Warsaw University of Technology

FACULTY OF
MATHEMATICS AND INFORMATION SCIENCE

Bachelor's diploma thesis

in the field of study Data Science

AI regulations database with the analytical user interface module

Piotr Piątyszek

Student record book number 298833

Jakub Wiśniewski

Student record book number 298850

Hanna Zdulska

Student record book number 298852

thesis supervisor

dr hab. inż. Przemysław Biecek, prof. uczelni

WARSAW 2022

Abstract

As the need for regulating artificial intelligence (AI) increases, countries and non-governmental organizations produce more and more AI policies and regulations. For policymakers and data scientists, the number of documents, specific language, and structure may be overwhelming. Sometimes it might be hard even to know where to get particular documents. As our thesis, we wanted to tackle those problems. We introduce a high-quality database with an analytical user interface module. The number of documents in the database is continuously increased with AI regulations and policies from internationally recognized sources. The analytical module is available through a web page. Users can upload their documents, read them, view sentences connected with popular topics in the AI community, and explore definitions extracted with the NLP model. We believe that our work will benefit the key users by providing a web tool to analyze documents and give them access to a high-quality resource of raw documents for detailed analysis.

AI regulations database with the analytical user interface module

Keywords: Database, AI, regulation, policies, documents, keywords, definitions, scraping

Abstract

Wraz ze wzrostem potrzeby regulacji sztucznej inteligencji (AI), kraje i organizacje pozarządowe publikują coraz więcej polityk i regulacji dotyczących AI. Dla politologów oraz data scientistów ilość dokumentów, ich specyficzny język oraz struktura, może być przytłaczająca. Czasem samo znalezienie odpowiednich dokumentów może stanowić problem. W naszej pracy inżynierskiej chcieliśmy rozwiązać ten problem. Jako odpowiedź na ten problem przedstawiamy wysokiej jakości bazę danych wraz z analitycznym interfejsem użytkownika. Liczba dokumentów jest automatycznie powiększana poprzez dodawanie regulacji i polityk AI z międzynarodowo uznawanych źródeł. Moduł analityczny jest dostępny w postaci strony internetowej, gdzie użytkownicy mogą wgrywać dokumenty, czytać je, oglądać popularne tematy dotyczące AI oraz eksplorować definicje znalezione przez moduł NLP. Wierzymy, że nasza praca przyniesie korzyści kluczowym użytkownikom, poprzez udostępnienie narzędzia do analizy regulacji i polityk oraz da dostęp do bazy danych wysokiej jakości dokumentów w celach pogłębionych analiz i eksploracji.

Baza danych regulacji AI z analitycznym interfejsem użytkownika

Sowa kluczowe: Baza danych, sztuczna inteligencja, AI, regulacje, polityki, dokumenty, słowa kluczowe, definicje, scraping

Contents

Introduction

Artificial Intelligence (AI) will be a vital part of our life. It either is or will be incorporated into hospitals, road traffic, social media, and many other environments that we have contact with daily. AI is currently in power to solve our problems, but unfortunately, it can generate more.

On the one hand, AI can be seen as a solution to many of our problems. In many countries, there are not enough physicians and nurses. Incorporating AI into patient care could potentially alleviate this burden through various ways like limiting human error, making therapy more home-based, and analyzing the mental state of a patient Shaheen [2021]. AI is also helpful in many branches of pharmacy like drug discovery, where it can make this process easier, faster, and much less expensive Paul et al. [2021]. It can even measure the severity and enable the prevention of suicide based on text from social media Zhang et al. [2021]. AI is also becoming more omnipresent on the roads. Self-driving cars may increase safety and mobility and decrease carbon footprint by driving more optimal way Urmson and Whittaker [2008].

On the other hand, AI can generate problems, feedback loops, and create harm. One problem with the adoption of AI in healthcare facilities is checking if algorithms trained with local data are not discriminating underrepresented populations Panch et al. [2019]. Such unfairness may create feedback loops where the algorithm's decisions are affecting the natural world and aggravating the differences between privileged and unprivileged individuals Barocas et al. [2019]. In Stahl et al. [2016] authors, after examining 809 papers concerning computation systems, found that in 177 of them, there was the issue of privacy. They also identified trust, security, misuse, design, etc. This also applies to AI, where systems learn from data, and their decisions might not be inherently understandable by humans. In Chuan et al. [2019] after analyzing 2,485 AI newspaper articles from the United States, authors found out that the topic of AI ethics and morality was increasingly discussed through the years, including good and bad aspects of it. This shows that the threats of AI are known to regular people and therefore should be known and acknowledged by policymakers.

The paragraphs above show that there are risks and opportunities for AI. Therefore to minimize these threats and draw from the possibilities of AI, there is a need for regulation, and in

fact, countries and organizations have become to do just that. In 2019 42 countries adopted new OECD Principles on AI noa [2019] where recommendations and principles of safe and beneficial use of AI were stated. However, not always the motivations and actions of countries are unanimous. Authors of Gesley et al. had made a collection of different regional codes regarding AI policies and regulations. Their comparative summary has shown that there was much focus on regulating data privacy, transparency, surveillance, and lethal weapons, but the most advanced were regulations in autonomous vehicles. Their document covers policies from many countries, and it involved the work and expertise of many people from different backgrounds to aggregate this amount of knowledge.

Finding and analyzing such documents require expert knowledge. This knowledge may exceed a typical machine learning engineer or data scientist who creates such systems. This type of expertise takes much time, as it requires an in-depth document analysis. On the other hand, lawmakers may not have enough time for an in-depth analysis of all the documents. Therefore as our thesis, we propose a system for gathering, storing, and analyzing documents and policies about Artificial intelligence.

We present an overview of the solution to this problem in chapter 1. Furthermore, the contents of the thesis include the description of data in chapter 2 and database in chapter 4. We analyze the current state of the database and its contents. We then describe how we process the data. Then we go through our orchestration pipelines, describing the Airflow pipeline in chapter and Redis queue in 6. Later we describe models that analyze the sentences - definition extraction model and key issues model and their implementation and evaluation.

1. Proposed solution overview

The product of our work is a database of AI documents and policies and a user interface hosted on a website to enable intuitive interaction where a user may analyze the document.

In detail, our solution is based around the database, described in detail in chapter 2. The database is composed of documents from various sources and their metadata. Those documents are being scrapped weekly from the existing policy websites (EUR-Lex and OECD) (chapter 6). Then the text is extracted, and they are processed and segmented into sentences (chapter 3). In the end, it puts the text, the segments, and the sentences into the database. The other more user-centered flow is to upload a document through UI (chapter 10), where it will be analyzed by models that extract definitions and topics (chapter 7). The flow of data can be seen in figure 1.1.

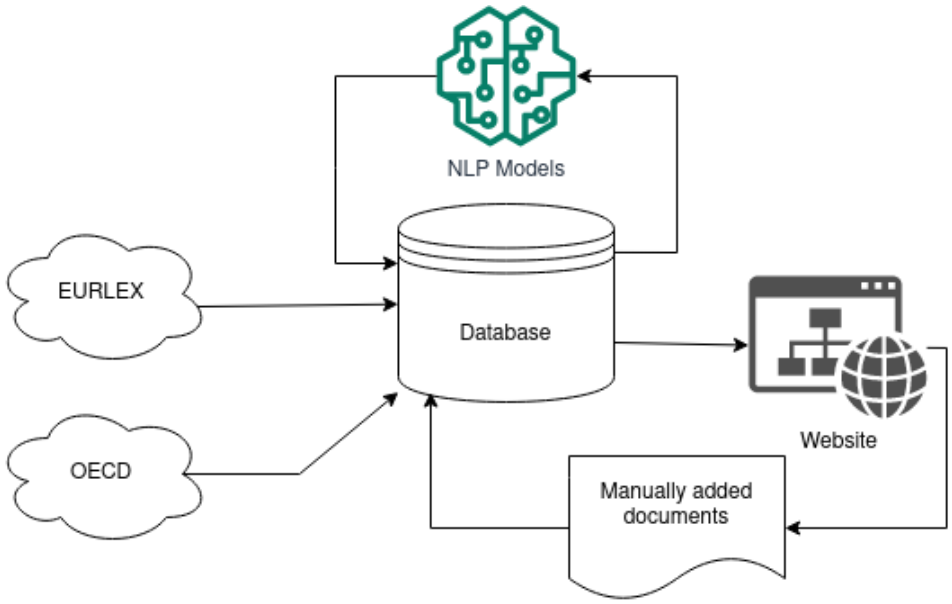

Figure 1.1: Flow of data in the system. Documents from OECD, EUR-Lex, and those manually added will be stored in the database. Machine learning models will be taking texts from the database, and predictions regarding topics and definitions will be made. Then those predictions will enrich documents metadata. Later the acquired labels will be presented along with documents on the website.

2. Acquisition and exploration of data

Many countries publish their policies and national strategies on the government website. However, each website has a different design and formatting. To obtain documents at their source would require either developing scrapers for each of them to get correct data or attempting to create a universal scraper, but with chances of getting incorrect documents. That is why we decided to use websites that aggregate said documents and share them publicly.

Two main sources for policies and documents are:

1. EUR-Lex.europa.eu - a source of European Union Law. As stated in their main website: *EUR-Lex is your online gateway to EU Law. It provides the official and most comprehensive access to EU legal documents. It is available in all of the EU's 24 official languages and is updated daily.* Many types of documents are available at the site: treaties, legal acts, EU case-law, etc. The records from their API have their metadata like data of adoption and entry, country of origin, and many more.

2. OECD.ai - AI Policy Observatory. As stated at their website: *The OECD AI Policy Observatory (OECD.AI) builds on the momentum of the OECDs Recommendation on Artificial Intelligence (OECD AI Principles) the first intergovernmental standard on AI adopted in May 2019 by OECD countries and adhered to by range of partner economies. The OECD AI Principles provided the basis for the G20 AI Principles endorsed by Leaders in June 2019. OECD.AI combines resources from across the OECD, its partners and all stakeholder groups. OECD.AI facilitates dialogue between stakeholders while providing multidisciplinary, evidence-based policy analysis in the areas where AI has the most impact.*. As in the previous case, the API returns metadata about the policy apart from the document content.

