [43] and maximize the gain from their usage by aligning the tokenized words. These pre-trained word embedding models were mainly related to the sequential embeddings of Word2Vec and GloVe, where Gensim offered a wide variety of them. For this study, the *word2vec-google-news-300* and *glove-wiki-gigaword-300*

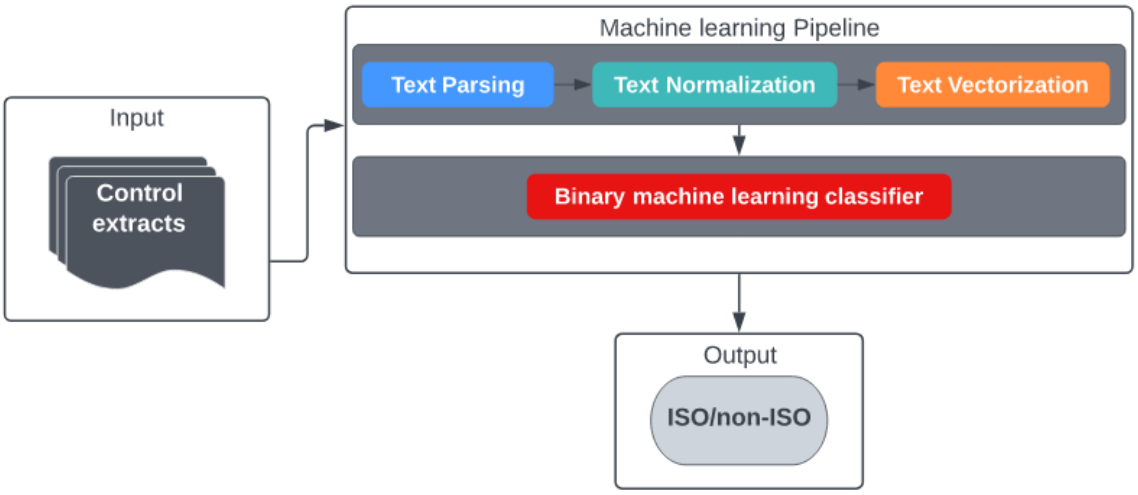


Figure 4.6: The overall machine learning pipeline. From documents, into the machine learning pipeline which consists of an NLP pipeline and a machine learning classifier, and yields an output.

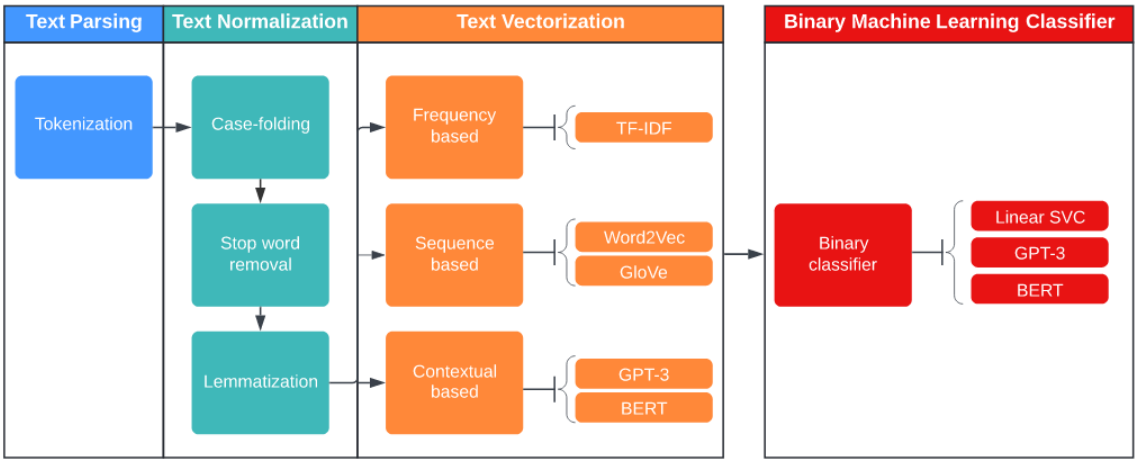


Figure 4.7: Detailed view of the machine learning pipeline in Figure 4.6 including sub-tasks and models used.

pre-trained embeddings were selected [50]. The word2vec-google-news-300 was a collection of 300-dimensional pre-trained word vectors and used Word2Vec as a basis for its learning technique. The model had been extensively trained on text corpora from Google News and had a total of three million vectors. Similarly, the glove-wiki-gigaword-300 was also a 300-dimensional pre-trained collection of word vectors but used GloVe as a basis for the learning technique. The model had been extensively trained on text corpora from Wikipedia, an online encyclopedia, and English Gigaword, which was an archive of newswire text data. Finally, glove-wiki-gigaword-300 had a total of 400 000 vectors [50]. These two pre-trained models are referred to as pre-trained Word2Vec and pre-trained GloVe henceforth.

4.2.4 Using GPT-3

In this section, the methodology for using GPT-3 is presented. Firstly, the method for the few-shot mode is shown, which includes how the data was uploaded to the model and the different parameters. Secondly, the method for how the fined-tuned model was used is presented.

4.2.4.1 Uploading data

For both the fine-tuning mode of GPT-3 and the few-shot mode, the data that the models used needed to be uploaded to OpenAI. At the time, OpenAI only supported the use of JSONL[2] files which were similar to usual JSON files where the difference is that each line in the file is a valid JSON value. These were structured in the following way.

```
{"text":"Text data of ISO certified policy +
\n\n===\n\n", "label":"iso"}
{"text":"Text data of a policy that is not certified +
\n\n===\n\n", "label":"oth"}
```

Where $\backslash n\ \backslash n === \backslash n\ \backslash n$ was used as the separator.

4.2.4.2 Few-shot mode

The data was uploaded to OpenAI for classification. Time.sleep(20) was used since it took some time for OpenAI to upload and process the text.

```
def _upload_training_file(self,path):
 response =  openai.File.create(file=open(path), purpose="classifications")
 time.sleep(20)
 self.training_file_id = response.id
```

After the data was uploaded, a model was created, the hyperparameters for the model can be seen in table 4.2. The hyperparameter *File id* corresponded to the uploaded data and changed when using K-fold cross-validation, since one id was needed for each *fold*. *Prompt* was the input data that was classified.

Setting	Value
Model name	curie
Prompt	The text to be classified
Search model	ada
Logprops	2
Max Examples	3
Expand	Completion
Training file id	The id of the uploaded files.

Table 4.2: The hyperparameters of GPT-3, in its few-shot setting, when classifying Information Security Policies. In this table, logprobs = 2 indicated that the model returned the logarithmic probability values.

[2]https://jsonlines.org/

4.2.4.3 Fine-tuning mode

GPT-3's fine-tuning mode worked similarly to the few-shot mode, where the main difference was that the fine-tuning mode was not yet supported by the OpenAI Python library, and therefore the OpenAI CLI and CURL commands needed to be used instead. The process can be described as follows,

1. **Prepare the data:** The OpenAI CLI provided commands which prepared the training data. The command established that the training data had a separator at the end of each prompt, that the label started with a whitespace, and that the JSONL file was formatted in the correct way.
2. **Create fine-tuned model** Using the OpenAI CLI, creating a model was done by executing the command

   ```
   openai API fine_tunes.create -t
   <TRAIN_FILE_ID_OR_PATH> -m <BASE_MODEL>
   ```

   OpenAI responded with a model name which was saved, and this made it so that the model could be reused unless the model needed to be re-trained.
3. **Using fine-tuned model** Using the fine-tuned model was done by the following OpenAI CLI command

   ```
   openai API completions.create
   -m <FINE_TUNED_MODEL> -p <PROMPT>
   ```

   Where *FINE_TUNED_MODEL* was the model name that was retrieved from the previous step, and *PROMPT* was a selected text from the testing dataset. To evaluate the model, this command needed to be executed for each value in the testing dataset. Then the returned value was compared to the true value to compute the F1-score.

4.3 Treatment Validation

In this section, the metrics that were used to evaluate the models are presented, and methods of validation are described. The model validation in this study was made in three steps in order to answer **RQ2** and a few of its facets. This section follows the same structure as those steps, with the exception of explaining the metrics and tools used for comparison first.

