Safety Desk: Extraction and analysis of textual information to build a reporting system
Information Extraction for Report Generation
Bruno Carlos Luís Ferreira
1 2 9 0 UNIVERSIDADE D COIMBRA

UNIVERSIDADE D
COIMBRA

Bruno Carlos Luís Ferreira

Safety Desk: Extraction and analysis of textual information to build a reporting system

Information Extraction for Report Generation

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems, advised by Professor Catarina Helena Branco Simões da Silva and Professor Hugo Ricardo Gonçalo Oliveira and presented to the Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2022

Faculty of Sciences and Technology

Department of Informatics Engineering

Safety Desk: Extraction and analysis of textual information to build a reporting system

Information Extraction for Report Generation

Bruno Carlos Luís Ferreira

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems advised by Prof. Catarina Helena Branco Simões da Silva and Prof. Hugo Ricardo Gonçalo Oliveira and presented to the
Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2022

UNIVERSIDADE D
COIMBRA

Abstract

As the amount of available data grows, working with large amounts of text data has become hectic and more time-consuming. Therefore, companies and organizations need to rely on techniques and algorithms to automate manual work with intelligent algorithms in order to reduce human effort, reduce expenses, and make the process less error-prone and more efficient.

The Safety Desk project outlined in this dissertation, in collaboration with Instituto Pedro Nunes and Talent Ingredient, aims to optimize the current reporting generation process of chemical substances done by the Talent Ingredient company, both in terms of saving human resources as in time saving. This process is very important for the company since the reports generated are the selling product in their business model, so the integration of an automatized system in the platform currently used (Cosmedesk) is a objective of the Talent Ingredient company.

That said, this thesis discusses the importance of Information Extraction (IE) and Machine Reading Comprehension (MRC) in the acquisition of information from unstructured data, in the case of this project PDFs documents, and exposes the work developed in the implementation of the pipeline proposed for the Safety Desk project.

The proposed pipeline is made up of five phases: (1) the Preprocessing Phase where the document is divided into sections in order to provide the right inputs to the Question Answering (QA) models used. (2) The IE Process that uses Extractive QA models that, given a context, *i.e.*, the sections obtained from the first phase of the pipeline, and question, it extracts the answer that predicts to be right. (3) The Data Verification Phase is where the information extracted from the second phase is clean and (4) Data-to-text (D2T) Phase generates a toxicological profile of the chemical substance. In last, the Safety Desk service can be integrated via a (5) RESTfull API implemented, where endpoints were created to establish the communication in the actual platform, Cosmedesk, and the Safety Desk work.

In the evaluations performed, the work developed presented solid results (0.74 F-Score, 0.78 Precision, 0.71 Recall and 0.77 Accuracy) for the documents used, although in terms of execution time the Safety Desk took an average of 191 tokens/second analysed, which in a average document with 30000 tokens takes 2'30 minutes.

Keywords

Natural Language Processing, Information Extraction, Machine Reading Comprehension, Question Answering, Transformers, Natural Language Generation, Data-to-text Generation.

Resumo

À medida que a quantidade de dados disponíveis cresce, trabalhar com grandes quantidades de dados de texto tornou-se agitado e mais demorado. Portanto, empresas e organizações precisam contar com técnicas e algoritmos para automatizar o trabalho manual com algoritmos inteligentes, a fim de reduzir o esforço humano, reduzir despesas e tornar o processo menos propenso a erros e mais eficiente.

O projeto Safety Desk detalhado nesta dissertação, em colaboração com o Instituto Pedro Nunes e Talent Ingredient, visa otimizar o atual processo de geração de relatórios de substâncias químicas feito pela empresa Talent Ingredient, tanto em termos de economia de recursos humanos como em economia de tempo. Esse processo é muito importante para a empresa, pois os relatórios gerados são o produto de venda no modelo de negócios, portanto a integração de um sistema automatizado na plataforma atualmente utilizada (Cosmedesk) é um objetivo da Empresa Talent Ingredient.

Dito isso, esta dissertação discute a importância da Extração de Informação (IE) e da Compreensão de Leitura de Máquina (MRC) na aquisição de informações a partir de dados não estruturados, no caso deste projeto documentos PDFs, e expõe o trabalho desenvolvido na implementação do pipeline proposto para o projeto Safety Desk.

O pipeline proposto é composto por cinco fases: (1) a Fase de Pré-processamento onde o documento é dividido em seções para fornecer as entradas corretas para os modelos Questão Resposta (QA) utilizados. (2) O processo EI que usa modelos Extrativos QA que, dado um contexto, *i.e.*, as seções obtidas da primeira fase do pipeline, e pergunta, extrai a resposta que prevê estar correta. (3) A Fase de Verificação de Dados é onde as informações extraídas da segunda fase são limpas e (4) a Fase Geração de Linguagem Natural gera um perfil toxicológico da substância química. Por fim, o serviço Safety Desk pode ser integrado através de uma (5) RESTfull API implementada, onde foram criados endpoints para estabelecer a comunicação na plataforma, Cosmedesk, e o Safety Desk.

Nas avaliações realizadas, o trabalho desenvolvido apresentou resultados sólidos (0.74 F-Score, 0.78 Precision, 0.71 Recall e 0.77 Accuracy) para os documentos utilizados, embora, em termos de execução, o Safety Desk processou em média 191 tokens/segundo, que numa média de 30.000 tokens por documento demora 2'30 minutos a processar.

Palavras-Chave

Processamento de Linguagem Natural, Extração de Informação, Compreensão de Leitura de Máquina, Resposta a Perguntas, Transformers, Geração de Linguagem Natural, Geração de Dados para Texto.

Dedicated to my father, Carlos, to my mother Graça, to my sister, Filipa, to my grandmothers Maria and Carmita and to my grandfathers Jacinto and Saul.

Acknowledgements

I would like to thank all of those that, throughout this long journey, were present in this project, either as part of the IPN team, or part of the Cosmedesk team.

To Professor Catarina Helena Branco Simões da Silva and Professor Hugo Ricardo Gonçalo Oliveira I would like to thank all of the positive addition of informative sources, the weakly checks and the additional suggested challenges, for example the participation in the SLATE'22 conference.

In personal and professional terms, appreciate this collaborative project between the University of Coimbra and IPN, not only allowed me to grow and strengthen important technical components but also to grow my soft skills. This project allowed me to fortify my responsibilities, my communication skills and evolve my professional intelligence.

Contents

Acronyms

AICIS Australian Industrial Chemicals Introduction Scheme. xvii, xviii, xx, xxii, 46–48, 52–58, 63, 68, 70, 72, 77–82, 85, 91

ALBERT A Lite BERT for Self-supervised Learning of Language Representations. 12

ATSDR Agency for Toxic Substances and Disease Registry. xvii, 46–48, 51, 52, 55, 56, 68, 70, 72, 91

BERT Bidirectional Encoder Representations from Transformers. 12–14, 26–28, 61, 64, 85, 86, 90

BioBERT BioBERT: a pre-trained biomedical language representation model for biomedical text mining. 61, 64, 85, 86

BLEU Bilingual Evaluation Understudy. 14–16, 59, 63

BOS Begin Of Sequence. 18

CIR Cosmetic Ingredient Review. 46–48, 57, 91

CNN Convolutional Neural Network. 12, 16, 27, 28

CR Coreference Resolution. 7, 9

CRF Conditional Random Field. 10

D2T Data-to-text. iii, 2, 3, 23, 24, 26, 38, 39, 42–44, 68–70, 90

EOS End Of Sequence. 18

FN False Negative. 82

FP False Positive. xx, 82–84

GAN Generative Adversarial Network. 26

IE Information Extraction. iii, xvii, xviii, 1–3, 5, 7–10, 27, 28, 30, 31, 33, 38–43, 48, 57, 59, 60, 65, 66, 69, 73–75, 77, 82, 90

IR Information Retrieval. 10

KB Knowledge Base. 10

LCS Longest Common Subsequence. 15, 16

LSTM Long Short-Term Memory. 12

ML Machine Learning. 10, 26

MRC Machine Reading Comprehension. iii, 10–14, 23, 29, 33, 38, 40

NEL Named Entity Linking. 7

NER Named Entity Recognition. 7, 8, 13, 28, 31, 33

NLG Natural Language Generation. 5, 16, 23–26, 33

NLP Natural Language Processing. 5–7, 10–12, 16, 22, 30–33, 59, 90

OECD Organization for Economic Cooperation and Development. 46–48, 57, 91

OpenQA Open-domain Question Answering. 10

POS Parts-of-Speech. 6, 8, 13, 31, 32

POS-T Part-of-Speech Tagger. 8

POST Parts-of-Speech Tagging. 6, 7

QA Question Answering. iii, 2, 5, 7–11, 21, 27–30, 33, 46, 59–64, 66, 90

RE Relation Extraction. 7, 9

RIFM Research Institute for Fragrance Materials. 46–48

RNN Recurrent Neural Network. 12, 13, 16, 18, 26

RoBERTa Robustly Optimized BERT Pretraining Approach. 12, 23, 30, 61, 64, 85, 86, 88, 90

ROUGE Recall-Oriented Understudy for Gisting Evaluation. xxii, 14, 15, 59, 60, 63–65, 85, 86, 90

SCCS Scientific Committee on Consumer Safety. xvii, xviii, xx, xxii, 46–48, 51, 52, 55–57, 65–68, 70–72, 77–82, 85, 91

SMT Statistical Machine Translation. 15

SQuAD Stanford Question Answering Dataset. 23, 61, 62

SVM Support Vector Machine. 10

TIE Temporal Information Extraction. 7, 9

TN True Negative. 82

TOC Table of Contents. xvii, xviii, xxii, 40, 47–52, 55–57

TP True Positive. xx, 82–84, 87

List of Figures

List of Tables

Listings

Chapter 1

Introduction

With the increasing volume of available information, companies need to develop processes for mining information that may be essential for their business. Unfortunately, much of this information is not present in structured databases, but rather in unstructured or semi-structured texts. Humans are capable of doing this process of extracting information from texts, however, it can take a long time to complete. IE emerged as a solution to deal with this problem (Cvitaš, 2010).

In the case of Safety Desk (CENTRO-01-0247-FEDER-113485), the problem in question emerged from the necessity of the company to optimise the time it takes to elaborate a report of a chemical substance. The process currently consists of a human searching information about the chemical compound and preparing a report with all the relevant information. The research process is done in different types of databases, including structured (*e.g*, websites) and unstructured (*e.g*, PDFs, articles).

In order to decrease required resources or time, the proposed solution to this problem is to use the main sources of information, PDFs from regulated sources, and with the right technology build an automated solution capable of extracting information and building reports.

This document is a dissertation of the Masters in Informatics Engineering of the University of Coimbra. The work is developed in the scope of the project Safety Desk, a partnership between the University of Coimbra, Instituto Pedro Nunes, and Talent Ingredient.

Throughout this chapter, the goals of the work, the motivations inherent to it, the contributions built throughout this journey, as well as the organization of the document, will be introduced.

1.1 Context and Motivation

The main subject to be analyzed and resolved in this project is the question from Safety Desk: how to extract information from human written PDFs regarding physicochemical and toxicological properties of chemical compounds?

At Talent Ingredient, an security advisor in the chemical field is responsible for researching, comparing and labeling information about chemical compounds. This process of "information extraction" is done manually, and, with the higher number of documents

and data sources consulted, the time spent in this task may take weeks. Not finished yet, the security advisor needs to write a toxicological profile of the substance, which has all the information acquired about the physicochemical and toxicological properties of chemical compounds.

The security advisor resorts to multiple data sources with different formats, *i.e.*, websites, xlsx files, PDF files, all of them have with relevance for the security advisor, either in terms of quantity or quality of information. The problem for the security advisor is that the PDFs data sources contain much information written by humans in an unstructured format, *i.e.*, natural language. These PDF may contain relevant information that needs to be compared with the information from the other sources, but acquiring the information from PDFs is very time consuming. This project is motivated by the challenge of trying to optimize the process of extracting information from PDFs.

After analysing the keywords in the job developed by the security advisor we can assume that, in a first stage, we are looking to extract the information that is present in the documents relying on an IE system. At a later stage, we are aiming to generate a toxicological profile with the information extracted, using Data-to-text (D2T) generation algorithms. So, we can say that IE and D2T are the main components in the resolution of this challenge, that by itself is already a great approach in developing this system.

1.2 Goals

From the business previously contextualized, a set of clear objectives emerge:

1. To explore automatic tools for extracting physicochemical and toxicological information from semi-structured and unstructured documents from relevant data sources;
2. To develop tools for generating the text of the toxicological profile automatically;
3. To develop an API for making the previous easily accessible and enable their integration in Talent Ingredient's platform (Cosmedesk);

These objectives have as main goal the optimization of time and resources in the elaboration of a report of chemical substances.

1.3 Contributions

Many contributions were produced throughout the duration of the project in order to complete challenges proposed or, for clear reasons, achieve the goals enumerated in Chapter 1.2.

Regarding practical contributions for the Safety Desk project we implemented the pipeline proposed in this dissertation, *i.e.*, the Preprocessing, the IE using Question Answering (QA) models, the Data Verification and the D2T processes. In the implementation we provide the IE service by requests using a RESTfull API.

In terms of challenges, it was proposed to write an article (Ferreira et al., 2022) for the Symposium on Languages Applications and Technologies (SLATE) conference where we explain the general IE approach suggested in this dissertation and present exploratory results obtained. The article was accepted in the conference where a presentation of the article was made followed by its publication.

1.4 Structure of the document

This document reports the work of this dissertation and is structured in the following chapters:

- In Chapter 2, a survey of all the theoretical aspects necessary to consolidate the goals mentioned in the previous subsection will be carried out. Yet, the different technologies needed within the pipeline will be raised, as well the work related to the application of these techniques and technologies for the same purpose as this project;
- In Chapter 3, the project problem is detailed as well as the proposed approach for the Safety Desk project. The risks of the approach are assessed and the scope of the project is detailed;
- In Chapter 4, the initial Phase of the pipeline, the Preprocessing Phase, is deeply detailed, from the initial exploratory work, the development of the Preprocessing Phase for different types of documents, to mentioning the problems found and further work needed in the Preprocessing Phase;
- In Chapter 5, the IE process, from the exploratory work done in the Hugging Face Hub to the models used in this phase, and the Data Verification Process are explained, mentioning the results obtained in experiments done for the article "Question Answering For Toxicological Information Extraction" (Ferreira et al., 2022);
- In Chapter 6, the D2T process for the toxicological profile is explained, also mentioning its limitations, and the RESTfull API and Evaluation Webpage implementation is exposed;
- In Chapter 7, the results of the evaluation carried out by the security advisor are stated and discussed;
- In the last chapter, Chapter 8, a brief conclusion of the work is presented as well an overview of the work developed and the difficulties in each phase. Final considerations regarding future work are also made.

Chapter 2

Background

The tasks of extracting information from documents and generating text from the information extracted has a deep and complex study background. Therefore, in this chapter, we are going to explore the theoretical contents that can be used to tackle the problem in hand, *i.e.*, Natural Language Processing (NLP) tasks for the first part of the Safety Desk problem, Information Extraction (IE) and Question Answering (QA), and Natural Language Generation (NLG) tasks for the second part, generating text from the information extracted.

2.1 Natural Language Processing

NLP is an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech. Applications of NLP include a number of fields of study, such as machine translation, summarization, user interfaces, multilingual and cross-language information retrieval, speech recognition, expert systems, etc (Chowdhury, 2003).

All the applications of NLP can be applied in various business models, *e.g.*, ecommerce, understanding which words the consumers most frequently use in reviews, speech recognition, virtual assistants, and in many known daily technologies that we utilize, like web search engines, social media, auto correct and spell check (Bahja, 2020). With the advance of technology and computers, NLP applications are each day more present in our society.

For the problem at hand, we are going to explore the background work and approaches in the NLP fields that are typically used in pipelines of related works (Gui et al., 2017; Nguyen et al., 2019, 2021b, 2020b, 2021a; Arici et al., 2022; Li et al., 2020) of the Safety Desk problem, *i.e*, extracting information from unstructured data using QA models. The fields that we are going to delve into are IE and QA.

2.1.1 Natural Language Processing Tasks

Many high-level NLP tasks, *e.g.*, IE, QA, Sentiment analysis, etc., involve syntactic and semantic analysis, used to break down human language. Syntactic analysis identifies the syntactic structure of a text and the dependency relationships between words. Semantic analysis focuses on identifying the meaning of language. However, since language is polysemic and ambiguous, semantics is considered one of the most challenging areas in

NLP.

Semantic tasks analyze the structure of sentences, word interactions, and related concepts, in an attempt to discover the meaning of words, as well as understanding the topic of a text. Some of the tasks of NLP are *Tokenization*, Parts-of-Speech Tagging (POST) and *Parsing*.

Tokenization

Tokenization is the process of tokenizing or splitting a string of words into semantically useful units called tokens. Sentence tokenization splits sentences within a text, and word tokenization splits words within a sentence (Singh, 2018). Generally, word tokens are separated by blank spaces, and sentence tokens by stops. However, high-level tokenization can be performed for more complex structures, *i.e.*, words that often go together, otherwise known as collocations, *e.g.*, New York.

Parts-of-speech tagging

POST labels sequences of words in natural language with their Parts-of-Speech (POS) such as noun, verb, adjective, preposition,etc (Lin et al., 2016). POST is a fundamental step in various NLP tasks, such as speech recognition, speech synthesis, machine translation, information retrieval and information extraction (Singh, 2018).

POST approaches can generally fall into two categories: Rule-based approaches and statistical approaches. Rule-based approaches apply language rules to improve the accuracy of tagging. The limitation of this approach lies in requirement of large annotated data which require expert linguistic knowledge, labor and cost. In order to overcome the shortcoming of this approach the "transformation based approach" as proposed in which rules are automatically learned from corpora (Singh, 2018). On the other hand, statistical methods use Decision Trees (Dzunic et al., 2006), Hidden Markov Model (Miller et al., 1999), Maximum Entropy classifier (Nigam et al., 1999), Support Vector Machine (Giménez and Marquez, 2004) and deep learning based POST (Singh, 2018; Deshmukh and Kiwelekar, 2020).

Parsing

Parsers can generally be divided into two broad categories based on their underlying grammatical formalism: constituency parsers and dependency parsers. Constituency parsers (also known as tree-bank parsers) produce syntactic analysis in the form of a tree that shows the phrases comprising the sentence and the hierarchy in which these phrases are associated. Constituency parsers have been used for pronoun resolution, labeling phrases with semantic roles and assignment of functional category tags. Constituency parsers overlook functional tags when training. Therefore, they cannot use them when labeling unseen text. Dependency parsers analyze the sentence as a set of pairwise word-to-word dependencies. Each dependency has a type that reflects its grammatical function. Dependency parsers model language as a set of relationships between words and construct a graph for each sentence, and each arc in the graph represents a grammatical dependency connecting the words of the sentence to each other (Singh, 2018; Entwisle and Powers, 1998).

2.1.2 Information Extraction

The significant growth of data provides a chance for humans to approach information from many sources. To address this opportunity, IE can be considered as an appropriate solution for converting unstructured and semi-structured data to structured data (Nguyen et al., 2020c). In detail, IE is the process of analyzing text and identifying mentions of semantically defined entities and relationships within it. Hence, the goal of IE is to extract salient facts about pre-specified types of events, entities, or relationships, in order to build more meaningful, rich representations of their semantic content, which can be used to populate databases that provide more structured data.

IE is most valuable in applications where the volume of textual data to be studied simply overwhelms the reader. For example, medical and biomedical literature is growing at a rate of more than 500,000 articles per year, and hospitals and medical practices generate large volumes of electronic medical records to be reviewed at each patient visit or admission (Grishman, 2015). So from the business point of view, IE is a crucial step for digital transformation (Herbert, 2017).

IE refers to the use of computational methods to identify relevant pieces of information in document generated for human use and convert this information into a representation suitable for computer based storage, processing, and retrieval (Wimalasuriya and Dou, 2010). The input to IE system is a collection of documents (email, web pages, news groups, news articles, business reports, research papers, blogs, resumes, proposals, and so on) and the output is a representation of the relevant information from the source document according to some specific criteria (Singh, 2018).

IE technologies help to efficiently and effectively analyze free text and to discover valuable and relevant knowledge from it in the form of structured information. Hence, the goal of IE is to extract salient facts about pre-specified types of events, entities, or relationships, in order to build more meaningful, rich representations of their semantic content, which can be used to populate databases that provide more structured input (Singh, 2018). Figure 2.1 is an example of IE representations that can be obtained.

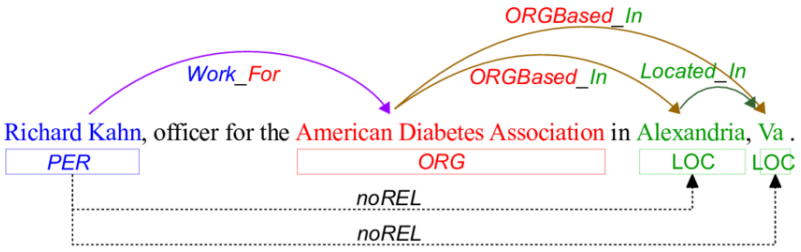


Figure 2.1: Example of IE. Adapted from (Gupta, 2019).

Information Extraction Architecture

Information Extraction is often an early stage in the pipeline for various high level tasks, such as QA Systems, Machine Translation, event extraction, user profile extraction, and so on. Various subtasks involved in IE are: Named Entity Recognition (NER), Named Entity Linking (NEL), Coreference Resolution (CR), Temporal Information Extraction (TIE), Relation Extraction (RE). Various low level tasks in NLP such as POST, chunking, parsing, NER, are fundamental building blocks of complex NLP tasks such as Knowledge

Base construction, text summarization, QA systems, and so on (Singh, 2018). Hence, the effectiveness of these low level tasks highly determines the performance of high level tasks. Error in low level tasks gets propagated to high level tasks, degrading the overall performance. In this section, we will discuss various sub-tasks in the field of Information Extraction.

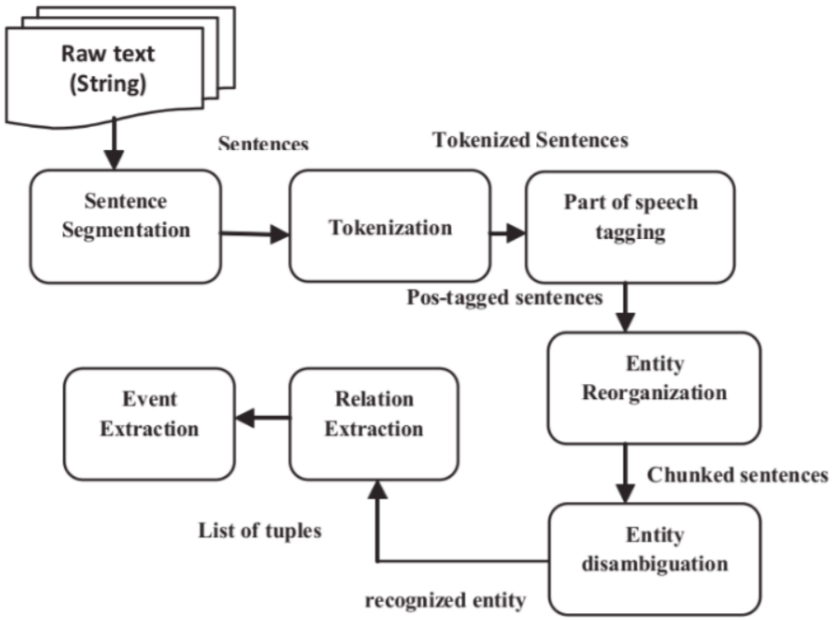


Figure 2.2: General Information Extraction Architecture. Adapted from (Singh, 2018).

The effectiveness of various IE tasks down the pipeline highly depends upon pre-processing stages such as *Tokenizer*, Part-of-Speech Tagger (POS-T), and *Parser*. Tokenizer extracts tokens from the text. Tokenizer can be treated as a classifier which classifies tokens into orthographic classes. POS-T assigns one tag to each word from various POS classes, *e.g.*, "Sam" is a proper-noun and "they" is personal-pronoun. Noun Phrase Recognizer finds the noun phrases from the text. For example, in "the president of Portugal", the president is a noun and it refers to a person, whereas Portugal is a noun phrase and refers to name of the country. NER, finally assigns a particular named entity class from various classes such as: person, organization, location, date, time, money, percent, e-mail address and web-address. (Singh, 2018).

Named Entity Recognition

NER is the task of recognizing Named Entities occurring in the text, *i.e.*, to find Person (PER), Organization (ORG), Location (LOC) and Geo-Political Entities (GPE). For instance, in the sentence "Cristiano Ronaldo lives in the United Kingdom", NER system extracts "Cristiano Ronaldo" which refers to name of the person and "United Kingdom" which refers to name of the country. NER serves as the basis for various crucial areas in Information Management, such as Semantic Annotation, QA and Ontology Population (Singh, 2018; Mohit, 2014; Guo et al., 2009).

Coreference Resolution

CR is the task that determines which noun phrases (including pronouns, proper names and common names) refer to the same entities in documents (Kong et al., 2010). For instance, in the sentence, "I have seen the annual report. It shows that we have gained 15% profit in this financial year". "I" refers to name of the person, "It" refers to annual report and "we" refers to the name of the company in which that person works. CR plays vital role in tasks as natural language understanding, text summarization, information extraction, textual entailment, etc (Singh, 2018; Ng and Cardie, 2002).

Relation Extraction

RE is the task of detecting and classifying predefined relationships between entities identified in the text. In other words, it is a way of transforming unstructured text into a structural form which can be used in web-search, QA (Gardner and Mitchell, 2015; Pawar et al., 2017; Singh, 2018). The notion of a relation is inherently ambiguous and there is often an inherent ambiguity about what a relation "means", which is often reflected in high inter-annotator disagreements. As the expression of a relation is largely language-dependent, it makes the task of RE language dependent (Pawar et al., 2017).

Temporal Information Extraction

TIE refers to the task of identifying events, *i.e.*, information which can be ordered in a temporal order, in free text and deriving detailed and structured information about them(Ling and Weld, 2010; Singh, 2018). For instance, in the statement, "Yesterday the president Marcelo Rebelo de Sousa visited the Azores", "Yesterday" is a noun phrase which refers to temporal information. Temporal information is important when we want to extract structured information from natural language text according to some temporal criteria such as news, organization of events date-wise or biographies.

Methods for Information Extraction

The various approaches used in IE can be broadly categorized into three main categories:

- **Pattern matching based approach**: In this approach, extraction patterns are defined using formalisms, normally Regular Expressions. These patterns can be easily matched directly with the given input text and the matched text is extracted, which corresponds to an occurrence of that entity. For example, if we want to extract corporate news, then we define simple regular expressions with cue words such as "Inc.", "Co.", "Company", "Limited" and so on. Though it provides a quick and easy process, this approach has limitations as it is usually not possible to provide all the cue words related to particular domain. In order to make it more exploratory, Regular Expressions patterns are enriched by incorporating lexical information and incorporating special cases and domain knowledge. Despite these limitations, this approach is widely used in practice (Grishman, 2015; Singh, 2018).

- **Gazetteer based approach**: This approach makes use of a predefined list of all possible values of an named entity, called a gazetteer. Gazetteer is only possible for

those named entities which have a finite number of possible values. Though this approach is fast and accurate, the limitation lies in preparing complete and accurate gazetteers (Rijhwani et al., 2020; Singh, 2018).

- **Machine Learning based approach**: In this approach, Machine Learning algorithms automatically learn the IE patterns by generalizing from a given set of examples. First we have to create a training data-set, which is a collection of documents in which all occurrences of named entities of interest are manually marked or tagged. Machine Learning (ML) algorithms such as Decision Trees, Naive Bayes classifier, Maximum Entropy (MaxEnt), Conditional Random Field (CRF), Support Vector Machine (SVM), and, more recently, *Transformers* use features such as word surrounding an occurrence of named entity (Singh, 2018).

Given the amount of data sources and, consecutively, documents that we are going to use in this project, we will focus in the Machine Learning based approach, specifically the state-of-the art model, the *Transformer* model, given the ability of generalization of the referred models.

2.1.3 Question Answering in IE

QA is a research area that combines research from different, but related, fields which are Information Retrieval (IR), IE and NLP (Allam and Haggag, 2012). QA aims to provide precise answers in response to questions in natural language. Nowadays, many web search engines like Google[1] and Bing[2] have been evolving towards higher intelligence by incorporating QA techniques into their search functionalities. Empowered with these techniques, search engines now have the ability to respond with high precision to some types of questions such as:

-Q: "Who is the president of Portugal?" -A: "Marcelo Rebelo de Sousa"

The whole QA landscape can roughly be divided into two parts: textual QA and Knowledge Base (KB) QA, according to the type of information source where answers are derived from. Textual QA mines answers from unstructured text documents while KB-QA extracts answers from a predefined structured KB that is often manually constructed. Textual QA is generally more scalable than the latter, since most of the unstructured text resources it exploits to obtain answers from are fairly common and easily accessible, such as Wikipedia[3], articles, books, etc. Textual QA is studied under two task settings based on the availability of contextual information, *i.e.*, Machine Reading Comprehension (MRC) and Open-domain Question Answering (OpenQA) (Zhu et al., 2021).

MRC, which originally took inspiration from language proficiency exams, aims to enable machines to read and comprehend specified context passages for answering a given question. In comparison, OpenQA tries to answer a question without been given a limited context. It usually requires the system to, first, search for the relevant documents as the context can be from either a local repository or the *World Wide Web*, and then generate the answer (Zhu et al., 2021). In the end, MRC can be considered as a step to OpenQA. Given the characteristics of our problem, we will focus in MRC domain and how to build a MRC system.

[1] https://www.google.com/
[2] https://www.bing.com/
[3] https://www.wikipedia.org/

Machine Reading Comprehension

MRC aims to teach machines to understand a text like a human. The machine should consider both the story and the question, and answer the question after necessary interpretation and inference (Zhang et al., 2019) (Table 2.1). The goal of MRC systems is to learn the predictive function f, which extracts or generates the appropriate answer A by receiving the context C and the related question Q:

$$f : (C, Q) => A$$

Context
Computational complexity theory is a branch of the theory of computation in theoretical computer science that focuses on classifying computational problems according to their inherent difficulty,and relating those classes to each other. A computational problem is understood to be a task that is in principle amenable to being solved by a computer, which is equivalent to stating that the problem may be solved by mechanical application of mathematical steps, such as an algorithm.
Question
What branch of theoretical computer science deals with broadly classifying computational problems by difficulty and class of relationship?
Answer
Computational complexity theory

Table 2.1: Example MRC system objective.