We believe that such a combination of sources will give a holistic view of AI policies and their development.

2.1. Methodology of data retrieval

Using python libraries - selenium and requests we wrote scripts, that automatically download raw documents. Some scripts download from

1. EUR-Lex - using selenium, we can select keywords- "Artificial intelligence" or "AI" and languages - English. Using regexes, we obtain CELEX numbers - unique identifiers of each document in EUR-Lex. Later, using those CELEX numbers, we can easily access them.

   Using EUR-Lex API, we obtain metadata about each document.

2. Oecd.ai - using Oecd API, we can obtain all metadata about their documents, including public access URLs. Using selenium, we download those documents.

Both scripts use scraper build using selenium, which handles errors and 404 response codes gracefully by taking a screenshot and saving the website's source code that caused a given situation - diagram can be seen in Figure 2.1.

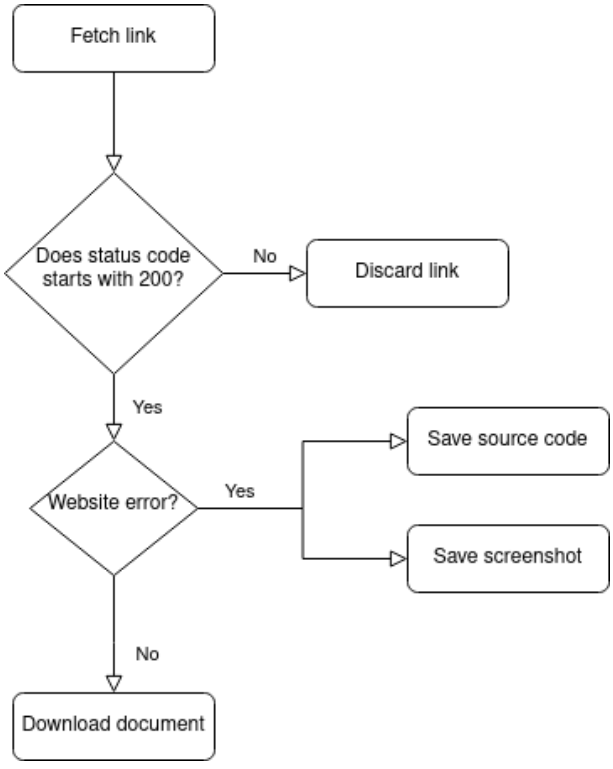

Figure 2.1: Diagram of error handling in scraper.

1. OECD.AI does not make its API public. However, after inspection, we saw that the website sent a GET request to link in domain api.oecd.ai. For example, to see a list of Polish

initiatives, the browser would send a query `country=Poland` [1]. Following this pattern, we can obtain initiatives from all countries sending GET requests without defining a specific country [2]. Respond links to original documents and metadata like country, author, and publication date of the document. The downside is that some of the referenced documents are no longer available, or the link provided is incorrect and returns status code 404 - not found.

2. EUR-Lex.europa.eu offers a search engine to find specific documents - users can input the desired country, language, date of publication, keywords, and others. We use the keyword "Artificial Intelligence" and English language. EUR-Lex uses CELEX numbers to identify documents. To see the manual for navigating on EUR-Lex, see Office [2018] These numbers give access to the full text of the document in HTML under a specific link. For example document with CELEX number 52019IP0081 - A comprehensive European industrial policy on artificial intelligence and robotics - can be read under known link [3]. Metadata can be similarly obtained via EUR-Lex API by sending GET requests.

In the project's first iteration, documents were saved to the local machine, and their location was uploaded to the databasehowever, this created problem. It occurred when the script was executed on a machine other than the database host. Files were saved locally and not on the database machine. To solve this problem, we decided to use WebDAV Whitehead and Goland [1999], which allowed us to read and write remote files using the python library. Later they are parsed and split into segments and sentences as described in chapter 3.

In the second iteration, we started downloading the data continuously from OECD and EUR-Lex and saving them in the database on the server that we set up. After setting up this step, we ended up with around 6,300 documents. However, some were empty, obsolete, or invalid (with a different tag). We then verified this, filtered out the inadequate documents, and ended up with 199 documents in the database and correct scraping and pre-processing steps.

2.2. Exploration of source data

Throughout our work, we have gathered a lot of documents and data. This section aims to describe them in detail. In our work, we used the following data:

[1] https://api.oecd.ai/ws/AIPO/API/dashboards/policyInitiatives.xqy?conceptUris=undefined&country=Poland

[2] https://api.oecd.ai/ws/AIPO/API/dashboards/policyInitiatives.xqy?conceptUris=undefined

[3] http://publications.europa.eu/resource/celex/52019IP0081

1. https://eur-lex.europa.eu - 5730 legal documents were obtained up to this date, all in HTML file format.

2. https://oecd.ai - we obtained a metadata list of 705 documents. However, they were missing correct prefixes like "www" or "HTTP" in some cases. What is more, URLs were outdated or broken beyond repair in some cases. Five hundred thirty-two documents were obtained in either PDF files or HTML files. Number of documents from each country can be seen in Figure 2.2.

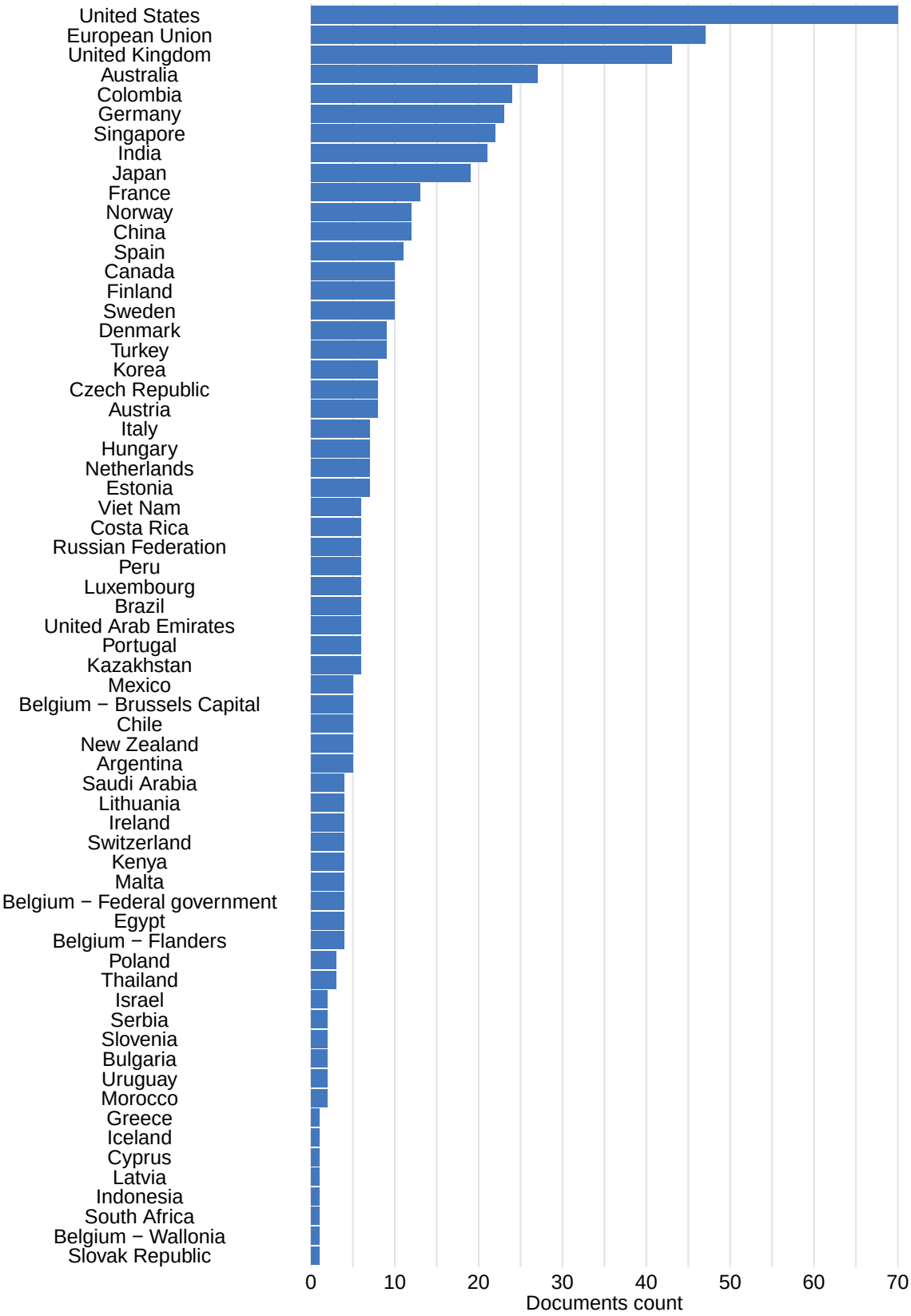

Figure 2.2: Number of documents from each country from oecd.ai. Belgium has multiple occurrences, because each of Belgium's regions has it's own independent government.

3. Processing of documents text

In this chapter, we described each step in our data processing pipeline. Firstly, raw text is extracted from the document, then it is split into smaller pieces, and in the end, our models analyze its contents. We described each step in detail in the sections below and linked our GitHub repository where steps are implemented.

Each document is split into segments 3.3 and sentences 3.1. Firstly document is split into segments, and then each segment is split into sentences. Information about the order of segments and sentences is saved to the database.

Definition 3.1. Sentence - a group of words, usually containing a verb, that expresses a thought in the form of a statement, question, instruction, or exclamation and starts with a capital letter when written.[1]

Definition 3.2. Header - a group of words in a document, written in a separated line using a bigger font. They indicate a topic of the text below it.