The first step in the model validation process was to compare models and word embedding techniques in order to investigate which model features were best suited for each dataset. To perform this step, a collection of models were trained on a training set and then evaluated on a testing set. The results were then compared between the models in terms of F1-scores and accuracy and then analyzed in order to provide an answer for **RQ2.1** and partially for **RQ2.2**. The second step was to validate the performance of the framework in combination with GPT-3 by comparing its results to an annotated validation dataset that had been annotated by a policy expert. This step was mainly performed to evaluate whether the framework and model functioned

as intended, i.e., if it could aid or replace an expert in determining completeness. The result from this step complemented the previous results and together provided a complete answer for **RQ2.2**.

The third, and final step, was to investigate the use of GPT-3 in other domains, such as biomedicine, and compare them to the achieved performance in this study in order to determine whether GPT-3 was a good fit for information security classification. The results from the other domains was attained from various research papers that have used GPT-3 for text classifications. The result from this step provided an answer to **RQ2.3** and thus also allowed for a discussion of **RQ2**.

4.3.1 Metrics and tools

The metrics and tools used to evaluate and validate the various models are defined in this section. Besides checking the accuracy of how a model performed on a testing set, the metrics of precision, recall, and F1-score were used along with the method of K-fold cross-validation.

4.3.1.1 Precision, Recall and F1-score

When dealing with imbalanced datasets evaluation metrics such as Precision and Recall is preferred [62]. The reason for this is that analyzing how many true positives a model produces can lead to high accuracy even though the model labels all texts the same. Therefore in this paper the chosen evaluation metrics were *Precision*, *Recall*, and *F1-score*. These are based on the Confusion Matrix seen in Table 4.3, and are defined in 4.1, 4.2 and 4.3.

	Predicted Positives	**Predicted Negatives**
Real Positives	TP	FN
Real Negatives	FP	TN

Table 4.3: Confusion Matrix, here TP = True Positives, FN = False Negatives, FP = False Positives, TN = True Negatives.

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.1}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.2}$$

$$\text{F1-score} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{4.3}$$

4.3.1.2 K-fold cross-validation

K-fold cross-validation is a method for validating a model by partitioning the dataset into k subsets, training the model on $k - 1$ subsets, and testing the model on the

remaining subset [63]. This is repeated until all subsets have been used as the testing set. K-fold cross-validation was used across all data sets, with 75% of the total dataset being partitioned as a training set and the remaining 25% as a testing set. The final result was calculated as a mean of all results across the k subsets.

4.3.2 Model comparisons

The model comparisons was performed by comparing GPT-3 to various benchmark models in order to validate its performance and answer the question of which characteristics were desired for the classification of the datasets. These benchmark models are defined in Table 4.4.

Model name	**Word embedding type**	**Word embedding model**	**Classifier**
ZeroR	-	-	Zero Rule Classifier
TF-IDF	Frequency	TfidfVectorizer (scikit-learn)	LinearSVC
Word2Vec	Sequential	Word2Vec (Gensim)	LinearSVC
Word2Vec (pre-trained)	Sequential	word2vec-google-news-300 (Gensim)	LinearSVC
GloVe (pre-trained)	Sequential	glove-wiki-gigaword-300 (Gensim)	LinearSVC
BERT	Contextual	bert_en_uncased (TensorFlow)	BERT

Table 4.4: Table of benchmark models with their word embedding types and models as well as the combined classifier.

In Table 4.4, the model names are given along with their word embedding type, word embedding model, and classifier coupled with. The same model names is referenced in the upcoming sections. An additional model that has previously been left unnamed is also present in the table. The model, referred to as the *ZeroR*-model, is a Zero Rule Classifier. A ZeroR Classifier is a classifier model that predicts all data points to the most frequent class and is a common model baseline to validate performance against. Additionally, it is also a method for determining if the model in question is a useful predictor. To illustrate, a ZeroR model predicts the same as voting just ones or zeros in favor of the class that has the most dominance in any given dataset and ignores any possible predictors [64]. In the case of this study, the classifier predicted all data points as "Non-ISO" as the dataset was slightly leaning more towards Non-ISO labeled data points rather than ISO.

The validation of GPT-3, on the other hand, consisted of evaluating the two models, few-shot and fine-tuned, on the annotated dataset. For both models, there was one common parameter, the *Model* parameter. These consisted of Davinci, Curie, Babbage, and Ada. The differences between the models were which tasks they could perform well on, how costly they were to use, and how efficient they were when it

came to training. Overall, Davinci was the best performing model, but it was more costly and slower to train. On the other hand, Ada was the simplest model and therefore cheaper to use and trained in less time than Davinci.

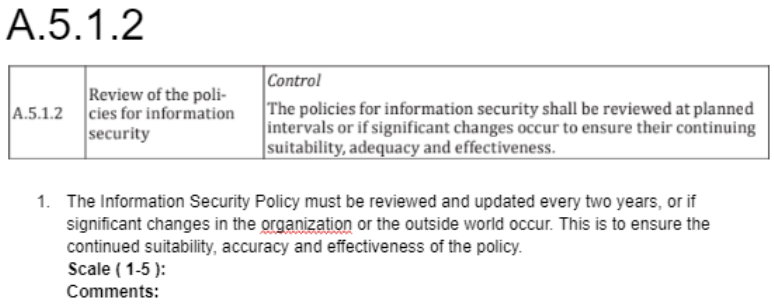
In this study, for GPT-3 Few-shot, Davinci and Curie were used as the classifier models, and Ada and Curie were used as search models. While for GPT-3 fine-tuned, Davinci and Ada were used as classifier models.

For the few-shot model, three parameters were changed during the validation process. There were *max_examples*, which determined how many examples GPT-3 was given during interference time, and K, which determined how many subsets the dataset was divided into (see section 4.3.1.2), and lastly the model name. For the fined tuned model, only the model was changed.

4.3.3 Expert validation with the case company

The expert validation was carried out with policy experts from the case company. This was done to determine how GPT-3's predictions compared to experts in information security policies. The process consisted of the creation of a new dataset for each control, referred to as a validation dataset. These datasets each consisted of ten extracts from information security policies that corresponded to each selected control which no model had previously seen nor been evaluated on.

The expert validation session was carried out in the form of a demo session where an expert was quickly briefed on the control in question and was asked to annotate ten extracts relating to that same control. The annotation was carried out while the expert used a thinking-out-loud method and ranked each of the extracts on a Likert scale from one to five. Whereas one represented that the expert strongly disagreed with the extract being an ISO-level defined control, and five represented that the expert strongly agreed. For a visual representation of the template that was presented to the expert, observe Figure 4.8.



A.5.1.2

A.5.1.2	Review of the policies for information security	*Control* The policies for information security shall be reviewed at planned intervals or if significant changes occur to ensure their continuing suitability, adequacy and effectiveness.

1. The Information Security Policy must be reviewed and updated every two years, or if significant changes in the organization or the outside world occur. This is to ensure the continued suitability, accuracy and effectiveness of the policy.
   Scale (1-5):
   Comments:

Figure 4.8: The template used for expert validation session with an example for the sake of illustration.

Afterward, the expert was also given a few features in the form of words and segments of words to rank on a Likert scale. The features were ranked in terms of how

well they represented the ISO label for the control in question and was extracted from vectorizers such as the TF-IDF vectorizer. However, the feature needed to have a certain level of statistical significance since using all of the features would be highly inefficient. To accomplish this, the chi2 function from the scikit-learn package was used to compute the chi-squared stats between the classes and then ranked them in the order of highest value.

The results was evaluated by comparing the expert's answers to the label probability given from GPT-3. The label probability was the probability that the given input corresponded to the predicted label. In order to compare the expert's answers to GPT-3's label probability, the expert's answers were normalized to values between zero and one, and was compared to the ISO label probability from GPT-3.

To answer **RQ2**, the expert's answers were seen as the ground truth, and GPT-3's answers were compared to these. If GPT-3's answers aligned with the expert's, then it was able to determine the completeness to a high degree, however if the answers differed, then GPT-3 was able to determine completeness to a low degree.