MRC is a useful benchmark to evaluate natural language understanding of machines and has been a challenging task in the NLP field with considerable research in recent years. For measuring the machine comprehension in a piece of natural language text, a set of questions about the text is given to the machine, and its responses are evaluated against the gold standard. Even though MRC is routinely referred to as QA, they are different in the following ways (Baradaran et al., 2020):

- The main objective of QA systems is to answer the input questions, while in an MRC system the main goal is to understand natural languages by machines;
- The only input to QA systems is the question, while the inputs to MRC systems are the question and the corresponding context that should be used to answer the question. For this reason, MRC is referred to as QA from text;
- The main information source that is used to answer questions in MRC systems are natural language texts, while QA systems use structured and semi-structured data sources such as knowledge-bases;

The approaches used for developing MRC systems can be grouped into three categories: rule-based methods, classical machine learning-based methods, and deep learning-based methods.

The traditional rule-based methods use the rules handcrafted by linguistic experts. These methods suffer from the problem of the incompleteness of the rules. Also, this approach is domain specific where for any new domain, a new set of rules should be handcrafted.

The second approach is based on classical machine learning. These methods rely on a set of human-defined features and train a model for mapping input features to the output.

The third approach uses deep learning methods to learn features from raw input data automatically. These methods require a large amount of training data to create high accuracy models. Because of the growth of available data and computational power in recent years, deep learning methods achieved state-of-the-art results in many tasks. In the MRC task, most of the recent research falls into this category (Baradaran et al., 2020; Zhang et al., 2019; Sun et al., 2021). Two main deep learning architectures used by MRC researchers are the Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN).

RNNs are often used for modeling sequential data by iterating through the sequence elements and maintaining a state containing information relative to what have seen so far. A type of RNNs are Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and in MRC systems, like other NLP tasks, these architectures have been commonly used in different parts of the pipeline, such as for representing questions and contexts. But in recent years, the attention-based *Transformer* (Vaswani et al., 2017) has been emerged as a powerful alternative to the RNN architecture.

CNN is a type of deep learning model that is universally used in computer vision applications. It utilizes layers with convolution filters that are applied to local spots of their inputs. In MRC systems, CNN is used in the embedding phase (character embedding) (Baradaran et al., 2020).

In recent years, with the advent of attention-based *transformer* architecture (Vaswani et al., 2017) as an alternative to common sequential structures, new transformer-based language models, such as Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018), Robustly Optimized BERT Pretraining Approach (RoBERTa) (Liu et al., 2019b), A Lite BERT for Self-supervised Learning of Language Representations (ALBERT) (Lan et al., 2019), have been introduced. They are used as the basis for new state-of-the-art results in the MRC task.

Machine Reading Comprehension Phases

Most of the recent deep learning-based MRC systems have the following phases: embedding phase, reasoning phase, and prediction phase. Figure 2.3 presents a typical neural machine reading comprehension system, which takes the context and question as inputs and the answer as output. In Figure 2.3, "Embeddings" represents phase 1 (embedding phase), "Feature Extraction" and " Context-Question Interaction" represents the reasoning phase and "Answer Prediction" represents the prediction phase.

In the **embedding phase**, input characters, words, or sentences are represented by real-valued dense vectors in a meaningful space. The goal of this phase is to provide question and context embedding. Different levels of embedding are used in MRC systems, *i.e.*, character-level, word-level embeddings can capture the properties of words, and higher level representations, hybrid word-character embedding and sentence embedding, can represent syntactic and semantic information of input text (Baradaran et al., 2020).

Character embedding is useful to overcome unknown and rare words problems (Dhingra et al., 2016). To generate the input representation, deep neural network models are commonly used. Word embedding consists of the words in a numeric vector space, which is performed by two main approaches: 1) non-contextual embedding, and 2) contex-

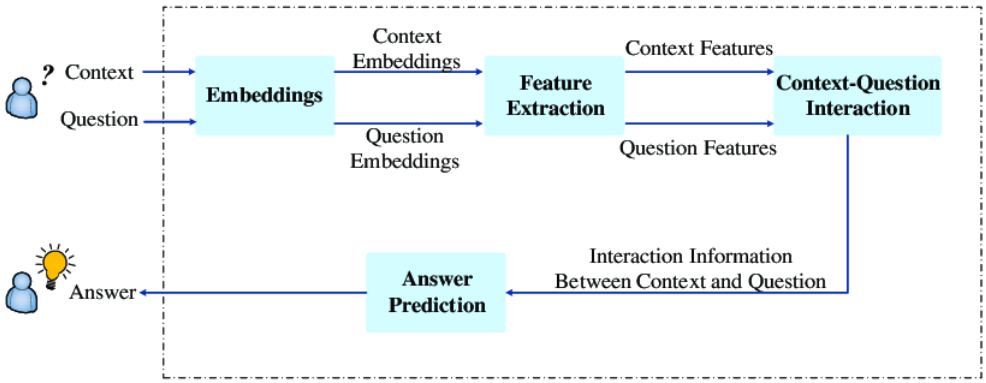


Figure 2.3: Machine Reading Comprehension System Architecture. Adapted from (Liu et al., 2019a).

tual embedding. Non-contextual word embeddings present a single general representation for each word, regardless of its context. Contextual word embedding move beyond word-level semantics and represent each word considering its context (surrounding words). For learning the contextual word embedding, a sequence modeling method, usually a RNN, is used.

Hybrid word-character embedding is a combination of word embedding and character embedding. Hybrid embedding tries to use the strengths of both word and character embeddings. A simple approach is to concatenate the word and character embeddings. This approach suffers from a potential problem. Word embedding has better performance for frequent words, while it can have negative effects for representing rare words. The reverse is true for character embedding. To solve this problem, some researchers introduced a gating mechanism which regulates the flow of information. A fine-grained gating mechanism for dynamic concatenation of word and characters embedding was proposed in (Yang et al., 2016), where the mechanism uses a gate vector, which is a linear multiplication of word features (POS and NER) to control the flow of information of word and character embeddings. Sentence embedding is a high-level representation in which the entire sentence is encoded in a single vector. It is often used along with other embeddings. However, sentence embedding is not so popular in MRC systems, because the answer is often a sentence part, not the whole sentence (Baradaran et al., 2020).

In the **reasoning phase** the goal is to match the input query (question) with the input document (context). In other words, this phase determines the related parts of the context for answering the question by calculating the relevance between question and context parts. The attention mechanism (Chorowski et al., 2015), originally introduced for machine translation, is used for this phase. The attention mechanism used in MRC systems can be explored in three perspectives: direction, dimension, and number of steps. Direction can be divided in two approaches: 1) one directional and bi-directional. One directional signifies which query words are relevant to each context word while bi-directional signifies which context words have the closest similarity to one of the query words and are hence critical for answering the question. In transformer-based MRC models like BERT-based models, the question and context are processed as one sequence, so the attention mechanism can be considered as bi-directional attention (Baradaran et al., 2020; Sun et al., 2021; Zhang et al., 2019).

There are two attention dimensions: 1) one-dimensional and 2) two-dimensional attentions. In one-dimensional attention, the whole question is represented by one embedding vector, which is usually the last hidden state of the contextual embedding. In two-

dimensional attention, every word in the query has its own embedding vector (Baradaran et al., 2020; Sun et al., 2021; Zhang et al., 2019). One-dimensional does not pay more attention to important question words unlike the two-dimensional attention.

There are three types of number of steps in MRC systems: 1) single-step reasoning, 2) multi-step reasoning with fixed number of step and 3) multi-step reasoning with dynamic number of steps. In the single step reasoning, question and passage matching is done in a single step. However, the obtained representation can be processed through multiple layers to extract or generate the answer. In multi-step reasoning, question and passage matching is done in multiple steps such that the question-aware context representation is updated by integrating the intermediate information in each step. The number of steps can be static or dynamic. Dynamic multi-step reasoning uses a termination module to decide whether the inferred information is sufficient for answering or more reasoning steps are still needed. Therefore, the number of reasoning steps in this model depends on the complexity of the context and question (Baradaran et al., 2020; Zhang et al., 2019).

In the **prediction phase** the final output of the MRC system is specified. The output can be extracted from context or generated according to context. In some cases, multiple choices are presented to the system, and it must select the best answer according to the question and context (Greco et al., 2017; Baradaran et al., 2020; Zhang et al., 2019). The extraction mode is implemented in different forms. If the answer is a span of context, the start and end indices of the span are predicted in many studies by estimating the probability distribution of indices over the entire context (Duan et al., 2017; Min et al., 2017). In other studies (Sachan and Xing, 2018), the candidate answers are extracted first, which are ranked by a trained model. These outputs can be sentences or entities.

2.1.4 Similarity Metrics

Text similarity aims at determining how "close" two pieces of text are. There are two types of similarity, semantic similarity and lexical similarity. Semantic similarity is a metric where the distance between text snippets is based on the strength of the proximity of meaning between them, *i.e.*, considering that words can have different meanings and different words can be used to represent a similar concept. On the other hand, the lexical similarity is the primitive form of text similarity where two text snippets are considered similar if they contain the same words/characters.

Both similarity types have their own purpose where each has useful use cases, *e.g.*, semantic similarity is useful in cases that relationships, meanings and contexts are important and lexical similarity is preferable in cases that purely words/strings comparisons are necessary.

Multiple similarity metrics have been created for both types of text similarity. In the field of semantic similarity the most popular metrics are *Knowledge-based semantic similarity methods* and *Corpus-based semantic similarity methods*. *Knowledge-based semantic similarity methods* calculate semantic similarity between two terms based on the information derived from one or more underlying knowledge sources, such as ontologies/lexical databases and dictionaries, *e.g.*, *Edge-counting Methods*, *Feature-based Methods* and *Information Content-based Methods*. *Corpus-based semantic similarity methods* measures semantic similarity between terms using the information retrieved from large corpora, *e.g. Word2Vec*, *BERT Score*, *fastText*. On the other hand, some of the most popular lexical similarity metrics are *Damerau–Levenshtein distance*, *Bilingual Evaluation Understudy (BLEU) Score*, *Recall-Oriented Understudy for Gisting Evaluation (ROUGE) Score* and

Cosine Similarity (Chandrasekaran and Mago, 2021).

For the Safety Desk problem we are intending to use similarity metrics in order to compare the results of the information extracted, *i.e.*, the words/strings. Given the use case we opted to use the a lexical similarity approach, exploring deeply the ROUGE and BLEU Score.

BLEU Score

The BLEU metric is designed to measure how close Statistical Machine Translation (SMT) output is to that of human reference translations. It is important to note that translations, SMT or human, may differ significantly in word usage, word order, and phrase length. To address these complexities, BLEU attempts to match variable length phrases between SMT output and reference translations (Wołk and Marasek, 2015). In the BLEU metric, scores are calculated for individual translated segments and then those scores are averaged over the entire corpus to reach an estimate of the translation's overall quality. The BLEU score is always a number between 0 and 1.

The BLEU metric works by counting matching n-grams in the candidate translation to n-grams in the reference text, where 1-gram or unigram would be each token and a bigram comparison would be each sequence of two words. The comparison is made regardless of word or n-gram order. The counting of matching n-grams is modified to ensure that it takes the occurrence of the words in the reference text into account, not rewarding a candidate translation that generates an abundance of reasonable words. This is referred to as modified n-gram precision (Papineni et al., 2002).

ROUGE Score

ROUGE is a package for automatic evaluation of summaries and its evaluations and it includes several automatic evaluation methods that measure the similarity between summaries, *i.e.*, *ROUGE-N*, *ROUGE-L*, *ROUGE-W* and *ROUGE-S*.

ROUGE-N: N-gram Co-Occurrence Statistics

ROUGE-N is an n-gram recall between a candidate summary and a set of reference summaries where "n" stands for the length of the n-gram. *ROUGE-N* is a recall-related measure, closely related to the BLEU Score (Lin, 2004).

ROUGE-L: Longest Common Subsequence

Given two sequences X and Y, the Longest Common Subsequence (LCS) of X and Y is a common subsequence with maximum length. LCS has been used in identifying cognate candidates during construction of N-best translation lexicon from parallel text. LCS can be used pairwise to compare similarity between two texts (Lin, 2004).

ROUGE-W: Weighted Longest Common Subsequence

LCS has a problem of not differentiating LCSs of different spatial relations within their embedding sequences, *e.g.*, given a reference sequence X and two candidate sequences Y_1 and Y_2, as in Figure 2.4, Y_1 and Y_2 have the same *ROUGE-L* score, however, in this case, Y_1 should be the better choice than Y_2 because Y_1 has consecutive matches. To improve the basic LCS method, we can simply remember the length of consecutive matches encountered so far to a regular two dimensional dynamic program table computing LCS. This is called

weighted LCS (WLCS) (Lin, 2004).

$$X: \quad [\underline{A}\ \underline{B}\ \underline{C}\ \underline{D}\ E\ F\ G]$$
$$Y_1: \quad [\underline{A}\ \underline{B}\ \underline{C}\ \underline{D}\ H\ I\ K]$$
$$Y_2: \quad [\underline{A}\ H\ \underline{B}\ K\ \underline{C}\ I\ \underline{D}]$$

Figure 2.4: *ROUGE-L* example sequences. Adapted from (Lin, 2004).

ROUGE-S: Skip-Bigram Co-Occurrence Statistics

Skip-bigram is any pair of words in their sentence order, allowing for arbitrary gaps. Skip-bigram co-occurrence statistics measure the overlap of skip-bigrams between a candidate translation and a set of reference translations. One advantage of skip-bigram vs. BLEU is that it does not require consecutive matches but is still sensitive to word order. Comparing skip-bigram with LCS, skip-bigram counts all in-order matching word pairs while LCS only counts one longest common subsequence (Lin, 2004).

2.2 Transformers

Since its introduction (mid-2017), the Transformer (Vaswani et al., 2017) has rapidly become the dominant architecture for NLP, surpassing alternative neural models such as CNN and RNN in performance for tasks in both NLP and NLG. The architecture scales with training data and model size, facilitates efficient parallel training, and captures long-range sequence features (Wolf et al., 2020). Model pretraining (McCann et al., 2017; Howard and Ruder, 2018) allows models to be trained on generic corpora and subsequently be easily adapted to specific tasks with strong performance. The Transformer architecture is particularly conducive to pretraining on large text corpora, leading to major gains in accuracy on downstream tasks including text classification, language understanding, machine translation, coreference resolution, commonsense inference, and summarization among others.

The *Transformers* library, maintained by HuggingFace[4], is dedicated to supporting Transformer-based architectures and facilitating the distribution of pre-trained and fine-tuned models. At the core of the library is an implementation of the Transformer which is designed for both research and production.The philosophy is to support industrial-strength implementations of popular model variants that are easy to read, extend, and deploy. On this foundation, the library supports the distribution and usage of a wide-variety of pre-trained models in a centralized model hub. This hub supports users to compare different models with the same minimal API and to experiment with shared models on a variety of different tasks (Wolf et al., 2020; Braşoveanu and Andonie, 2020).

2.2.1 Transformer Architecture

The original Transformer architecture (Figure 2.5) was introduced in June 2017. It is composed by an encoder-decoder structure where the encoder maps an input sequence of

[4]https://huggingface.co/docs/transformers/index

symbol representations to a sequence of continuous representations and the decoder then generates an output sequence of symbols one element at a time. At each step the model is auto-regressive, consuming the previously generated symbols as additional input when generating the next (Vaswani et al., 2017; Organization, 2021b). The Transformer follows this overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder respectively.

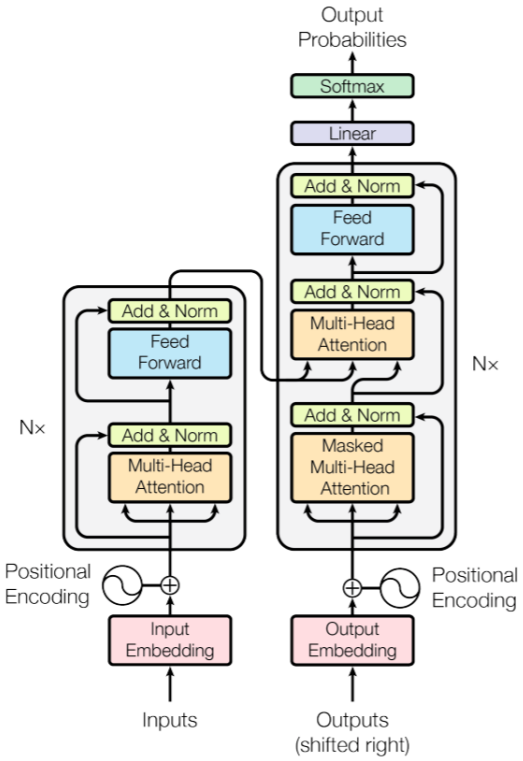


Figure 2.5: The Transformer - model architecture. Adapted from (Vaswani et al., 2017).

Each of these parts, encoder and decoder, can be used independently, depending on the task (Organization, 2021b; von Platen, 2021):

- Encoder-only models: Good for tasks that require understanding of the input, such as sentence classification and named entity recognition;
- Decoder-only models: Good for generative tasks such as text generation;
- Encoder-decoder models or sequence-to-sequence models: Good for generative tasks that require an input, such as translation or summarization;

The key feature of Transformer models is that they are built with special layers called attention layers (Vaswani et al., 2017; Organization, 2021b). These layers tell the model to pay specific attention to certain words in the sentence when dealing with the representation of each word. On the other hand, the attention layers can also be used in the encoder/decoder to prevent the model from paying attention to some special words. For example, in the sentence:

"Bumblebee plays some catchy music and dances along to it."

Humans can easily identify that the word "catchy" refers to the "music". If the tracking is only done between two consecutive words, we may end up in a situation where the word

"it" at the end of the sentence loses its reference. To a human reader the "it" in the sentence is clearly referring to "music". With the attention layers the encoder looks for clues in the other elements of the sentence as it processes them. In this way self-attention can be used to extract understanding of each of the processed elements in the sequence.

Encoder and Decoder Stacks

An important feature of RNN-based encoder-decoder models is the definition of special vectors, such as the End Of Sequence (EOS) and Begin Of Sequence (BOS) vector. The EOS vector often represents the final input vector to warn the encoder that the input sequence has ended and also defines the end of the target sequence. The BOS vector represents the input vector fed to the decoder RNN at the very first decoding step. To output the first logit, *i.e.*, the non-normalized probability, an input is required and since no input has been generated at the first step a special BOS input vector is fed to the decoder RNN.

The **Encoder** is a stack of residual encoder blocks, where each encoder block consists of a bi-directional self-attention layer, followed by two feed-forward layers. A residual connection is employed around each of the two sub-layers, followed by layer normalization. The bi-directional self-attention layer puts each input vector in relation with all input vectors ($x'_{j1:n}$) and by doing so transforms the input vector to a more "refined" contextual representation of itself (x''_j) (von Platen, 2021). Figure 2.6 represents the encoder process given the input "I want to buy a car EOS" to a contextualized encoding sequence. In Figure 2.6 the input sequence is represented by $X_{1:n}$ and the second encoder block is shown in more detail in the red box. The bi-directional self-attention mechanism is illustrated by the fully-connected graph in the lower part of the red box and the two feed-forward layers are shown in the upper part of the red box (von Platen, 2021).

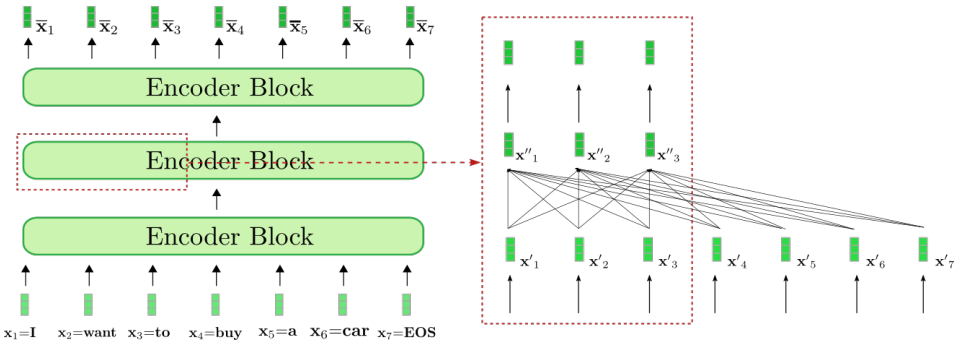


Figure 2.6: Visualization encoder block example. Adapted from (von Platen, 2021).

The **Decoder** is a stack of decoder blocks followed by a dense layer (Language Model Head). The stack of decoder blocks maps the contextualized encoding sequence ($\overline{X}_{1:n}$) and a target vector ($\overline{Y}_{0:i-1}$) sequence, preceded by the BOS vector and cut to the last target vector (Vaswani et al., 2017; Organization, 2021b; von Platen, 2021). Then, the Language Model Heads maps the encoded sequence of target vectors to a sequence of logit vectors ($L_{1:n}$) whereas the dimensionality of each logit vector (l_i) corresponds to the size of the vocabulary. The Language Model Head layer compares the encoded output vector ($\overline{y}_i$) to all word embeddings in the vocabulary ($\overline{y}^{1:vocab}$) so that the logit vector (l_{i+1}) represents the similarity scores between the encoded output vector and each word embedding (von Platen, 2021; Vaswani et al., 2017). Figure 2.7 represents the decoder process given the input $Y_{0:5}$ "BOS", "Ich", "will", "ein", "Auto", "kaufen" that is the German translation for "I want to buy a car". The red box on the right shows a decoder block for the first three target

vectors and in the lower part, the uni-directional self-attention mechanism is illustrated. In the middle part, the cross-attention mechanism is illustrated (von Platen, 2021).

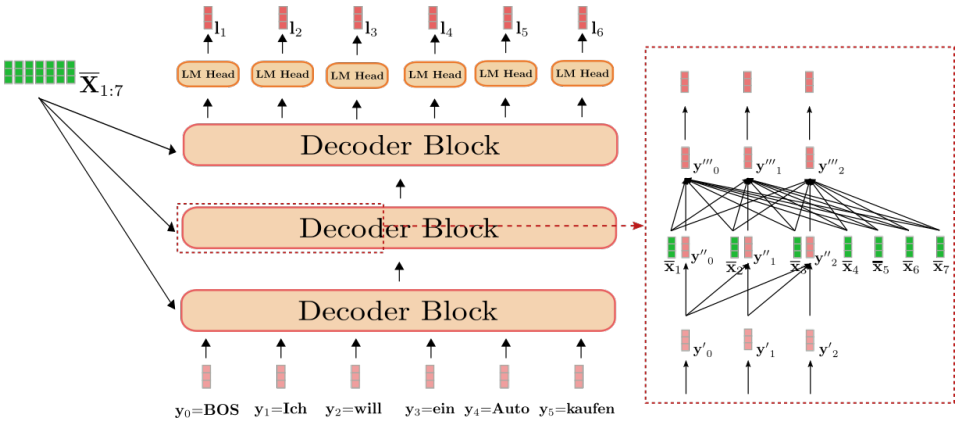


Figure 2.7: Visualization decoder block example. Adapted from (von Platen, 2021).

Attention

The attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key (Vaswani et al., 2017; Organization, 2021a; Cristina, 2021). The Transformer architecture revolutionized the use of attention by dispensing of recurrence and convolutions.

The main components in use by the Transformer attention are the following (Cristina, 2021):

- q and k denoting vectors of dimension, d_k, containing the queries and keys, respectively;
- v denoting a vector of dimension, d_v, containing the values;
- Q, K and V denoting matrices packing together sets of queries, keys and values, respectively;
- W^Q, W^K and W^V denoting projection matrices that are used in generating different subspace representations of the query, key and value matrices;
- W^O denoting a projection matrix for the multi-head output;

A scaled dot-product attention was proposed and then built on to propose multi-head attention (Vaswani et al., 2017). Within the context of neural machine translation, the query, keys and values that are used as inputs to the these attention mechanisms, are different projections of the same input sentence. Therefore, the proposed attention mechanisms implement self-attention by capturing the relationships between the different elements of the same sentence (Cristina, 2021).

Scaled Dot-Product Attention

The scaled dot-product attention (Figure 2.8) first computes a dot product for each query, q, with all of the keys, k. It, subsequently, divides each result by $\sqrt{d_k}$ and proceeds

to apply a softmax function obtaining the weights that are used to scale the values v (Cristina, 2021; Vaswani et al., 2017; Organization, 2021b).

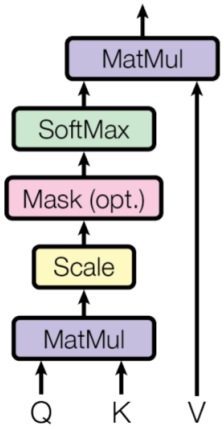


Figure 2.8: Scaled dot-product attention. Adapted from (Vaswani et al., 2017).

In practice, the attention function (Figure 2.9) computes a set of queries simultaneously, packed together into a matrix Q. The keys and values are also packed together into matrices K and V.

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

Figure 2.9: Attention Matrix Computation. Adapted from (Vaswani et al., 2017).

Scaled Dot-Product Attention is similar to dot-product attention except for the added scaling factor of $1/\sqrt{d_k}$ (Vaswani et al., 2017). This scaling factor was added because large values of d_k the dot products grow large in magnitude, pushing the softmax function into regions where it has extremely small gradients (Vaswani et al., 2017), that would lead to the infamous vanishing gradients problem. The scaling factor, therefore, serves to pull the results generated by the dot product multiplication down, hence preventing this problem (Cristina, 2021).

Multi-Head Attention

Their multi-head attention mechanism linearly projects the queries, keys and values h times, each time using a different learned projection. The single attention mechanism is then applied to each of these h projections in parallel, to produce h outputs, which in turn are concatenated and projected again to produce a final result (Cristina, 2021).

The idea behind multi-head attention is to allow the attention function to extract information from different representation subspaces, which would, otherwise, not be possible with a single attention head.

The multi-head attention function is represented in Figure 2.11 here each $head_{i1:h}$ implements a single attention function characterized by its own learned projection matrices (Cristina, 2021; Vaswani et al., 2017; Organization, 2021b). In the proposed Multi-Head Attention, the authors (Vaswani et al., 2017) used h= 8 parallel attention layers, where for each they used d_k=d_{model}/h= 64. Due to the reduced dimension of each head, the

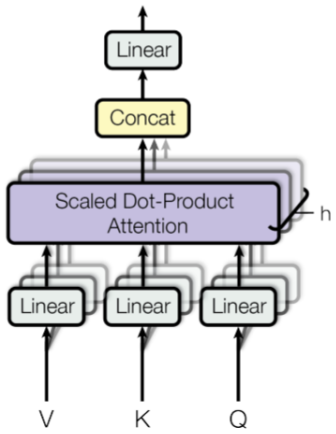


Figure 2.10: Multi-Head Attention. Adapted from (Vaswani et al., 2017).

total computational cost is similar to that of single-head attention with full dimensionality (Vaswani et al., 2017; Cristina, 2021).

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Figure 2.11: Multi-Head Attention Function. Adapted from (Vaswani et al., 2017).

In practice, the multi-head attention is used in the Transformer in three different ways (Vaswani et al., 2017):

1. In the encoder-decoder attention layers where the queries come from the previous decoder layer and the memory keys and values come from the output of the encoder;
2. Self-attention layers in the encoder that allows all of the keys, values and queries that come from the same place to be the output of the previous layer in the encoder;
3. Self-attention layers in the decoder allow each position in the decoder to attend to all positions in it up to and including that position.

2.2.2 Transformers Library

Transformers library[5] is dedicated to supporting Transformer-based architectures and facilitating the distribution of pre-trained and fine-tuned models. At the core of the library is an implementation of the Transformer which is designed for both research and production. Each model is made up of a Tokenizer, Transformer, and Head. The model is pre-trained with a fixed head and can then be further fine-tuned with alternate heads for different tasks (Figure 2.12).

Heads allow a Transformer to be used for different tasks, such as classification, QA, translation, and more (see Table 2.2). Here we assume the input token sequence is *x1:N* from a vocabulary $\mathcal{V}$, and y represents different possible outputs, possibly from a class set $\mathcal{C}$ (Wolf et al., 2020). Each Transformer can be paired with one out of several ready-implemented heads with outputs amenable to common types of tasks. These heads are

[5]https://huggingface.co/

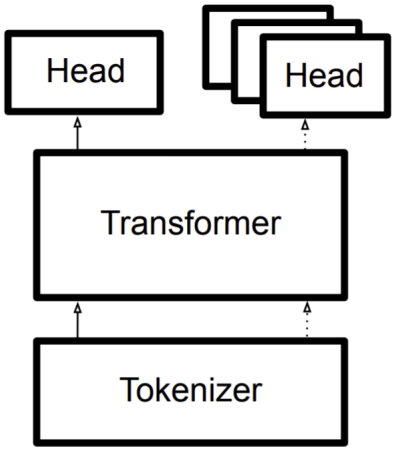


Figure 2.12: The *Transformers* library. Adapted from (Wolf et al., 2020).

		Heads		
Name	Input	Output	Tasks	Ex. Datasets
Language Modeling	$x_{1:n-1}$	$x_n \in \mathcal{V}$	Generation	WikiText-103
Sequence Classification	$x_{1:N}$	$y \in \mathcal{C}$	Classification, Sentiment Analysis	GLUE, SST, MNLI
Question Answering	$x_{1:M}$,$x_{M:N}$	y span $[1:N]$	QA, Reading Comprehension	SQuAD, Natural Questions
Token Classification	$x_{1:N}$	$y_{1:N} \in \mathcal{C}^N$	NER, Tagging	OntoNotes, WNUT
Multiple Choice	$x_{1:N}, \mathcal{X}$	$y \in \mathcal{X}$	Text Selection	SWAG, ARC
Masked LM	$x_{1:N \setminus n}$	$x_n \in \mathcal{V}$	Pretraining	Wikitext, C4
Conditional Generation	$x_{1:N}$	$y_{1:M} \in \mathcal{V}^M$	Translation, Summarization	WMT, IWSLT, CNN/DM, XSum

Table 2.2: *Transformers* heads. Adapted from (Wolf et al., 2020)

implemented as additional wrapper classes on top of the base class, adding a specific output layer, and optional loss function, on top of the Transformer's contextual embeddings (Wolf et al., 2020).