Definition 3.3. Segment - text between two headers.

3.1. Extracting text segments from documents

There are two file types of documents in the database - HTML files and PDF files. To split them into headers and segments, we use different strategies. HTML files are already split, tagged as headers and paragraphs. There is no universal way to split PDF files. Therefore we determine the most used font size and treat it as a paragraph. Bigger fonts are treated as headers. [2]

[1] Definition by Cambridge Dictionary

[2] The implementation of this module is available at mars/segmentation/segmentation.py

3.2. Partitioning segments into sentences

To split a text into sentences, we used the English pipeline from the Python package spaCy by Montani et al. [2020]. It has knowledge about the English language, punctuation, and grammar. This library provides information about where the former sentence ends and a new one begins. As stated in spaCy documentation:

> The dependency parser jointly learns sentence segmentation and labeled dependency parsing and can optionally learn to merge tokens over-segmented by the tokenizer. The parser uses a variant of the non-monotonic arc-eager transition-system described by Honnibal and Johnson [2015], with the addition of a break transition to performing the sentence segmentation. Nivre and Nilsson [2005]s pseudo-projective dependency transformation is used to allow the parser to predict non-projective parses.

We have to clean the text from residue characters initially. We split the raw text into a list of strings and used the spaCy pipeline to determine sentences. [3]

3.3. Curating documents by their relevance

At this point, we have many documents that seem to be concerning AI. However, not all of them can be analyzed and used by the end-user from a technical and substantive point of view. For example, not every 'AI' stands for 'Artificial Intelligence' - almost 5 thousand documents contain the keyword 'AI', but not 'Artificial Intelligence' on EUR-Lex. To ensure the quality of our database, we decided to apply constraints to documents. The numbers of discarded documents in each of the steps described in steps below are presented in Figure 3.1.

1. Remove documents that have no segments or no sentences. We cannot avoid errors with unit-testing segmentation modules since our sources cannot always be trusted. Successful segmentation requires HTML websites to have a specific structure based on correct HTML tags, or websites will return the correct status code if the file was moved or no longer exists. Segmentation may also fail in the case of parsing PDF, which is a challenge described in detail by Bast and Korzen [2017].

2. Keep only documents with more than 20 sentences. Some documents that do not contain actual documents may still be parsed and put in the database - for example, oecd.ai can

[3] The implementation of this module is available at mars/sentences_splitting.py

provide a URL to the home page of an organization that works in the field of monitoring AI, but not to the specific document.

3. Remove documents with a small number of mentions of AI. In order to approximate how much each document concerns AI, we decided to see how many sentences phrases 'Artificial Intelligence' occurred. This data is presented in Figure 3.2. We decided to set the threshold to at least three occurrences as the number of documents stabilized.

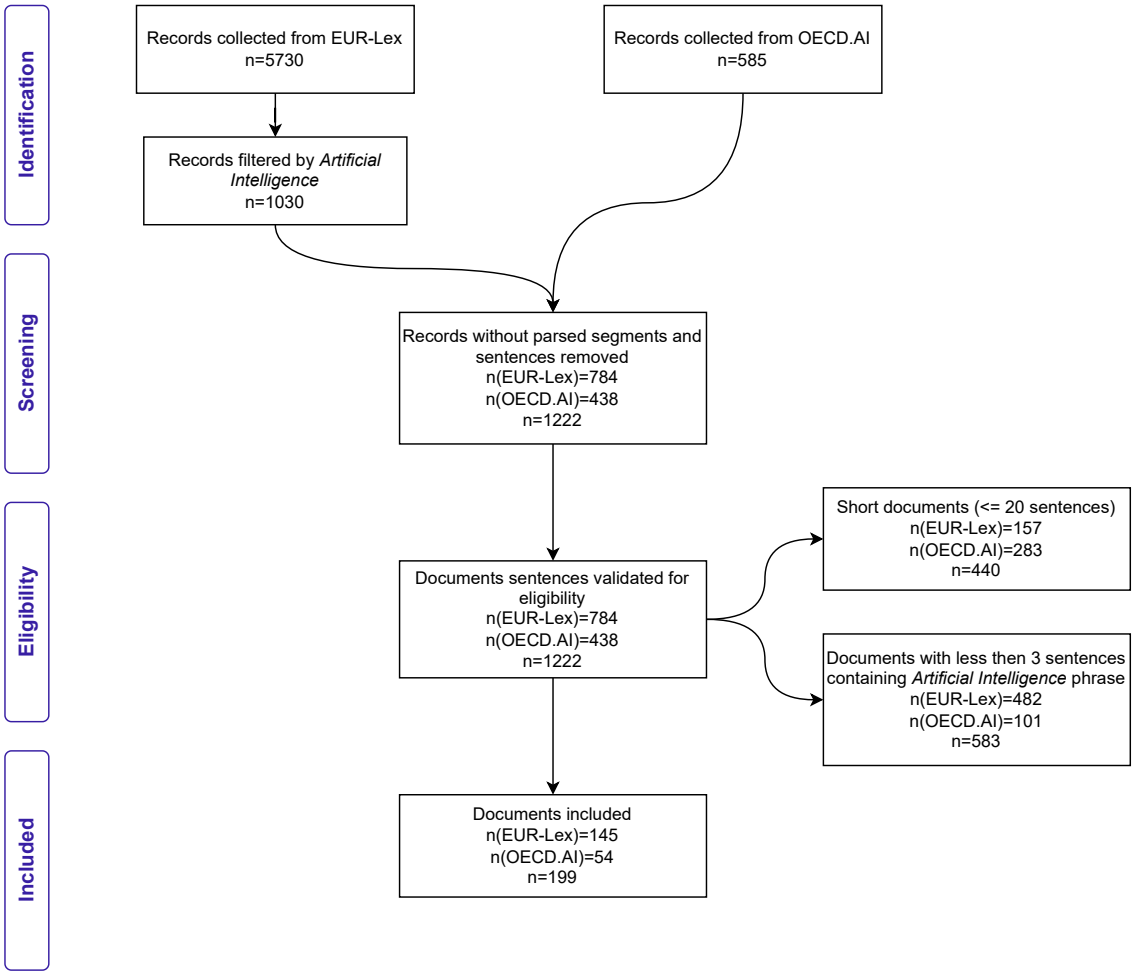

Figure 3.1: Prism chart of curating documents. We start with 6315 raw documents from EUR-Lex and oecd.ai. After filtering EUR-Lex documents, we are left with 1615 documents. Next, we remove documents that were not successfully segmented. At this point, we have 1222 documents. From this group, we include 199 that are longer than 20 sentences, and at least three sentences contain the phrase "Artificial Intelligence".

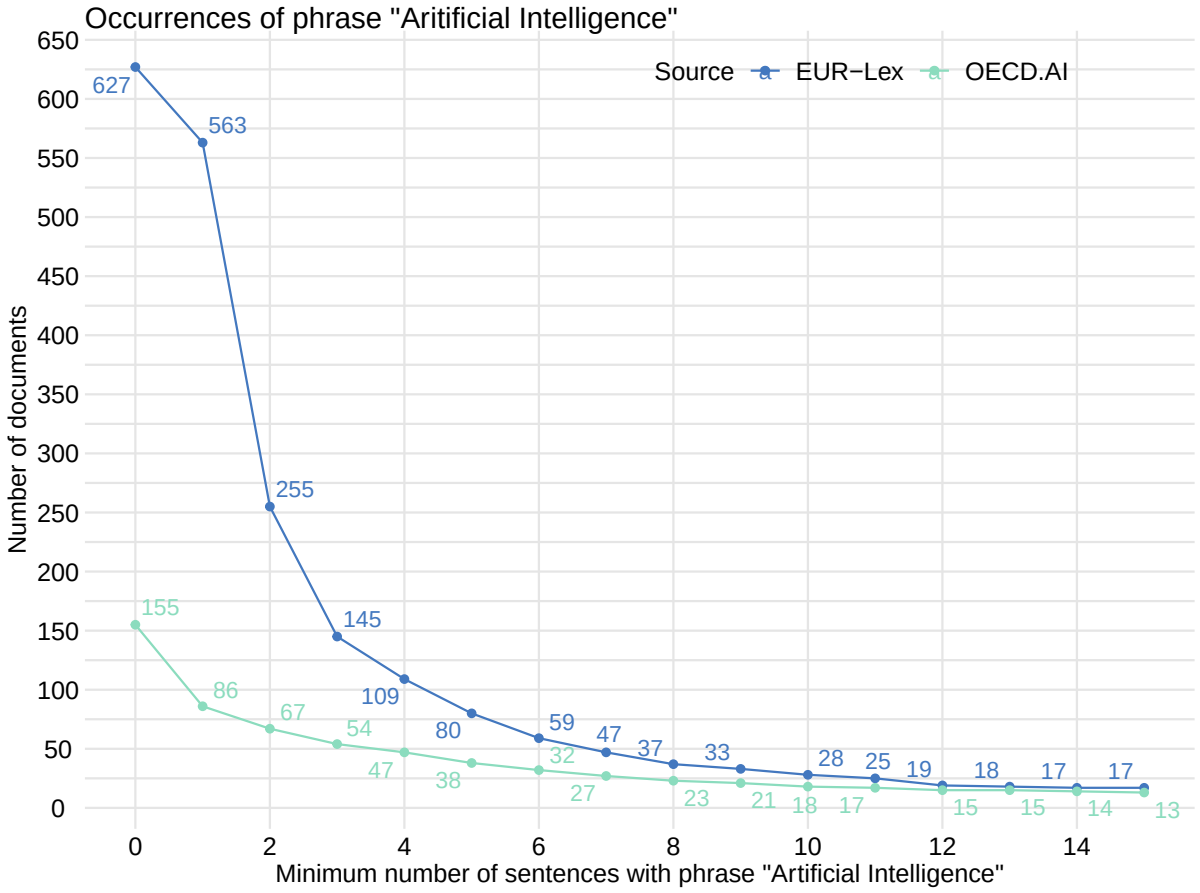

Figure 3.2: Number of documents with the minimum number of sentences with phrase 'Artificial Intelligence'. EUR-Lex returns documents by automatically matching keywords, and oecd.ai selects their documents manually - this can explain the difference in the initial drop of numbers.