4.3.4 Domain result comparison

The result comparison, in terms of domains, was performed as the absolute last step of validation as it depended on the results from the previous step defined in 4.3.2. Five different datasets were used for comparison with GPT-3 in its few-shot setting. Four of the datasets were from the biomedical domain and the last consisted of abstracts from research papers. The datasets used were described as the following:

BioText, is a multi-label dataset that contains 3500 snippets from abstracts in the biomedical domain. Each abstract is classified using one of eight labels. This dataset was included in a study by Moradi et al. [6]

MedSts, is a multi-label dataset that consists of 1000 pairs of sentences, where a label between 0-5 is assigned based on their similarity. This dataset was included in a study by Moradi et al. [6]

PubMed-RCT, is a multi-label dataset that consists of 200,000 abstracts with sentence classification. Each sentence is either classified as: Background, Objective, Method, Result, or Conclusion. This dataset was included in a study by Moradi et al. [6]

ADE, is a binary-label dataset that contains extracted sentences from medical case reports related to adverse drug effects. Sentences are labeled as either being related to adverse drug effects or not. This dataset was included in a study by Alex et al. [7]

NIS - NeurIPS, is a binary-label dataset that contains broader impact statements from papers submitted to the NeurIPS 2020 conference. The authors annotated the

dataset depending on whether the statement mentioned that the research done in the paper could be used for possibly harmful applications. This dataset was included in a study by Alex et al. [7]

In the studies by Moradi et. al [6] and Alex et. al [7], GPT-3 in its few-shot setting was evaluated on the datasets. Due to three of these datasets containing multiple labels, Macro and Micro F1-score were used as an evaluation metric. Macro and micro F1-score are designed to be used for models evaluated on datasets containing multiple labels. However, the calculations are roughly the same as the F1-score used in this study, and therefore these were directly compared.

RQ2.3 was answered by comparing the F1-scores from the previously mentioned studies, to the F1-scores from GPT-3 used on the datasets in this study. Then a discussion was held on whether or not GPT-3 performed better in the information security domain or the compared domains.

4.4 Weekly meetings with case company

At least once every week, a meeting was held together with various data science and policy experts from the case company. Additional meetings with specific experts were also held if it was deemed necessary. At these meetings, the progress from the design cycle steps were discussed and an opportunity to validate the findings and/or ask for guidance was offered. Among the ones who attended regularly were the lead data scientist, the Chief Information Security Officer (CISO), and Chief Operating Officer (COO).

The data scientists mainly provided ideas in terms of data processing and design ideas for the framework and machine learning models. But they also remained critical to the ideas of the CISO as to keep the study both feasible and implementable. Meanwhile, the CISO and COO (but mainly CISO) provided knowledge in terms of information security, policies, and their processes. An important benefit of their participation were their instructions in terms of what to look for when extracting policy texts in relation to the ISO controls. For example, elements that characterized the A.5.1.2 control (defined in 4.1), were definitions of update frequency and reviews or updates in relation to significant changes. The extracts for A.5.1.2 should therefore have aimed to have these included. Furthermore, the CISO also provided constant validation of the findings from the various design cycle steps. If a specific extract was found to be confusing or if there were conflicts in terms of what extract better represents the control, the CISO helped sort these odd cases out.

5

Results

This section presents the results of the different natural language processing models and word embedding models. The main metrics that were used for evaluation were F1-score and accuracy. Firstly, the results of GPT-3 is presented, then BERT, and lastly, SVM, which was used in combination with different word embedding models.

5.1 Model comparisons

In this section, GPT-3 is compared to various benchmark models that were defined in 4.3.2.

5.1.1 GPT-3

This section presents the results from applying GPT-3 to the three different datasets A512, A722, and A923. Firstly, the results from GPT-3 in its few-shot setting is presented, and then the results from fine-tuning GPT-3.

5.1.1.1 Few-shot

For the hyperparameter *Model*, Curie and Davinci were used. Similarly, for the hyperparameter *Search model*, Ada and Curie were used. However, Davinci as the classifier model and Curie as the search model was only used for on run for each dataset. These models are more expensive in terms of pricing and training time, and using these models costed 2$ for each run. Therefore these were not used for all runs, since the budget for this project did not allow it. The results from evaluating GPT-3 in its few-shot setting can be seen in table 5.1. In all cases using K_fold = 1 yielded the best results, and using *Davinci* as the classifier model and *Curie* as the search model yielded the best F1-score on two out of three datasets.

Dataset	Max examples	Model	Search model	K_fold	F1-score	Accuracy
A512	5	Curie	Ada	5	0.5764	0.6
A512	25	Curie	Ada	5	0.5958	0.62
A512	50	Curie	Ada	5	0.6576	0.66
A512	50	Curie	Ada	1	**0.7273**	**0.7**
A512	50	Davinci	Curie	1	0.6154	0.5
A722	5	Curie	Ada	5	0.5325	0.58
A722	25	Curie	Ada	5	0.2914	0.54
A722	50	Curie	Ada	5	0.517	0.52
A722	50	Curie	Ada	1	0.5456	0.5
A722	50	Davinci	Curie	1	**0.8235**	**0.7**
A923	5	Curie	Ada	5	0.3603	0.56
A923	25	Curie	Ada	5	0.3467	0.48
A923	50	Curie	Ada	5	0.2633	0.46
A923	50	Curie	Ada	1	0.6667	**0.7**
A923	50	Davinci	Curie	1	**0.7142**	0.6

Table 5.1: Results of applying GPT-3 few-shot to the three different datasets. The bold text is used to show the highest value for each category. For values of $K > 1$ the average score was calculated across the runs.

5.1.1.2 Fine-tuning mode

For the fine-tuned model, only the parameter *Model name* was changed, which is the model that GPT-3 used when classifying the input. The F1-scores and accuracy score for the fine-tuned model can be seen in table 5.2. On the dataset A923, GPT-3 achieved the highest accuracy score when using *Davinci*, however the highest F1-score was achieved when using *Ada*, in this case it was hard to determine which the best performing model was. However since *Ada* was the simplest and cheapest model, it can be deemed the most efficient for these tasks, since it was able to produce the same or better scores compared to *Davinci*, for roughly one tenth of the cost.

Dataset	Model	F1-Score	Accuracy
A512	Ada	0.6154	0.4857
A722	Ada	0.2857	**0.4285**
A923	Ada	**0.5714**	0.4709
A512	Davinci	**0.6667**	**0.5**
A722	Davinci	0.2857	0.4118
A923	Davinci	0.5	**0.6970**

Table 5.2: F1-score and accuracy score for the fine-tuned GPT-3 model. The scores in bold are the best performing for that model and dataset.

5.1.2 Comparison with benchmark models

In this section the results from evaluating the benchmark models on the datasets is presented, and a comparison is made with the results from GPT-3. For each dataset a figure has been created, showing the different models' F1-score and accuracy. However, when evaluating Word2Vec in combination with SVC on the dataset A923 it failed to calculate a F1-score, perhaps due to precision or recall being zero in Equation 4.3. Therefore, the F1-score is absent in Figure 5.3.
For the dataset A512, the results can be seen in Figure 5.1, the purple dotted line is the accuracy from the ZeroR baseline. Ideally the models should achieve a much better accuracy than the ZeroR, however as can be seen in the figure, BERT failed to perform better, and TF-IDF with the SVC classifer only performed slightly better. This indicates that these models may not be a good fit for this task. On the other hand, the sequential embedding models perform satisfactory results above the ZeroR baseline. This could imply that prompts from the A512 control has a sequential characteristic. The best performing model is GPT-3 and achieves an accuracy of 0.7 and a F1-score of 0.727.

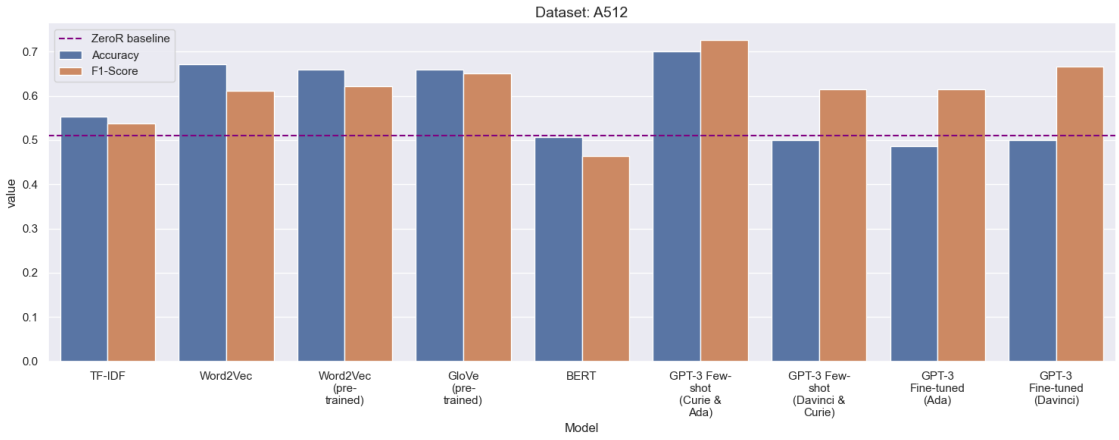


Figure 5.1: The results of using the models on the A512 dataset. GPT-3 in it's few shot setting with Ada as search model and Curie as the classification model is the best performing by achieving an accuracy of 0.7 and a F1-score of 0.727.