2.2.3 Transformers Applications

Since 2018, hundreds of papers and language models inspired by Transformers were published, the best known being BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019b), ALBERT (Lan et al., 2019), XLNet (Yang et al., 2019), DistilBERT (Sanh et al., 2019) and Electra (Clark et al., 2020). Many of these models are complex and include significant architectural improvements compared to the early Transformer and BERT models (Braşoveanu and Andonie, 2020).

Many of these Transformer models can be imported from the Transformers library with pre-trained and fine-tuned models for various NLP taks (Figure 2.13). The Transformers also make it easy for users to utilize the same core Transformer parameters with a variety of other heads for fine-tuning. The library also includes a collection of examples that show each head on real problems. These examples demonstrate how a pre-trained model can be adapted with a given head (Wolf et al., 2020).

[6]https://huggingface.co/models

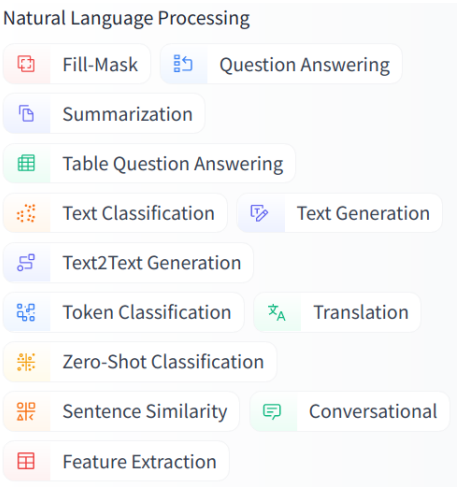


Figure 2.13: NLP tasks supported in the Transformers library. Adapted from Huggingface[6].

2.2.4 Transformer model for MRC task

For the Safety Desk problem we are going to use the MRC task for extracting the information from the given unstructured data sources, and we are going to utilize the Transformer model. The "Question Answering" models[7] present in the HuggingFace Hub are the exact models that we need for the MRC task. Figure 2.14 is an example of MRC using the RoBERTa trained with theStanford Question Answering Dataset (SQuAD) dataset. These models were mainly trained with the SQuAD obtaining the state-of-the-art F1-scores (about 89%) (Rajpurkar et al., 2018).

The SQuAD dataset has two versions:

- SQuAD V1.1: contains 100,000+ question-answer pairs on 500+ articles;
- SQuAD V2.0 combines the 100,000 questions in SQuAD V1.1 with over 50,000 unanswerable questions written adversarially by crowdworkers to look similar to answerable ones;

That means that SQuAD V2.0 not only answers questions when possible, but also determine when no answer is supported by the paragraph and abstains from answering (Rajpurkar et al., 2018).

2.3 Natural Language Generation

NLG is characterized as the subfield of artificial intelligence and computational linguistics that is concerned with the construction of computer systems than can produce texts in human languages from some underlying representation of information (Gatt and Krahmer, 2018).

NLG is present in many technologies and applications available to the general user, as translations, automatic spelling and text correction, weather and financial reports, etc. These applications are examples of what is usually referred to as Data-to-text (D2T)

[7]https://huggingface.co/models?pipeline_tag=question-answering
[8]https://huggingface.co/deepset/roberta-base-squad2

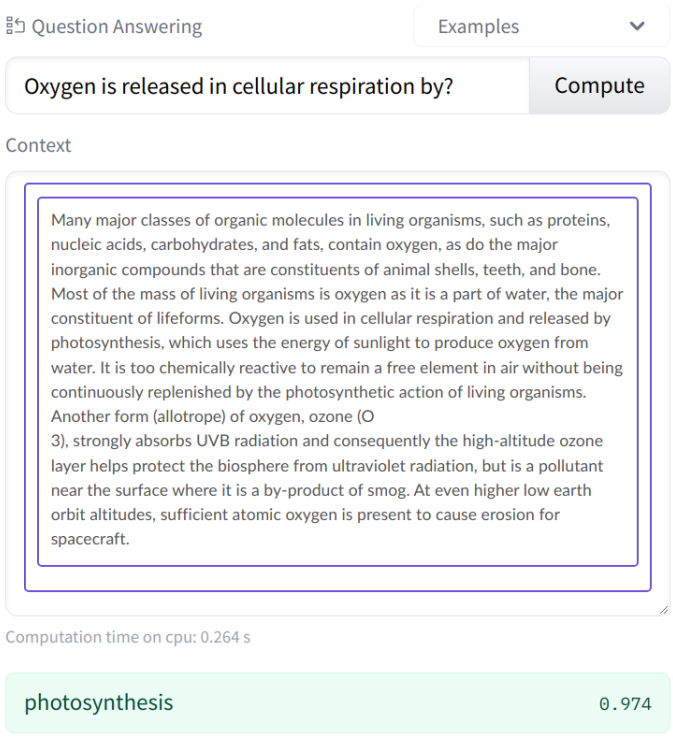


Figure 2.14: Example of MRC task using the RoBERTa trained with the SQuAD dataset. Adapted from Huggingface[8].

generation. These systems may differ considerably in the quality and variety of the texts they produce, their commercial viability and the sophistication of the underlying methods, but all are examples of D2T generation.

For the problem at hand, we are going to explore the background work and approaches in the NLG fields that are typically used for the Safety Desk problem, *i.e*, generating text from structured information.

2.3.1 NLG Subproblems

The NLG problem of converting input data into output text was addressed by splitting it up into a number of subproblems. The following six are frequently found in many NLG systems (Reiter and Dale, 1997):

- Content determination: deciding which information to include in the text under construction;
- Text structuring: Determining in which order information will be presented in the text;
- Sentence aggregation: Deciding which information to present in individual sentences;
- Lexicalisation: Finding the right words and phrases to express information;
- Referring expression generation: Selecting the words and phrases to identify domain objects;
- Linguistic realisation: Combining all words and phrases into well-formed sentences.

Content Determination

Typically, more information is contained in data than we want to convey through text, or the data is more detailed than we care to express in text. In order to solve this problem, as a first step in the generation process, the NLG system needs to decide which information should be included in the text under construction. The selection of what information to include depends on the target audience, *i.e.*, expert or casual, and on the overall typology of the text, *e.g.*, a manual guide, a biography, a clinical report, etc. Though content determination is present in most NLG systems the approaches are typically closely related to the domain of application (Gatt and Krahmer, 2018).

Text Structuring

Having determined what messages to convey, the NLG system needs to decide on their order of presentation to the reader. Once again, the order is dependent on the text typology and as the result of this stage is a discourse, text or document plan, which is a structured and ordered representation of messages (Gatt and Krahmer, 2018).

Sentence Aggregation

By combining multiple messages into a single sentence, the generated text becomes potentially more fluid and readable, although there are also situations where it has been argued that aggregation should be avoided (Gatt and Krahmer, 2018). This process of related messages being grouped together in sentences is known as sentence aggregation. In general, aggregation is difficult to define, and has been interpreted in various ways, ranging from redundancy elimination to linguistic structure combination.

Lexicalisation

Having all the content of the sentence finalised as a result of aggregation at the message level, the system can start converting it into natural language. The complexity of this lexicalisation process depends on the number of alternatives that the NLG system can have. One straightforward model for lexicalisation is to operate on preverbal messages, converting domain concepts directly into lexical items. This is feasible in well-defined domains, but for the other domains, lexicalisation is hard because it can involve selection between semantically similar, near-synonymous or taxonomically related words and it is not always straightforward to model lexicalisation in terms of a crisp concept-to-word mapping (Gatt and Krahmer, 2018).

Referring Expression Generation

Referring Expression Generation is the task of selecting words or phrases to identify domain entities. Typically, there are multiple entities which have the same referential category or type in a domain. Referring Expression Generation content determination algorithms can be thought of as performing a search through the known properties of the referent for the right combination that will distinguish it in context (Gatt and Krahmer, 2018).

Linguistic Realisation

This task involves ordering constituents of a sentence, as well as generating the right morphological forms. Often, realisers also need to insert function words and punctuation marks. An important complication at this stage is that the output needs to include various linguistic components that may not be present in the input. This generation task can be thought of in terms of projection between non-isomorphic structures so different approaches have been proposed (Gatt and Krahmer, 2018):

- Human-crafted templates: this approach is ideal when the application domain is small, variation is expected to be minimal and outputs can be specified using templates, making the implementation a relatively easy task. An advantage of templates is that they allow for full control over the quality of the output and avoid the generation of ungrammatical structures;
- Human-crafted grammar-based systems: this approach makes some or all of the choices on the basis of a grammar of the language under consideration. This grammar can be manually written (hand-coded) where hand-crafted rules with the right sensitivity to context and input are difficult to design;
- Statistical approaches: these approaches evolved during the last 20 years. An initial approach was a hand-crafted grammar that is used to generate alternative realisations from which a stochastic re-ranker selects the optimal candidate. With the emergence of ML and deep neural networks, some approaches were made using different architectures (Dong et al., 2021; Gatt and Krahmer, 2018; Celikyilmaz et al., 2020), as RNN Seq2Seq (Sutskever et al., 2014), Copy and Pointing Mechanisms (See et al., 2017), Generative Adversarial Network (GAN) (Goodfellow et al., 2014), and pre-trained models, *e.g.* the already mentioned BERT.

In the case of the Safety Desk problem, these six NLG subproblems can be defined as:

1. Content determination: physicochemical and toxicological properties of chemical compounds;
2. Text structuring: the report already has a structure currently in use (introduction, description of each information extracted and conclusion);
3. Sentence aggregation, 4. Lexicalisation, 5. Referring expression generation and 6. Linguistic realisation: the Talent Ingredient company has in place a predefined human-crafted template that has the majority of the sentence aggregation, lexicalisation, referring expression generation and linguistic realisation problems solved. Figure 2.15 is a excerpt of the human-crafted templates that is already been used in the creation of the toxicological profile;

> "The **$toxicologicalProperty** carcinogenic potential of the ingredient **$compound** was evaluated in a study. **$species** were fed in diet with **$doseLevel**; The substance was classified as **$PropertyClassification**."

Figure 2.15: Example of Safety Desk human-crafted template.

Since the existence of a well defined problem with a structure and templates made, in this project we will utilize templates in D2T generation in order to achieve objectives of the

Safety Desk problem. The end result will be similar to the example in Figure 2.16, where the structured data that we obtain from the IE task is transformed in a human-readable text.

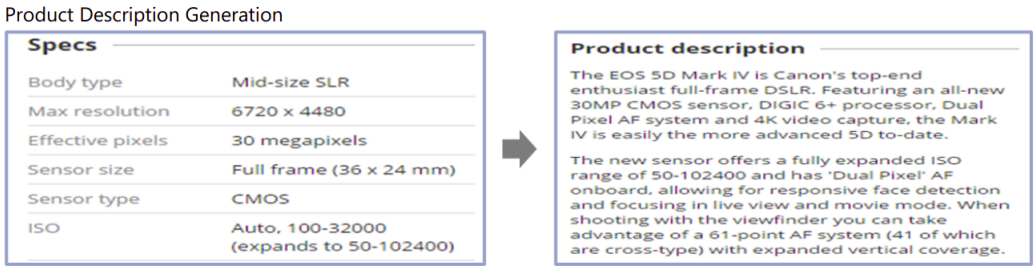


Figure 2.16: D2T example. Adapted from Microsoft[9].

2.4 Related Work

The Safety Desk project is a project where there is the necessity of extracting information from documents and formulating a toxicological profile of chemical substances.

So regarding related works with the general definition of the Safety Desk project we will discuss two types of works found, using QA models for IE and IE for the chemical domain.

2.4.1 IE using QA models

In terms of works formulating the IE problem in a QA problem, just in the last years, since the emergence of the Transformers models, is when some works started to appear.

In the papers (Nguyen et al., 2021a,b, 2020b; Minh-Tien Nguyen, 2020) the authors formulated their IE problem as a QA task. All the papers have the same problem identified: a limited data for domain-specific documents where information needs to be extracted from them. All the four papers are related to the system "AURORA", which extracts information from domain-specific Business Documents with Limited Data.

In the approach, see Figure 2.17, the papers proposed the usage of pre-trained model BERT, combined with CNN to learn the localization of the context of each document. The proposed model has three main components: the input vector representations of input tokens, BERT for learning hidden vectors for every token from the input tag and the document, a convolution layer for capturing the local context and a *softmax layer* to predict the value location.

[9]https://www.microsoft.com/en-us/research/project/data2text-automated-text-generation-from-structured-data/

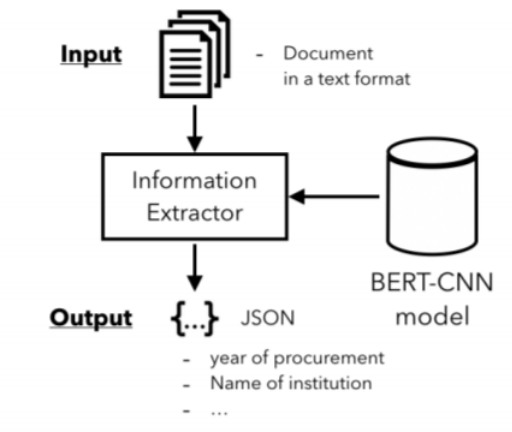


Figure 2.17: Overview of the system proposed by the author. Adapted from (Nguyen et al., 2021a)

In the model, BERT learns the context of the document given the tag and produces hidden vectors for every token and the CNN layer adjusts the vectors towards the domain.

IE is reformulated as a QA task where the value is pulled from the document by querying the tag, *i.e.*, a list of required information is defined and represented as tags, *e.g.*, "Name of institution" or "Deadline for bidding" (Nguyen et al., 2021a). Using the tags, they can be considered as a question, or part of one, the model, BERT learns the context of the document given the tag and produces hidden vectors for every token and the CNN layer adjusts the vectors towards the domain. Finally, a *softmax layer* is used to predict the location of the value (Nguyen et al., 2021a,b, 2020b; Minh-Tien Nguyen, 2020).

The authors used public texts of competitive bids for development projects in Japan in their work, specifically, 78 documents used for training and 22 used for testing. They used the F-score to evaluate the performance of the model as well as the baselines. The extracted outputs were matched to ground-truth data to compute precision, recall, and F-score.

Regarding the results obtained by the authors we can affirm that they obtained various results, some very low with an F-Score lower than 0.30 but other methods obtained very good results, including the method just using the BERT model, a method that we can also use.

In (Arici et al., 2022), the authors used a QA approach to quantity extraction trying to solve a Price Per Unit (PPU) problem. In the approach, they first predict the unit of measure (UoM) type, *e.g.*, volume, weight or count, in order to formulate the desired questions, *e.g.*, "What is the total volume?", and then use this question to find all the relevant answers.

The authors opted to use a QA approach instead of NER because NER solutions do not enable the model to couple start and end indices explicitly, and check for their compatibility during training. Furthermore, they are prone to small variations in the tokens, for *e.g.*, "fluid ounce" or "fl oz", which need to be explicitly tagged. Other span characteristics such as shorter answers are more likely to occur, can not be learned by the model. Also, UoM type information, which is important for quantity extraction, cannot be efficiently fed to NER model other than learning different token representations for each UoM type. To overcome these limitations, the authors introduced a span-image architecture that works at a character level and uses a QA approach to quantity extraction which conditions the

extractor model with UoM type information.

The model architecture consists of two subnetworks for the two subtasks: a classifier to predict UoM type, *i.e.*, the question, and an Questions Extractor to extract the relevant quantities. The UoM classifier consists of the following stages: (1) character embedding layer that maps each character to a k dimensional vector, (2) convolutional layers that consist of multiple layers with filter sizes 3 and 5, (3) an attention module that computes an attention vector from all input attributes, (4) categorical embeddings vectors are created by embedding categorical indices into a high dimensional space and (5) product-description vector and category-embedding vectors are concatenated and passed to classification layers to produce logits for UoM type (Arici et al., 2022).

The Quantity extraction model has the following stages: (1) a character embedding layer, (2) 1D convolutional layers are applied to obtain an encoded sequence y without any strided pooling. Resultant sequence is batch normalized and dropout is used during training. (3) Each vector in y is concatenated with UoM softmax outputs, and fed into two different 1D convolutional layers to compute two vector sequences s and e of length n, with a shrunken depth d allowing specialization for start and end index prediction. (4) s is tiled horizontally and e is tiled vertically to produce two tensors of size $\mathbf{n} \times d$. These two tensors are multiplied element-wise to create a span-image of width and height equal to and depth of d. 2D convolutional filters are applied on the span-image to produce an image of size $\mathbf{n} \times d$ and depth 2. (5) Softmax normalization is applied on the depth dimension as opposed to the sequence dimension. Post-processing is done on the extracted quantities above a certain threshold to obtain the final quantity (Arici et al., 2022).

In (Li et al., 2020), the authors formulates event extraction as multi-turn QA approach, MQAEE. Typically, event extraction can be divided into two subtasks: trigger extraction (trigger identification and classification) and argument extraction (argument identification and classification), and approaches to event extraction can thus be roughly categorized into two groups: (1) pipelined approaches that perform trigger extraction and argument extraction in separate stages and (2) joint approaches that perform all subtasks simultaneously in a joint learning fashion.

Most of these approaches, whether pipelined or joint, formulate event extraction as classification tasks, by classifying event triggers into pre-defined event types, and further event arguments into pre-defined argument roles. By treating event types and argument roles directly as golden labels, such classification-based approaches suffer from two limitations. First of all, they cannot explicitly model the semantics of these golden labels and also fail to capture the rich interactions among them, which could be extremely useful for event extraction. The second limitation lies in the generalization ability. By taking event types and argument roles as golden labels, classification-based approaches are not able to be generalized to new event types or argument roles without additional annotations.

To address the mentioned limitations, the authors propose a new paradigm that formulates event extraction as multi-turn QA, MQAEE. The approach splits event extraction into three sub-tasks: trigger identification, trigger classification, and argument extraction. These subtasks are modeled by a series of MRC based QA templates. Trigger identification is cast into an extractive MRC problem, identifying trigger words from given sentences. Trigger classification is formalized as a YES/NO QA problem, judging whether or not a candidate trigger belongs to a specific event type and argument extraction is also solved via extractive MRC, with questions constructed iteratively by a target event type and the corresponding argument roles.

Table 2.3 provides an example and overview of the MQAEE framework where the

Passage: Saddam's family left that city three days ago.
Trigger identification Q_1: Which word is the trigger word? A_1: left
Trigger classification Q_2: The trigger word is left $\langle pos \rangle$ 2 $\langle /pos \rangle$, movement: transport? A_2: YES
Argument extraction Q_3: left $\langle pos \rangle$ 2 $\langle /pos \rangle$. Movement:transport, time-within? A_3: three days ago
Q_4: left $\langle pos \rangle$ 2 $\langle /pos \rangle$. Movement:transport, artifact? A_4: Saddam's family
Q_5: left $\langle pos \rangle$ 2 $\langle /pos \rangle$. Movement:transport, destination? A_5: NULL
...

Table 2.3: Example and overview of MQAEE framework. Adapted from (Li et al., 2020)

sentence is taken as the passage and each turn contains a question (Qi) and an answer (Ai) and $NULL$ means there is no answer to the question.

The authors mention that the advantages of MQAEE are that the multi-turn QA infrastructure provides an effective way to model rich interactions among triggers, event types, and arguments, which has shown to be beneficial to event extraction, and by converting event types and argument roles as questions rather than golden labels, MQAEE can be easily generalized to new types and roles.

Others (A., 2022) suggest a practical approach on how to use QA models for automating IE. They describe step by step how to use QA models in order to extract information from documents. The publication is present in the Deepset[10] site, a company present in NLP bussiness and that provides some open source libraries and models. The Deepset company published the version of the RoBERTa model[11] used in the implementation of our IE process.

2.4.2 IE for the chemical domain

Extracting chemical information from documents is a challenging task, but an essential one for dealing with the vast quantity of data that is available, requiring increasingly sophisticated approaches for less structured documents, such as PDFs.

Some existing works that we found relevant to mention regarding the extraction of

[10] https://www.deepset.ai/
[11] https://huggingface.co/deepset/roberta-base-squad2

Run	Exact-Match			Relaxed-Match		
	P	R	F	P	R	F
Melaxtech-run1	**0.9201**	**0.9147**	**0.9174**	**0.9319**	**0.9261**	**0.9290**
NextMove/Minesoft-run1	*0.8492*	*0.7609*	*0.8026*	*0.8663*	*0.7777*	*0.8196*
NextMove/Minesoft-run2	0.8486	0.7602	0.8020	0.8653	0.7771	0.8188
NextMove/Minesoft-run3	0.8061	0.7207	0.7610	0.8228	0.7371	0.7776
OntoChem-run1	0.7971	0.3777	0.5126	0.8407	0.3984	0.5406
OntoChem-run2	0.7971	0.3777	0.5126	0.8407	0.3984	0.5406
OntoChem-run3	0.7971	0.3777	0.5126	0.8407	0.3984	0.5406
Baseline	0.2104	0.7329	0.3270	0.2135	0.7445	0.3319
Melaxtech-run2	0.2394	0.2647	0.2514	0.2429	0.2687	0.2552
Melaxtech-run3	0.2383	0.2642	0.2506	0.2421	0.2684	0.2545

Table 2.4: Overall performance of the ChEMU system. Adapted from (He et al., 2021)

information in the chemical domain are ChEMU (Nguyen et al., 2020a; He et al., 2021), ChemDataExtractor (Swain and Cole, 2016), ChemEx (Tharatipyakul et al., 2012) and papers and surveys (Abdelmagid et al., 2014; Zimmermann et al., 2005) detail IE technologies and challenges of extracting information from chemical compound literature.

The papers regarding IE for chemical compound literature mention the NLP tasks exposed in this Chapter, namely, *tokenization*, POS and NER as well syntactic analysis. Based on that same NLP tasks, works were developed, ChEMU, ChemDataExtractor and ChemEx.

ChEMU uses NER to identify chemical compounds as well as their types of context and Event Extraction to extract chemical reactions from Patents. The goal of ChEMU is to automatically identify compounds and extract chemical reaction events to construct cheminformatics databases, capturing key information about chemicals, *e.g.*, the temperature at which the reaction was carried out, the reaction time of the reaction, and how they are produced, from the patent resources, *e.g.*, the starting material that is consumed in the course of a chemical reaction, the reagent catalyst that is a compound added to a system to cause or help with a chemical reaction, etc (Nguyen et al., 2020a).

The system was evaluated, each task alone, *i.e.*, task one is NER and task two is Event Extraction, and overall results were obtained, Figure 2.4.

ChemEx is a system for extracting information from chemical data curation that consists on four main modules: Document Preprocessor, 2D Chemical Structure Image Recognition, Text Annotator, and Information Viewer. First, the Document Preprocessor transforms and segments each input literature into textual and visual data. The 2D Chemical Structure Image Recognition module then translates the visual data (images) into machine readable string whereas the Text Annotator module tags words in a subject domain using a component called Analysis Engine (AE) from Unstructured Information Management Applications (UIMA) to analyse document in four steps: Tokenizer, Tagger, Phase Parser and Identification, and Coordination Resolution. A user can visualize extracted information using the Information Viewer (Tharatipyakul et al., 2012).

ChemEx is able to extract compound, organism, and essay entities from text content

	Exact Matches	Partial Matches	False Positive	False Negative	Precision	Recall
Compounds	203	15	41	105	83.20%	62.85%
Organisms	91	21	3	5	96.81%	77.78%
Assays	80	0	0	15	100.00%	84.21%

Table 2.5: Overall performance of Chemex. Adapted from (Tharatipyakul et al., 2012)

	precision	recall	F-score
chemical identifier records	94.1%	92.7%	93.4%
spectrum records	88.3%	85.4%	86.8%
chemical property records	93.5%	89.6%	91.5%

Table 2.6: Overall performance of ChemDataExtractor. Adapted from (Swain and Cole, 2016)

automatically. It also finds the 2D chemical structure of each compound from images embedded in full text, and converts the 2D chemical structure images to machine readable format (Tharatipyakul et al., 2012).

ChemEx was evaluated, see Figure 2.5, using 89 publications with terms "fungus Thailand" from ACS Publications where only 74 publications reported compounds with 2D chemical structures. Compounds, organisms, and assays were extracted from text content and compared with manually listed entities.

Finally, ChemDataExtractor is a tool created for extracting chemical information from chemistry literature, *e.i.*, chemical compounds and their relations and properties, specifically text from HTML, XML and PDF sources. For HTML and XML ChemDataExtractor uses semantic markup of headings, paragraphs, captions and tables and for PDF sources it uses a layout analysis tool built on top of the PDFMiner framework[12], to use the positions of images and text characters to group text into headings, paragraphs, and captions. The text extracted then is processed in a NLP pipeline where the text is first split into sentences and then into individual tokens. The POS tagger and entity recognizer outputs are combined to assign a single tag to each token, which is then parsed using a rule-based grammar to produce a tree structure. This structure is interpreted to extract individual chemical records for the respective sentence, which are then combined with records from throughout the document to resolve data interdependencies and produce unified records for depositing in a database (Swain and Cole, 2016).

ChemDataExtractor was evaluated in 50 open-access chemistry articles from academic journals, ACS, RSC, and Springer, where chemical entities, spectra, and properties, namely melting points, oxidation and reduction potentials, were extracted from the abstract, main text, tables, and figure captions. The overall results obtained, Figure 2.6, show that the ChemDataExtractor demonstrate good performance in the extraction of chemical entities and their associated experimental properties and spectroscopic data (Swain and Cole, 2016).

[12]https://pypi.org/project/pdfminer/

2.5 Conclusion

In this Chapter we analysed all the theoretical aspects necessary to consolidate the goals mentioned in the Chapter 1.2. We explored NLP tasks, *e.g.*, IE and QA, as well as NLG tasks. Related works to IE using QA models demonstrated limitations of existing implementations of some NLP tasks for IE, namely, NER and event extraction, and why approaching the IE problem as a QA problem using MRC systems is a state-of-the-art implementation.

For the chemical domain, related works were mentioned in order to gather information about existing "competitors" and the results that those were able to obtain..

With the knowledge acquired we have sufficient theoretical notions to implement a practical solution to the Safety Desk problem.

Chapter 3

Problem Analysis and Approach

In this project, the goal is to extract physicochemical and toxicological properties of chemical compounds from unstructured sources, *i.e.*, PDFs from renown data sources. The human expert, *i.e.*, the security advisor, uses a set of data sources, see Table 3.1, where some are mandatory and others are optional or complementary. This criteria is based on the relevance that the data source has, *i.e.*, quantity of data (documents and information contained in the documents), adequacy of the information present in the documents, and what type of document format is available. Which data source to use depends on the type of chemical compound, *e.g.* natural compound, industrial compound, etc.

The security advisor, when producing a report of physicochemical and toxicological properties of chemical compounds, proceeds to search in the data sources for information and documents regarding the chemical compound that they want to report on. Then, considering multiple data sources, the security advisor searches for the information regarding each property that is necessary. This search is done manually, *i.e*, the security advisor traverses all the data sources and reads them to find results. Obviously, the process of searching data and having a large amount of data to analyse is very slow, and it takes hours to complete a report on a chemical compound.

With the help of the Talent Ingredient team, that showed us their workflow and resources, see Figure 3.1, we were able to identify all the physicochemical and toxicological properties that should be extracted from the data sources. Table 3.2 shows all the properties that are usually present in the reports.

The Cosmedesk platform is an application developed for the Talent Ingredient company. It contains a database with all the chemical compounds reported by the security advisor and provides a simple interface with multiple fields for the insertion of the information regarding the physicochemical and toxicological properties of the chemical compound.

The last part of the process is to write the toxicological profile of the chemical compound. This is an extended human-readable text that contains all the information about

[1]https://ec.europa.eu/health/scientific-committees/
[2]https://www.cir-safety.org/ingredients
[3]https://www.industrialchemicals.gov.au/chemical-information
[4]https://hpvchemicals.oecd.org/UI/Default.aspx
[5]https://echa.europa.eu/information-on-chemicals
[6]https://www.atsdr.cdc.gov/toxprofiledocs/index.html
[7]https://www.rifm.org/#gsc.tab=0
[8]https://ntp.niehs.nih.gov/
[9]https://www.efsa.europa.eu/en

Source	Description	Relevance	Document Format
SCCS (Scientific Committee on Consumer Safety)[1]	The SCCS provides Opinions on health and safety risks (chemical, biological, mechanical and other physical risks) of non-food consumer products (e.g. cosmetic products and their ingredients, toys, textiles, clothing, personal care and household products) and services (e.g. tattooing, artificial sun tanning).	High	PDF
CIR (Cosmetic Ingredient Review)[2]	The CIR was established by the industry trade association (then the Cosmetic, Toiletry, and Fragrance Association, now the Personal Care Products Council), with the support of the U.S. Food and Drug Administration and the Consumer Federation of America. The CIR studies individual chemical compounds as they are used in cosmetic products.	High	PDF
AICIS (Australian Industrial Chemicals Introductions Scheme)[3]	The AICIS regulates the importation and manufacture (introduction) of industrial chemicals in Australia. The AICIS conduct scientific risk assessments on the introduction and intended use of industrial chemicals in Australia.	High	PDF
OECD (Organisation for Economic Co-operation and Development)[4]	The OECD Existing Chemicals Database track the status of chemical and chemical categories, obtain published OECD assessments, find a SIDS contact point, or view a variety of useful reports and lists on chemicals within the OECD Cooperative Chemicals Assessment Programme.	High	PDF
ECHA (European Chemicals Agency)[5]	The ECHA is the driving force among regulatory authorities in implementing the EU's chemicals legislation.	High	Website
ATSDR (Agency for Toxic Substances and Disease Registry)[6]	The ATSDR is a federal public health agency of the U.S. Department of Health and Human Services. ATSDR protects communities from harmful health effects related to exposure to natural and man-made hazardous substances.	Low	PDF
RIFM (Research Institute for Fragrance Materials)[7]	The RIFM is a nonprofit member supported organization to ensure the safe use of fragrance ingredients by consumers. RIFM gathers and analyzes scientific data, engages in testing and evaluation, distributes information, and maintains open communication with all related official international agencies. RIFM maintains the most comprehensive online database of fragrance materials in the world.	Low	PDF
US National Toxicology Program (NTP)[8]	The NTP provides the scientific basis for programs, activities, and policies that promote health or lead to the prevention of disease. Founded in 1978, NTP plays a critical role in generating, interpreting, and sharing toxicological information about potentially hazardous substances in our environment. NTP strives to remain at the cutting edge of scientific research and the development and application of new technologies for modern toxicology and molecular biology.	Low	Website
EFSA (European Food Safety Authority)[9]	EFSA is a European agency funded by the European Union that operates independently of the European legislative and executive institutions (Commission, Council, Parliament) and EU Member States. As the risk assessor, EFSA produces scientific opinions and advice that form the basis for European policies and legislation on food safety.	Low	xlsx and PDF

Table 3.1: Data sources used by Talent Ingredient.