4. Architecture of database

We decided to use ArangoDB which is NoSQL Strauch et al. [2011] database. Using a non-relational database does not restrict us by tables schemas - we can easily add another column or metadata. Different sources have different identifiers, and in the case of a new source, we cannot predict its type. So if a structure changes or some fields are empty, it is all covered by ArangoDB. Up to this day, we have several collections in our database, which are shown in Figure 4.1 and described underneath:

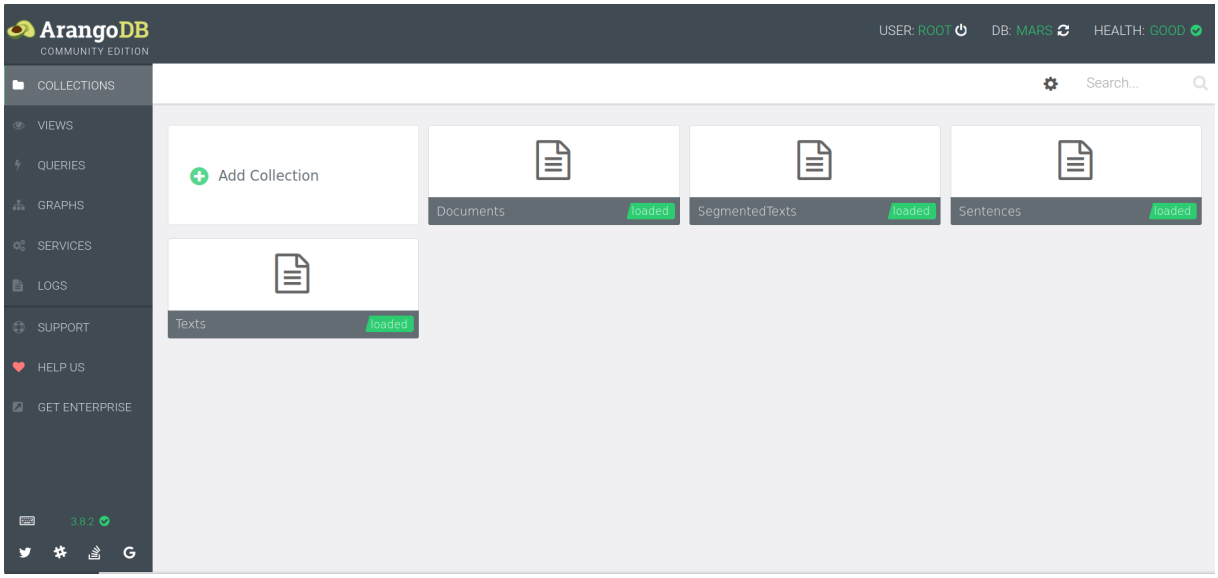

Figure 4.1: The view from administrator panel of our database

1. `Documents` - identifies each document and metadata. Consists of:
    - `url` - link to source of the document
    - `file_type` - PDF or HTML
    - `filename` - referencing where file with contents is located
    - `source_website` - oecd or eurlex
    - `lang` - language of the document
    - `keyword` - if document is obtained via eurlex, what keyword was used in search

- `celex` or `oecdId` - unique identifier of source website
- `country` - country where document was created
- `title` - title of the document
- `startDate` - when document is in effect
- `endDate` - when document is in effect

2. `Texts` - contents of documents in plain text. Consists of:
   - `source_doc_id` - id of the document in `Documents` collection.
   - `extraction_method` - how content was obtained
   - `content` - content of the document in plain text
3. `SegmentedTexts` - documents split into segments. Information about order of those segments is kept as an attribute. Consists of:
   - `source_doc_id` - id of the document in `Documents` collection.
   - `html_tag` - indication of header size or paragraph
   - `content` - content of segment in plain text
   - `sequence_number` - indicates the order of given segment in document
4. `Sentences` - documents split into sentences. It has an additional information from which section does it come from. Consists of:
   - `source_doc_id` - id of the document in `Documents` collection.
   - `source_segment_id` - id of the document in `SegmentedTexts` collection.
   - `html_tag` - indication of header size or paragraph
   - `sentence` - sentence in plain text
   - `sentence_number` - indicates the order of given sentence in segment

5. Exploration of documents collected in database

After thorough filtering of obtained documents, we analyzed them. We created histograms of sentence counts and segments per document. They can be seen in Figures 5.2 and 5.1 respectively. As we can see, most documents are short, the mean number of segments per document is 550.95, and the mean number of sentences per document is 1168.11. To inspect the contents of documents, we created word clouds and described their contents in detail in captions. We obtained them following the steps below:

1. extraction of words separated by space
2. filtering only letters with regex
3. filtering out stopwords with the help of Dabbas [2018]

We split the sources into long and short based on their word count medians. It was 35102 for OECD.ai and 207313 for EUR-Lex. We then decided to inspect documents in groups based on their source and length. Wordclouds can be viewed in Figures: 5.9, 5.10, 5.11, 5.12. The most popular words in EUR-Lex are pretty different from OECD.ai. In the first source, there is a big emphasis on Europe and regulation and policy. In OECD.ai, however, there is more focus on data, research, and the development of AI-related products.

We were also interested in the date of documents' entry into force. It can be seen in 5.5. What is worth mentioning is that oecd.ai was created in 2020. They may focus on the most recent documents rather than past ones.

The filtering described in Section 3.3 allowed us also to investigate the length of documents based on the number of sentences with the phrase "Artificial Intelligence". This data is presented in Figure 5.3.

The percentage of file types per source can be seen in Figure 5.4. Dominating filetype is HTML, which is expected, as all of the documents from EUR-Lex are in HTML format.

As EUR-Lex provides metadata about the legal type of documents they share and their authors, we decided to see what types of documents occurred most in our database, which can be seen in Figure 5.6 and Figure 5.7. On the other hand OECD.ai provides metadata about country relevant to documents, which can be seen in Figure 5.8.

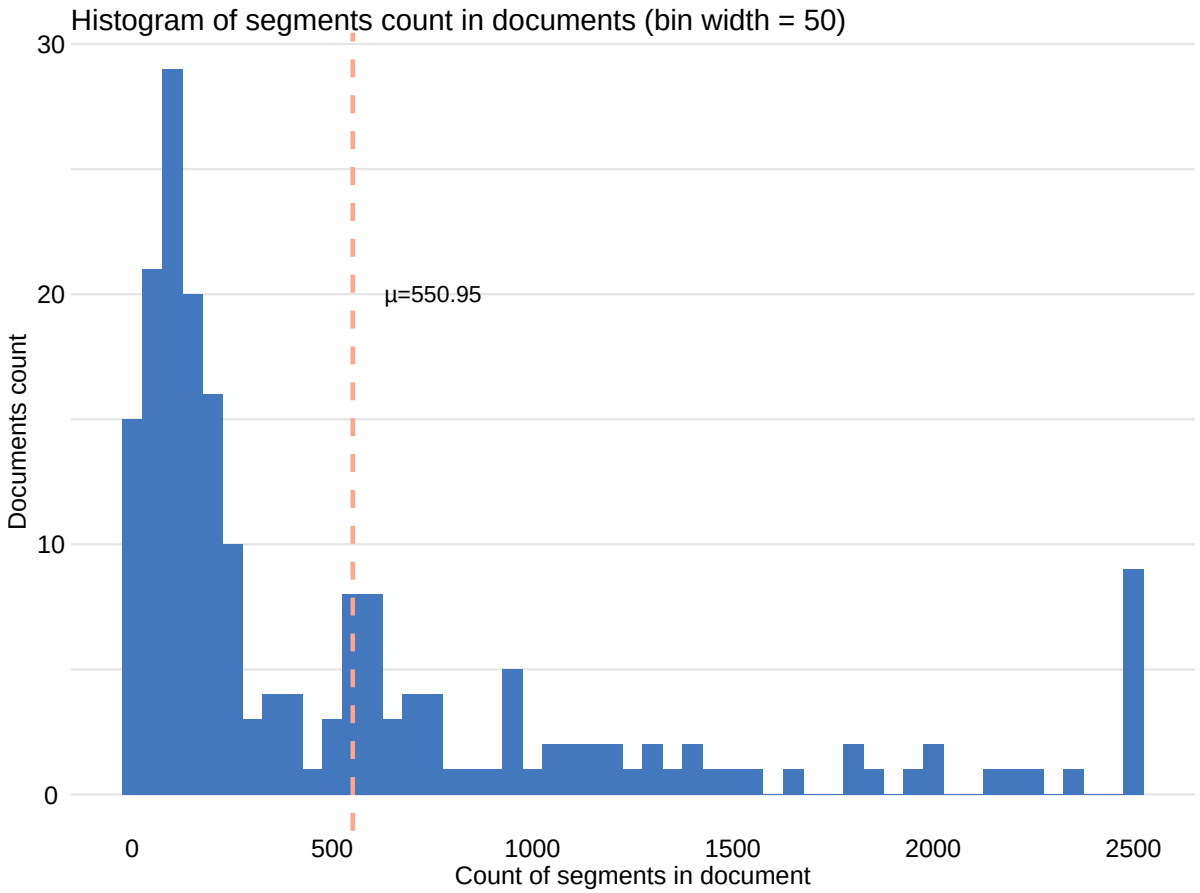

Figure 5.1: Histogram of segments count per document. Mean is marked by orange line. For readability we limited upper values to 2500 - any document that exceeds this number will be counted as 2500.