The results from evaluating the models on the dataset A722 can be seen in Figure 5.2. For this dataset, four models failed to achieve a better accuracy than the ZeroR baseline, and the Word2Vec (not pre-trained) model with the SVC classifier only managed to perform slightly better. Although the pre-trained GloVe model had a higher accuracy (0.725) than GPT-3, the F1-score fell a bit too short. Therefore, for this dataset GPT-3 in its few-shot setting with *Davinci* as the classifier model, and *Curie* as the search model performed the best, with an accuracy of 0.7 and a F1-score of 0.824.
The last dataset that was evaluated was A923 and the results can be seen in Figure 5.3. Again, many models either fell short of the ZeroR baseline or performed just slightly better. GPT-3 was the best performing model, where the combination of *Davinci* and *Curie* achieved the best F1-score of 0.714, and using *Curie* as the

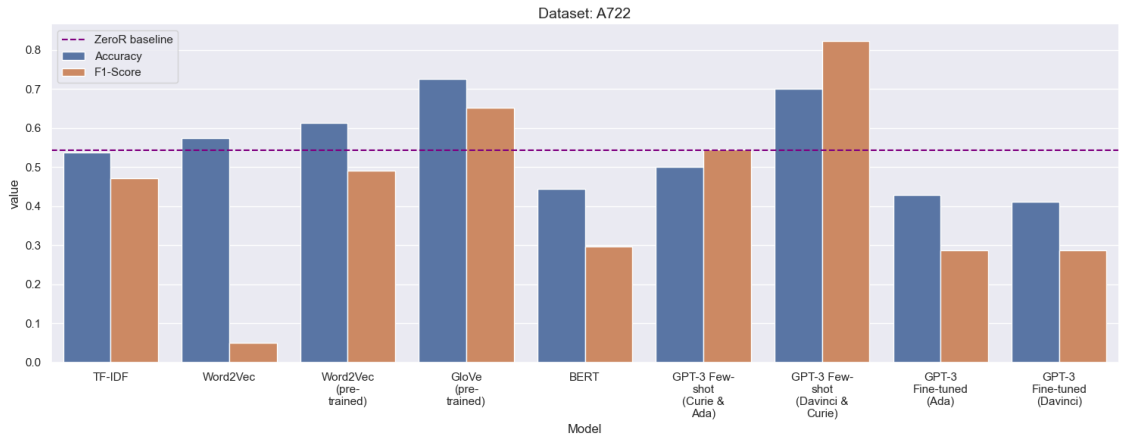


Figure 5.2: Results from all models evaluated on the dataset A722. The purple dotted line shows the ZeroR baseline.

classifier model, with *Ada* as the search model achieved the best accuracy of 0.7.

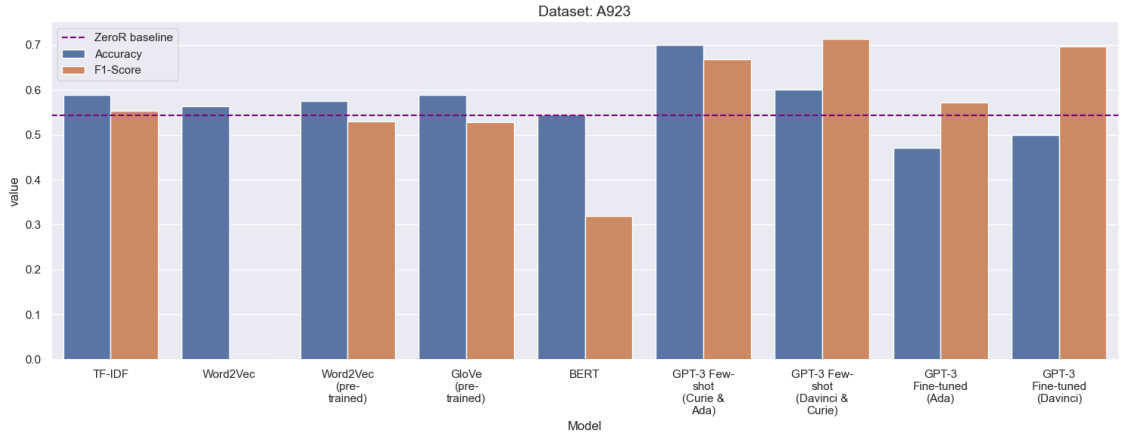


Figure 5.3: Results from all models evaluated on the dataset A923. For the model Word2Vec a F1-score was unable to be calculated. The purple dotted line shows the ZeroR baseline.

5.2 Expert validation with the case company

To fully answer **RQ2.2**, a validation session was conducted with an expert on information security policies. In this section the results from the expert validation is presented and compared to GPT-3's answers. Ideally this validation would be done with at least five or more experts, however due to time limitations, the validation was only conducted with one expert from the case company.

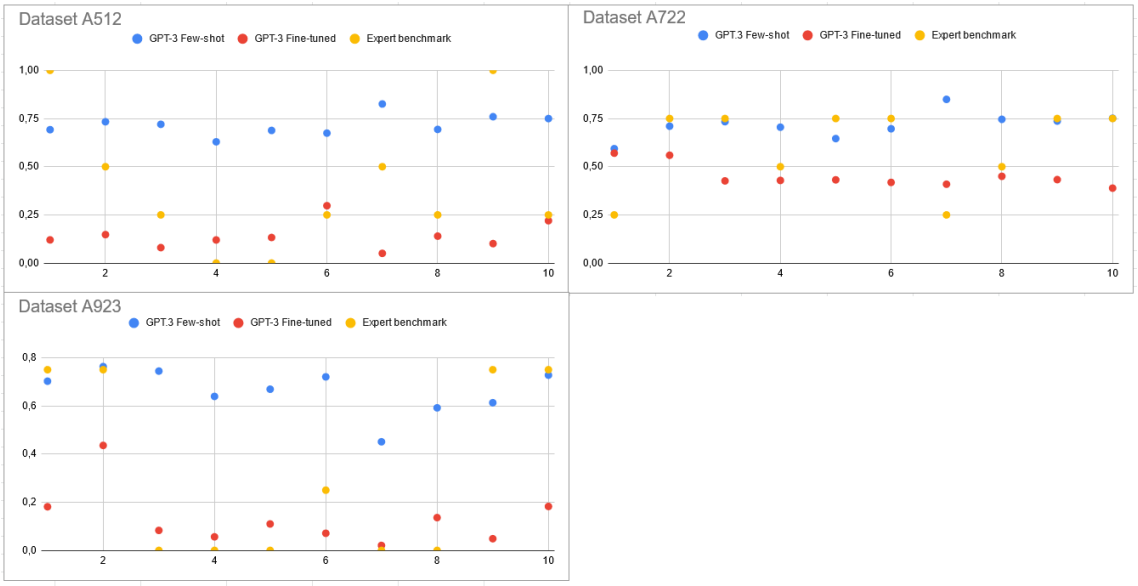


Figure 5.4: Scatter plots for each dataset, where the yellow dots are the expert's answers to how complete a policy text extract was towards a given ISO control. The red and blue dots represents GPT-3's probability towards the text extract being related to an ISO certified policy. The X-axis represents which text was used. The Y-axis represents the answers towards how complete the text was towards ISO 27001:2013.

Since the expert answered on a Likert scale, the values were normalized to values between zero and one. This was done to better compare the answers from the expert with the confidence given from GPT-3. In Figure 5.4, the answers from the expert can be seen in yellow, and the answers from GPT-3 can be seen in red and blue. Looking at the first value for the dataset A512, the expert labeled it with a one, meaning that the expert strongly agreed that the text was complete towards the ISO control, while the fine-tuned version disagreed with this, and GPT-3 in its few-shot setting agreed with the expert. However, it can also be seen that GPT-3 few-shot labeled all prompts as complete towards the ISO control, and GPT-3 fine-tuned labeled almost all of the texts as not complete. This is problematic since GPT-3 will be correct some of the time, but it performs somewhat like a ZeroR classifer.