Property type	Property	Specific information related to the property
Physicochemical	Molecular Formula	
	Molecular Weight	
	Density	
	Log Pow	
	Degree of Ionization	
	Topological Surface Area	
	Melting Point	
Toxicological	Dermal absortion / Bioavailability	
	NOAEL	
	Acute Toxicity	Species used in study
		OECD Guideline
		Exposure route
	Dermal / Eye Irritation	Species used in study
		Classification: non-irritating; moderately irritating; mildly irritating; irritating
		OECD Guideline
	Sensitization	Classification: non-sensitizing; sensitizing
		OECD Guideline
		Concentration used in study
	Mutagenicity	Classification: non-mutagenic; mutagenic
		OECD Guideline
	Carcinogenicity	Classification: non-carcinogenic; carcinogenic
		Species used in study
		OECD Guideline
	Photo-induced Toxicity	Classification: non-phototoxic; phototoxic
		OECD Guideline
	Reproductive Toxicity	OECD Guideline
		Species used in study
		Classification: non-reprotoxic; reprotoxic

Table 3.2: List of Physicochemical and Toxicological properties.

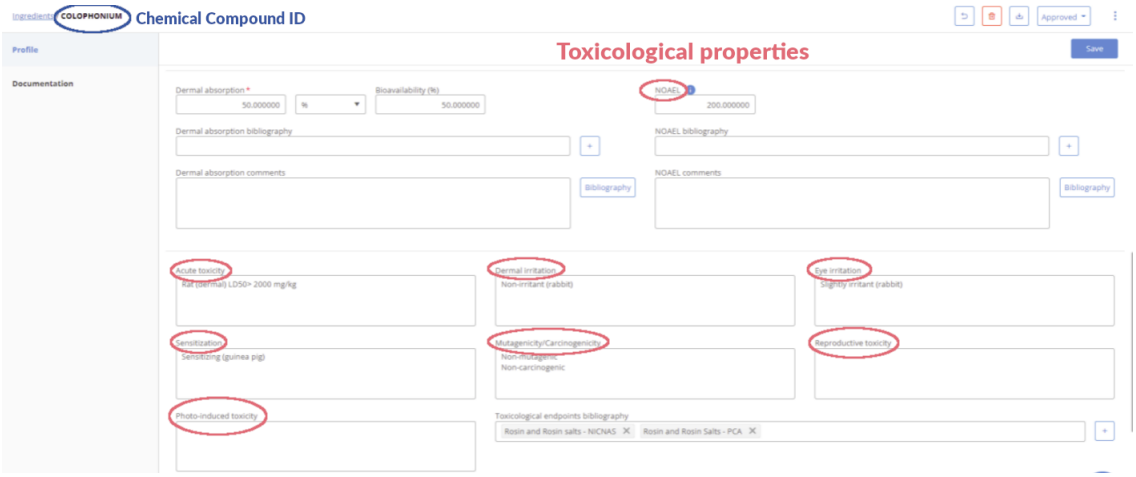


Figure 3.1: Talent Ingredient reporting platform, Cosmedesk, with some properties of the chemical compound identified.

the chemical compound written for non-expert to understand the overall report.

In this chapter we will mention the challenges present in the Safety Desk project, the approach that we will take and dive into probable risks that can occur and the scope of the project, *i.e.*, the direct influence and responsibilities of the Safety Desk project in the Cosmedesk platform.

3.1 Challenges

Given the state of the art presented in Chapter 2, *i.e.*, Information Extraction (IE), Machine Reading Comprehension (MRC), and Data-to-text (D2T) generation, and taking into account the objectives of this project, we can identify the main challenges of this work:

- Multiple information needs to be extracted, so each information needs a specific configuration for the IE algorithm;
- The context regarding each property can be in different locations depending on the document, which consequently means that multiple document structures need to be taken into account;
- Different documents can contain different results regarding each information, so there is a problem of divergent information;
- The documents have multiple context sizes, *i.e.* various number of pages and multiple paragraphs, and the *Transformers*, used in the IE algorithm, just support a limited input size;

3.2 Proposed Approach

Taking those challenges into account, possible solutions were idealized. Therefore, what is proposed within this project, and based on what was presented in the Chapter 2, is a pipeline adapted to the needs of the problem that we have in hands. A *pipeline* in software and computational terms is a collection of computational processes or programs

that are connected in series, and the output of one element is the input of the next one. By establishing the previous theoretical foundations as the main points to solve this problem, we can consider that all of the needed functions and operations over the system are going to be used in different stages of the final system. We can assume that this system is a pipeline of multiple stages that will implement different technologies for different and specific goals.

To achieve the project goals, the final system must be able to, among the different data sources, extract each existing information regarding each property of the compound from the PDF. With all the information extracted, it is necessary to do a verification step because, depending on the amount of data sources and information regarding a property of the compound, it is possible to obtain multiple information for a compound property. Having the information verified is also necessary to transform the data obtained in a human-readable text, *i.e.*, to generate a summary that is correctly written and thus ready to be read by a human. As for the end result, it is necessary to provide a method to share all the compound properties extracted and the text generated.

The approach idealized, taking into account the previous problems identified and the scheme of the project, has five phases: Preprocessing, IE, Data Verification, D2T and Rest API. In Figure 3.2 it is possible to check the first version of the proposed pipeline model for this project.

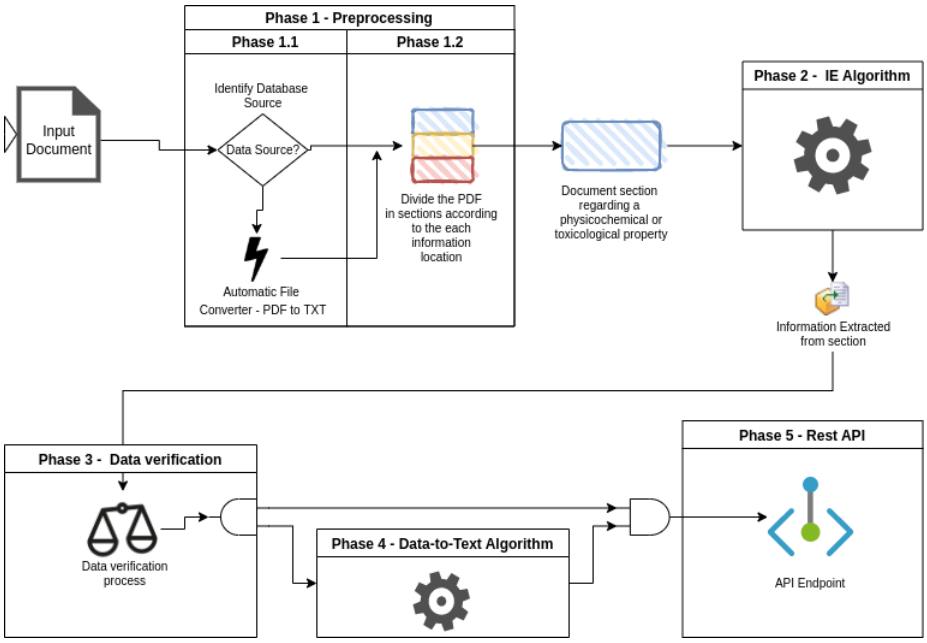


Figure 3.2: Proposed Project Architecture Pipeline.

Phase 1 (figure 3.3) is the initial and a defying phase of the project. This phase is divided in two sub-phases:

- Phase 1.1: Identify from which database is the Input Document and convert the PDF document to textual format;
- Phase 1.2: Divide the text in sections. The division process differs from database to database because each one follows their specific template. This step is essential in order to limit the area where to search for in the next phase.

After the initial phase, we have identified from which database is the document and have sets of text to extract information from. In Phase 1.2, the division into sections is

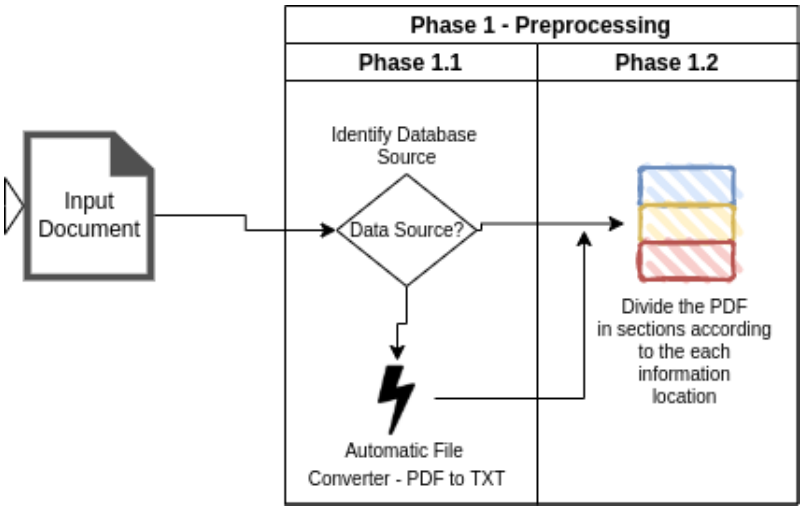


Figure 3.3: Phase 1 Architecture.

based on the localization of the properties in the document. As an example, if we have the properties A and B and the property A has information in section 1 and 3 and property B has information in section 2 it is not necessary to search for information regarding the property B in all the sections but search just in section 2. Figure 3.4 is a graphical example of the Phase 1.2 algorithm.

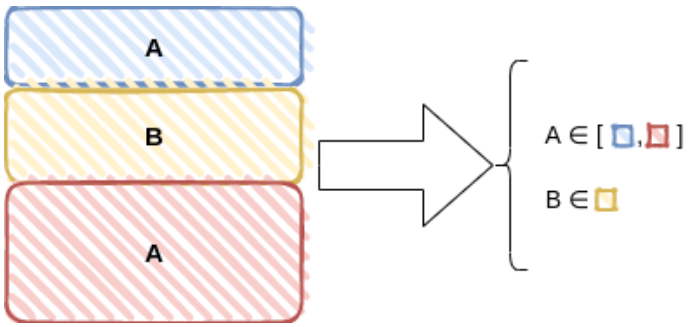


Figure 3.4: Graphical Example Phase 1.2 Algorithm.

In practical terms, these section divisions need to follow schemes. The example in Figure 3.4 is an ideal case. However, there is a high possibility that the properties can be mentioned all over the document. In order to do a correct identification of what and where is each section regarding each property, we propose taking advantage of the documents structure/ Table of Contents (TOC) whenever possible.

In this project, the documents with information regarding chemical compounds follow templates. The templates differ according to each database, that is why the Phase 1.1 has a relevant importance in identifying the pair database-template to use in the division of the document in sections (Phase 1.2). Each template needs to be studied and analyzed in order to implement an algorithm suitable to the retrieval of sections. At the end of Phase 1.2 we expect to achieve the correct set of pairs section-property. Figure 3.5 represents the objective of the Phase 1.2.

The next step of the pipeline, Phase 2 (Figure 3.6), converges all knowledge analyzed in the previous Chapter 2, about IE, MRC and *Transformers*, to build an algorithm to extract all the relevant information from the sections obtained from the Phase 1.2. The approach that we propose for this phase is based on the MRC system described in Chapter

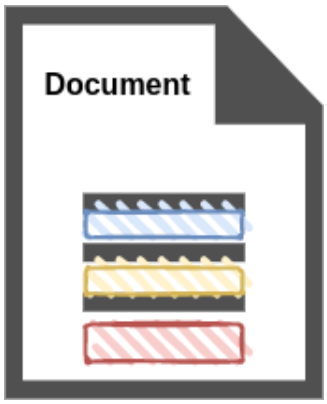


Figure 3.5: Objective of the Phase 1.2: Document divided in sections.

2. This means that we must provide the following to the transformer-based model:

   - The context (sections returned from Phase 1.2);
   - The right set of questions to obtain the right answers for each context given;

Each question depends on the information that we want to obtain for each property and also the document used. That means that this Phase needs to take into account the set Document-Property.

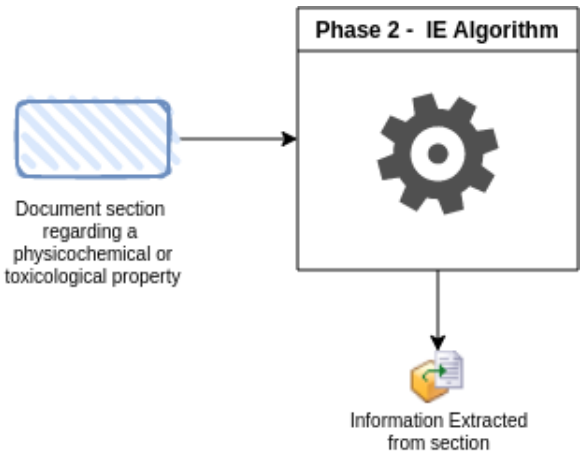


Figure 3.6: Phase 2 Architecture.

Eventually, after extracting all the existing information from the document, we need to validate the information extracted (Phase 3 represented in Figure 3.7). To do so, we propose a cross-validation method where we verify between the information extracted for each property if there are multiple returns of the same information extracted by the IE phase. With this phase we try to filter the information extracted from the IE phase, *i.e.*, remove the incorrect information and just return the correct information extracted.

After having all the information regarding a compound, there are two more phases in the pipeline, however Phase 5 is not fully dependent on Phase 4. The later, see Figure 3.9, is a specific phase to generate a summary of the information regarding the compound. Phase 4 consists of an algorithm to, using templates, generate human-readable text structured as follows:

1. Introduction;

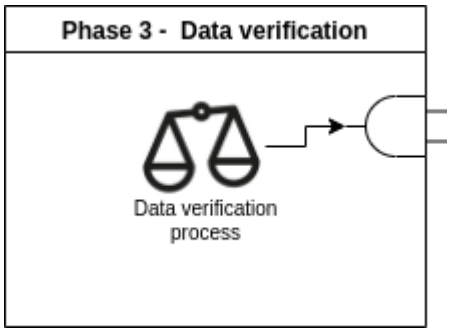


Figure 3.7: Phase 3 Architecture.

2. Specific description of each compound property;

3. Discussion and Conclusion;

Each topic of the structured information summary has a predefined human-crafted template, that in conjunction with being a small domain with little variation, means that we can use templates for the D2T task, as mentioned in Chapter 2. Figure 3.8 is an example of templates already predefined that we can use in D2T generation.

> "The **$toxicologicalProperty** carcinogenic potential of the ingredient **$compound** was evaluated in a study. **$species** were fed in diet with **$dose-Level**; The substance was classified as **$PropertyClassification**."

Figure 3.8: Example of human-crafted template.

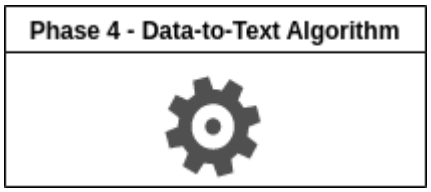


Figure 3.9: Phase 4 Architecture.

Finally, Phase 5 (Figure 3.10) consists of implementing an API endpoint to ease the communication between this project with existing technologies, *e.g.* website or database, that the company already uses. The API should provide methods to access all the information resulting from Phase 3 and the generated text created in Phase 4.

3.3 Risk Analysis

In the proposed pipeline, the developed IE process is constituted by multiple phases with the main objective of extracting information about toxicological properties of chemical compounds. In this process we can identify in advance multiple risks that are possible to occur throughout the multiple phases. As the pipeline is connected in series, and the output of one element is the input of the next one, it is clear that if an incorrect output appears, the next phase will also return an incorrect output, like the quote "garbage in, garbage out".

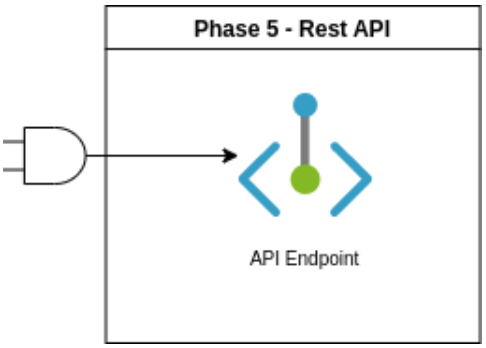


Figure 3.10: Phase 5 Architecture.

Going phase by phase, in the first phase, *i.e.*, Preprocessing, there is mainly one risk: the Preprocessing phase fails leading to a total collapse of all pipeline. This risk can occur in different degrees, where the Preprocessing phase fails totally or just the detection of a specific section is not correct, where the risk does not lead to a total collapse but rather a fail in the extraction of information regarding a specific toxicological property.

In the IE phase there are three risks identified:

- The Phase does not obtain information extracted while there was information to be extracted;
- The information extracted is incorrect;
- The IE Phase does not extract all the information that needs to be extracted;

All the three risks have the same impact in the next phase, the Data Verification Phase, and the whole pipeline, "garbage in, garbage out". But the risks have different levels of impact according to the level of test, results proven and human control, *i.e.*, in an initial phase, where the pipeline is implemented, high level of human control and verification is required in order to verify the work developed as it is to improve the IE process developed. So initially, the three risks have similar consequences, where the security advisor needs to consult and verify the information extracted.

In a long term, arriving to a point where the pipeline is accepted, the three risks have different impact levels:

- If there is information to be extracted but is not then the report is incomplete;
- The report contains incorrect information;
- The report is incomplete;

As the Data Verification phase consists of verifying the information extracted from the IE phase, the risk that can occur is the incorrect verification leading to two possibilities, removing a correct information extracted or not removing an incorrect information extracted. As the phase is a process created to just accept the correct information extracted, there is a risk of removing a correct information extracted and decreasing the recall of the IE process.

The D2T Phase is a very heavy conditional algorithm due to the usage of templates, therefore, if no error is made in the implementation (code) then no risk is evident. Sure

that the toxicological profile generated can be with incorrect information but that depends on previous phases, not the D2T phase. The same thing happens in the Phase 5, where the API is developed and deployed, if no error in implementation is made, no setback in running the service, then no risks are apparent.

3.4 Scope

The Safety Desk project has the objective of extracting information from a given document regarding properties of chemical compounds. That definition creates well defined boundaries about what the Safety Desk project needs to develop, defining the responsibilities that the Safety Desk project has.

In terms of databases, the safety desk project will just work with the document given, providing a way to communicate the results obtained with other services, *i.e.*, REST API. So in the Safety Desk project we are not required to save and store data extracted from the documents.

Also, an important point to reference is that the Safety Desk project does not intervene with the Cosmedesk database, so we do not have access to the data regarding chemicals compounds already stored, something necessary mainly for the creation of a complete toxicological profile in the D2T Phase.

Chapter 4

Preprocessing

The pipeline proposed in Chapter 3.2 starts off with the Preprocessing phase. The Preprocessing Phase is a very important phase of the project, where any error has a direct impact in the rest of the pipeline.

This Phase consists of dividing the input document into sections, where each section contains the information regarding a specific property of the chemical substance. That way, we minimize the context given to the Question Answering (QA) models, eliminating noise, *i.e.*, parts of the document not relevant for each property. So any incorrect Section defined means that the information about a property of the chemical substance will be also incorrect.

In this Chapter we delve into what documents and what methods for the division of the sections we used and the implementation developed, the challenges encountered and the problems that we still need to fix.

4.1 Exploratory Work

The first work developed in this Phase was studying the data sources (Table 3.1) and the documents provided from each one in order to create and develop the algorithms and methods for the Preprocessing Phase.

We analysed documents from the following data sources, Scientific Committee on Consumer Safety (SCCS)[1], Cosmetic Ingredient Review (CIR)[2], Australian Industrial Chemicals Introduction Scheme (AICIS)[3], Organization for Economic Cooperation and Development (OECD)[4], Agency for Toxic Substances and Disease Registry (ATSDR)[5] and Research Institute for Fragrance Materials (RIFM)[6], in that order since the security advisor has a relevance defined, as detailed in Table 3.1.

According to the security advisor, the SCCS source is the most important source because the SCCS is a Committee that provides Opinions on health and safety risks (chemical, biological, mechanical and other physical risks) of non-food consumer products (*e.g.* cos-

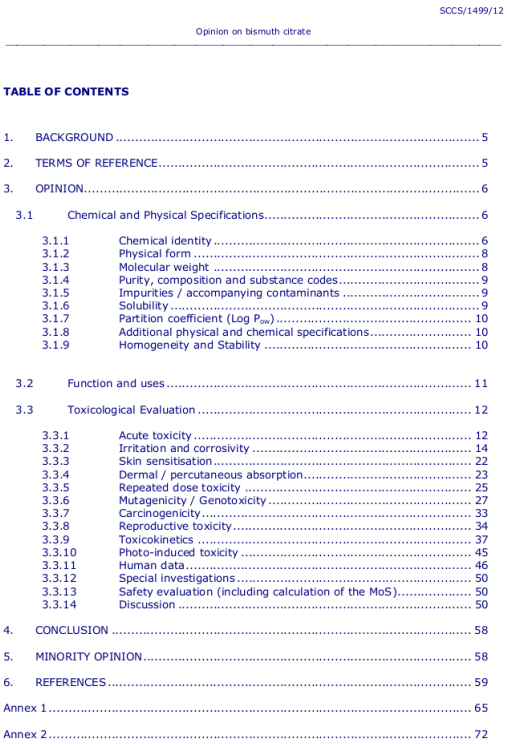
[1]https://ec.europa.eu/health/scientific-committees/
[2]https://www.cir-safety.org/ingredients
[3]https://www.industrialchemicals.gov.au/chemical-information
[4]https://hpvchemicals.oecd.org/UI/Default.aspx
[5]https://www.atsdr.cdc.gov/toxprofiledocs/index.html
[6]https://www.rifm.org/#gsc.tab=0

metic products and their ingredients, toys, textiles, clothing, personal care and household products) and services (*e.g.* tattooing, artificial sun tanning) (Commission, 2022) in the European Union. The security advisor uses all the information from the SCCS Opinions as much as possible. But when there is no information in the SCCS data source, the security advisor needs to rely on other sources, so after verifying that there is no SCCS Opinions the security advisor checks the existence of CIR reports, that is the equivalent to the SCCS but in the USA, and AICIS Human Health Assessments. Just after analysing that three main sources the security advisor consults the OECD data source, or in case of frangrances, the RIFM data source. The ATSDR is an unusual data source to use, it is just used in case information regarding Health effects on Cancer is needed.

From the multiple documents analysed from each data source, we could identify documents that we could develop the Preprocessing Phase more easily than others. We divided the documents in three main groups considering the following characteristics:

1. Documents with Table of Contents (TOC) and defined structure;
2. Documents without TOC but with a defined structure;
3. Documents without TOC and defined structure;

In our work, we defined a document with TOC and a defined structure as a document that contains a TOC, *i.e.*, the list of chapters identifiers at the beginning of a document with a defined structure, and the list of chapters present in the TOC is present in the same order in the document. From the data sources analysed, SCCS Opinions (Figure 4.1) and ATSDR Toxicological Profiles (Figure 4.2) documents fitted in this group. For these documents, since we had the TOC, our approach was to, using regular expressions, extract the TOC, chapter identifiers and hierarchy.

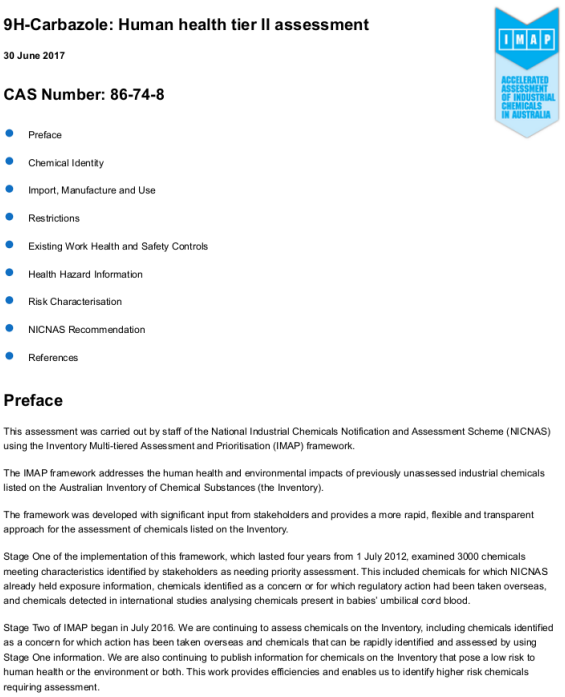


SCCS/1499/12

Opinion on bismuth citrate

TABLE OF CONTENTS

Figure 4.1: SCCS Opinion document example with a defined TOC



9H-Carbazole: Human health tier II assessment

30 June 2017

CAS Number: 86-74-8

- Preface
- Chemical Identity
- Import, Manufacture and Use
- Restrictions
- Existing Work Health and Safety Controls
- Health Hazard Information
- Risk Characterisation
- NICNAS Recommendation
- References

Preface

This assessment was carried out by staff of the National Industrial Chemicals Notification and Assessment Scheme (NICNAS) using the Inventory Multi-tiered Assessment and Prioritisation (IMAP) framework.

The IMAP framework addresses the human health and environmental impacts of previously unassessed industrial chemicals listed on the Australian Inventory of Chemical Substances (the Inventory).

The framework was developed with significant input from stakeholders and provides a more rapid, flexible and transparent approach for the assessment of chemicals listed on the Inventory.

Stage One of the implementation of this framework, which lasted four years from 1 July 2012, examined 3000 chemicals meeting characteristics identified by stakeholders as needing priority assessment. This included chemicals for which NICNAS already held exposure information, chemicals identified as a concern or for which regulatory action had been taken overseas, and chemicals detected in international studies analysing chemicals present in babies' umbilical cord blood.

Stage Two of IMAP began in July 2016. We are continuing to assess chemicals on the Inventory, including chemicals identified as a concern for which action has been taken overseas and chemicals that can be rapidly identified and assessed by using Stage One information. We are also continuing to publish information for chemicals on the Inventory that pose a low risk to human health or the environment or both. This work provides efficiencies and enables us to identify higher risk chemicals requiring assessment.

Figure 4.2: AICIS Assessment document example with a defined structure

In our definition, documents without TOC but with a defined structure are documents that do not contain a TOC all of them follow a template from document to document, *i.e.*, the name of the chapters are constant and normally, not always, containing the same chapters. AICIS Human Health Assessments (Figure 4.2) fitted in this group, where, not having a defined TOC, the structure of the document was almost always similar. Since the structure of these documents was almost always similar, we extracted the identifiers of the chapters using statistical data regarding the text of the documents.

Documents without TOC and defined structure are documents that do not contain a TOC and where the structure pattern is more difficult to identify when compared to the two previous groups defined. CIR Final and Published Reports, OECD Final Assessment Reports and RIFM Ingredient Safety Assessments are documents that do not provide any TOC and their structure is harder to process.

As the main priority of our work is to provide the client with an Information Extraction (IE) solution for the most number of sources, we needed to define priorities in our development for the right compromise between the development of the different Phases of the pipeline. So, regarding this initial phase, *i.e.*, the Preprocessing Phase, we tackle three documents, the SCCS Opinions, ATSDR Toxicological Profiles and AICIS Human Health Assessments. Due to a higher complexity, *i.e.*, no TOC and complex document structure, of the CIR Reports, OECD Assessment Reports, and RIFM Ingredient Safety Assessments we decided to, for the timeline that we had, not advance in those documents, being a future work objective.

As mentioned, regarding the algorithms to proceed with the development of the Preprocessing Phase, namely the division of the document in sections, we had three main ideas in mind:

- Use regular expressions for documents with TOC;

- Use text statistics for documents with a defined structure;
- Use Context Analysis;

In our preliminary work we already had contact with regular expressions, so using regular expressions was always a valid option for us. Using text statistics was an approach that occurred to us when analysing the various documents and we thought that the compromise between implementation and time was a good compromise. The last option that we had in mind was methods for context Analysis, *e.g.*, bag of words, TF-IDF (Term Frequency - Inverse Document Frequency), etc.

Our approach was to try create the Preprocessing Phase for the maximum number of data sources that we could in our available time, so we started to implement the methods using regular expressions and text statistics. Unfortunately, for the time of deliver of this report we did not delved into methods for Context Analysis but it is our intention for future work.

4.2 SCCS Opinions & ATSDR Toxicological Profiles

As mentioned in the Exploratory Work Section 4.1, these two documents have in common two characteristics:

- Both contain a TOC;
- The document structure is defined and always similar;

In order to divide the document into sections we first extracted the TOC of the document, where our objective was to get the identifiers of the chapters, *i.e.*, the number and title of each chapter and also the hierarchy position of the chapter. For example, Figure 4.3 details visually what components of the TOC of the documents we search for, *i.e.*, the hierarchy level, the Section title number and Section title name.

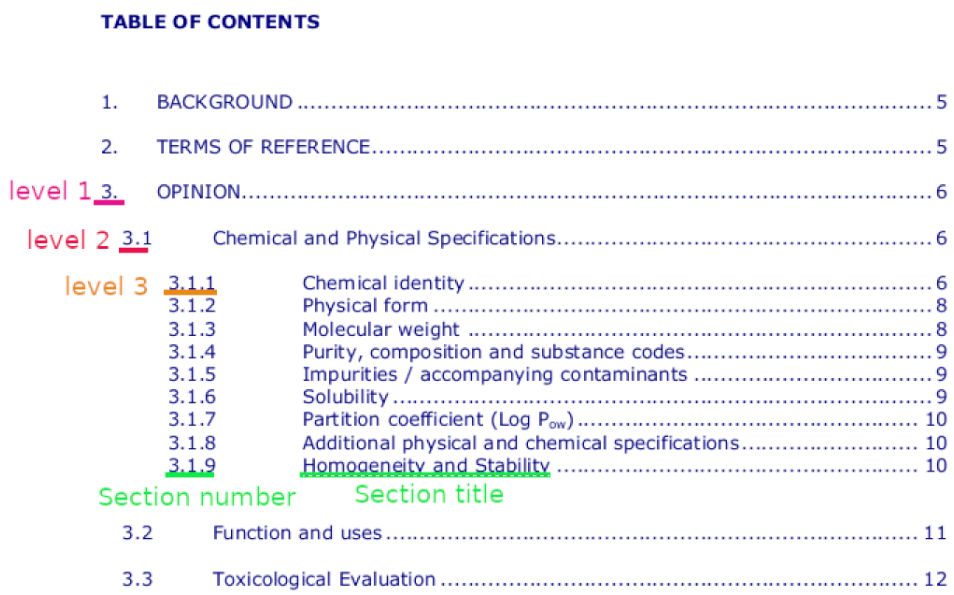


Figure 4.3: Elements necessary to extract from the TOC

This process of extracting the TOC is similar to that of a human when navigating and searching the document using the Index/TOC. The input PDF documents were converted to text with the *pdfplumber*[7] parser, and, combined with Regular Expressions, we could obtain the TOC of the document. The usage of the TOC allows us to find the start and the end of each section, *i.e.*, by considering the number and title of the sections, where the start corresponds to the section title obtained from the obtained TOC and the end of the section is the starting of the next section with the same hierarchical level. Figures 4.4 and 4.5 are visual representations the Preprocessing Phase, where the information obtained from the TOC (Figure 4.4) helps us divide the document into sections (Figure 4.5).