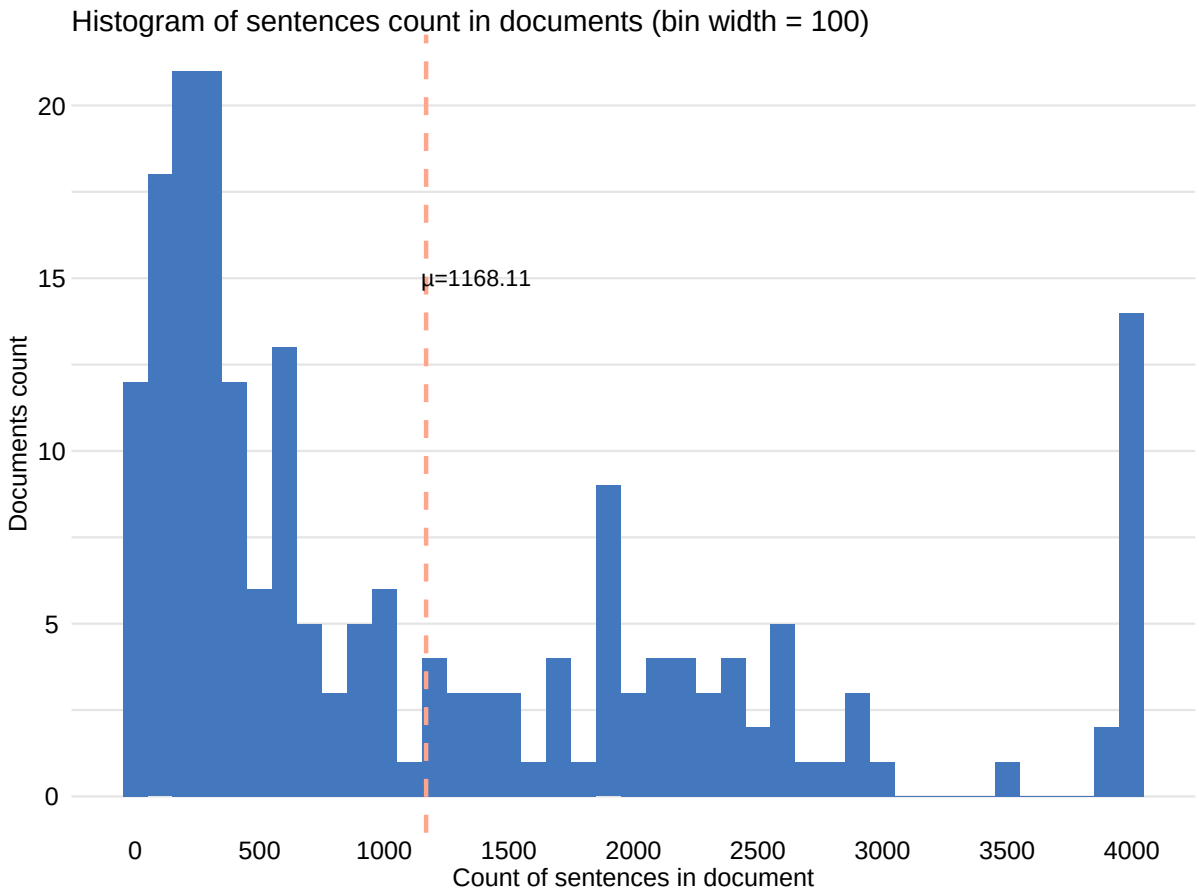

Figure 5.2: Histogram of sentences count per document. Mean is marked by orange line. Most of documents are relatively short, the highest bars are those with 200-300 sentences. The highest bar is made for 150-200 segments. For readability we limited upper values to 4000 - any document that exceeds this number will be counted as 4000.

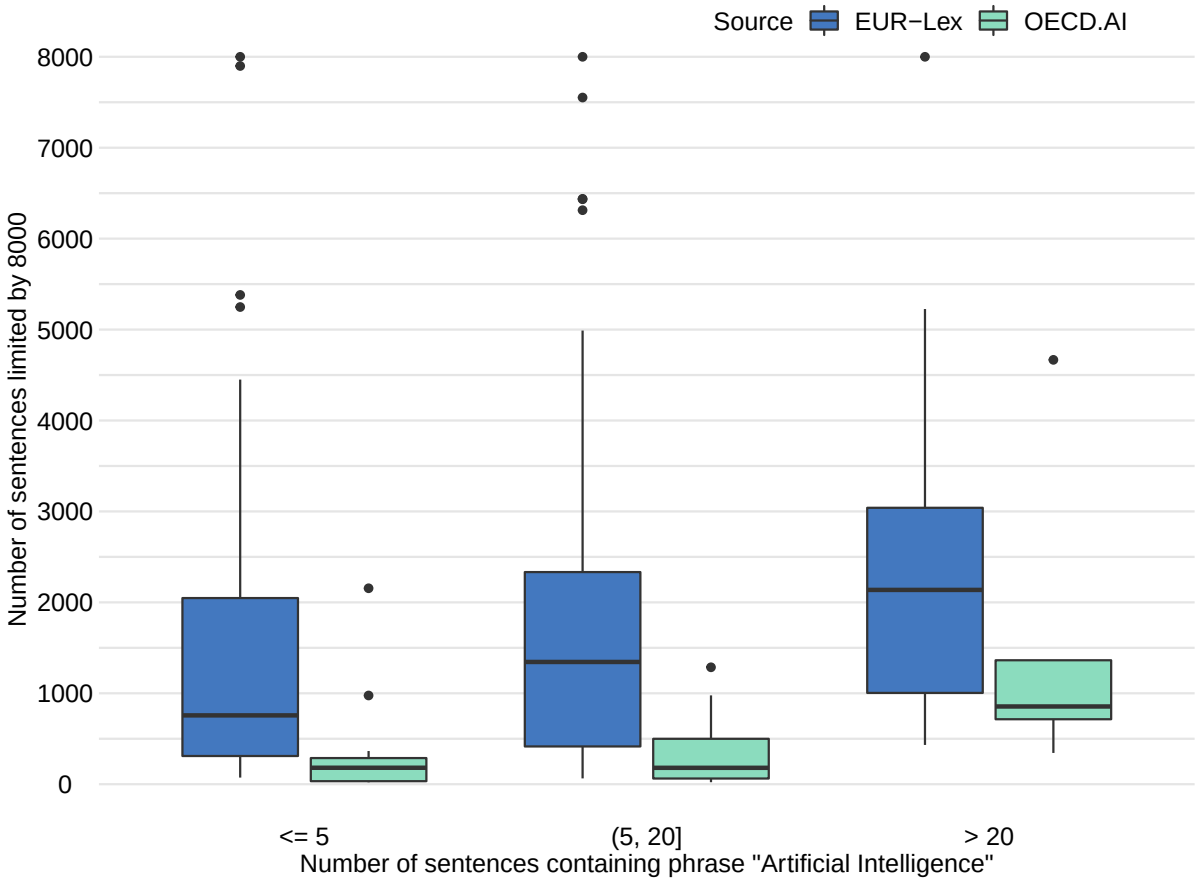

Figure 5.3: The plot shows the number of sentences containing the phrase "Artificial Intelligence". Documents from EUR-Lex are significantly longer than those from OECD. From this, we can conclude that the documents from OECD are more frequently using this phrase. They are more monothematic than the ones from EUR-Lex, which, despite, for example, mentioning the phrase from 5 to 20 times, is a few times longer than the typical OECD document.

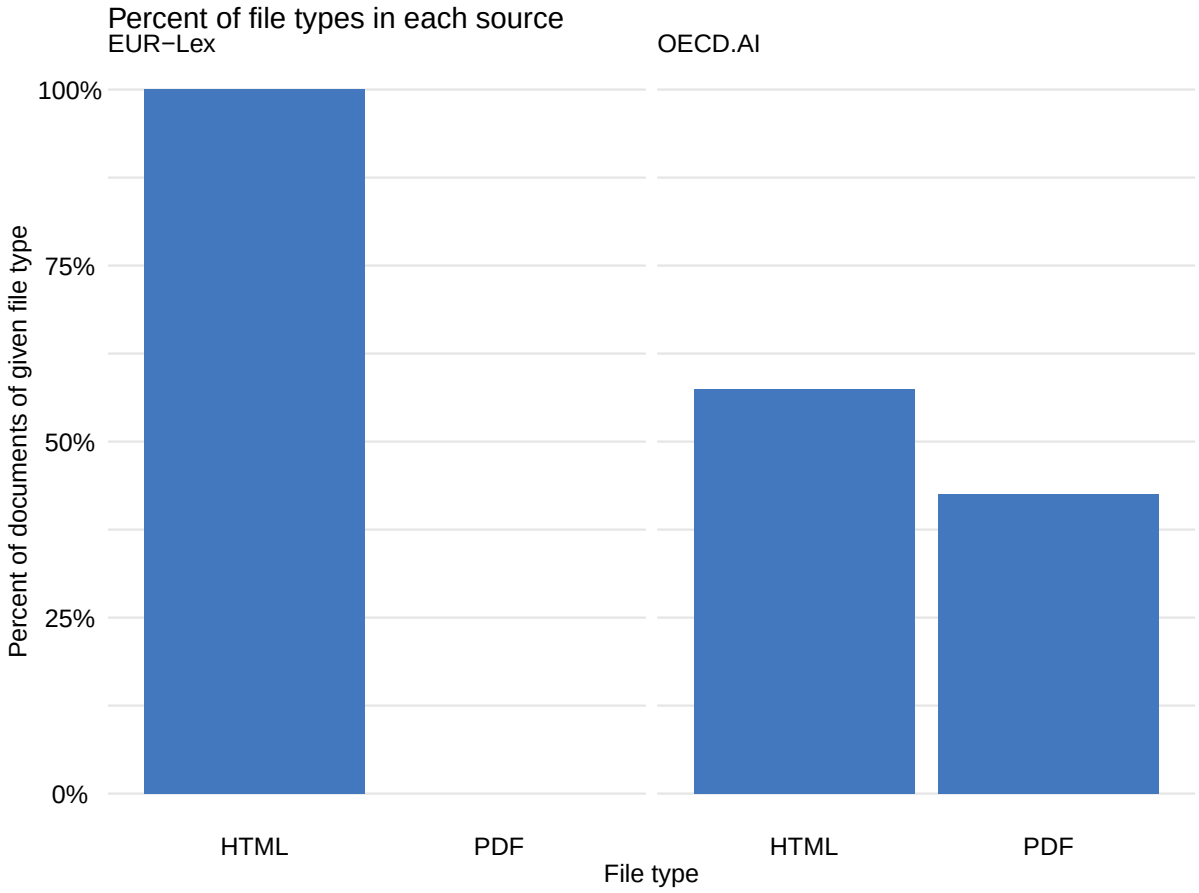

Figure 5.4: Percentage of file types splitted by the source. The HTML is dominating source of the document, especially in EUR-Lex, where all files in this format. In OECD the fractions are similar, but still with HTML's being more popuar ones.