A similar conclusion can be drawn when looking at the residual plot seen in Figure 5.5, where both GPT-3 models often produced large errors, with a few correct answers. The residual plots also show a trend of not being entirely random in comparison to the expert benchmark, i.e. there seem to be patterns in the residual plots. This often implies that the GPT-3 models are a bad fit to the expert's ground-truth. Therefore based on these results, and to answer **RQ2**, GPT-3 both in it's fine-tuned and few-shot setting, is not suitable for classifying information security policies. As it performed much worse than the experts and at least for the validation dataset, seemed to only predict one label.

The expert also ranked various features extracted from the tf-idf vectorizer. After the chi2 function had been applied the statistical significance level was varied to get

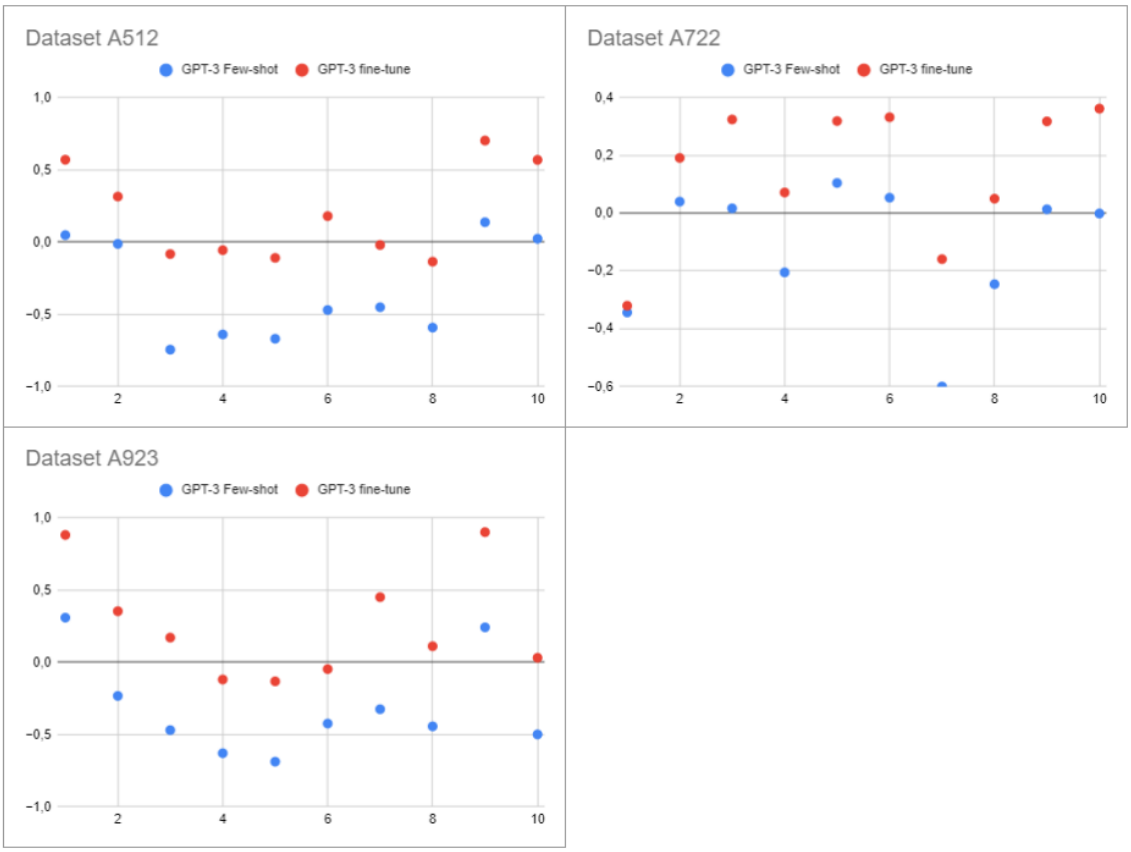


Figure 5.5: Residual plots for each dataset, where the error is calculated by using the expert's answers as the true value. If the dots are closer to the 0.0 line, then they are more aligned with the expert's opinion. The Y-axis shows the error, and the X-axis shows for which text the error was calculated.

a sense of which features were considered important at what statistical significance. Also due to a few controls not having established any features at all at the higher values for significance. Observe Table 5.3 for an output of these results and also observe Table 5.4 for the expert rankings.

In Table 5.3, the lower the significance level sinks the more features that seem unrelated to the control are discovered. For example, at 75% for both the A512 and A923 controls, the feature *university* is present. This has less to do with the actual control and is highly likely a direct result of the dataset containing more information security policies that come from universities, implying a low diversity in the industry types. Furthermore, there are very few features that are discovered at 85% significance and above. This may suggest that the features for the controls are hard to characterize at high confidence since the data is too sparse or has high variability.

In Table 5.3, the A512 and A722 datasets showed early signs of establishing high valued features already at 90% statistical significance. Two out of three of these were also highly ranked in the expert rankings in Table 5.4. However, A923 seemed to struggle with finding any defining features and even the ones that were found at a 75% statistical significance scored poorly in the expert rankings. At the lowest

Statistical significance	Features for A512	Features for A722	Features for A923
95%			
90%	year, access	awareness training	
85%			
80%	review year, data, procedure, information security policy, management, annually	access, security training, data, information security awareness	
75%	university, board, include, approve	information security training, awareness program, program	user access, base, university

Table 5.3: Features found with an increasing significance level for each control.

statistical significance levels that were tested, many features that were later ranked low were found. This was expected out of most controls, but unexpectedly, A722 seemed to find several of its highest expert ranked features at this level.

The motivation for the features that were ranked as 1 & 2 were, according to the expert, that they were too general and poorly representative of the control. The ones ranked at 3 made sense to include in the prompts. Finally, 4 & 5 seemed to be essential to exist, or at least to some extent.

Score (1-5)	Features for A512	Features for A722	Features for A923
5	-	information security awareness, information security training, security training	-
4	year, annually, review year	awareness training, awareness program	-
3	require, approve, management, board	-	user access
2	-	program, staff	
1	access, procedure, include, data, university	data, access	base, university

Table 5.4: Expert ranked features based on most characterizing for each control using a Likert scale.

5.3 Domain result comparison

To answer **RQ2.3** comparisons were made between GPT-3 trained in the information security domain, to GPT-3 used in other domains. As can be seen in table 5.5, GPT-3 in the information security domain is performing better than GPT-3 applied to the datasets in the other chosen domains.

Dataset	**Metric**	**Score**
BioText	Micro F1	0.53
MedSts	Micro F1	0.26
PubMed-CRT	Micro F1	0.43
ADE	Macro F1	0.686
NIS - NeurIPS	Macro F1	0.679
A512	F1-score	0.7273
A722	F1-score	0.8235
A923	F1-score	0.7142

Table 5.5: The scores from GPT-3 applied on different datasets. BioText, MedSTS and PubMedCRT are from the study by Moradi et. al [6]. ADE and NIS are from the study by Alex et. al [7]. The datasets A512, A722, and A923 are taken from the results of this study, whre the highest F1-score were chosen for each dataset.

6
Discussion

In this section the results, potential threats to validity, and future work are discussed.

6.1 Framework evaluation

The constructed framework that was established in 4.2.3 does indeed seem useful since it was able to outperform a ZeroR baseline on several occasions with a range of models. Hence, the framework can be deemed as a working approach in terms of analyzing information security policy completeness in relation to ISO. Additionally, the results found from looking at the higher statistical significance features of the controls found in 5.4 seemed to imply that the data points were simply too few or did not have enough characterizing features to provide high value features. Therefore, given the amount of data that was found, the selected level of data granularity, i.e. controls, was a good choice for this study. If, on the other hand, control objective or domain level had been used, which are on average much longer, the features would be much harder to detect no matter the statistical significance and would have been an indication of poor ability to classify.

6.2 GPT-3 and information security policy completeness

To validate the GPT-3 performance within the domain of information security policies and deem whether it could have a real-world application, this study applied two test. Whereas the first was to compare GPT-3 to other existing models and measure how well it performs in comparison and also what type of model was best fitted for information security policy completeness. The second step was to validate GPT-3's performance using a policy expert from the case company and determine whether the model had the potential of evaluating on par with an expert.

6.2.1 GPT-3 compared to benchmark models

A few conclusions can be drawn based on the results gathered in 5.1.2. In the comparisons across every dataset, some version of a GPT-3 model was shown to have the best performance. However, the pre-trained GloVe and pre-trained Word2Vec models had seemingly good overall performance across all controls by continuously performing better than the ZeroR baseline and may indicate that utilizing pre-trained

weights for the embedding models is preferable. Additionally, all the sequential word embedding models showed positive results across all datasets and may indicate that information security policies have more of a sequential structure.