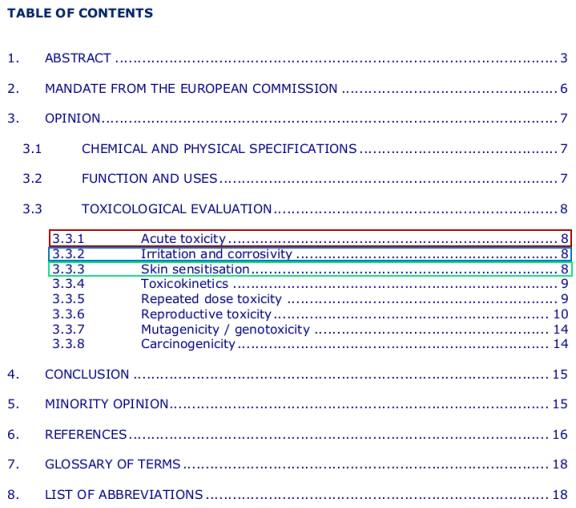


Figure 4.4: Graphical example of the Preprocessing process (1).

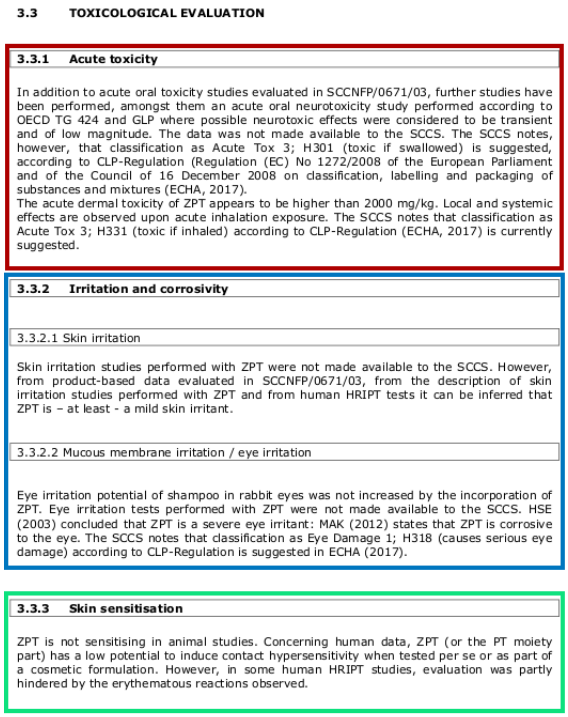


Figure 4.5: Graphical example of the Preprocessing process (2).

Regarding the practical implementation, to extract the TOC we firstly needed to detect

[7] `https://github.com/jsvine/pdfplumber`

the TOC page in the document. For that we used the function *search_for()* of the library *PyMuPDF* to search for the specific string. In the SCCS Opinions we search for the page containing the expression "Table of Contents" in order to find the page that contains the TOC, see Figure 4.6. In the ATSDR reports we search for the page that contains the expression "CONTENTS" two times, as in Figure 4.7.

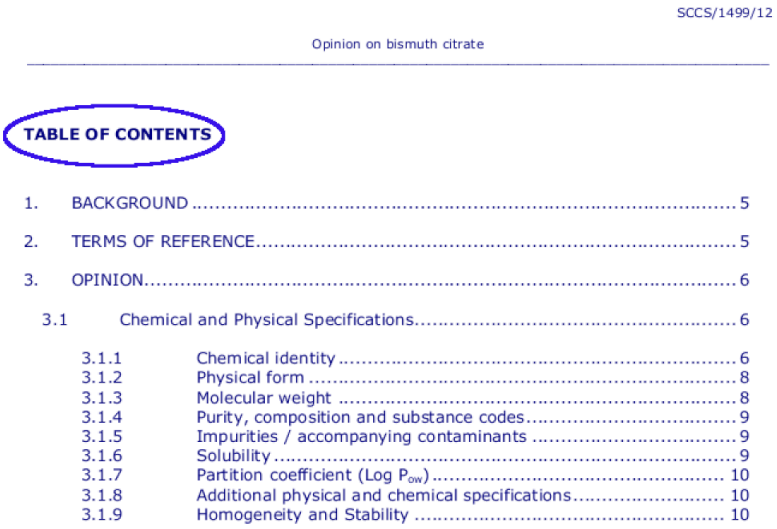


Figure 4.6: How we identified the TOC page in the SCCS Opinions

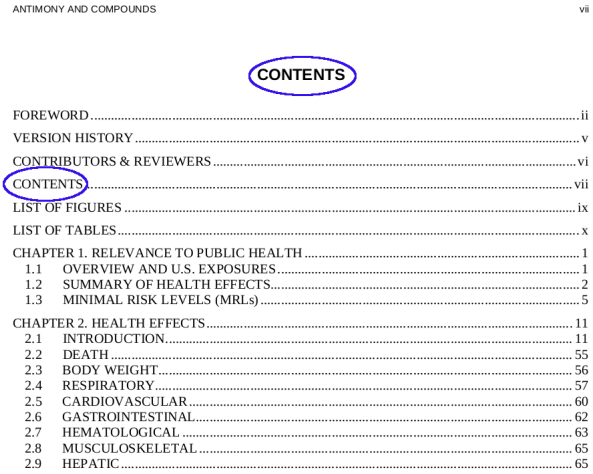


Figure 4.7: How we identified the TOC page in the ATSDR reports

After having the page of the TOC identified, we verified in the ATSDR reports if the TOC was only present in one or multiple pages. We needed to do this verification because in multiple documents the TOC was present in multiple, two or three, pages. For that verification we used the same function *search_for()* and we searched for the expression "LIST OF FIGURES" in the document. The page containing this expression is the first page that appears after the TOC, so the TOC was contained between the page that contains the "CONTENTS" expression and the page that contains the "LIST OF FIGURES" expression.

From the pages where the TOC is present, we extract the text using the function *extract_text()* from the library *PdfPlumber* and regular expressions, see Code 4.1, we extracted the number and title of the sections.

```
# Build TOC from the TOC pages Text
def buildTOC(text):
    TOC = []
    for line in text.splitlines():
        TOC.append(line)
    TOCTable = []  # Table Of Contents
    for line in TOC:
        if len(line) > 10:
            # obtain complete Section identifier (number + title)
            reg = "(\d)(\d)*(\.)*((\d)(\d)*(\.)*)*(\s)*[A-z][A-z\s\/\(\)\,\-]*"
            acceptableLine = search(reg, line)
            try:
                # obtain title number
                reg = "(\d)(\d)*(\.)*((\d)(\d)*(\.)*)*"
                titleNumber = search(reg, acceptableLine.group())
                # obtain title
                reg1 = "(\s)*[A-z\s\/\(\)\,\-]*"
                titlereg = search(str(titleNumber.group())+reg1, line)
                title = split(str(titleNumber.group()) +"(\s)*", titlereg.group())
                entry_dict = {"number": titleNumber.group(), "title": title[2]}
                TOCTable.append(entry_dict)
            except:
                # print(f"No Acceptable line in: {line}")
                pass
    return TOCTable
```

Listing 4.1: Build TOC function using regular expressions

Having the TOC extracted enabled us to search for the chapter identifiers in the middle of the PDF documents. We search for the number and title of the sections, where the start corresponds to the section title obtained in the TOC and the end of the section is the starting of the next section with the same hierarchical level. Otherwise, if the section is the last section in that hierarchical level, the next section is the first section with a lower hierarchical level. For the SCCS Opinions and ATSDR reports we identify if the next section has the same hierarchical level or lower by analysing the number of the sections, *i.e.*, we verify how many components the number has, Figure 4.8.

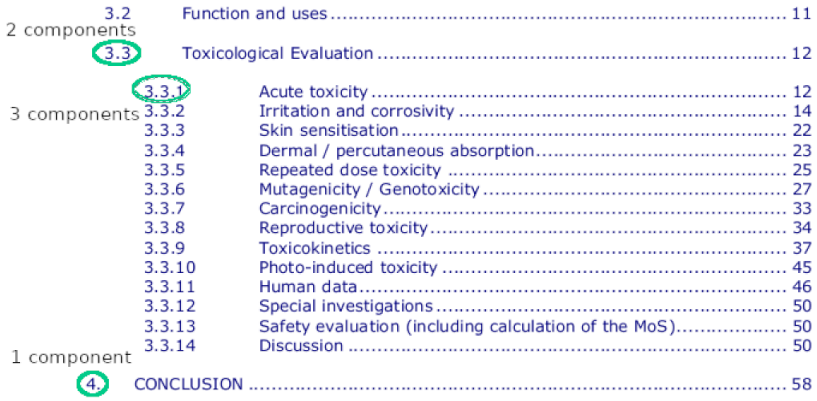


Figure 4.8: Components that we search in the sections numbers

All the implementation mentioned of this Preprocessing Phase using the TOC works for both documents of the different data sources, *i.e.*, SCCS and ATSDR because the documents share the two main characteristics, as mentioned before.

4.3 AICIS Human Health Assessments

The AICIS Human Health Assessments do not contain a TOC as the SCCS and ATSDR reports, so for the development of the Preprocessing Phase we needed to adopt a different

approach, that was when we had the idea of using a text statistics approach. For us, the definition of text statistics is literally the statistics of the characters present in the document, *i.e.*, characters size, types of font, bold and not bold text, etc. We also recognized two recurring patterns in the AICIS Assessments documents:

- Presence of redirectional links for the beginning of each section;
- Document with a defined structure;

So our approach was to: 1) identify the section titles through the existing page links and 2) use text statistics to found in the text those section titles.

To extract all the page links we used the function *get_ links()* from the library *PyMuPDF*, Code 4.2, where we obtained the name of the section titles, Figure 4.9.

```
# function that returns a list with the lines that
# contains links
def extractTextWithLinks(docPage, docName):
    doc = fitz.open(docName)
    page = doc.load_page(docPage)
    links = page.get_links()
    linksList = []
    for link in links:
        # limit the size the area to extract
        finalRect=link["from"]
        finalRect[2] = 1000

        titles = page.get_text("blocks", finalRect)
        # obtain section title
        title=titles[0][4].split("\n")[0]
        linksList.append(title)
    doc.close()
    return linksList
```

Listing 4.2: Function to extract Sections links from AICIS Assessments

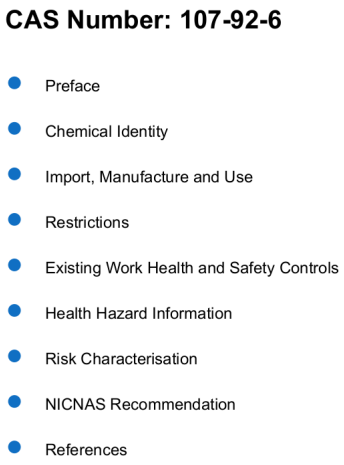


Figure 4.9: Example of Sections links from AICIS Assessments

Having the names of the main sections, we used text statistics where our idea was to identify patterns in the document, mainly identify the font type and size that are used in the diverse types of texts, *i.e.*, title characters, sections title characters, subsection title characters and main text characters. If we could identify those statistics, we could identify precisely in the document where a section starts and ends, being the end of the section the beginning of another one.

So the first step was to identify the Font Size of characters that is more used in the document, *i.e.*, the mode, so we can assume that it is the Font Size of the text inside the sections, Figure 4.10. Having the mode we search for the names of sections identified in the previous step that have the Font Size bigger than the mode size. With the function *search_for()* we can search for the title name in the document and extract their exact location, *i.e.*, page and location in page, Code 4.3 .

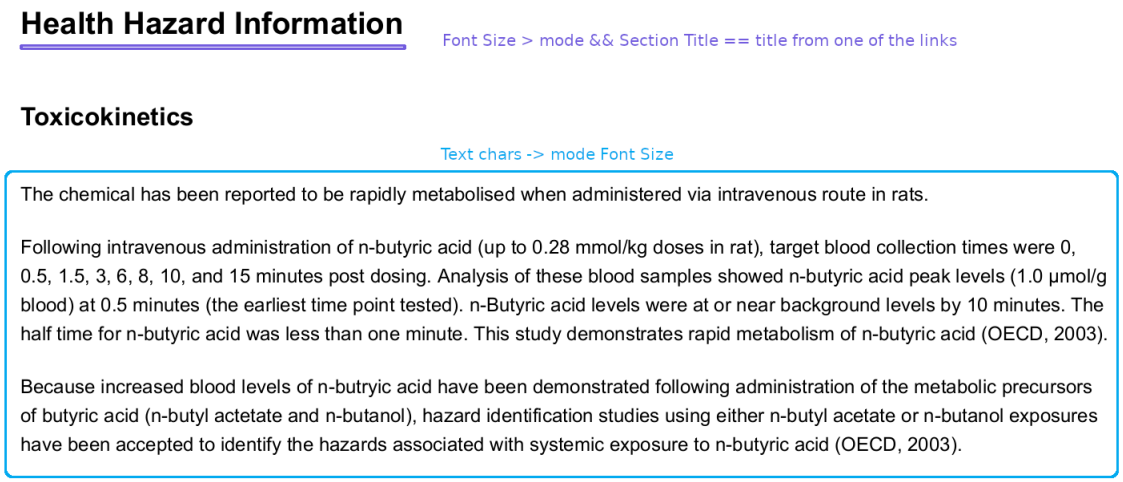


Figure 4.10: Visual example of approach using Font size statistics

```
# function that retrieves list of know titles with fontsize > modeFont,
# their respective font size, their page and their location in the document
def searchTitlesFonts(listOfKnownTitles, modeFont, docName):
    doc = fitz.open(docName)
    pageCount = doc.page_count
    listTitles = []
    for i in range(0, pageCount):
        page = doc.load_page(i)
        for title in listOfKnownTitles:
            areas = page.search_for(title)
            if len(areas) > 0 :
                for area in areas:
                    textExtracted = page.get_text("text", clip=area)
                    extracted = page.get_text("dict", clip=area)
                    blocks = extracted["blocks"]
                    for block in blocks:
                        line = block["lines"]
                        firtsLine = line[0]
                        if round(firtsLine["spans"][0]["size"],2) > modeFont["size"]:
                            title = {
                                "section": textExtracted.split("\n")[0],
                                "fontsize": round(firtsLine["spans"][0]["size"], 2),
                                "location" : area,
                                "page" : i
                            }
                            listTitles.append(title)
    doc.close()
    return listTitles
```

Listing 4.3: Function to obtain exact location of section titles in the document

In the AICIS Assessments, the important information regarding the toxicological properties is present in the "Health Hazard Information" section. Having the exact location where that section begins and ends, *i.e.*, the start of the next section title, our objective is to find in that section the subsections related to each toxicological property desired. For that we use the exact same algorithm that we used to extract the sections titles, we search in the "Health Hazard Information" section for the text that has characters with bigger font size than the mode font size but with smaller font size than the section titles. In Figure 4.10 the title "Toxicokinetics" is the beginning of that toxicological property, so using the

method of searching for chars with bigger font size than the font mode but smaller font size than the section titles, in this case "Health Hazard Information", we can detect all the titles that define the toxicological properties, just like the "Toxicokinetics" title. With that information, we basically can create our own TOC with all the information about the sections and subsections regarding the toxicological properties, *i.e.*, exact location in the document of the begin and end of sections and subsections, Figure 4.11.


```
{'subsection': 'Toxicokinetics', 'fontsize': 11.82, 'location': (41.999996185302734, 367.72430419921875, 123.22437286376953, 380.92755126953125), 'page': 3, 'hierarchy': 0}
{'subsection': 'Acute Toxicity', 'fontsize': 11.82, 'location': (41.999996185302734, 191.3878936767578, 121.90251922607422, 204.59112548828125), 'page': 4, 'hierarchy': 0}
{'subsection': 'Oral', 'fontsize': 10.69, 'location': (41.999996185302734, 235.92807006835938, 62.200363159179688, 247.8738555908203), 'page': 4, 'hierarchy': 1}
{'subsection': 'Dermal', 'fontsize': 10.69, 'location': (41.999996185302734, 332.7254333496094, 76.45376586914062, 344.67120361328125), 'page': 4, 'hierarchy': 1}
{'subsection': 'Inhalation', 'fontsize': 10.69, 'location': (41.999996185302734, 483.5491638183594, 88.36531066894531, 495.49493408203125), 'page': 4, 'hierarchy': 1}
{'subsection': 'Corrosion / Irritation', 'fontsize': 11.82, 'location': (41.999996185302734, 660.3673706054688, 156.25723266601562, 673.5706176757812), 'page': 4, 'hierarchy': 0}
{'subsection': 'Skin Irritation', 'fontsize': 10.69, 'location': (41.999996185302734, 704.9075317382812, 104.39371490478516, 716.8533325195312), 'page': 4, 'hierarchy': 1}
{'subsection': 'Eye Irritation', 'fontsize': 10.69, 'location': (41.999996185302734, 113.61849212646484, 102.01992797851562, 125.56427764892578), 'page': 5, 'hierarchy': 1}
{'subsection': 'Sensitisation', 'fontsize': 11.82, 'location': (41.999996185302734, 276.93011474609375, 115.55509185791016, 290.13336181640625), 'page': 5, 'hierarchy': 0}
{'subsection': 'Skin Sensitisation', 'fontsize': 10.69, 'location': (41.999996185302734, 321.4700622558594, 126.38866424560547, 333.41583251953125), 'page': 5, 'hierarchy': 1}
{'subsection': 'Observation in humans', 'fontsize': 10.69, 'location': (41.999996185302734, 551.0823364257812, 151.9426727294922, 563.0281372070312), 'page': 5, 'hierarchy': 1}
{'subsection': 'Repeated Dose Toxicity', 'fontsize': 11.82, 'location': (41.999996185302734, 139.61225891113328, 174.45001220703125, 152.81549072265625), 'page': 6, 'hierarchy': 0}
{'subsection': 'Oral', 'fontsize': 10.69, 'location': (41.999996185302734, 184.15267944335938, 62.200363159179688, 196.0984649658203), 'page': 6, 'hierarchy': 1}
{'subsection': 'Dermal', 'fontsize': 10.69, 'location': (41.999996185302734, 677.1438598632812, 76.45376586914062, 689.0896606445312), 'page': 6, 'hierarchy': 1}
{'subsection': 'Inhalation', 'fontsize': 10.69, 'location': (41.999996185302734, 327.4725036621094, 88.36531066894531, 339.41827392578125), 'page': 7, 'hierarchy': 1}
{'subsection': 'Genotoxicity', 'fontsize': 11.82, 'location': (41.999996185302734, 398.48870849609375, 113.58512115478516, 411.69195556640625), 'page': 7, 'hierarchy': 0}
```


Figure 4.11: Excerpt of TOC created in the AICIS Assessments

As visible in Figure 4.11, the hierarchy level is also present in our information. *Level 0* corresponds to all the section titles of the toxicological properties, like "Toxicokinetics" in Figure 4.10 and *level 1* corresponds to all the sub properties, see Figure 4.12.

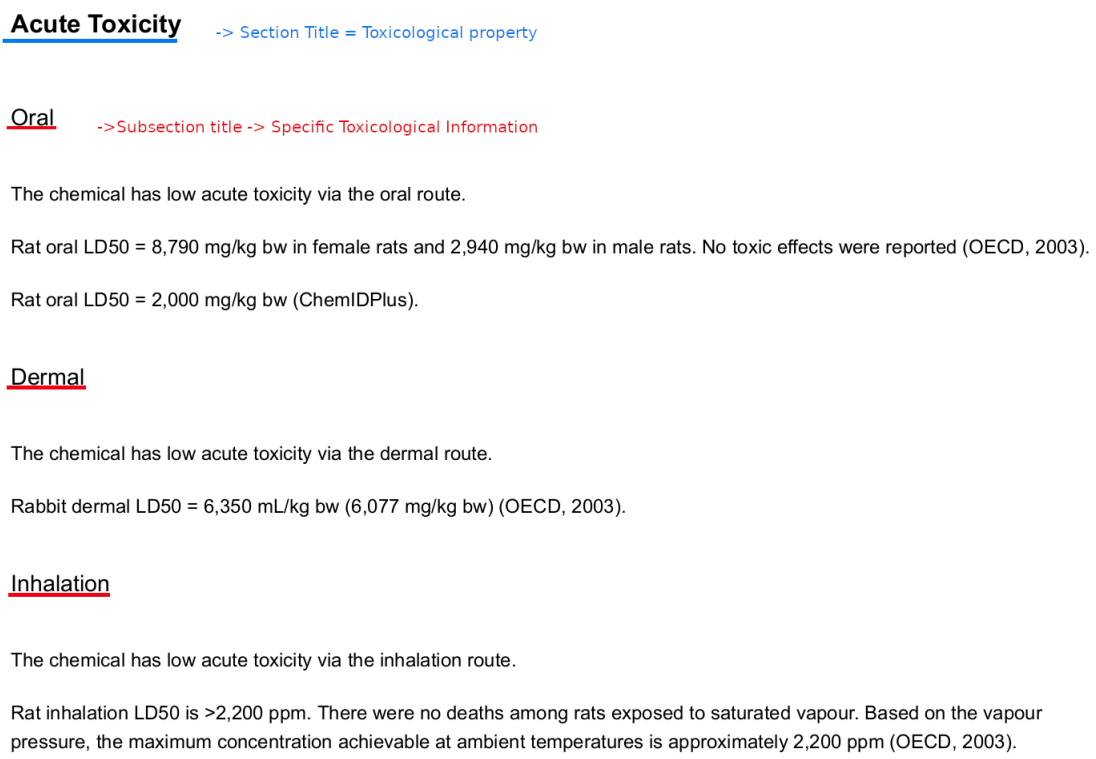


Figure 4.12: AICIS Assessment Sections and Subsections identification

4.4 Configuration Files

Once we can search for the section identifiers in the document, we need to know which section titles to search for in the document. For that we created a different configuration file for each document type, *i.e.*, SCCS Opinions (Figure 4.13), ATSDR Toxicological Profiles and AICIS Human Health Assessments (Figure 4.14). Those configuration files contain

data necessary for other Phases of our project but, regarding this Preprocessing Phase, they contain the titles of the sections and subsections that we want to analyse.

All the config files have the same structure:

$$Section; Subsections; Questions; Models$$

Regarding this first Phase of the pipeline we just use the "Section" and "Subsections" information from the config files. They contain all the sections that we need to look for and their respective subsections. In the case that there are no subsections we simply do not introduce any in the config files. The files are *csv* files and the different data are separated by ",", used when are multiple subsections, questions or models, and ";" in order to separate the different data, *i.e.*, section, subsections, questions and models. Figures 4.13 and 4.14 are excerpts of the final config files used.

```
Dermal / percutaneous absorption;;What is the dermal absorption,What is the oral absorption;roberta,bert,biobert
Acute Toxicity;Acute dermal toxicity,Acute oral toxicity,Acute Intraperitoneal Toxicity,Acute Inhalation Toxicit
Irritation and Corrosivity;Skin Irritation,Eye Irritation,Airways Irritation,Mucous Membrane Irritation;What is
Skin Sensitisation;;What is the Guideline?,What is the study?,What is the species?,What is the concentration?,Wh
Mutagenicity;Genotoxicity in Vitro,Genotoxicity in Vivo;What is the Guideline?,What is the study?,What is the sp
Carcinogenicity;;What is the Guideline?,What is the study?,What is the species?,What is the conclusion?;roberta,
Photo-induced Toxicity;Phototoxicity,Photomutagenicity;What is the Guideline?,What is the study?,What is the spe
Reproductive Toxicity;Two generation,Two-generation,Teratogenicity and other data on fertility,Developmental Tox
Repeated dose toxicity;;What is the NOAEL value?,What is the Guideline?,What is the study?;roberta,bert,biobert
Safety Evaluation;;What is the NOAEL value?;roberta,bert,biobert
```

Figure 4.13: Excerpt of the config file for the SCCS Opinions

```
Toxicokinetics;;What is the dermal absorption?,What is the oral absorption?;roberta,bert,biobert
Acute Toxicity;Oral,Dermal,Inhalation,Observation in humans;What is the guideline?,What is the study?,What is th
Acute Toxicity;;What is the guideline?,What is the study?,What is the species?,What is the LD50?,What is the LC5
Corrosion / Irritation;Skin Irritation,Eye Irritation,Respiratory Irritation,Observation in humans;What is the g
Corrosion / Irritation;;What is the guideline?,What is the study?,What is the species?,What is the concentration
Sensitisation;Skin Sensitisation;What is the guideline?,What is the study?,What is the species?,What is the conc
Sensitisation;;What is the guideline?,What is the study?,What is the species?,What is the concentration?,What is
Repeated Dose Toxicity;Oral,Dermal,Inhalation,Observation in humans;What is the guideline?,What is the study?,Wh
Repeated Dose Toxicity;;What is the guideline?,What is the study?,What is the NOAEL value?;roberta,bert,biobert
```

Figure 4.14: Excerpt of the config file for the AICIS Assessments

4.5 Current and Future Work

Both our approaches for the SCCS Opinions/ ATSDR Reports and AICIS Assessments tackle a large number of files from those data sources. However some limitations and errors were find over the course of the project while more and more files were used for testing.

Regarding the SCCS Opinions we identified two main problems:

1. the TOC is not completely defined;
2. the section titles are present in the document multiple times;

For both of these problems we could integrate a text statistics approach to fix them. In the first case, where the TOC is not completely defined, as in Figure 4.15, we could have a similar approach to the AICIS documents, *i.e.*, we search for the characters mode font size and try to obtain the titles of the sections and subsections presents in the Chapter "Toxicological Evaluation".

For the second case, Figure 4.16, where the section identifiers are present more than two times, *i.e.*, in the TOC and in the beginning of the section, we could have the same

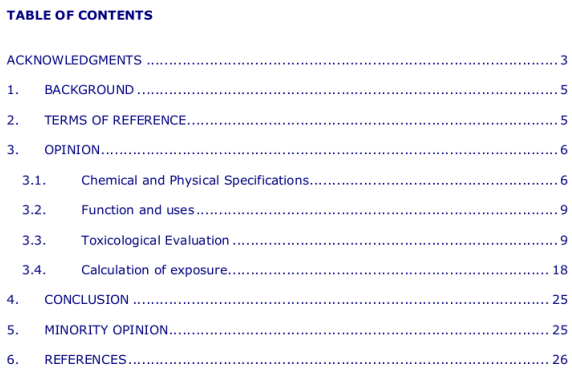


TABLE OF CONTENTS

Figure 4.15: Example of SCCS Opinion document with TOC not completely detailed

approach, where we just accept as section title the text with a similar font size to rest of the sections titles. That way we could guarantee that mentions of the section titles throughout the document text would be ignored.

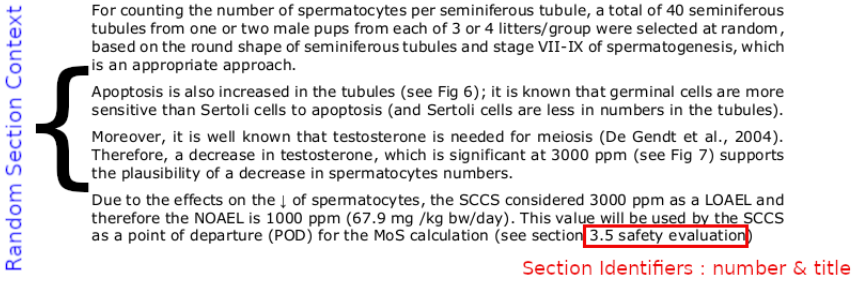


Figure 4.16: Example of Section identifier being present in a different location than the normal beginning of section

Regarding the AICIS Human Health Assessments documents we identified a problem: sometimes in the beginning of the page, the date was written with the same font size as the subsections, see Figure 4.17. That results in the context extracted from that section or subsection being incomplete or entirely wrong.

The fixes mentioned were not developed in time because other priorities emerged or the problems were just found in the later process of evaluation. In our evaluation process we evaluate the Preprocessing Phase where we verify if the context extracted from each section is the correct one. More information and concrete results is mentioned and discussed in the Chapter 7.

Regarding further work that can be developed, as mentioned in this approach's of IE the most important Phase is the Preprocessing Phase, so for new and different types of documents this is the phase that needs the new development regarding each document. About the CIR and OECD documents, we want to see in the future if we can develop any Context Analysis approaches to find the sections regarding each toxicological property.

Corrosion / Irritation

Respiratory Irritation

The chemical is considered to cause respiratory tract irritation, warranting hazard classification (see **Repeat dose toxicity: Inhalation**).

Data on acute inhalation toxicity are limited. In the single study available (see **Acute toxicity: Inhalation**), no respiratory tract effects were reported in rats apart from breathing through the mouth.

Repeated dose toxicity studies in rats indicated minimal damage to the nasal olfactory epithelium after exposure to vapours of the chemical starting from 1 ppm in a 13-week study and 2 ppm in a four-week study (see **Repeat dose toxicity: Inhalation**). The damage was more pronounced at higher exposure concentrations (SCOEL, 2010).

After long-term exposure (see **Carcinogenicity**), nasal and lung inflammation were observed in rats and mice at all doses of 10 ppm (50 mg/m^3) and above.

Skin Irritation

The chemical is considered to be a slight skin irritant. The reported effects were not sufficient to warrant hazard classification.

In a well-conducted study (as reported by the EU RAR, 2003), six rabbits were exposed to a single dose of 500 mg of the chemical for four hours and observed for six days. Slight to well defined erythema was seen in three rabbits, appearing 30 minutes after exposure, with a slight fissuring of the skin appearing 72 hours after exposure. No oedema was reported and all signs had cleared within six days (EU RAR, 2003).

In another study (reported as similar to OECD TG 404), the chemical was applied in occlusive patches, at a dose of 500 mg, to the skin of six New Zealand White rabbits for 24 hours. Mean erythema and oedema scores were <2 for individual animals. Effects had not completely cleared within 48 hours (REACH).