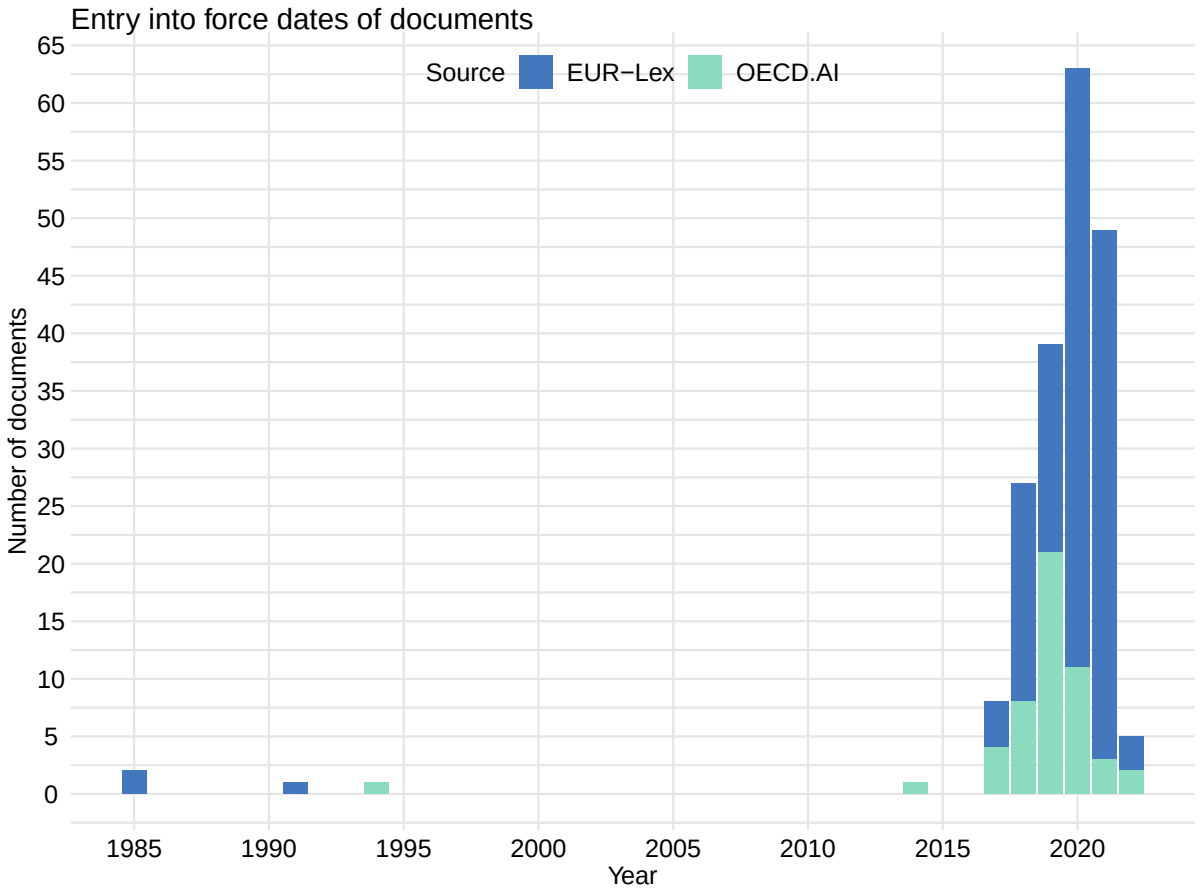

Figure 5.5: The histogram of documents entry into force. The first documents entered into force in the year 1985 - *Council Decision of 11 February 1985 adopting the 1985 work program for the European Strategic Programme for Research and Development in Information Technologies: ESPRIT* from February and *Community-COST Concertation Agreement on a concerted-action project in the field of artificial intelligence and pattern recognition (COST project 13)* from December. Until 2015's only three more documents went into force. After that, the interest rose significantly. Three documents from oecd.ai were excluded, as oecd.ai did not provide the entry date into force.

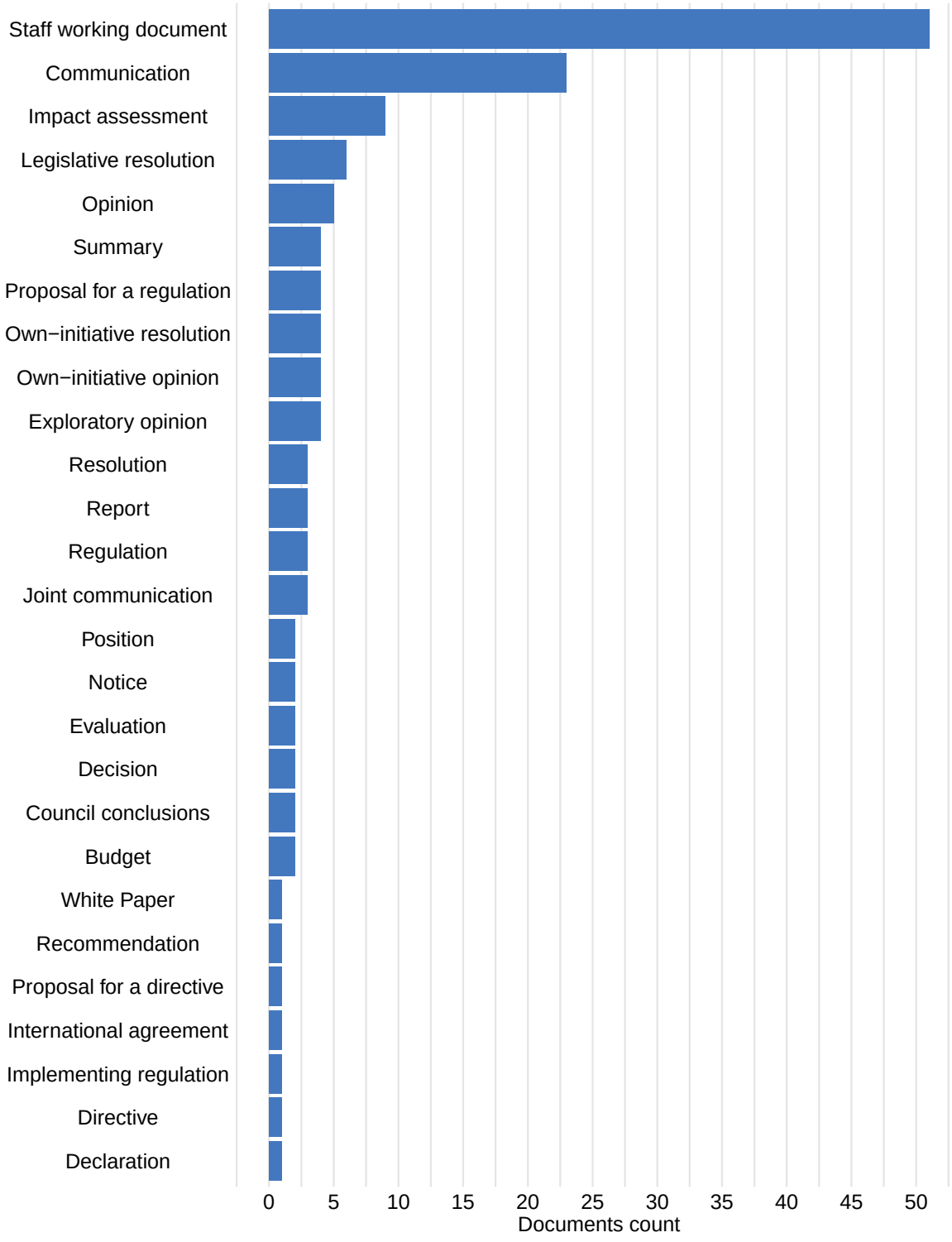

Figure 5.6: Legal type of documents from EUR-Lex. As expected, documents on lower legislative level, like Staff working documents and Communications, appear more often then documents on higher legislative level, for example Regulations and Resolutions.