The TF-IDF model often performed just above or below the ZeroR baseline and does not seem like a good fit for this task. Similarly, the BERT contextual model performed worse than the ZeroR baseline, and its counterpart GPT-3, in all datasets and is also an indication of not being a good fit for this specific task. However, with more data and model fine tuning it may be possible to increase its performance.

In conclusion, for the GPT-3 comparisons with the benchmark models, some format of GPT-3 had a superior performance than the other embedding models. There are also indications that controls have different characteristics in terms of semantic properties and the choices for models to use should be based on an one model per control approach rather than one model for all controls to maximize the prediction performance for each control.

6.2.2 GPT-3 compared to expert results

As the expert validation results showed, GPT-3 does not perform well when compared to how an expert would evaluate the given texts. This could be due to numerous reasons. Firstly, the GPT-3 model and the policy expert may have conflicting perspectives on how to define completeness. In other words, the features that the GPT-3 model considers important in comparison to the systematic approach of the expert to analyze ISO factors may be too different from one another. For example, when the TF-IDF features were presented to the expert at 75% statistical significance, only about a third were considered to be highly agreeable or agreeable as features that could characterize that specific control. Additionally, since features such as *university* seemed to appear at 75% significance, there are implications that the dataset is not diversified enough in terms of industry. Lastly, when classifying the inputs, the expert used a check-box approach, for example for the control A512, the expert verified that it was stated that it was the policy itself that needed to be reviewed, that it was clearly stated how often the policy was reviewed, and that the policy was updated in case of significant changes. If all these were stated in the policy, then the expert considered the policy to be complete towards the control A.5.1.2. Based on these, modifications might be needed to the framework in order to more closely emulate how an expert would classify the policies.

In conclusion, the results from the expert validation session indicated that the machine learning application within the domain of information security policy is still far from having a real-world implementation although showing promising results in comparison to the benchmark models.

6.3 GPT-3 compared to other domains

GPT-3 in the information security policy domain is better performing than the results of GPT-3 in the biomedical and research domain. In particular, GPT-3 performed better than chosen datasets from the studies by Moradi et al. [6] and Alex et al. [7] which can be seen in section 5.3. On average, the best performing GPT-3 model in this study had a higher F1-score of 0.4168 compared to GPT-3 in the study by Alex et al. [7], and a higher F1-score of 0.141 compared to GPT-3 in the study by Moradi et al. [6]. Therefore, in comparison to the other domains, GPT-3 is more suitable for the policy domain, and its contextual weights may be more aligned with the content in information security policies. Another reason for GPT-3 performing better in this study can be related to the fact that all of the chosen datasets from Alex et al. [7] included multi-label classification. This can point to the fact that GPT-3 performs better in binary classification. This is also supported by GPT-3 scoring higher on the datasets chosen from Moradi et al. [6], which were both datasets with binary labels.

6.4 Threats to validity

In this sections the threats to validity are discussed.

6.4.1 External Validity

In total the dataset consisted of 50 policies, which were further divided into datasets for each control. For the fine-tuning mode of GPT-3, OpenAI recommended 100-200 examples for each label. The data collection stage did not reach this number mainly due to the reason that companies generally do not want to share their information security policy, even though it should not contain anything sensitive to the company.

One approach to mitigate the lack of data was if the authors were unable to find text related to the control, then arbitrary text was chosen instead and labeled as Non-ISO. This meant that even though the policy itself lacked text for the specific control, models were still given extra data to be trained on. However, it is possible that inserting more arbitrary texts and labeling them as Non-ISO would have yielded different results, but the authors decided to keep a more balanced dataset instead.

Another possible threat is the fact that certain information security policies may still have an ISO-level defined control even if they are not certified. Possibly because of attaining certification towards other standards or simply because gaining ISO-certification has not been a priority for that specific organization. To mitigate this the authors also used their newfound expertise and by comparing to other prompts in order to validate the ISO label of each prompt.

The information security policies gathered for this study was mostly gathered through gray literature, and thus may introduce a bias as certain organizations may have a

higher online exposure than others. To mitigate this threat the authors also listed a few randomly selected universities and organizations from top-100 lists to visit their company pages and search for a publicly accessible information security policy.
Furthermore, the choice of manually extracting texts from the policies can also affect the outcome of this study. The goal of this study was to automate the process of determining the completeness of an information security policy toward the ISO 27001:2013 standard. However, if this process includes manually extracting text and inputting it into the model, it can hardly be seen as automated, as it still requires manual work. In the ideal scenario, a text extraction tool should have been developed in tandem with the GPT-3 model, but due to time limitations, this was not done.

6.4.2 Internal Validity

Even though this thesis was written on the topic of ISO 27001:2013 and information security policies, the authors did not have extensive knowledge on this topic. Efforts were made to validate each step of the process with the company's policy expert during the weekly meetings, but there is a possibility that some texts were labeled to either the wrong ISO 27001:2013 control objective, or wrongly labeled as ISO, and vice versa. Ideally, the annotations should have been performed by experts in the field but coming across experts that have the time to annotate several documents in relation to ISO controls is not realistic and thus had to be performed by the authors.

Finally, several word embedding models in this study were not parameter tuned nor combined with different classifiers to explore and maximize their performance and could negatively impact their accuracy results in this study. However, as the purpose of this study was to attempt and measure the success of applying machine learning methods within a software engineering domain that is also business critical and deem whether it shows promise rather than an optimal performance based study, the authors did not spend extensive time on finding the optimal parameters nor combination of models.

6.5 Future work

This study was heavily affected by the lack of publicly available information security policies and even though a considerable amount of time out of the limited time budget was spent on collecting the data, the resulting dataset was far from satisfactory. Therefore, the first suggestion for future work is the collection of more data to attain results with higher confidence and that can also be relied on to train sequential models from scratch that is only trained on information security policies. Furthermore, with more data, differences between the policies in terms of industry type and geographic location could be analyzed. Preferably, the dataset should also be annotated by experts. The authors suggestion is to possibly establish a collaboration with a certification body which also deals with the ISO standard. Reasoning behind this is that the chances of them having the data and experts to offer is high.

The authors spent little time on incorporating more classifiers and fine tuning the models as it did not align with the purpose of the study. Hence, attempting grid-searches to find the optimal model combinations may also be attempted.

While GPT-3 may not be ready to assist policy experts with their completeness assessment, it did perform well in the policy domain in comparison to the other domains it had been applied in. Hence, it may be interesting to see if GPT-3 could handle other policy areas such as privacy policies in relation to GDPR.

As discussed in 6.4.1, the process of manually extracting texts from policies was labor intensive, time exhausting, and error-prone. Hence, a suggestion for further work is to build a text extraction tool that retrieves the section of text in a policy that is related to the control of interest. This could be performed by using NLP techniques such as LDA (topic modeling) or keyword based searches. Furthermore, this study provides a framework and proof-of-concept for the absolutely lowest level of the automation process, i.e. the ISO control completeness check. This can be further developed into including all 114 controls and compose a completeness pipeline that can assess entire information security policies. The authors would then recommend to begin with the GloVe pre-trained model for all controls as it seemed to perform well across all results.

7
Conclusion

In this study the focus has been on the application of a language model within the software engineering and the business critical topic of information security. More specifically, the task has been to determine information security policy completeness in relation to the ISO 27001:2013 standard by utilizing the latest within language models, namely OpenAI's GPT-3. By using the research method of design science methodology, this study investigated a design problem, designed a treatment, and evaluated that treatment in iterations together with a case company that specializes in software solutions and policy compliance.

The treatment design was structured as a framework for assessing information security completeness in relation to ISO, and incorporated NLP and machine learning models to complement it. This also enabled an answer to **RQ1:** *What characterizes a good quality machine learning framework based on factors such as the amount of data and manual labor needed for information security policy?*. The constructed framework was based on the finest data granularity in the ISO 27001:2013 standard, which is also referred to as a control. This choice was made together with experts from the case company and required less data but more manual labor from the authors. To evaluate the framework, a few controls were selected together with a policy expert in order to establish a proof-of-concept. The controls were then annotated from a dataset of 50 information security policies and labeled as either ISO or Non-ISO. The choice of classifying information security policies on control-level can be deemed as a good quality since GPT-3 was able to perform well on the testing sets.