Eye Irritation

The chemical is a slight eye irritant. The reported effects were not sufficient to warrant hazard classification.

In an eye irritation study (reported as similar to OECD TG 405), six rabbits were exposed to the chemical (0.1 mg in one eye of each rabbit) for 24 hours and then observed for seven days. Only minor effects were reported. One rabbit had an iris reaction on day two after dosing (grade 1), five rabbits showed conjunctival redness (grade 1) over a period of two days and slight chemosis (grade 1) was noted in one rabbit on day one after dosing. All effects had cleared by day three after dosing (REACH).

Sensitisation

Figure 4.17: AICIS Assessments data font size problem

Chapter 5

Information Extraction and Data Verification

We can use the Extractive Question Answering (QA) models for extracting information. In order to do so, we need to identify the set of questions related to the context and to the information that we want to obtain. So we need to define which models to use and which set of questions to use for each property. If the Information Extraction (IE) process works as supposed we also need to verify the information extracted in order to clean the information extracted until now.

In the Preprocessing Phase, we obtain the sections from the documents that we use as the context that we give to the Extractive QA Models in the IE process. The Extractive QA Models return the answers that they found in the context given the set of questions that we provide.

We explored multiple models in an initial work phase, ending up implementing the IE process with one model and since we just used the models as we imported them from the Hugging Face Hub, we realised that using multiple models was one possibility because we did not have to invest time in training or fine-tuning the models.

We realised that there are two points that we needed to fix with the Data Verification process:

- For each question, each extractive QA Model returns all the answers that predicts that is correct;
- As we can use multiple models, we obtain the sum of multiple answers that each model returns;

That generates two options that we can follow. In first option we do not develop any process of verification and we just simply use all the answers that the models provide. This way we can guarantee the maximum recall possible. This is a valid option but we wanted to guarantee some level of confidence of the information extracted. For that we needed to fix the two points previously mentioned, *i.e.*, multiple answers from the same and multiple models.

To resolve both the problems we used Natural Language Processing (NLP) Similarity Metrics. As mentioned in the Chapter 2, there are multiple Similarity Metrics available, *e.g.*, Recall-Oriented Understudy for Gisting Evaluation (ROUGE) Score, Bilingual Evaluation Understudy (BLEU) Score, Damerau-Levenshtein Distance, etc. Our approach was

to use Similarity Metrics, in specific the ROUGE Score, to eliminate or maintain similar answers, *i.e.*, returns from the extractive QA models, process explained in detail throughout the chapter.

So in this Chapter we analyse and clarify which models we use, how to we resolve the two points mentioned and the experiments done in order to test the work developed in this two phases mentioned, *i.e.*, the IE process and the Data Verification process.

5.1 Exploratory Work

The initial step in the development of this project was to define a way that we can use to complete our main goal, extract information from documents. In that regard we had some boundaries that we wanted to set in our approach:

- Define a approach that is flexible for documents from different domains;
- Use existing techniques and technologies;

Having that in account we started our research and we found great advances in extraction of information using Transformer Models, namely Models for extractive QA. In our research we did initial tests, as ilustrated in Figures 5.1 and 5.2, in the Hugging Face[1] website. From those initial tests we could see that, for those cases tested, the results obtained showed good impressions. Since those models are trained and already fine-tuned we could use those same models and approach for developing future IE processes for other documents.

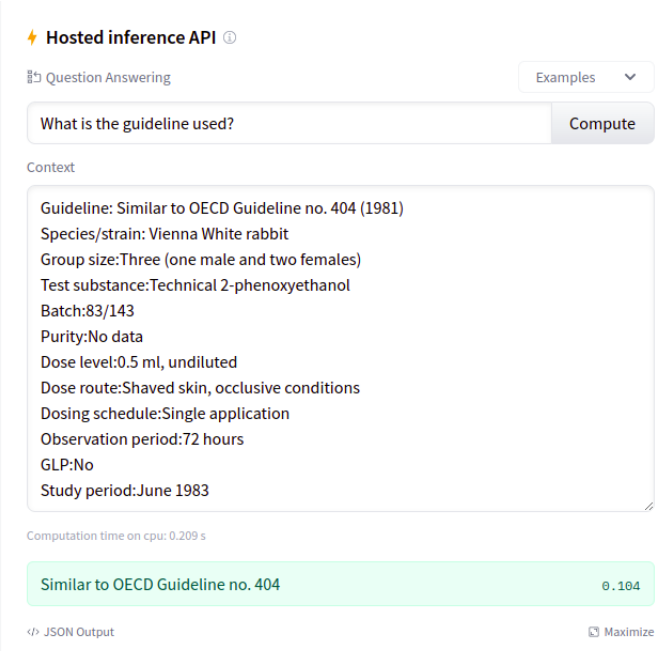


Figure 5.1: Example 1 - test using a semi-structured context

[1] https://huggingface.co/

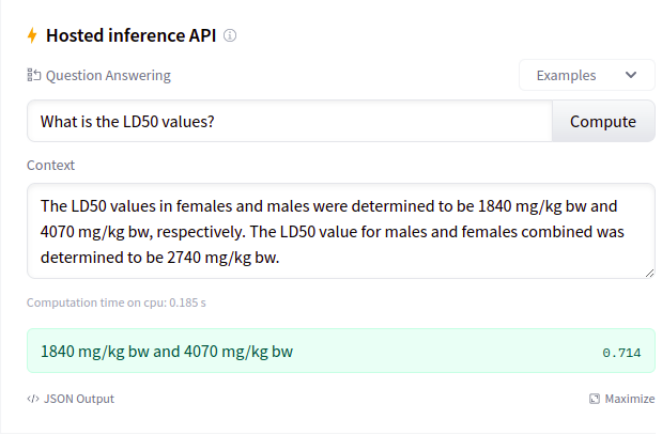


Figure 5.2: Example 2 - test using a unstructured context

5.2 IE Process

After the initial tests we needed to choose which models to use. From the Hugging Face Hub we imported the following models:

- BERT [2]
- BioBERT [3]
- RoBERTa [4]
- MINILM [5]
- AlBERT [6]
- Chemical BERT [7]
- ELECTRA [8]

All these seven models are models trained for Extractive QA using the Stanford Question Answering Dataset (SQuAD) 2.0 dataset. From the seven models mentioned models we, in the final implementation, ended up using just three models: Bidirectional Encoder Representations from Transformers (BERT), BioBERT: a pre-trained biomedical language representation model for biomedical text mining (BioBERT) and Robustly Optimized BERT Pretraining Approach (RoBERTa). We used three models because of the combination process created, process explained in Chapter 5.4, and also due to the compromise that we wanted to achieve between performance and time.

The models take time to extract answers from the context. Given that the documents contain multiple pages and the sections used, *i.e.*, the sections with information regarding the toxicological properties can contain multiple paragraphs with multiple lines, the time

[2]https: // huggingface.co/deepset/bert-base-cased-squad2
[3]https://huggingface.co/ktrapeznikov/biobert_v1.1_pubmed_squad_v2
[4]https://huggingface.co/deepset/roberta-base-squad2
[5]https://huggingface.co/deepset/minilm-uncased-squad2
[6]https://huggingface.co/mfeb/albert-xxlarge-v2-squad2
[7]https://huggingface.co/recobo/chemical-bert-uncased-squad2
[8]https://huggingface.co/deepset/electra-base-squad2

used by the models ended up being notable, *i.e.*, taking more and more seconds the bigger the document.

Having decided which models to use the next step was to define the right set of questions to use with the models. The questions are directly related to the information that we want to extract.

With the help of the security advisor we could identify which information that we needed to extract. In Table 5.1 we present the identified toxicological properties and respective information that is necessary to extract from the documents.

Substance Property	Information to Extract
Repeated Dose Toxicity	NOAEL[9] value; OECD[10] Guideline used
Acute Toxicity	Species used in study; OECD Guideline used; Exposure route; LD50 value; LC50 value
Irritation	Species used in study; OECD Guideline used; Exposure route; Concentration used in study; Classification
Mutagenicity	Species used in study; OECD Guideline used; Classification
Skin Sensitization	OECD Guideline used; Classification; Concentration used in study
Carcinogenicity	Species used in study; OECD Guideline used; Classification
Photo-induced Toxicity	Species used in study; OECD Guideline used; Classification; Concentration used in study
Reproductive Toxicity	Species used in study; OECD Guideline used; Classification
Absorption	Dermal Absorption; Oral Absorption

Table 5.1: Substances properties information

Substance Property	Questions
Repeated Dose Toxicity	What is the NOAEL value? What is the guideline?; What is the study?
Acute Toxicity	What is the guideline?; What is the study?; What is the species? What is the LD50?; What is the LC50?
Irritation	What is the guideline?; What is the study?; What is the species?; What is the concentration?; What is the conclusion?
Mutagenicity	What is the Guideline?; What is the study?; What is the conclusion?
Skin Sensitization	What is the Guideline?; What is the study?; What is the conclusion?; What is the concentration?
Carcinogenicity	What is the species?; What is the Guideline?; What is the study?; What is the conclusion?
Photo-induced Toxicity	What is the Guideline?; What is the study?; What is the conclusion?; What is the concentration?
Reproductive Toxicity	What is the Guideline?; What is the study?;What is the species?; What is the conclusion?
Absorption	What is the dermal absorption?; What is the oral absorption?

Table 5.2: Set of questions per property

To extract that information we needed to formulate questions to use with the extractive QA models. As all the chosen models are fine-tuned in the SQuAD, we formulated the questions using similar questions formats that are present in this dataset. As SQuAD uses the Six W's (Who, What, When, Where, Why and How) in the formulation of the questions, we also created questions of this kind, regarding each information that we want to extract. For example, in the sentence present in one of the PDF documents used, "Eye irritation potential of shampoo in rabbit eyes was not increased by the incorporation of ZPT" we want to obtain the species that the test applies to, so we can formulate a question

[9]No Observed Adverse Effect Level
[10]Organisation for Economic Co-operation and Development

as "What is the species?". Given the sentence (as the context) and the question, we hope to obtain from the QA models the right answer, in this case, "rabbit". In Table 5.2 we present the set of questions used for each specific property.

As in the Phase 1, the Preprocessing phase, the config files created have important data for the application of this Phase 2 of the pipeline. Concretely, the config files include which questions to use for each section. The main toxicological properties are the same but the identification of those in the different document sources are different, as shown in Chapter 4. Figure 5.3 is a excerpt of the Australian Industrial Chemicals Introduction Scheme (AICIS) Assessments config file containing the questions used.

```
Toxicokinetics;;What is the dermal absorption?,What is the oral absorption?;roberta,bert,biobert
Acute Toxicity;Oral,Dermal,Inhalation,Observation in humans;What is the guideline?,What is the study?,What is th
Acute Toxicity;;What is the guideline?,What is the study?,What is the species?,What is the LD50?,What is the LC5
Corrosion / Irritation;Skin Irritation,Eye Irritation,Respiratory Irritation,Observation in humans;What is the g
Corrosion / Irritation;;What is the guideline?,What is the study?,What is the species?,What is the concentration
Sensitisation;Skin Sensitisation;What is the guideline?,What is the study?,What is the species?,What is the conc
Sensitisation;;What is the guideline?,What is the study?,What is the species?,What is the concentration?,What is
Repeated Dose Toxicity;Oral,Dermal,Inhalation,Observation in humans;What is the guideline?,What is the study?,Wh
Repeated Dose Toxicity;;What is the guideline?,What is the study?,What is the NOAEL value?;roberta,bert,biobert
```

Figure 5.3: Excerpt of AICIS Assessment config file

5.3 Repeated Answers from Models

These Extractive QA models are trained with a *max position embeddings* of 512 or 1024. The *max position embeddings* is the maximum sequence length that this model might ever be used with. Since the sections can contain lengths greater than the mentioned, we needed to iteratively traverse the context with cycles. For each cycle the model returns what is the best answer, however, as their is crossover of contexts in the cycles, the same answer can be returned, ending with repeated answers from the same context.

In order to eliminate repeated or similar answers, *e.g.* "OECD TG 414 (2001)" and "OECD TG 414" are the same answer but given the iteratively cycling of the context they could be present in multiple contexts given to the model and one was completely extracted, "OECD TG 414 (2001)', and the other, "OECD TG 414', has the year, "(2001)", missing. In this example the answers extracted are the same but not 100% equal, so we need to use Similarity Metrics in order to compare the extracted answers. For that we could have used any of the previous metrics mentioned, *i.e.*, ROUGE Score, BLEU Score, Damerau-Levenshtein Distance. We tested the three mentioned metrics, all with similar performance so we just started the implementation using the ROUGE Score, obtained good results and we settled in that metric.

Code 5.1 is the implementation of the ROUGE Score in order to just preserve the different answers given by each model. We firstly remove the "!not an answer!" and "[CLS]" returns because they are a identifier of not any answer returned for a given context (explained in Chapter 5) and a token that is used for the classification of relations (a return that the models give), respectively. After deleting those returns that do not provide any relevant information we remove the duplicates, the answers that are 100% similar. After that the process of eliminating repeated answers that are similar using the ROUGE Score initiates. We rank the answers by descending length in order to remove the smaller elements, then we iteratively compare the answers with the ROUGE Score and mark as similar and eliminate the ones that have a ROUGE Score higher than the define threshold. We empirically defined a 0.8 out of 1 threshold because we wanted to preserve most of the answers, just eliminating those that are extremely similar.

```
# use the rouge score in order to compare similar
# answers from the same model
def rougeScoreComparison(Answers):
    try:
        while True:
            Answers.remove("[CLS]")
    except:
        pass
    try:
        while True:
            Answers.remove("!not an answer!")
    except:
        pass
    # remove duplicates
    Answers = list(dict.fromkeys(Answers))
    if len(Answers) == 0:
        Answers.append(NAA)
    # if there are multiple answers verify if they are similar
    elif len(Answers) >= 2:
        # sort list by descending length in order to remove smaller elements
        Answers.sort(key=len, reverse=True)
        threshold = 0.8
        listRepeatedAnswers = []
        for a, b in itertools.combinations(Answers, 2):
            # compare(a, b)
            scorer = rouge_scorer.RougeScorer(['rouge1'], use_stemmer=True)
            scores = scorer.score(a,b)
            finalScore = scores['rouge1']
            if finalScore[-1] >= threshold:
                listRepeatedAnswers.append(b)
        # remove duplicates -> similar answers
        listRepeatedAnswers = list(dict.fromkeys(listRepeatedAnswers))
        for elem in listRepeatedAnswers:
            try:
                Answers.remove(elem)
            except:
                pass
    return Answers
```

Listing 5.1: Function to remove similar answers given using the ROUGE Score

5.4 Combination Process

As mentioned, we ended up using three Extractive QA models, the BERT, BioBERT and RoBERTa models. But why did we use three models? One wasn't enough for our implementation to work? In short, yes, our implementation and pipeline would work with just one model.

However there were two strong points to use multiple models. The first point is that, in our exploratory work, we obtained different answers to the same question using different models. The second point is, as the models in the Hugging Face Hub are already trained and fine-tuned, we did not invest any time in the development of those models, so we could invest that time in experiment and testing the models.

So we wanted to extract the most information possible but also create a method to achieve some level of confidence in the information extracted, because, when comparing the answers from different models, if the same information is returned by more than one model, it means more confidence on the suitability of such answer, something that we otherwise would not be able to guarantee with just one model.

```
# use the rouge score in the combination process
def rougeCombinationProcess(Answers, threshold):
    # remove '!not an answer!' answers and empty answers
    try:
        while True:
            Answers.remove("")
    except:
        pass
```

```
try:
    while True:
        Answers.remove("!not an answer!")
except:
    pass
if len(Answers) == 0:
    Answers.append(NAA)
# if there are multiple answers verify if they are similar
elif len(Answers) >= 2:
    # sort list by descending length in order to remove smaller elements
    Answers.sort(key=len, reverse=True)
    listRepeatedAnswers = []
    # using the itertools
    for a, b in itertools.combinations(Answers, 2):
        scorer = rouge_scorer.RougeScorer(['rouge1'], use_stemmer=True)
        scores = scorer.score(a, b)
        finalScore = scores['rouge1']
        if finalScore[-1] >= threshold:
            if len(listRepeatedAnswers) == 0:
                listRepeatedAnswers.append(a)
            # verify if there is any answer similar in the returns
            # if not add answer
            else:
                for elem in listRepeatedAnswers:
                    scorerList = rouge_scorer.RougeScorer(['rouge1'], use_stemmer=True)
                    scoresList = scorerList.score(elem, a)
                    finalScoreList = scoresList['rouge1']
                    if finalScoreList[-1] <= 0.8:
                        listRepeatedAnswers.append(a)
    # remove duplicates
    listRepeatedAnswers = list(dict.fromkeys(listRepeatedAnswers))
    Answers = listRepeatedAnswers
    if len(Answers) == 0:
        Answers.append(NAA)
return Answers
```

Listing 5.2: Combination Process Function the ROUGE Score

In order to compare the answers returned by the multiple models we created a simple comparison method that we called "Combination Process". The process is extremely similar to the process of eliminating repeated answers from the same model. The difference is that, instead of returning the most different answers, our objective is to return those that are similar. Code 5.2 is the implementation of the Combination Process, very similar to the one of elimination repeated answers. We defined a threshold of 0.6 out of 1 for the similar answers. We tested with various values between 0.5 until 0.9, and by our tests with some documents we found that with a threshold higher than 0.6/0.7 we could obtain similar answers. Also, this comparison is between every answer, so we need to guarantee that we just return one of the similar answers. For that we do another verification but with an increased threshold (0.8) just to verify if very similar answers are not returned.

5.5 Experiments

As we were developing these two phases, we run some initial tests in order to evaluate our work and approach, so we gathered ten random Scientific Committee on Consumer Safety (SCCS) Opinions documents and executed the process of IE developed so far, *i.e.*, Phase 1 and Phase 2 and Phase 3. In these tests we just used a portion of the substance properties and we did not use all the questions identified because, at the time, not all the properties sections of the SCCS document were identified. Table 5.3 presents the set of questions per property used in our experiments.

The whole process of initial tests run can be deeply analyzed in paper "Question Answering For Toxicological Information Extraction" (Ferreira et al., 2022) written in the middle of this project. In the paper we used the same approaches of Phase 1, Phase 2 and Phase 3 and we evaluated the results obtained, Tables 5.4 and 5.5. At the time of the writing of the paper we performed the Combination process manually, *i.e.*, comparing the

Substance Property	Questions
Repeated Dose Toxicity	What is the NOAEL value?
Acute Toxicity	What is the guideline?;What is the species?
Irritation	What is the guideline?;What is the species?
Mutagenicity	What is the Guideline?;What is the conclusion?
Skin Sensitization	What is the Guideline?;What is the conclusion?;What is the concentration?
Carcinogenicity	What is the species?;What is the Guideline?;What is the conclusion?
Photo-induced Toxicity	What is the Guideline?;What is the conclusion?
Reproductive Toxicity	What is the Guideline?;What is the species?;What is the conclusion?

Table 5.3: Set of questions per property tested.

information extracted manually.

	BERT			BioBERT			RoBERTa		
	F1	P	R	F1	P	R	F1	P	R
Acute Toxicity Information	0.77	0.87	0.70	0.74	0.59	1.00	1.00	1.00	1.00
Irritation Information	0.68	0.76	0.61	0.76	0.61	1.00	0.85	0.85	0.85
Skin Sensitisation Information	0.86	0.82	0.92	0.72	0.56	1.00	0.85	0.84	0.87
Mutagenicity Information	0.67	0.53	0.92	0.57	0.40	1.00	0.71	0.55	1.00
Carcinogenicity Information	0.84	0.78	0.91	0.66	0.50	1.00	0.58	0.43	0.90
Photo-induced Toxicity Information	0.44	0.40	0.50	0.58	0.41	1.00	0.54	0.37	1.00
Reproductive Toxicity Information	0.80	0.66	1.00	0.70	0.54	1.00	0.73	0.57	1.00
Repeated Dose Toxicity Information	1.00	1.00	1.00	0.80	0.66	1.00	1.00	1.00	1.00
Micro Average	0.76	0.68	0.85	0.64	0.48	1.00	0.76	0.65	0.94

Table 5.4: Individual evaluation of QA models on SCCS documents

By first analysing each QA model individually, see Table 5.4, we were able to understand that some optimizations can be developed even though some strong results were obtained. In some cases the precision and the recall were close to perfect, which can be due to the disposition of the information in the document, *i.e.*, better results can be achieved if the information is present in bullet points than if it is in the middle of the sentences.

By analysing Table 5.5, we confirm that there are gains when the models are combined, both in terms of precision and recall, when compared with the individual models results. In general, and despite the limited set of questions, solid results were obtained.

Writing the paper provided us some important knowledge of some strong points that we need to improve, namely the evaluation process. For the paper the evaluation process was done by the team that developed and implemented the IE process, bringing a bias aspect that is not correct. So our objective was to eliminate that bias and improve our evaluation process, work done and mentioned in Chapter 7. Also big improvements were made since the date of the writing of the paper and the combination process, as mention throughout this Chapter, is totally automatic and without any manual intervention.

	BERT + BioBERT + RoBERTa		
	F1	Precision	Recall
Acute Toxicity Information	1.00	1.00	1.00
Irritation Information	0.89	0.96	0.83
Skin Sensitisation Information	0.96	0.96	0.96
Mutagenicity Information	0.78	0.65	0.96
Carcinogenicity Information	0.84	0.80	0.88
Photo-induced Toxicity Information	0.75	0.60	1.00
Reproductive Toxicity Information	0.85	0.74	1.00
Repeated Dose Toxicity Information	1.00	1.00	1.00
Micro Average	0.87	0.82	0.93

Table 5.5: Combination process evaluation on SCCS documents

Chapter 6

Integration Tools

The process of extracting information from the documents is done in the first three Phases of the pipeline, however there are two important steps required to complete our pipeline:

- Generate a toxicological profile of the substance with the information extracted;
- Enable communication between our work and future services;

The Data-to-text (D2T) process of generating a toxicological profile is an important step in the completion of the toxicological report of the substance in the Cosmedesk platform. This toxicological profile is a summary with information about the substance analysed. It profile provides a quick way for humans, without expertise in the domain, to acquire a little information and knowledge about that particular substance analysed. In this Chapter we explain the work developed, the current limitations and what the future work and final objectives are with the generation of the toxicological profile.

But in the end we need a way to share the information extracted and toxicological profile with other services that Cosmedesk uses. This project, Safety Desk, will be used by the Cosmedesk platform, so we needed to create a bridge in order to those two to communicate. For that we ended up building a RESTfull API with some endpoints regarding the information extracted and the generated toxicological profile. In Chapter 6.2 the RESTfull API will be explained, what are the concrete objectives and future work in the integration in the Cosmedesk platform.

The developed pipeline has the objective of extracting information from documents, in this case and developed until now, Scientific Committee on Consumer Safety (SCCS) Opinions reports, Australian Industrial Chemicals Introduction Scheme (AICIS) Human Health Assessments and Agency for Toxic Substances and Disease Registry (ATSDR) reports, but all the implementation was developed without having any method of evaluating the quality of the information extracted from the documents.

In order to evaluate our work we developed a webpage with some functionalities for considering the opinion and evaluation of the security advisor. That webpage provided the security advisor a simple user interface and with the resources to evaluate the work developed, *i.e.*, evaluate the quality of the information extracted for each property of the substance. That evaluation was saved and detailed in the next Chapter 7.

In this Chapter we explain deeply the D2T process implemented, the RESTfull API and the evaluation webpage created.

6.1 D2T Generation Process

The D2T Generation Process, as mentioned in the Chapter 2, is a process based in templates and we just need to fill the variables with the information extracted from the document. The templates were provided by the security advisor and those templates are the same that they usually use in the construction of the toxicological profile. Some examples of the templates provided by the security advisor, Figures 6.1 and 6.2, show how the templates are simply variable dependent.

> **Mutagenicity** was investigated in accordance to **$Guideline Used$**, using **&Species Used$**, in the ***presence/absence*** of metabolic activation; The test substance was classified as **$Classification$**.

Figure 6.1: Template for the Mutagenicity property

> The **Irritation** potential was assessed in a study performed in accordance to **$Guideline Used$**, where **&Species Used$** were exposed to **$Concentration$**; The test substance was classified as **$Classification$**.

Figure 6.2: Template for the Irritation property

This in terms of implementation we just need to realise conditional verifications in order to check what information the Information Extraction (IE) process extracted and conjugate that information extracted with the templates provided by the security advisor.

As a result we obtain texts that contain the templates completed with the information extracted in the variable positions, Figure 6.3. When none information is present in the document or extracted regarding a property, we simply do not use the template related to that property.

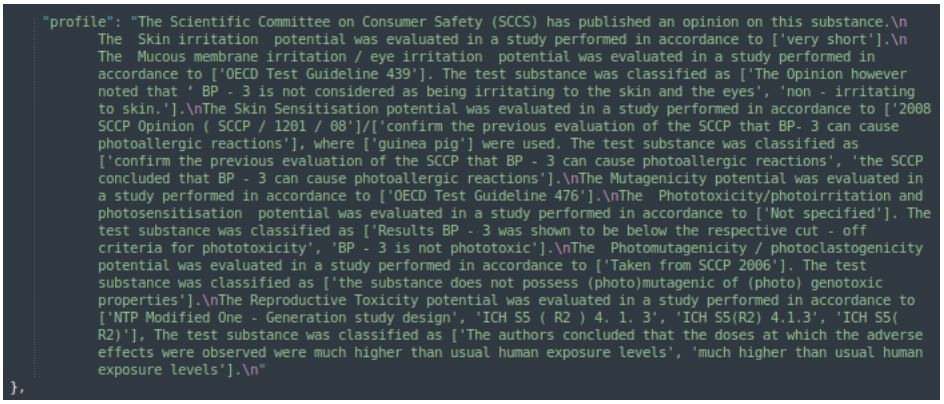


Figure 6.3: Example of toxicological profile generated

In terms of final result and direct use in the Cosmedesk platform, the application is more complicated given the scope of this project. Some reasons for that is that this D2T process just has in account the information extracted from one document at a time. The Safety Desk service just extracts information from the documents, not in any way saving it in a database, so the D2T process just has in account the information extracted from one document, while the ideal goal is to create a toxicological profile that contains all the information extracted from all the documents used. This means that this D2T generation process needs to be implemented from the side of the Cosmedesk service.

The D2T generation process being in the Cosmedesk service gives some flexibility and a better final user experience. As the Cosmedesk has a database with all the information extracted related to the substances, the completion of the templates used for the D2T is more complete. If the D2T process has all the data from multiple documents that provides the possibility of the creation of a good user interface where the user can define what the final text in the toxicological profile can be.

6.2 Rest API

Although not the final product given that there are more sources to be implemented and a continuous work in the Safety Desk service, we developed a RESTfull API in order to provide a way between our Safety Desk service and others, *i.e.*, Cosmedesk, to communicate.

For each source, *i.e.*, SCCS Opinions, ATSDR reports and AICIS Assessments, we implemented endpoints using the Flask[1] python library that returns a Json response with the information extracted from the document. Code 6.1 is the implementation of one of the routes of the RESTfull API. The request must be a *POST* request in order to send a file and with the parameter "Option" we choose one of three options:

1. *Option=all* - Extract the information from all the toxicological properties in the document;
2. *Option=$toxicological property$* - Extract the information from all specific property passed in the parameter;
3. *Option=d2t* - Return the toxicological profile produced with the information extracted from all the toxicological properties;

The process of defining what Sections of the documents are used to extract information from is present in the config files explained in detail in Chapters 4 and 5.

```
# API route related to SCCS Opinions Documents
# option = all -> extract information from all toxicological properties
# option = $name of toxicological property$ -> extract information
# regarding $name of toxicological property$
# option =  d2t -> produce toxicological profile based in the extraction of
# information from all toxicological properties
@app.route('/structured-toc/cosing', methods=["POST"])
def sccsRoute():
    section = request.args.get('Option')
    if request.method == 'POST':
        if 'file' not in request.files:
            return jsonify(returnError)
        file = request.files['file']
        if file and allowed_file(file.filename):
            completeName = os.path.join(app.config['UPLOAD_FOLDER'], secure_filename(file.filename))
            file.save(completeName)
            document = completeName.replace("tmp/", "")
            # if request to extract all information
            if section.lower() == "all":
                returnAPI, listSizeSections = IEtocCosing.cosingAll(CosingStructure, completeName)
            if section.lower() == "d2t":
                returnAPI, listSizeSections = IEtocCosing.cosingAll(CosingStructure, completeName)
                toxicologicalProfile=TOCFunctions.toxicologicalProfile(returnAPI, 'sccs')
            # if request to extract information from 1 specific question
            else:
                for line in CosingStructure:
                    if section.lower() in line["Section"].lower():
                        returnAPI = IEtocCosing.cosingSection(CosingStructure, section,
                        completeName)
            # after processing is done, delete file
            os.remove(completeName)
            # Returns
            if toxicologicalProfile:
                return jsonify(toxicologicalProfile, returnSuccess)
            if returnAPI:
```

[1] https://flask.palletsprojects.com/en/2.2.x/

```
            return jsonify(returnAPI, returnSuccess)
        else:
            return jsonify(returnError)
    elif file and not allowed_file(file.filename):
        return jsonify(returnError)
```

Listing 6.1: Implementation of SCCS Opinions related API Route

In term of returns of the information extracted, there are two types of returns per toxicological property:

- Toxicological properties with the information searched in a Section of the document, Figure 6.4;
- Toxicological properties with the information searched in Subsections of a Section of the document, *i.e.*, more detail obtained, Figure 6.5;

As explained in Chapter 6.2, we implemented an endpoint for the toxicological profile that returns a human-readable text, *i.e.*, a summary, of the information extracted from a document about a certain substance that produces a return, *i.e.*, text as in Figure 6.3.


```
{
    "Extracted": {
        "Context": " \n \nAfter oral exposure of rats to 14C-furfural, at least 90% is
            absorbed in the gastro-intestinal \ntract. After inhalatory exposure to
            furfural, pulmonary retention in humans was 78%. When \nhumans are exposed to
            furfural vapours (30 mg/m3), the dermally absorbed quantity of \nfurfural is
            about 30% of the amount absorbed through inhalation. After dermal exposure to \n
            liquid furfural, about 3 µg furfural per cm2 skin per minute is absorbed in
            humans. In the EU \nRisk Assessment Report, it was concluded that 90% oral and
            100% dermal and inhalation \nabsorption were to be used in the risk
            characterisation. \nRef.: 11 \n \nComment \nIn the absence of dermal absorption
            data relevant for the use in cosmetic products, the \nSCCS will use 100% dermal
            absorption and 90% for oral absorption for calculation of MoS. \n",
        "dermal absorption": [
            "100 %"
        ],
        "oral absorption": [
            "90 %"
        ]
    },
    "Models": [
        "roberta",
        "bert",
        "biobert"
    ],
    "Reference Pages": [
        10,
        11
    ],
    "Section": "Dermal / percutaneous absorption"
},
```


Figure 6.4: Json return of a Section regarding a toxicological property

6.3 Evaluation Webpage

The evaluation webpage created, Figure 6.6, is a simple webpage with basic functionalities in order to provide the security advisor a platform easy to use. Our initial idea was to use a secondary program to analyse the returns from the Rest API created, *i.e.*, Postman[2], but that created two problems. The first is that we needed to introduce the security advisor to using requests and read Json. Also, in that way the security advisor was totally responsible for saving the evaluation. Those two problems quickly eliminated that idea and we worked in the development of the evaluation webpage.