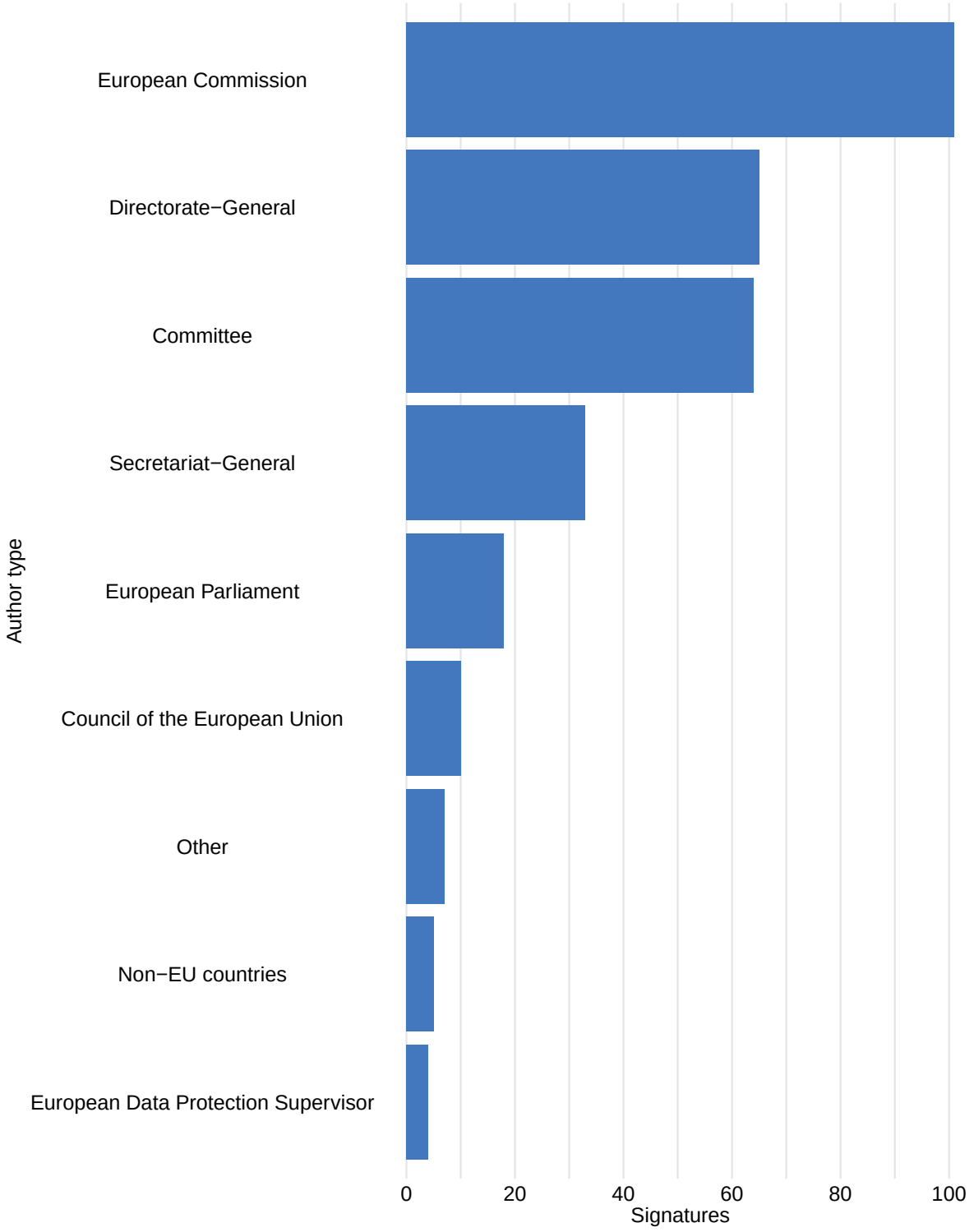

Figure 5.7: Visualization of authors of documents from EUR-Lex. Documents may have more than one author. For better visualisation we grouped together and added authors based by their rank - for example committee is sum of committees signed under all documents. The European Commission is highly active in creating documents regarding AI

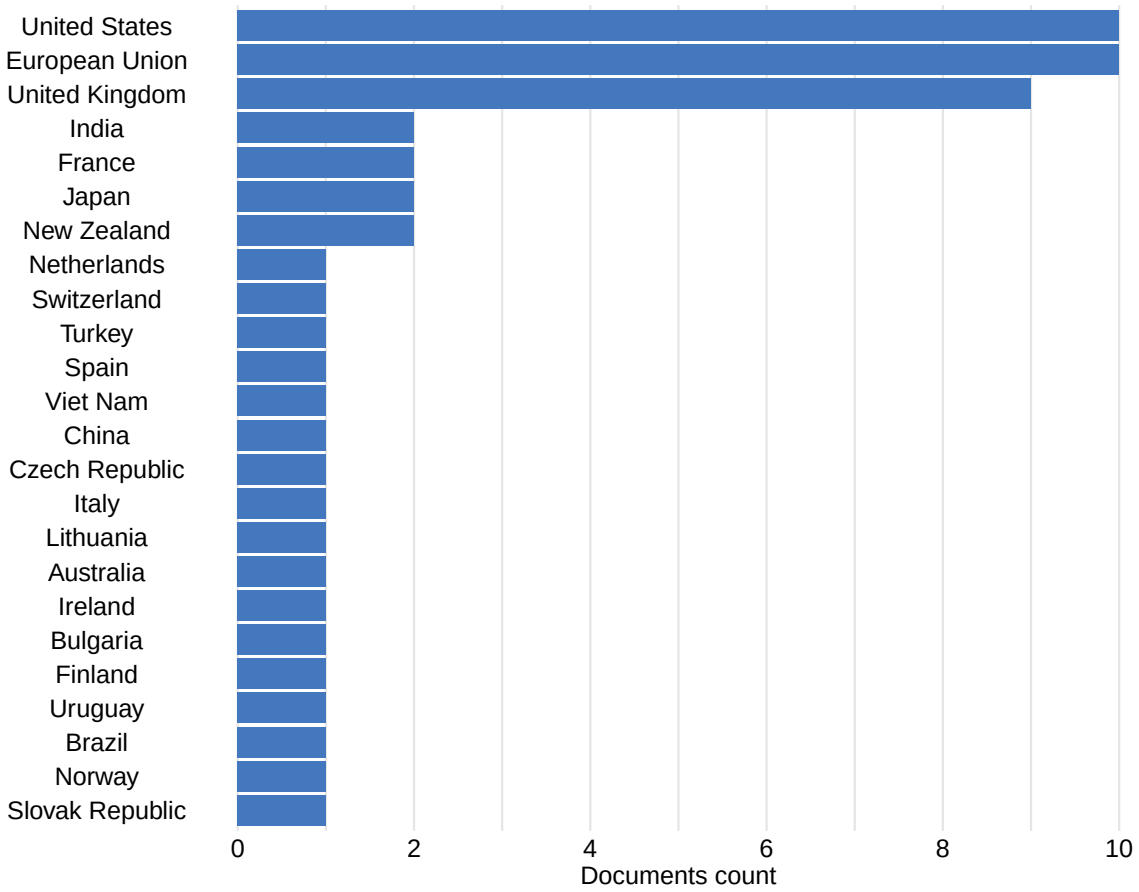

Figure 5.8: Number of documents from each country or union of countries from OECD.ai. As can be seen in the plot, the United States and European Union have produced the most documents, United Kingdom fell short of only one document.

Figure 5.9: EUR-Lex long documents word cloud. Words are more European and regulation-focused. Most frequent ones are "European", "financial", "appropriation", "member", data and union. Some are a little bit different, like "ojl" and "oj" that point to Original Journal, "sme" that stands for Small and Medium-sized Enterprises, or "ec" that point to articles by the European Commission (e.g., 2004/18/EC). These words were created with our pipeline, and sometimes regex filtered out numbers or space inside living only letters.

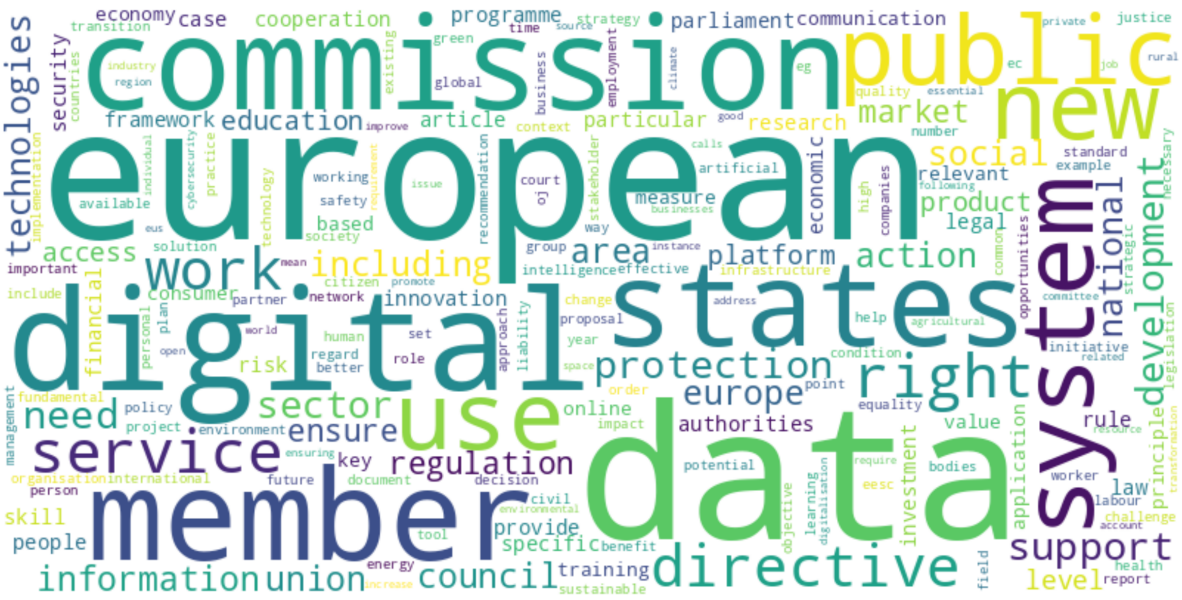

Figure 5.10: EUR-Lex short documents word cloud. The words are similar to those in Figure 5.9. Here, however, there are more words regarding data and system, the fact that it is digital, and its use. Most words here are well self-explanatory, or we already mentioned them in the caption of Figure 5.9. There was one word standing out - "eesc" which is the abbreviation for European Economic and Social Committee.