In order to validate the proof-of-concept, i.e. the framework and GPT-3 on controls, this study employed three methods. The first method was to compare its performance to various existing models, which was referred to as benchmark models. The second was to validate by comparing its assessment of policy extracts in comparison to a policy expert. Finally, the third method was to compare GPT-3's achieved performance to different domains to deem how GPT-3 within information security fared in terms of specific domain tasks.

The results that yielded from the model validation provided answers for **RQ2.1:** *Which machine learning model features are beneficial for determining document coverage and alignment in relation to ISO?* and **RQ2.2:** *To what degree can the features of GPT-3 enhance the document classification process, and how does it perform versus other algorithms?*. The answer to the latter question is that GPT-3 has the potential to automate the document classification process. It was able to achieve

F1-scores between 0.7 and 0.8 on the three different testing sets that it was applied on. The variations of sequential embeddings also showed a great overall performance and also indications of being a good general choice for any control. Therefore, to answer the former research question, contextual and sequential characteristics of embeddings have seemingly beneficial features to include in a model. However, when GPT-3 was compared to the performance of an expert, GPT-3 seemed to be a poor fit. In order to increase the performance, more granularity of the framework, or more training data is most likely needed.

In the final validation step, to answer **RQ2.3:** *How does the GPT-3 model trained and evaluated on information security policy documents compare to performances of GPT-3 models applied in other domains?*, GPT-3 performance in this study was compared to the biomedical domain and research paper domain. The result was that GPT-3 applied within information security policies outperformed the application of GPT-3 in the other two domains. Finally, the answers to **RQ2.1**, **RQ2.2**, and **RQ2.3** provide an answer to **RQ2:** *To which degree does a GPT-3 language model determine the ISO completeness of various organizations' information security documents?* The answer, as well as the conclusions drawn from this study, is that the application of language models, specifically GPT-3, within the software engineering and business critical domain of information security shows promise, but it is not quite there in terms of real world implementation.

Bibliography

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[2] Corby Rosset. Turing-nlg: A 17-billion-parameter language model by microsoft, 2020. URL `https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/`.

[3] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[4] 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. *CoRR*, abs/1907.11692, 2019. URL `http://arxiv.org/abs/1907.11692`.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

[6] Milad Moradi, Kathrin Blagec, Florian Haberl, and Matthias Samwald. GPT-3 models are poor few-shot learners in the biomedical domain. *CoRR*, abs/2109.02555, 2021. URL `https://arxiv.org/abs/2109.02555`.

[7] Neel Alex, Eli Lifland, Lewis Tunstall, Abhishek Thakur, Pegah Maham, C. Jess Riedel, Emmie Hine, Carolyn Ashurst, Paul Sedille, Alexis Carlier, Michael Noetel, and Andreas Stuhlmüller. RAFT: A real-world few-shot text classification benchmark. *CoRR*, abs/2109.14076, 2021. URL `https://arxiv.org/abs/2109.14076`.

[8] Surayahani Hasnul Bhaharin, Umi Asma' Mokhtar, Rossilawati Sulaiman, and Maryati Mohd Yusof. Issues and trends in information security policy compliance. In *2019 6th International Conference on Research and Innovation in Information Systems (ICRIIS)*, pages 1–6, 2019. doi: 10.1109/ICRIIS48246.2019.9073645.

[9] Joe Tidy and David Molloy. Twitch confirms massive data breach, 2021. URL https://www.bbc.com/news/technology-58817658.

[10] Joe Tidy. Swedish coop supermarkets shut due to us ransomware cyber-attack, 2021. URL https://www.bbc.com/news/technology-57707530.

[11] Georg Disterer. Iso/iec 27000, 27001 and 27002 for information security management. *Journal of Information Security*, 4:92–100, 2013. doi: 10.4236/jis.2013.42011.

[12] Mada Alassaf and Ali Alkhalifah. Exploring the influence of direct and indirect factors on information security policy compliance: A systematic literature review. *IEEE Access*, pages 1–1, 2021. doi: 10.1109/ACCESS.2021.3132574.

[13] Heru Susanto, Mohammad Nabil Almunawar, and Yong Chee Tuan. A novel method on iso 27001 reviews: Isms compliance readiness level measurement. *Computer Science Journal*, Volume 2, Issue 1, 2012.

[14] Mikko Siponen and Robert Willison. Information security management standards: Problems and solutions. *Information & Management*, 46:267–270, 06 2009. doi: 10.1016/j.im.2008.12.007.

[15] Heru Susanto and Mohammad Nabil Almunawar. *Information security management systems*. Apple Academic Press, 1 edition, 2018.

[16] Elisa Costante, Yuanhao Sun, Milan Petković, and Jerry den Hartog. A machine learning solution to assess privacy policy completeness: (short paper). In *Proceedings of the 2012 ACM Workshop on Privacy in the Electronic Society*, WPES '12, page 91–96, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450316637. doi: 10.1145/2381966.2381979. URL https://doi.org/10.1145/2381966.2381979.

[17] O. Amaral CEJAS, S. Abualhaija, D. Torre, M. Sabetzadeh, and L. Briand. Ai-enabled automation for completeness checking of privacy policies. *IEEE Transactions on Software Engineering*, pages 1–1, nov 2021. ISSN 1939-3520. doi: 10.1109/TSE.2021.3124332.

[18] SIS. *SS-EN ISO/IEC 27001:2017*. Svenska Institutet för Standarder (SIS), 1st edition, 2017. URL https://www.sis.se/en/produkter/sociology-services-company-organization/company-organization-and-management/management-systems/isoiec270012013/.

[19] SIS. *SS-EN ISO/IEC 27002:2017*. Svenska Institutet för Standarder (SIS), 1st edition, 2017. URL https://www.sis.se/en/produkter/standardization/information-sciences-publishing/documents-in-administration-commerce-and-industry/ssenisoiec270022017/.

[20] Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. Deep learning–based text classification: A com-

prehensive review. *ACM Comput. Surv.*, 54(3), apr 2021. ISSN 0360-0300. doi: 10.1145/3439726. URL https://doi.org/10.1145/3439726.

[21] Sahar A. El Rahman, Feddah Alhumaidi AlOtaibi, and Wejdan Abdullah Al-Shehri. Sentiment analysis of twitter data. In *2019 International Conference on Computer and Information Sciences (ICCIS)*, pages 1–4, 2019. doi: 10.1109/ICCISci.2019.8716464.

[22] Sikha Bagui, Debarghya Nandi, Subhash Bagui, and Robert Jamie White. Classifying phishing email using machine learning and deep learning. In *2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, pages 1–2, 2019. doi: 10.1109/CyberSecPODS.2019.8885143.

[23] Fang Miao, Pu Zhang, Libiao Jin, and Hongda Wu. Chinese news text classification based on machine learning algorithm. In *2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, volume 02, pages 48–51, 2018. doi: 10.1109/IHMSC.2018.10117.

[24] Seyyed Mohammad Hossein Dadgar, Mohammad Shirzad Araghi, and Morteza Mastery Farahani. A novel text mining approach based on tf-idf and support vector machine for news classification. In *2016 IEEE International Conference on Engineering and Technology (ICETECH)*, pages 112–116, 2016. doi: 10.1109/ICETECH.2016.7569223.

[25] Athanasios Tzimourtas, Spyros Bakalakos, Panagiota Tselenti, and Athanasios Voulodimos. An exploration on text classification using machine learning techniques. In *25th Pan-Hellenic Conference on Informatics*, PCI 2021, page 247–249, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450395557. doi: 10.1145/3503823.3503869. URL https://doi.org/10.1145/3503823.3503869.

[26] Yoon Kim. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014. URL http://arxiv.org/abs/1408.5882.

[27] Parul Sharma and Teng-Sheng Moh. Prediction of indian election using sentiment analysis on hindi twitter. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1966–1971, 2016. doi: 10.1109/BigData.2016.7840818.

[28] Peter Story, Sebastian Zimmeck, Abhilasha Ravichander, Daniel Smullen, Ziqi Wang, Joel Reidenberg, N Russell, and Norman Sadeh. Natural language processing for mobile app privacy compliance. 03 2019.

[29] Methus Narksenee and Kunwadee Sripanidkulchai. Can we trust privacy policy: Privacy policy classification using machine learning. In *2019 2nd International Conference of Intelligent Robotic and Control Engineering (IRCE)*, pages 133–137, 2019. doi: 10.1109/IRCE.2019.00034.