For that development, as we used the Flask Library[3] for the development of the Restfull API, we take advantaged of that library also providing a quick way of developing webpages

[2]https://www.postman.com/
[3]https://flask.palletsprojects.com/en/2.2.x/

```
{
    "Extracted": [
        {
            "Context": "\n \nRabbit LDLo:  620 mg/kg bw \nRef.: 1 \n \n",
            "LC50": [
                "!not an answer!"
            ],
            "LD50": [
                "620 mg / kg"
            ],
            "Subsection": "  Acute dermal toxicity ",
            "guideline": [
                "!not an answer!"
            ],
            "species": [
                "Rabbit"
            ],
            "study": [
                "!not an answer!"
            ]
        },
        {
            "Context": "\n \nRat LD50:  65 mg/kg bw \nMouse LD50:  400 mg/kg bw \nGuinea
                pig LD50:  541 mg/kg bw \nRabbit LD50:  800 mg/kg bw \nDog LD50:  950 mg/kg
                bw \nRef.: 1 \n \n",
            "LC50": [
                "950 mg / kg bw"
            ],
            "LD50": [
                "!not an answer!"
            ],
            "Subsection": "  Acute oral toxicity ",
            "guideline": [
                "!not an answer!"
            ],
            "species": [
                "!not an answer!"
            ],
            "study": [
                "!not an answer!"
            ]
        },
        {
            "Context": "\n \nRat LC50:  175 ppm/6 h \nMouse LC50:  350 ppm/6 h \nDog LC50:
                370 mg/6 h \nHuman TCLo:  0.31 mg/m³ \nRef.: 1 \n",
            "LC50": [
                "370 mg / 6 h"
            ],
            "LD50": [
                "370 mg / 6 h"
            ],
            "Subsection": "  Acute inhalation toxicity ",
            "guideline": [
                "!not an answer!"
            ],
            "species": [
                "Rat LC50 : 175 ppm / 6 h Mouse"
            ],
            "study": [
                "!not an answer!"
            ]
        }
    ],
    "Models": [
        "roberta",
        "bert",
        "biobert"
    ],
    "Reference Pages": [
        9,
        9
    ],
    "Section": "Acute Toxicity"
```


Figure 6.5: Json return of a Subsections regarding a toxicological property

using templates (Jinja[4] template).

The evaluation webpage provided the security advisor the possibility of evaluating documents from two of the three sources that we considered so far in this work, *i.e.*, SCCS Opinions and AICIS Human Health Assessments. We did not evaluate the ATSDR reports because, as mentioned int the Chapter 4, that source is only used for information regarding Health effects on Cancer, which is important but a very small quantity of substances and use cases need that information. So, with the active opinion of the security advisor, we decided to evaluate the work developed in the SCCS Opinions and AICIS Human Health Assessments. With the evaluation from the security advisor we have an impartial evaluation and a good data collection to analyse and discuss the work developed.

In the initial page, as show in Figure 6.6, we provide the user with the possibility of uploading the file depending of the source, *i.e.*, SCCS Opinions or AICIS Assessments. After the upload is concluded, the process of extracting information from the document starts, see Figure 6.7. The process consists of all the phases of the pipeline created, *i.e*

[4]https://jinja.palletsprojects.com/en/3.1.x/templates/

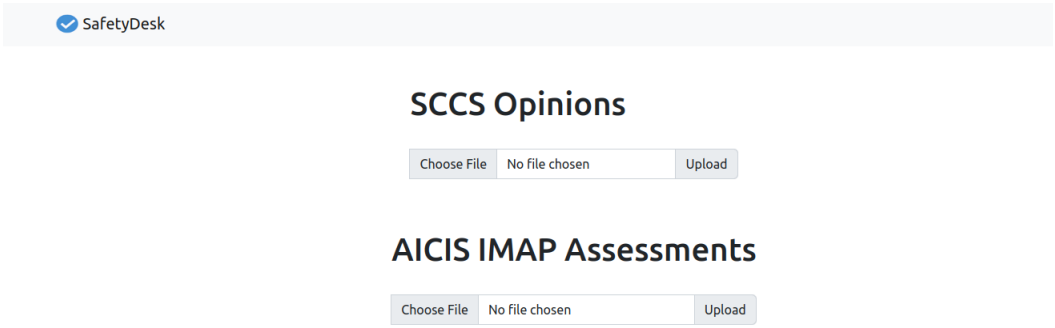


Figure 6.6: Evaluation Webpage initial page

preprocessing the document depending on the source (Phase 1), extract information from the sections created in the preprocessing phase (Phase 2) and verification and validation of the information extracted (Phase 3).

When the process is finished, the information extracted from the document is presented to the user in a new page, see Figure 6.8. In that page the information extracted is presented with the context used, *i.e.*, the section related to each toxicological property. In that page the user is presented with the evaluation mechanism (Likert Scale) created that consists of evaluating the information extracted with one of the five points:

1. Without Information;
2. Incorrect Information;
3. Incomplete Information;
4. Correct Information;
5. Incorrect Context;

We, along side the security advisor, discussed and outlined the rules that they would follow in order for the evaluation to be consistent throughout the process. So we decided to evaluate the information extracted as "*Incorrect Context*" when the context provided does not match the context related to that toxicological property in the document. That can occur due to an error in the Phase 1 of the pipeline, the Preprocessing Phase, in that way we are evaluating the results from that phase as well.

The evaluation "*Without Information*" was decided to be used when, the context is correct and the IE process does not find any correct information because that information

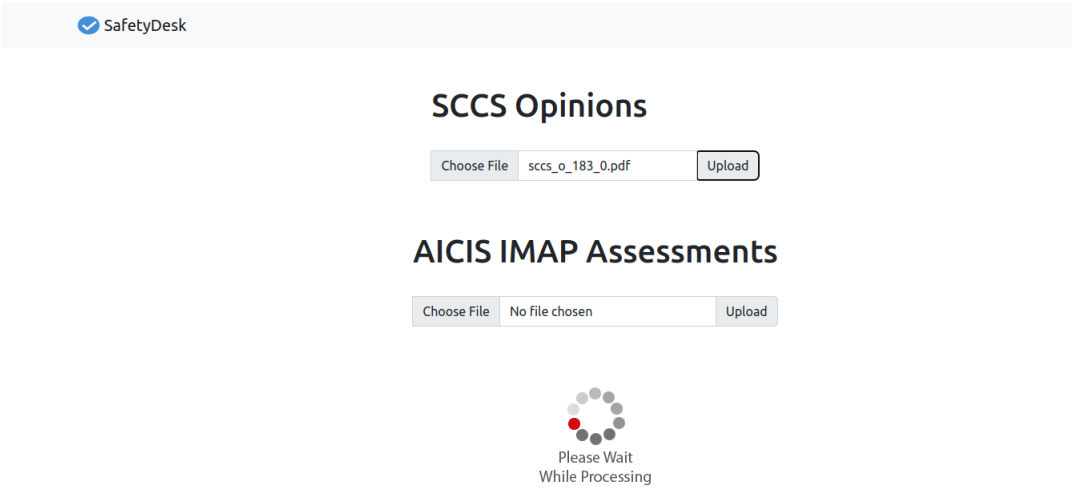


Figure 6.7: Evaluation Webpage with file chosen and process running in background

does not exist. This simply occurs when the degree of detail in the documents regarding a toxicological property is not complete and does not provide all the information that we aimed to extract.

"*Incorrect Information*" is used when the information extracted is totally wrong, *i.e.*, the security advisor does not validate that information as correct in the case it was extracted.

"*Incomplete Information*" was decided to be used when not all the information was extracted, *i.e.*, there are multiple expressions or values from the document that are relevant to a certain characteristic of a toxicological property and not all of them are extracted with our process, meaning that the information was partially extracted.

The last term of the evaluation that we provided was the "*Correct Information*" that was used when the information extracted was correct, *i.e.*, the IE process extracted all the expressions or values needed to be extracted that characteristic of the toxicological property.

Having those five options to evaluate the information extracted, like a Likert scale, was a good and essential point that we decided to use in the evaluation process but we wanted to know what was the correct information, *i.e.*, the information that the process should have extracted, when the security advisor marked the information extracted as "*Incorrect Information*" or "*Incomplete Information*". In those cases we added an input field, Figure 6.9, for the security advisor to specify the correct information that should have been extracted.

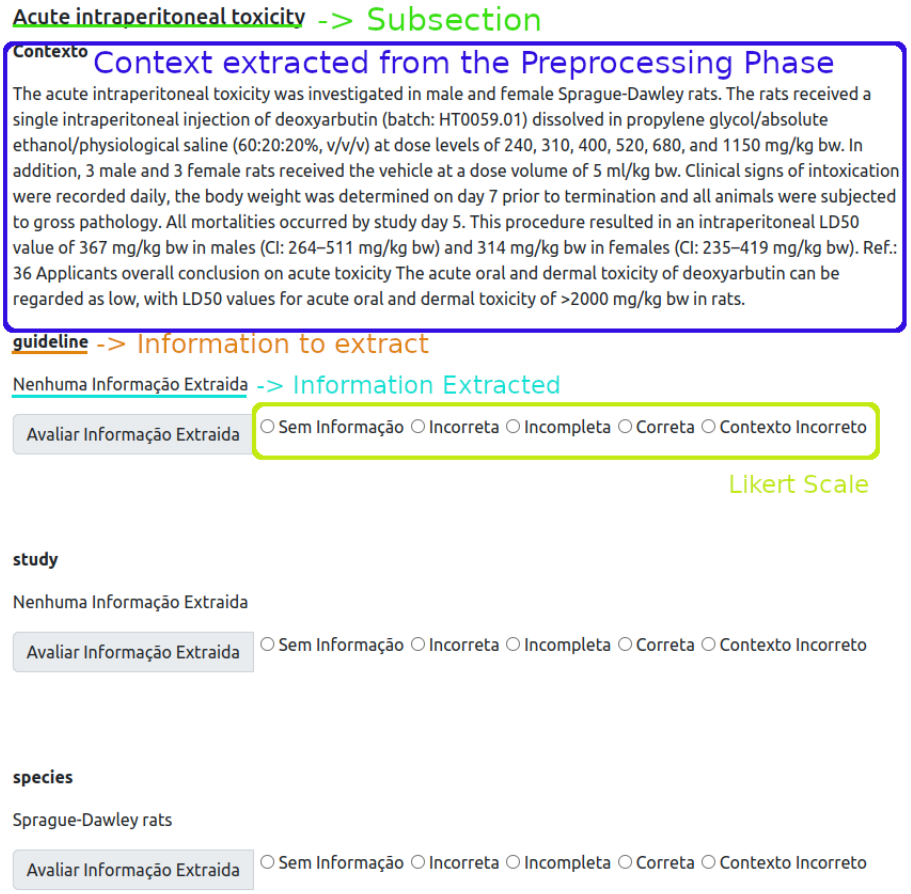


Figure 6.8: Evaluation Webpage after the IE process is completed

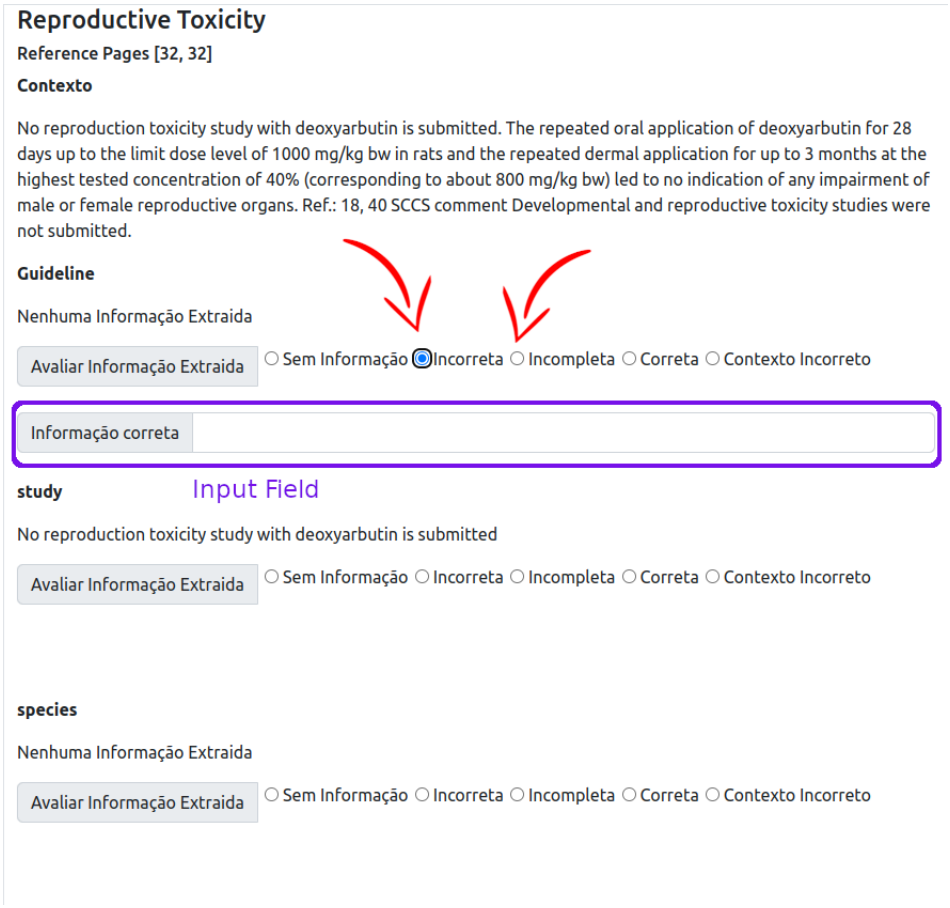


Figure 6.9: Evaluation Webpage Input Field popup

Chapter 7

Results and Discussion

In this Chapter we analyse the results obtained from the evaluation performed by the security advisor using the evaluation Webpage created. We provided the Webpage to the security advisor and they, according to their possibilities regarding time, performed the evaluation of the work developed.

We explain the configurations of the evaluation, *i.e.*, the number of documents used, how the information was saved and we mention some statistics of those documents and also some performance at the level of execution time.

We also analyse the results of the application of our Information Extraction (IE) service in the documents, with results scored according to common metrics, *i.e.*, precision, accuracy, recall and F-Score.

7.1 Results

The evaluation process was done only by the security advisor in order to guarantee consistent results throughout all the evaluation, which means that all the results are not in any way modified by the team that implemented the IE process. We wanted to be sincere and have a true evaluation of the work developed.

For the evaluation the security advisor used 33 documents, 15 Scientific Committee on Consumer Safety (SCCS) Opinions and 18 Australian Industrial Chemicals Introduction Scheme (AICIS) Human Health Assessments. The information of the evaluation was saved in two files, "*filesExecuted.csv*" and "*formSubmission.csv*".

In the first, "*filesExecuted.csv*", we saved for each document analysed the name of the document, the duration of the IE process, the date time of the execution of the IE process and the sizes of each sections. The document saves that information for us to later study and analyse the duration of the IE process and the sizes of each sections.

In the "*formSubmission.csv*" file we saved the submissions done by the security advisor in the evaluation webpage. That file contains all the information, *i.e.*, document name, toxicological property, characteristic of the toxicological property, information extracted, evaluation of the security advisor and the input field values added by the security advisor when the information extracted was evaluated as "*Incorrect Information*" or "*Incomplete Information*".

7.1.1 Documents statistics

From the "*filesExecuted.csv*" file we gathered important statistics about the files and sources used in order to compare them. Firstly we analysed the sections of each document, see Figures 7.1 and 7.2, where we calculated the average number of tokens present in each section. We noticed that the sizes of the sections in both sources had a great standard deviation that is why we also removed the outlier values. In some sections, both in the SCCS and AICIS source, there were not any outliers, visible in Figures 7.1 and 7.2 when the same average number of tokens is equal with and without outliers.

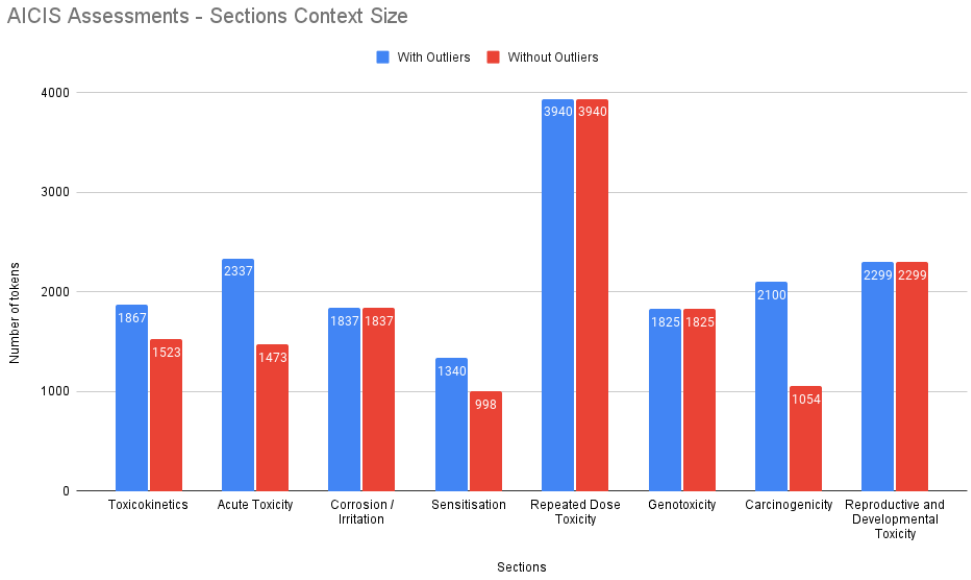


Figure 7.1: Average number of tokens per section in AICIS Assessment reports

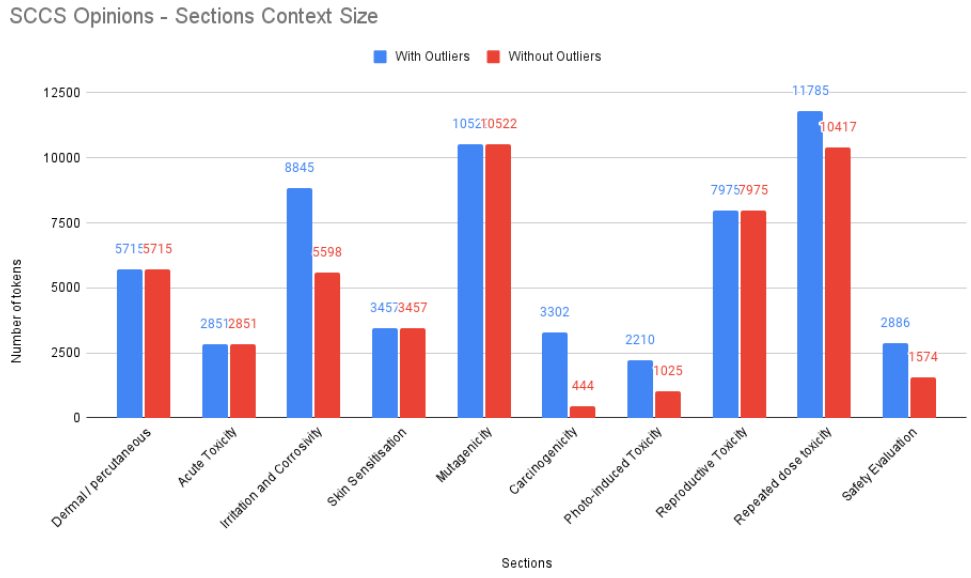


Figure 7.2: Average number of tokens per section in SCCS Opinions

In terms of total context used from each document, see Figure 7.3, we can verify that almost all SCCS Opinions contain more context than the AICIS Assessments. This is not a direct correlation with the size of the documents because we just gather the number of tokens used in the models, *i.e.*, the context for each section, that means that although in most cases the SCCS Opinions are longer than the AICIS Assessments (number of pages), we cannot directly correlate the size of document with the total context used.

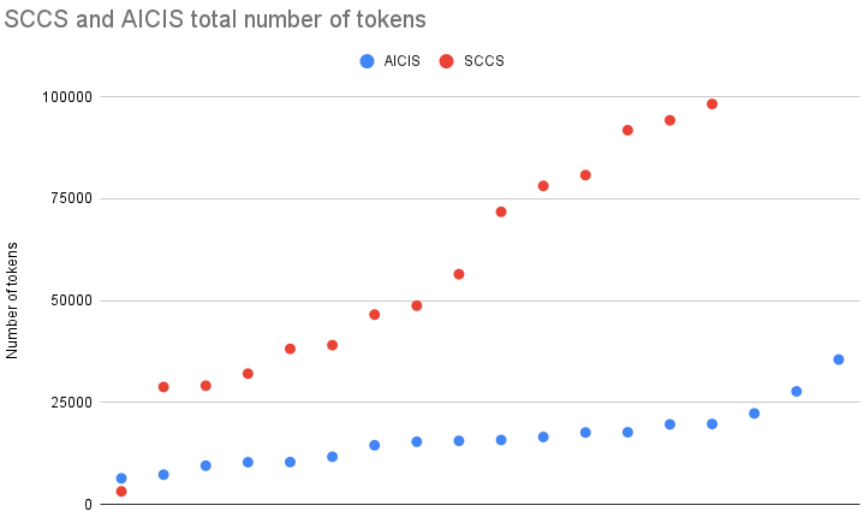


Figure 7.3: Total number of tokens in SCCS and AICIS reports

In terms of performance of our implementation, as the SCCS Opinions and AICIS assessments contain sections of different sizes and the total context used is also not directly related, we could not directly compare the total executions times. If we directly compared them, as in Figure 7.4, we would obtain a figure similar to Figure 7.3 where we can directly affirm that the number of tokens used is a direct contributor to better executions times of the AICIS Assessments.

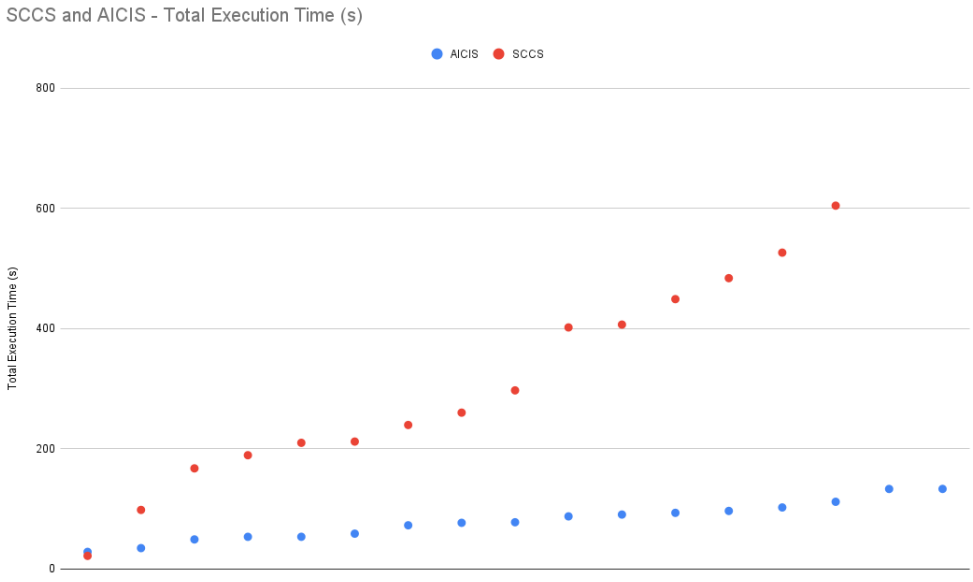


Figure 7.4: SCCS and AICIS Total execution time (s)

For the comparison to be fair we decided to create a metric to compare the performance of our implementation:

$$\frac{TotalContextSize(numberoftokens)}{TotalExecutionTime(s)}$$

Using that metric we can directly compare the performance of our implementation in both sources, Figures 7.5 and 7.6.

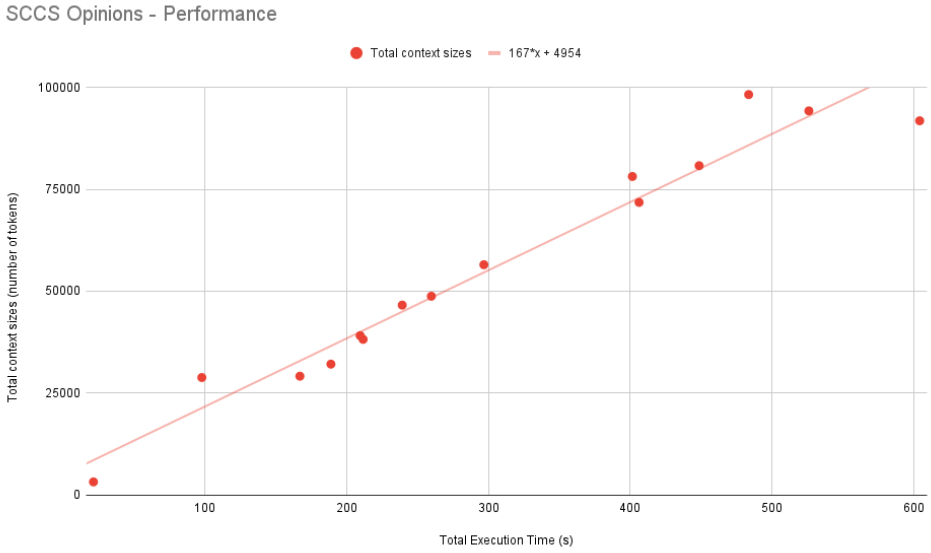


Figure 7.5: SCCS Opinions performance

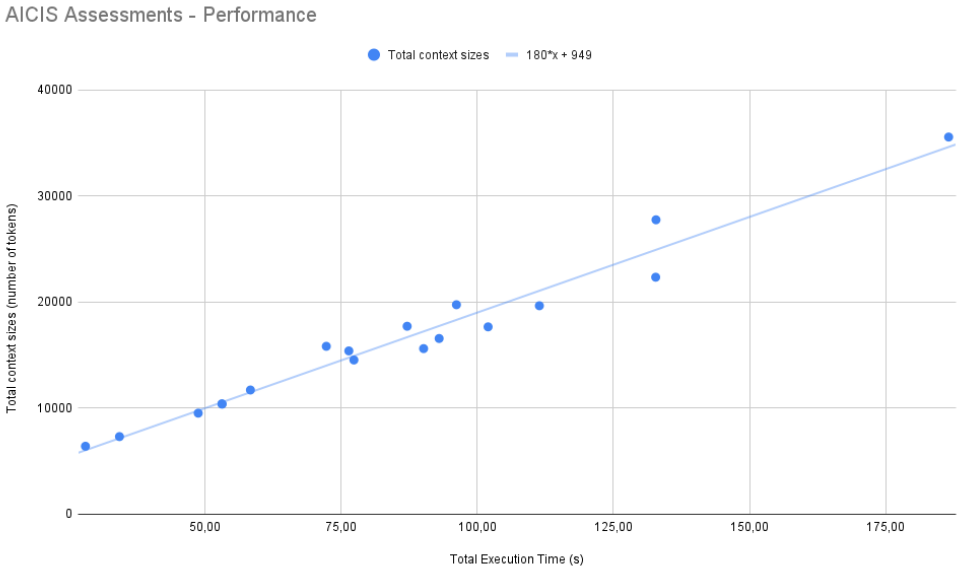


Figure 7.6: AICIS Assessments performance

In average, the AICIS Assessments had a performance of 195 tokens/second with a standard deviation of 17 tokens against the 187 tokens/second and 33 tokens of standard deviation of the SCCS Opinions. The greater standard deviation value of the SCCS

Opinions is the cause of the inclusion of all the 15 samples, *i.e.*, documents used, in the calculations, if the outliers values where removed the standard deviation value would reduce.

The existing difference in performance can be due to multiple factors:

- The different Preprocessing Method used for each source, *e.g.* approach, libraries, etc;
- The relevant information present in the Sections, *i.e.*, the number of information that is present in the Sections that is relevant therefore necessary to extract;
- The Data Verification and Combination Processes are directly related to the quantity of information extracted, *i.e.*, if more information is extracted more time is used in these processes;

But only comparing the average performance, 195 tokens/second (AICIS) vs 187 tokens/second (SCCS), we can conclude that there is a difference in performance. In two documents where 30000 tokens are used for context (approximately average between all documents used), in the end the AICIS Assessments would be approximately 6 seconds faster.

7.1.2 Evaluation

As introduced in the beginning of this Chapter, the security advisor performed the evaluation in a Likert Scale using five options: "*Without Information*", "*Incorrect Information*", "*Incomplete Information*", "*Correct Information*" and "*Incorrect Context*". Also, an Input Field was used when the information extracted was evaluated as "*Incorrect Information*" or "*Incomplete Information*" in order to better complete the evaluation. From the 33 documents used in the evaluation, the security advisor performed 3057 evaluations in total, 2301 with the Likert Scale and 756 with the Input Field.

Starting with the results of the Likert Scale evaluation, Figure 7.7, multiple observations can be detected.

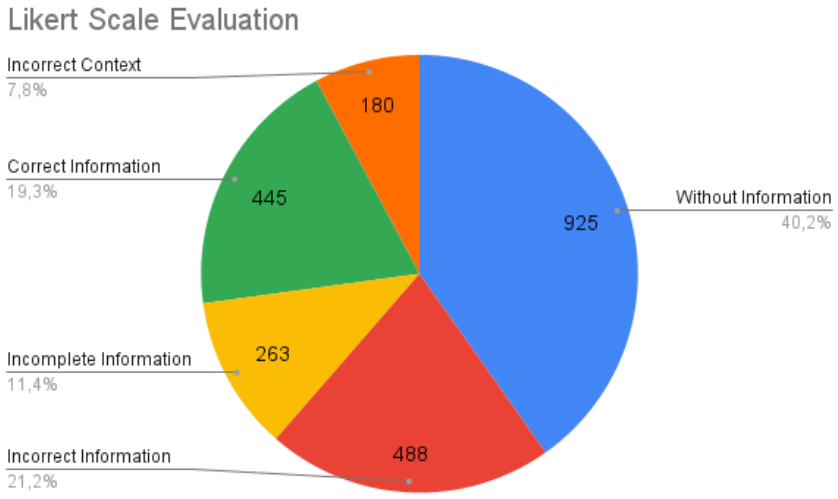


Figure 7.7: Likert Scale Evaluation Results

Firstly, the security advisor found 180 extractions where the wrong context was used and analysing the "*formSubmission.csv*" file we found that 88 where from AICIS Assessments documents and 92 from SCCS Opinions. Those 88 extractions from the AICIS file came from 18 different Sections and the 92 extractions from the SCCS Opinions from 8 different Sections.

In total, our Preprocessing Phase extracted in total 141 Sections from the 18 AICIS Assessments and 139 Sections from the 15 SCCS Opinions, meaning that our Preprocessing Phase failed 12.7% in the AICIS Assessments and 5.7% in the SCCS Opinions. As mentioned throughout the document the Preprocessing Phase is essential for the remaining Phases to work, so failing 9.3%(26 of the 280 Sections) is not a good of enough job. Although those errors, as explained in Chapter 4.5, could occur but some improvements need to be implemented in order to improve the performance of the Preprocessing Phase.

Using the remaining 2121 evaluations we can perform an evaluation of our remaining IE pipeline, *i.e.*, Phase 2 and Phase 3. For that evaluation we need to understand the remaining values of the security advisor evaluation and with them build a confusion matrix in order to achieve a quantitative evaluation. A confusion matrix uses the number of *True Positive (TP)*, *False Positive (FP)*, *False Negative (FN)* and *True Negative (TN)*. In the context of this work regarding IE, those are defined as:

- *TP*: There is information in the document to be extracted and the information extracted is correct;
- *FP*: There is no information in the document to be extracted but there is some information extracted or there is information in the document to be extracted but the information extracted is not correct;
- *FN*: There is information in the document to be extracted but there is no information extracted;
- *TN*: There is no information in the document to be extracted and there is no information extracted;

But we cannot directly convert the Likert Scale results into the confusion matrix because we firstly need to deeply analyse the results obtained. From those defined as "*Incorrect Information*" we need to analyse which ones are *FPs* and *FNs*. We assume that the *TPs* directly correspond to the "*Correct information*" and the *TNs* are the "*Without Information*". Regarding the "*Incomplete Information*" we need to analyse the document because that is the grey area where some limitations of our implementations are displayed.

Our implementation has the main objective to provide the security advisor with the most precise information extracted, that is the base of development of our Combination Process, where we just return the information extracted from multiple models, so in that way we attain a certain level of confidence. However, just returning information the information obtained from multiple models causes a problem of not returning all the information extracted, and that is directly visible in this evaluation in the "*Incomplete Information*" evaluations. The "*Incomplete Information*" evaluations are considered incomplete for two main reasons:

- The information extracted can be subject to subjectivity (examples Table 7.1);
- The information extracted is not complete (examples Table 7.2);

Information Extracted	Input Field Answer
the chemical was not sensitising skin sensitisation	The chemical does not produce
The chemical was considered negative for photo - sensitisation or a photosensitiser.	the chemical is not a sensitiser
In vivo studies : In an in vivo study conducted in rats (strain unspecified)	In vitro studies
two-generation combined reproductive toxicity (according to EPA OPPTS 870.3800) and developmental neurotoxicity (according to EPA OPPTS 83-6) study	repeated inhalation toxicity studies

Table 7.1: Examples of "*Incomplete Information*" evaluations where the information extracted is subjective to the evaluator

Information Extracted	Input Field Answer
Salmonella typhimurium	Salmonella typhimurium (S. typhimurium) strains TA 1535, TA 1537, TA 98, TA 100 and Escherichia coli (E. coli) WP2; Chinese hamster lung cell line (CHL); Chinese hamster lung fibroblasts (V29)
New Zealand White rabbits	New Zealand White rabbits; Wistar rats
Wistar	Wistar rats
Fischer 344 (F344) rats	Mice; Fischer 344 (F344) rats
albino rabbits	guinea pigs; mice; albino rabbits
1800 mg / kg bw 848 mg/kg; 1600 mg/kg bw	1800 mg / kg bw; 1270 mg/kg; 800 mg/kg;

Table 7.2: Examples of "*Incomplete Information*" evaluations where the information extracted is not complete

Regardless of the reason, we face a problem in our evaluation: how to we consider the "*Incomplete Information*" in our confusion matrix. We can face this problem in three different ways:

1. Consider the "*Incomplete Information*" as *FPs*;
2. Consider them as *TPs*;
3. Do not consider the "*Incomplete Information*" evaluations;

	AICIS	SCCS	**Total**
TP	184	261	**445**
FP	179	287	**466**
FN	184	101	**285**
TN	512	413	**925**
Precision	0.51	0.48	**0.49**
Recall	0.50	0.72	**0.61**
Accuracy	0.66	0.63	**0.65**
F-Score	0.50	0.57	**0.54**

Table 7.3: Results considering "*Incomplete Information*" as *FPs*

Considering the "*Incomplete Information*" evaluations as *FPs*, results present in Table 7.3, is not correct because the information extracted is actually correct but simply is not with all the information 100% extracted. This results in the evaluation of our work to achieve very low values in the Precision and in consequence the F-Score. Also, this totally evaluates our approach in the inverse way intended, because considering the "*Incomplete Information*" evaluations as *FPs* is the contrary of the intended work of the Combination Process Created.

	AICIS	SCCS	**Total**
TP	184	261	**445**
FP	71	132	**203**
FN	184	101	**285**
TN	512	413	**925**
Precision	0.72	0.66	**0.69**
Recall	0.50	0.72	**0.61**
Accuracy	0.73	0.74	**0.74**
F-Score	0.59	0.69	**0.65**

Table 7.4: Results not considering "*Incomplete Information*"

Not considering the ´*Incomplete Information*" evaluations at all, Table 7.4 results, is also, in our opinion, not validating the intention of the work developed. Our Intention was to try to provide the security advisor with just the correct information extracted.

	AICIS	SCCS	**Total**
TP	292	416	**708**
FP	71	132	**203**
FN	184	101	**285**
TN	512	413	**925**
Precision	0.81	0.76	**0.78**
Recall	0.61	0.81	**0.71**
Accuracy	0.76	0.78	**0.77**
F-Score	0.70	0.78	**0.74**

Table 7.5: Results considering "*Incomplete Information*" as *TPs*

If we consider the "*Incomplete Information*" evaluations as *TPs*, Table 7.5 results, we obtain better results, not amazing results but better results than the two previous considerations.

Comparing the results with the preliminary experiments mentioned in Chapter 5.5 we can detect that, in comparison with the experimental implementation of the Combination Process results, the results obtained in this evaluation are smaller, but in comparison with the individual models, the evaluation produced very similar results.

In order to further understand in which Sections we need to work more, we evaluated each Sections of the Documents, Tables 7.6 and 7.7, also considering the ´*Incomplete Information*" evaluations as *TPs*.

	Precision	Recall	Accuracy	F-Score
Dermal / percutaneous absorption	0.64	0.88	0.73	0.74
Acute Toxicity	0.82	0.76	0.86	0.79
Irritation and Corrosivity	0.76	0.80	0.74	0.78
Skin Sensitisation	0.85	0.78	0.75	0.81
Mutagenicity	0.73	0.84	0.73	0.78
Carcinogenicity	0.64	0.69	0.84	0.67
Photo-induced Toxicity	0.60	0.58	0.81	0.59
Reproductive Toxicity	0.69	0.73	0.73	0.71
Repeated dose toxicity	0.83	0.97	0.82	0.90
Safety Evaluation	0.67	0.91	0.70	0.77
Micro Average	0.76	0.80	0.78	0.78
Macro Average	0.73	0.80	0.77	0.76

Table 7.6: SCCS Opinions Sections evaluation results

	Precision	Recall	Accuracy	F-Score
Toxicokinetics	1.00	0.63	0.91	0.77
Acute Toxicity	0.74	0.66	0.81	0.70
Corrosion / Irritation	0.89	0.50	0.69	0.64
Sensitisation	0.81	0.57	0.71	0.67
Repeated Dose Toxicity	0.91	0.82	0.91	0.86
Genotoxicity	0.69	0.59	0.58	0.64
Carcinogenicity	0.73	0.67	0.72	0.70
Reproductive and Developmental Toxicity	0.84	0.59	0.62	0.70
Micro Average	0.80	0.61	0.76	0.70
Macro Average	0.82	0.63	0.74	0.71

Table 7.7: AICIS Assessments Sections evaluation results

From the analysis of the evaluation in each Sections we can assess those that produce better results and those that need further fine-tuning, *i.e.*, from the SCCS Opinions we achieved solid results in Sections "Skin Sensitisation" and "Repeated dose toxicity", and in the AICIS Assessments the better results were present in the "Toxicokinetics" and "Repeated Dose Toxicity" Sections. On the other hand, in the Section "Photo-induced Toxicity", from the SCCS Opinions, the results were subpar.

For the final analysis, we use the three models, *i.e.*, Bidirectional Encoder Representations from Transformers (BERT), BioBERT: a pre-trained biomedical language representation model for biomedical text mining (BioBERT) and Robustly Optimized BERT Pretraining Approach (RoBERTa), individually with the sames files evaluated, instead of using the combination process. With this, we can evaluate the models individually and compare the results with the combination process. In order to do so, we compare the information extracted from each model for the different toxicological properties with the results obtained from the evaluation of the security advisor with the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) score, *i.e.*, the information extracted returned by the combination process and the inputs provided by the security advisor when the information was incorrect or incomplete.

That process of comparing the answers extracted from the individual models with the answers from the combination process and the inputs provided by the security advisor was implemented by having the following steps: (1) comparing the answers extracted with

the answers from the Likert evaluation, and if there is a answer for the same property information, then compare them and if the ROUGE score is higher than 0.6, then consider the Likert input given by the security advisor. (2) If the first step fails, then verify if the security advisor introduced information in the input field, *i.e.*, in the cases that the information extracted was incorrect or incomplete, and compare the answers as in the previous step. In the third and final step, (3) we verified if the remaining answers matched with information of properties extracted, *i.e.*, toxicological property and their details, meaning that they are incorrect extractions.

We obtained 1216 answers from the BERT model, 2162 from the BioBERT model and 2218 from the RoBERTa model. As the combination process just returns the answers that at least two models have in common, in this individual evaluation some of those answers did not have any correspondence with the answers returned by the combination process. Specifically, 97 answers from the BERT, 187 from the BioBERT and 147 from the RoBERTa that did not have any match with the answers obtained from the evaluation process. Those answers without matches represent the information that is excluded by the combination model.

	BERT			BioBert			RoBERTa		
	AICIS	SCCS	**Total**	AICIS	SCCS	**Total**	AICIS	SCCS	**Total**
TP	98	170	**268**	262	323	**585**	222	298	**520**
FP	69	167	**236**	266	456	**722**	189	266	**455**
FN	167	99	**266**	141	53	**194**	307	148	**455**
TN	153	196	**349**	248	226	**474**	375	266	**641**
Precision	0.59	0.50	**0.53**	0.50	0.41	**0.45**	0.54	0.53	**0.53**
Recall	0.37	0.63	**0.50**	0.65	0.86	**0.75**	0.42	0.67	**0.53**
Accuracy	0.52	0.58	**0.55**	0.56	0.52	**0.54**	0.55	0.58	**0.56**
F-Score	0.45	0.56	**0.52**	0.56	0.56	**0.56**	0.47	0.59	**0.53**

Table 7.8: Individual Model Results

Table 7.8 shows the results obtained from the models individually where low performances of all three are visible. The low overall performance of the individual models is due to some factors.

The biggest factor, and that has a direct impact in the results, is the comparison that we needed to do between the answers extracted by the models individually and the answers evaluated by the security advisor. Although is a possible comparison, there are no doubts that the evaluation done by the security advisor directly to the answers of the combination process is more precise. In order to make the evaluation of the models individually we should have added more information in our files, specifically, which models returned which answer.

If we did not consider the answers mentioned in the third step of the comparison, *i.e.*, the answers that do not have a direct comparison with neither the inputs of the Likert and Input Field, then the results would be more similar to the one expected, as shown in Table 7.9.

Just considering the answers with direct comparison and consequently the expected results of the models, the comparison of the performances between the models individually and the Combination Process used, as show in Table 7.10, is much fairer. The considerable highlight of the Combination Process used is an higher precision obtained compared with the individual models. On the other hand, a considerable drop off in recall.

	BERT			BioBert			RoBERTa		
	AICIS	SCCS	**Total**	AICIS	SCCS	**Total**	AICIS	SCCS	**Total**
TP	98	170	**268**	262	323	**585**	222	298	**520**
FP	42	76	**118**	133	136	**269**	93	93	**186**
FN	57	32	**89**	80	25	**105**	140	66	**206**
TN	153	196	**349**	248	226	**474**	375	266	**641**
Precision	0.70	0.69	**0.69**	0.66	0.70	**0.69**	0.70	0.76	**0.74**
Recall	0.63	0.84	**0.75**	0.77	0.93	**0.85**	0.61	0.82	**0.72**
Accuracy	0.72	0.77	**0.75**	0.71	0.77	**0.74**	0.72	0.68	**0.75**
F-Score	0.66	0.76	**0.72**	0.71	0.80	**0.76**	0.66	0.79	**0.73**

Table 7.9: Individual Model Results just considering direct comparisons

Comparing the F-Scores of the individual models and the Combination Process, we can affirm that the F-Score of the Combination Process is not groundbreaking, much due to the less good result of the recall.

	BERT	BioBERT	RoBERTa	Combination Process
Precision	0.69	0.69	0.74	0.78
Recall	0.75	0.85	0.72	0.71
Accuracy	0.75	0.74	0.75	0.77
F-Score	0.72	0.76	0.73	0.74

Table 7.10: Comparison of Individual Models and Combination Process

7.2 Discussion

Having the results all presented and processed we can highlight some quick points that, in our opinion, deserve to be further discussed and addressed. In terms of our evaluation process, we think that, given the context of evaluating information extracted from documents, we presented a solid webpage that provided the evaluator, in our case the security advisor, with a simple user interface as well as a way of evaluating the information extracted without our bias.

The Processing Phase, as mentioned throughout the document, is one of the pillars in this work, but as explained in Chapter 4.5, and as proven by an efficiency of 90.7%, there is almost 10% of sections that the Preprocessing is missing to identify correctly. This suggests that, although a solid job, the margin for error in Preprocessing phase should be as minimal as possible.

Regarding the Phase 2 and Phase 3, and analysing the results considering the "*Incomplete Information*" evaluations as *TPs*, in Tables 7.5, 7.6 and 7.7, we can affirm that although we are not achieving in this evaluation the results that we hoped for and expected, we are still with solid results in terms of global overview. The main result that is unexpectedly lower is the recall. We were waiting for a recall value more similar to the one obtained in the preliminary experiments that we conducted, *i.e.*, approximately 90% and we ended up with a recall of 70%.

In terms of precision and accuracy obtained we achieved similar results, approximately

80%, very similar to the results obtained in the preliminary experimentation conducted. Not groundbreaking results but we consider that the results obtained demonstrate that this approach is viable if given the right variables, *i.e.*, the Preprocessing Phase must be on point in order to precisely extract the right contexts. Although this performance is not critical, because it continues to speed up the process for the security advisor, who can always examine the results.

Some questions that came up from the results obtained are: (1) why not just use the RoBERTa model since is the most balanced model in all performance terms? (2) The Combination Process has any advantages over the individual models?

Using only the RoBERTa model has the advantage of a faster execution time, as well as the fact that the performance results obtained were very solid, making the use of the RoBERTa model a strong possibility. However, the Combination Process has some advantages over just using the RoBERTa model, specifically, the Combination Process brings a level of confidence of the information extracted that only using one model cannot provide, and also, the Combination Process provided some improvements in terms of precision and accuracy over the individual models, which is an important point.

The final point that we wanted to address is the overall work developed as we implemented in one work various algorithms and technologies in order to achieve a final result that we feel somewhat proud of.

Chapter 8

Conclusion

Throughout this document we reported all the work developed during this long journey related to the development and implementation of the proposed pipeline for the Safety Desk project.

Safety Desk is a project, turned into a service, with the goals of extracting information regarding toxicological properties of chemical substances and, with the information extracted, create a toxicological profile of the substance. To turn the project into a service we needed to make use of Natural Language Processing (NLP) technologies and techniques in conjunction with the integration of web application technologies. The final pipeline consists of five processing phases.

The first phase is based in Preprocessing steps, where regular expressions and multiple python libraries are used to try to minimize the context given to the Extractive Question Answering (QA) models, eliminating noise and optimizing the contexts. This phase was evaluated, performing correctly 90% of the times, which demonstrates that improvements need to be made. We identified and described the problems found and recognize that improvements need to be implemented in order to achieve a better performance in this phase.

The second phase is based on extractive QA models, where using the Bidirectional Encoder Representations from Transformers (BERT), BioBERT and Robustly Optimized BERT Pretraining Approach (RoBERTa) Transformers fine-tuned tries to extract the information from the context given. The third phase is a cleaning phase where, using similarity measures, specifically the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) Score we resolve conflicts coming from the second phase of the pipeline, trying to clean repeated information extracted and obtain the correct information. Evaluations were performed also for this phase where our Information Extraction (IE) process achieved a 0.74 F-Score(0.78 Precision, 0.71 Recall and 0.77 Accuracy). We identified that the recall is lower than the expected and that is due to the Combination Process present in the third phase.

The fourth phase is based in Data-to-text (D2T) using templates where a toxicological profile is generated from the information obtained from the previous phase. The toxicological profile generated provides a quick way to humans, without expertise in the domain, to acquire a little information and knowledge about that particular substance. As the toxicological profile created just has into account one file, the document used in the extraction of the information, the text generated is reduced, so in order to integrate the toxicological profile in the Cosmedesk platform, the implementation must be on the side of

the Cosmedesk platform and not in the Safety Desk side in order to have access to the full data regarding the chemical substance, *i.e.*, information extracted from multiple document sources.

The last phase of the pipeline consists of a Rest API in order to integrate the Safety Desk service capacities, *i.e.*, extract toxicological information from documents, at the moment implemented for the Scientific Committee on Consumer Safety (SCCS) Opinions, Australian Industrial Chemicals Introduction Scheme (AICIS) Health Assessments and Agency for Toxic Substances and Disease Registry (ATSDR) Reports.

In terms of prospects of future work, as mentioned, there are improvements needed to be put in place, specifically in the Preprocessing Phase and in the Data Verification Phase. As the improvements are resolved, the focus of the future work is the development of the Preprocessing Phase for other sources, namely Cosmetic Ingredient Review (CIR) and Organization for Economic Cooperation and Development (OECD) documents.

References

A., A. (2022). Automating information extraction with question answering. `https://www.deepset.ai/blog/automating-information-extraction-with-question-answering`, accessed: 10/05/2022.

Abdelmagid, M., Himmat, M., Ahmed, A., and KANNAN, R. (2014). Survey on information extraction from chemical compound literatures: Techniques and challenges. *Journal of Theoretical and Applied Information Technology*, 67(2):284–289.

Allam, A. M. N. and Haggag, M. H. (2012). The question answering systems: A survey. *International Journal of Research and Reviews in Information Sciences (IJRRIS)*, 2(3).

Arici, T., Kumar, K., Çeker, H., Saladi, A. S., and Tutar, I. (2022). Solving price per unit problem around the world: Formulating fact extraction as question answering. *arXiv preprint arXiv:2204.05555*.

Bahja, M. (2020). Natural language processing applications in business. In *E-Business-Higher Education and Intelligence Applications*. IntechOpen.

Baradaran, R., Ghiasi, R., and Amirkhani, H. (2020). A survey on machine reading comprehension systems. *arXiv preprint arXiv:2001.01582*.

Braşoveanu, A. M. P. and Andonie, R. (2020). Visualizing transformers for nlp: A brief survey. In *2020 24th International Conference Information Visualisation (IV)*, pages 270–279.

Celikyilmaz, A., Clark, E., and Gao, J. (2020). Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*.

Chandrasekaran, D. and Mago, V. (2021). Evolution of semantic similarity—a survey. *ACM Comput. Surv.*, 54(2).

Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., and Bengio, Y. (2015). Attention-based models for speech recognition. *arXiv preprint arXiv:1506.07503*.

Chowdhury, G. G. (2003). Natural language processing. *Annual review of information science and technology*, 37(1):51–89.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Commission, E. (2022). Scientific committee on consumer safety. `https://health.ec.europa.eu/scientific-committees/scientific-committee-consumer-safety-sccs_en`, accessed: 19/04/2022.

Cristina, S. (2021). The transformer attention mechanism. `https://machinelearningmastery.com/the-transformer-attention-mechanism/`, accessed: 15/12/2021.

Cvitaš, A. (2010). Information extraction in business intelligence systems. In *The 33rd International Convention MIPRO*, pages 1278–1282.

Deshmukh, R. D. and Kiwelekar, A. (2020). Deep learning techniques for part of speech tagging by natural language processing. In *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, pages 76–81. IEEE.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dhingra, B., Zhou, Z., Fitzpatrick, D., Muehl, M., and Cohen, W. W. (2016). Tweet2vec: Character-based distributed representations for social media. *arXiv preprint arXiv:1605.03481*.

Dong, C., Li, Y., Gong, H., Chen, M., Li, J., Shen, Y., and Yang, M. (2021). A survey of natural language generation. *arXiv preprint arXiv:2112.11739*.

Duan, N., Tang, D., Chen, P., and Zhou, M. (2017). Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874.

Dzunic, Z., Momcilovic, S., Todorovic, B., and Stankovic, M. (2006). Coreference resolution using decision trees. In *2006 8th Seminar on Neural Network Applications in Electrical Engineering*, pages 109–114. IEEE.

Entwisle, J. and Powers, D. M. (1998). The present use of statistics in the evaluation of nlp parsers. In *New Methods in Language Processing and Computational Natural Language Learning*.

Ferreira, B. C. L., Gonçalo Oliveira, H., Amaro, H., Laranjeiro, A., and Silva, C. (2022). Question Answering For Toxicological Information Extraction. In Cordeiro, J. a., Pereira, M. J. a., Rodrigues, N. F., and Pais, S. a., editors, *11th Symposium on Languages, Applications and Technologies (SLATE 2022)*, volume 104 of *Open Access Series in Informatics (OASIcs)*, pages 3:1–3:10, Dagstuhl, Germany. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

Gardner, M. and Mitchell, T. (2015). Efficient and expressive knowledge base completion using subgraph feature extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1488–1498.

Gatt, A. and Krahmer, E. (2018). Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Giménez, J. and Marquez, L. (2004). Svmtool: A general pos tagger generator based on support vector machines. In *In Proceedings of the 4th International Conference on Language Resources and Evaluation*. Citeseer.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets in advances in neural information processing systems (nips).

Greco, C., Suglia, A., Basile, P., Rossiello, G., and Semeraro, G. (2017). Iterative multi-document neural attention for multiple answer prediction. *arXiv preprint arXiv:1702.02367*.

Grishman, R. (2015). Information extraction. *IEEE Intelligent Systems*, 30(5):8–15.

Gui, L., Hu, J., He, Y., Xu, R., Lu, Q., and Du, J. (2017). A question answering approach to emotion cause extraction. *arXiv preprint arXiv:1708.05482*.

Guo, J., Xu, G., Cheng, X., and Li, H. (2009). Named entity recognition in query. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 267–274.

Gupta, P. (2019). *PhD Thesis: Neural Information Extraction From Natural Language Text*. PhD thesis.

He, J., Nguyen, D. Q., Akhondi, S. A., Druckenbrodt, C., Thorne, C., Hoessel, R., Afzal, Z., Zhai, Z., Fang, B., Yoshikawa, H., et al. (2021). Chemu 2020: Natural language processing methods are effective for information extraction from chemical patents. *Frontiers in Research Metrics and Analytics*, 6:654438.

Herbert, L. (2017). *Digital transformation: Build your organization's future for the innovation age*. Bloomsbury Publishing.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Kong, F., Zhou, J., Zhou, G., and Zhu, Q. (2010). Dependency tree-based anaphoricity determination for coreference resolution. In *2010 International Conference on Asian Language Processing*, pages 215–218. IEEE.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Li, F., Peng, W., Chen, Y., Wang, Q., Pan, L., Lyu, Y., and Zhu, Y. (2020). Event extraction as multi-turn question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 829–838.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Lin, Y., Shen, S., Liu, Z., Luan, H., and Sun, M. (2016). Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133.

Ling, X. and Weld, D. S. (2010). Temporal information extraction. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Liu, S., Zhang, X., Zhang, S., Wang, H., and Zhang, W. (2019a). Neural machine reading comprehension: Methods and trends. *Applied Sciences*, 9:3698.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019b). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

McCann, B., Bradbury, J., Xiong, C., and Socher, R. (2017). Learned in translation: Contextualized word vectors. *arXiv preprint arXiv:1708.00107*.

Miller, D. R., Leek, T., and Schwartz, R. M. (1999). A hidden markov model information retrieval system. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 214–221.

Min, S., Seo, M., and Hajishirzi, H. (2017). Question answering through transfer learning from large fine-grained supervision data. *arXiv preprint arXiv:1702.02171*.

Minh-Tien Nguyen, Viet-Anh Phan, L. T. L. (2020). Transfer learning for information extraction with limited data. *CINNAMON LAB*.

Mohit, B. (2014). Named entity recognition. In *Natural language processing of semitic languages*, pages 221–245. Springer.

Ng, V. and Cardie, C. (2002). Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 104–111.

Nguyen, D. Q., Zhai, Z., Yoshikawa, H., Fang, B., Druckenbrodt, C., Thorne, C., Hoessel, R., Akhondi, S. A., Cohn, T., Baldwin, T., et al. (2020a). Chemu: named entity recognition and event extraction of chemical reactions from patents. In *European conference on information retrieval*, pages 572–579. Springer.

Nguyen, M.-T., Le, D. T., and Le, L. (2021a). Transformers-based information extraction with limited data for domain-specific business documents. *Engineering Applications of Artificial Intelligence*, 97:104100.

Nguyen, M.-T., Le, D. T., Linh, L. T., Hong Son, N., Duong, D. H. T., Cong Minh, B., Hai Phong, N., and Huu Hiep, N. (2020b). Aurora: An information extraction system of domain-specific business documents with limited data. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 3437–3440.

Nguyen, M.-T., Le, D. T., Son, N. H., Minh, B. C., et al. (2020c). Understanding transformers for information extraction with limited data. In *Proceedings of the 34th Pacific Asia Conference on Language, Information and Computation*, pages 478–487.

Nguyen, M.-T., Le, D. T., Son, N. H., Minh, B. C., Shojiguchi, A., et al. (2021b). Information extraction of domain-specific business documents with limited data. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Nguyen, M.-T., Phan, V.-A., Linh, L. T., Son, N. H., Dung, L. T., Hirano, M., and Hotta, H. (2019). Transfer learning for information extraction with limited data. In *International Conference of the Pacific Association for Computational Linguistics*, pages 469–482. Springer.

Nigam, K., Lafferty, J., and McCallum, A. (1999). Using maximum entropy for text classification. In *IJCAI-99 workshop on machine learning for information filtering*, volume 1, pages 61–67. Stockholom, Sweden.

Organization, H. F. (2021a). Attention mask. `https://huggingface.co/docs/transformers/glossary#attention-mask`, accessed: 12/12/2021.

Organization, H. F. (2021b). How do transformers work? `https://huggingface.co/course/chapter1/4`, accessed: 10/12/2021.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Pawar, S., Palshikar, G. K., and Bhattacharyya, P. (2017). Relation extraction: A survey. *arXiv preprint arXiv:1712.05191*.

Rajpurkar, P., Jia, R., and Liang, P. (2018). Know what you don't know: Unanswerable questions for squad.

Reiter, E. and Dale, R. (1997). Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

Rijhwani, S., Zhou, S., Neubig, G., and Carbonell, J. (2020). Soft gazetteers for low-resource named entity recognition. *arXiv preprint arXiv:2005.01866*.

Sachan, M. and Xing, E. (2018). Self-training for jointly learning to ask and answer questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 629–640.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Singh, S. (2018). Natural language processing for information extraction. *arXiv preprint arXiv:1807.02383*.

Sun, C., Yang, Z., Wang, L., Zhang, Y., Lin, H., and Wang, J. (2021). Biomedical named entity recognition using bert in the machine reading comprehension framework. *Journal of Biomedical Informatics*, 118:103799.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Swain, M. C. and Cole, J. M. (2016). Chemdataextractor: a toolkit for automated extraction of chemical information from the scientific literature. *Journal of chemical information and modeling*, 56(10):1894–1904.

Tharatipyakul, A., Numnark, S., Wichadakul, D., and Ingsriswang, S. (2012). Chemex: information extraction system for chemical data curation. In *BMC bioinformatics*, volume 13, pages 1–11. Springer.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

von Platen, P. (2021). Transformers-based encoder-decoder models. `https://huggingface.co/blog/encoder-decoder#encoder-decoder`, accessed: 12/12/2021.

Wimalasuriya, D. C. and Dou, D. (2010). Ontology-based information extraction: An introduction and a survey of current approaches.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wołk, K. and Marasek, K. (2015). Enhanced bilingual evaluation understudy. *arXiv preprint arXiv:1509.09088*.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Yang, Z., Dhingra, B., Yuan, Y., Hu, J., Cohen, W. W., and Salakhutdinov, R. (2016). Words or characters? fine-grained gating for reading comprehension. *arXiv preprint arXiv:1611.01724*.

Zhang, X., Yang, A., Li, S., and Wang, Y. (2019). Machine reading comprehension: a literature review. *arXiv preprint arXiv:1907.01686*.

Zhu, F., Lei, W., Wang, C., Zheng, J., Poria, S., and Chua, T.-S. (2021). Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*.

Zimmermann, M., Fluck, J., Thi, L. T., Kolarik, C., Kumpf, K., and Hofmann, M. (2005). Information extraction in the life sciences: perspectives for medicinal chemistry, pharmacology and toxicology. *Current Topics in Medicinal Chemistry*, 5(8):785–796.