[30] T.A.I. Thotawaththa, Y.T. Gamage, Damika Gamlath, Wei Chee, and D. Meedeniya. Automated categorization of privacy policies based on user perspective. In *2021 10th International Conference on Information and Automation for Sus-*

tainability (ICIAfS), pages 54–59, Aug 2021. doi: 10.1109/ICIAfS52090.2021.9606158.

[31] Abdulrahman Alabduljabbar, Ahmed Abusnaina, Ülkü Meteriz-Yildiran, and David Mohaisen. Automated privacy policy annotation with information highlighting made practical using deep representations. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 2378–2380, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450384544. doi: 10.1145/3460120.3485335. URL https://doi.org/10.1145/3460120.3485335.

[32] Decui Liang and Bochun Yi. Two-stage three-way enhanced technique for ensemble learning in inclusive policy text classification. *Information Sciences*, 547:271–288, 2021. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2020.08.051. URL https://www.sciencedirect.com/science/article/pii/S0020025520308112.

[33] Yiyu Yao. Three-way decisions with probabilistic rough sets. *Information Sciences*, 180(3):341–353, 2010. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2009.09.021. URL https://www.sciencedirect.com/science/article/pii/S0020025509004253.

[34] Jonas Thiergart, Stefan Huber, and Thomas Übellacker. Understanding emails and drafting responses - an approach using GPT-3. *CoRR*, abs/2102.03062, 2021. URL https://arxiv.org/abs/2102.03062.

[35] Ke-Li Chiu and Rohan Alexander. Detecting hate speech with GPT-3. *CoRR*, abs/2103.12407, 2021. URL https://arxiv.org/abs/2103.12407.

[36] Tingyu Zhang and Ruixia Zhang. Revealing the power of bert for text sentiment classification. In *2021 IEEE 4th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, pages 14–17, 2021. doi: 10.1109/AUTEEE52864.2021.9668704.

[37] Neil Doherty, Leonidas Anastasakis, and Heather Fulford. The information security policy unpacked: A critical study of the content of university policies. *International Journal of Information Management*, 29:449–457, 12 2009. doi: 10.1016/j.ijinfomgt.2009.05.003.

[38] Allen C Johnston, Merrill Warkentin, Maranda McBride, and Lemuria Carter. Dispositional and situational factors: influences on information security policy violations. *European Journal of Information Systems*, 25(3):231–251, 2016. doi: 10.1057/ejis.2015.15. URL https://doi.org/10.1057/ejis.2015.15.

[39] Michael Nieles, Kelley Dempsey, and Victoria Yan Pillitteri. An introduction to information security. *NIST Special Publication 800-12*, 1, 2017. doi: 10.6028/nist.sp.800-12r1.

[40] Johanes Widhi Candra, Obrina Candra Briliyant, and Sion Rebeca Tamba. Isms planning based on iso/iec 27001:2013 using analytical hierarchy process at gap analysis phase (case study : Xyz institute). In *2017 11th International*

Conference on Telecommunication Systems Services and Applications (TSSA), pages 1–6, 2017. doi: 10.1109/TSSA.2017.8272916.

[41] Julia Hirschberg and Christopher D. Manning. Advances in natural language processing. *Science*, 349(6245):261–266, 2015. doi: 10.1126/science.aaa8685. URL `https://www.science.org/doi/abs/10.1126/science.aaa8685`.

[42] D. Jurafsky and J.H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall series in artificial intelligence. Pearson Prentice Hall, 2nd edition, 2009. ISBN 9780131873216.

[43] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL `http://www.aclweb.org/anthology/D14-1162`.

[44] Edward Loper Bird, Steven and Ewan Klein. *Natural Language Processing with Python*. O'Reilly Media, Inc., 2009. ISBN 9780596516499.

[45] Syed Farhan Alam Zaidi, Faraz Malik Awan, Minsoo Lee, Honguk Woo, and Chan-Gun Lee. Applying convolutional neural networks with different word representation techniques to recommend bug fixers. *IEEE Access*, 8:213729–213747, 2020. doi: 10.1109/ACCESS.2020.3040065.

[46] C.C. Aggarwal. *Machine Learning for Text*. Springer International Publishing, 2018. ISBN 9783319735313. URL `10.1007/978-3-319-73531-3`.

[47] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. doi: 10.1017/CBO9780511809071.

[48] Zellig S. Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954. doi: 10.1007/978-94-009-8467-7_1.

[49] Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `https://aclanthology.org/P10-1040`.

[50] RaRe-Technologies. gensim-data, 2022. URL `https://github.com/RaRe-Technologies/gensim-data`.

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[52] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2015.

[53] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. *CoRR*, abs/1904.08082, 2019. URL http://arxiv.org/abs/1904.08082.

[54] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909, 2015. URL http://arxiv.org/abs/1508.07909.

[55] Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. *CoRR*, abs/1606.01933, 2016. URL http://arxiv.org/abs/1606.01933.

[56] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018.

[57] D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

[58] Yan Yang, Juan Wang, and Yongyi Yang. Improving svm classifier with prior knowledge in microcalcification detection1. In *2012 19th IEEE International Conference on Image Processing*, pages 2837–2840, 2012. doi: 10.1109/ICIP.2012.6467490.

[59] Roel J. Wieringa. *Design Science Methodology for Information Systems and Software Engineering*. Springer, Berlin, Heidelberg, 1 edition, 2014.

[60] Why there aren't more information security research studies. *Information & Management*, 41(5):597–607, 2004. ISSN 0378-7206. doi: https://doi.org/10.1016/j.im.2003.08.001. URL https://www.sciencedirect.com/science/article/pii/S0378720603000995.

[61] Rutu Mulkar-Mehta, Jerry Hobbs, and Eduard Hovy. Granularity in natural language discourse. In *Proceedings of the Ninth International Conference on Computational Semantics*, IWCS '11, page 360–364, USA, 2011. Association for Computational Linguistics.

[62] David C. Blair. Information retrieval, 2nd ed. c.j. van rijsbergen. london: Butterworths; 1979: 208 pp. price: $32.50. *Journal of the American Society for Information Science*, 30(6):374–375, 1979. doi: https://doi.org/10.1002/asi.4630300621. URL https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/asi.4630300621.

[63] Sanjay Yadav and Sanyam Shukla. Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, pages 78–83, 2016. doi: 10.1109/IACC.2016.25.

[64] Chitra Nasa and Suman Suman. Evaluation of different classification techniques for web data. *International Journal of Computer Applications*, 52:34–40, 08 2012. doi: 10.5120/8233-1389.

A

Appendix 1

A.1 Search words

- information security
- information security policy
- isp
- public information security policy
- security policy
- volvo information security policy
- ica information security policy
- *company name* information security policy
- information security policy pdf iso
- public information security policy pdf
- isp pdf

A.2 E-mail template

Hej,

Jag heter Hampus Lundblad och är en mastersstudent inom Software Engineering and Technology på Chalmers Tekniska Högskola i Göteborg. Jag skriver just nu mitt examensarbete om informationssäkerhetspolicyn och dess fullbordan av välkända standarder. Mer specifikt är tanken med arbetet att experimentera med verktyg för att eventuellt kunna automatiskt kontrollera en policys överensstämmelse med ISO:27001:2013. Då jag i nuvarande läge ägnar mig åt datainsamling undrar jag om Ni vill låta mig ta del av Er informationssäkerhetspolicy för att stödja mitt arbete, i utbyte ser jag till att dela med mig av rapporten när arbetet väl är klart. Er medverkan kommer att hållas anonym och mottagen data kommer inte att delas med någon tredje part.

Tack på förhand!
Med vänliga hälsningar,
Hampus

Hej,

Jag heter Pouya Faramarzi och är en masterstudent inom Software Engineering and

Technology på Chalmers Tekniska Högskola i Göteborg. Jag skriver just nu mitt examensarbete om informationssäkerhetspolicyn och dess fullbordan av välkända standarder. Mer specifikt är tanken med arbetet att experimentera med verktyg för att eventuellt kunna automatiskt kontrollera en policys överensstämmelse med ISO:27001:2013. Då jag i nuvarande läge ägnar mig åt datainsamling undrar jag om Ni vill låta mig ta del av Er informationssäkerhetspolicy för att stödja mitt arbete, i utbyte ser jag till att dela med mig av rapporten när arbetet väl är klart. Er medverkan kommer att hållas anonym och mottagen data kommer inte att delas med någon tredje part.

Tack på förhand!
Med vänliga hälsningar,
Pouya