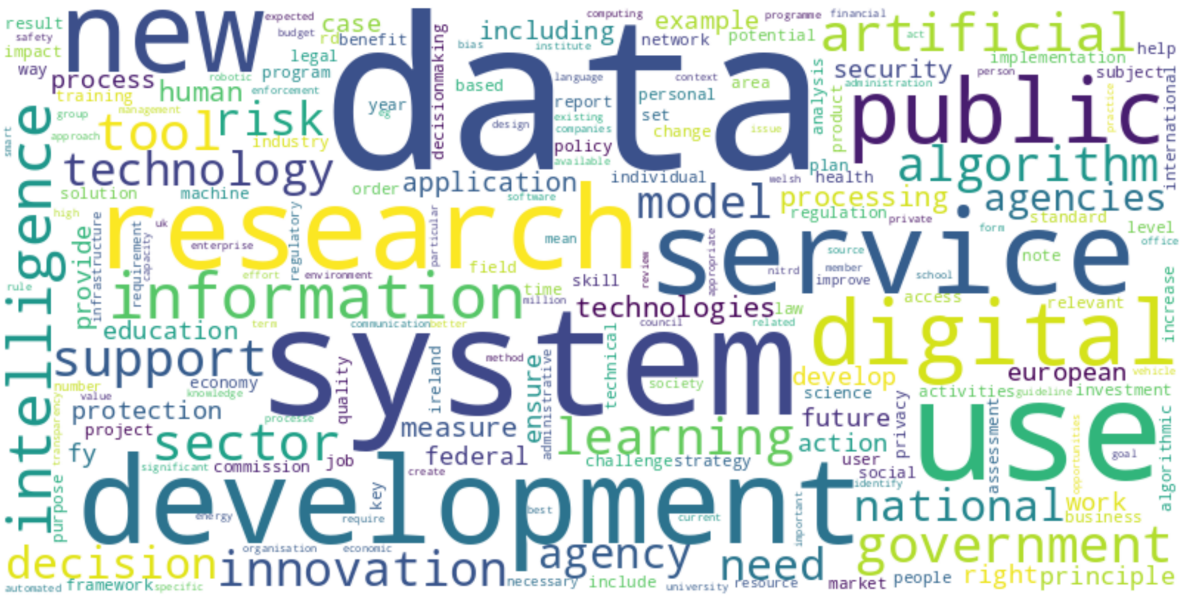

Figure 5.11: OECD.ai long documents wordcloud. One of the most common words are "data", "research", "system", "development" and "use". There also were a few words that were quite different like "fy" - Fiscal Year, "nitrd" - points to the side nitrd.gov. They were also created through regex.

Figure 5.12: OECD.ai short documents wordcloud. Ones of the most popular words are "intelligence", "technology", "data", "development". The odd words and their meanings are: "rd" - Research Development, "ppp" - Public-Private Partnerships, "gaiax" - Gaia-X, "nsf" - National Science Foundation.

6. Orchestration

We used several scrapers, APIs, databases, uploading, and pre-processing in our project. We needed to orchestrate those components in a scheduled fashion. We are using two tools which we describe in sections underneath.

6.1. Airflow

In order to maintain up-to-date documents, we used Apache Airflow [2022] for scheduling. The order and list of tasks are defined as DAG (Directed Acyclic Graph) in the definition file [1]. The flow consisted of 4 tasks in order - first, both scrapers 2.1 were triggered simultaneously, and after they were done, segmenting 3.1 and splitting 3.2 to sentences were done. The flow can be seen in Figure 6.1. The daily schedule can be seen in Figure 10.3.

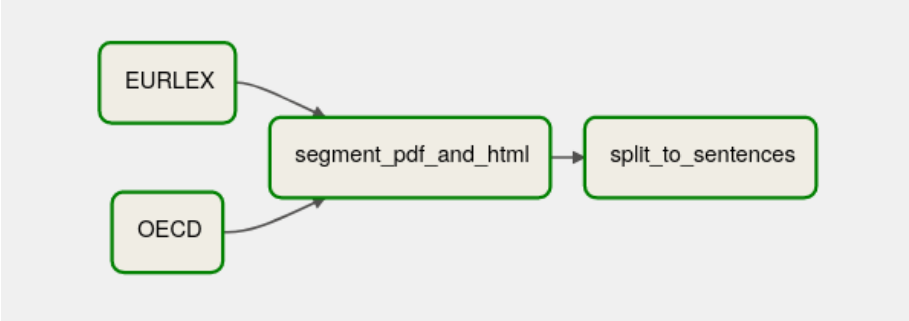

Figure 6.1: Airflow pipeline

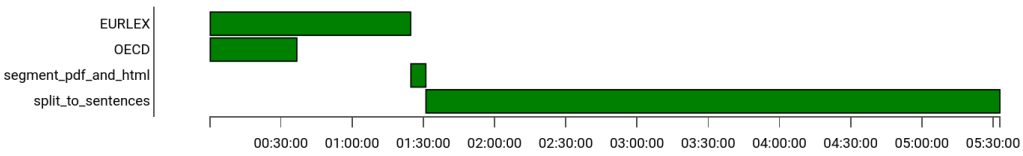

Figure 6.2: Airflow daily timeline

[1] Code is available at Github airflow/default.dag.py

6.2. Tasks pipeline implementation using Redis Queue

Redis Queue is used in handling tasks in the user interface (which is described in detail in chapter 10). Each analysis is based on a few transformations on the uploaded document. The transformations are in order: text extraction, segmenting and splitting to sentences, definition scoring, calculating embedding, and finding keyword issues. They can depend on each other. We use the Redis Queue Python package Driessen [2022] to manage those tasks. This tool allows us to schedule tasks easily and monitors their status. Figure 10.3 shows processed Redis Queue.

7. Modeling documents content

Nowadays, documents have much information to digest, and there is a need to summarise this knowledge. A valuable feature for users would be definition extraction from documents, which will search for definitions in possibly very long documents. Another added value could be automatically finding critical issues by the corresponding keywords. We implemented two models to do that - the definition scoring model and the keyword issue model. Both of them operate on the sentence level.

7.1. Definition Extraction Model

We needed a robust algorithm to process unstructured text to make a definition extraction model. We are using a transformer model that acts as a binary classifier. It predicts if a sentence is a definition or not. The model's output will be a score which is the probability of a sentence being a definition. Then in the user interface, the user will have the ability to view the most probable definitions.

Architecture

The sentence classification model is a transformer. Transformers are based on an attention mechanism that learns contextual relations between words. The model that we decided to use is called DistilBert Sanh et al. [2020], and it is a distilled version of BERT Devlin et al. [2019]. Authors claim that it has 40% fewer parameters is 60% faster while achieving 97% of performance of the original BERT model on GLUE Wang et al. [2018]. We decided to use it because it is an excellent combination of high performance, almost that of BERT (in our use case, we measure it in section 7.1) and low inference time (less than 0.1 seconds on the local machine with CPU).

Parameter fine-tuning

The model was originally pretrained on data prepared from big texts corpuses like Toronto Book Corpus and English Wikipedia, so it was necessary to fine-tune it to our specific use case.

We combined Wikipedia definitions dataset Spala et al. [2020] with DeftEval Navigli et al. [2010]. The test set combined 10% of the Wikipedia definitions dataset and a specific part of DeftEval prepared by their authors to test the models. We achieved around 0.89 accuracy, 0.84 F1 score, and only around 10% of sentences labeled as a definition were false positives (0.87 precision). Our training pipeline consists of training state-of-the-art model from HuggingFace Wolf et al. [2020] with the help of TensorFlow Abadi et al. [2015] which enables us to have production-grade solutions[1]. With such a pipeline, we obtained accurate predictions with a relatively short inference time[2].

7.2. Definition scoring script

After segments were split into sentences, we used the definition scoring model to classify whether the sentence was a definition or not. We define how the model works and how we trained it in the section about models - 7. Definition scoring model accepts a string as input (sentence) and returns a probabilistic score of being or containing a definition. The sentence is tokenized via previously initialized AutoTokenizer from HuggingFace Wolf et al. [2020]. Then the model scores the sentence and returns a float from the range [0,1]. [3]

7.3. Evaluation of Definition Extraction Model

Although the definition model was already evaluated on external data, it had similar distribution as the training set. Therefore it might be significantly better than running on real-life scenarios. Having this in mind, we evaluated the definition extraction system with our data from the database. We have extracted sentences from random 5 OECD and 5 EUR-Lex documents and scored them for being the definitions or not (sentences can also just include the definition(s)). The measure we focused on was precision, which tells us what fraction of definitions found by the model is actual definitions. Formally it is defined by the number of true positives (actual positives) divided by the number of positive predictions. We assumed that the sentence was a definition when points below were met:

1. Sentence provides the term which denotes a thing that definition defines. It can be some

[1]The code to train the model can be found in mars/models_training/train_definition_transformer.py and mars/definition_model.py

[2]Script scoring all sentences in the database can be found at scripts/extract_definitions.py

[3]The implementation of this module is available at mars/definition_model.py

proper noun, entity, or simply "it".

2. Sentence describes what the term is. It may be the description, abbreviations, and their longer forms or a set of actions that the defined term performs. If it is a set of actions, it must give enough information to identify the term. For example, the sentence *The AI Barometer highlights the potential for AI and data-driven technology to address societys greatest challenges.* provides insufficient information about *AI Barometer* to classify it as a definition. Sentence *The 'Public services and engagement' category includes tools that facilitate the provision of services to or communication with the public for regulatory or other purposes.* despite only providing information about what the category includes, the category can be now identified among other sections, so the information is sufficient.

We also need to add a caveat. The sentence can just contain a definition, so the sentence may not be a stand-alone definition. So, for example, a sentence *"AI, which is defined as an intelligent self-taught system, is transforming the landscape of software systems."* will be treated as a definition. A sample of true positives and false negatives can be found in Appendix A.

This way, we ended up with almost 700 definitions from 10 documents - 5 from EUR-Lex and 5 from OECD. We have annotated those definitions found by the model (sentences with more than 0.5 probability of being the definition) either as a proper definition or false definition (true positive or false positive). We then measured the precision of the model.

The results of the analysis can be found below in form of a table (Table 7.1) and a plot (Figure 7.1):

	True Positive	False Positive	Sum	Precision
OECD	280	114	394	0.71
EUR-Lex	176	120	296	0.59
Sum	456	234	690	
Fraction	0.66	0.34	1	

Table 7.1: Table showing how many sentences that were predicted to be definitions are actually definitions split by the source.

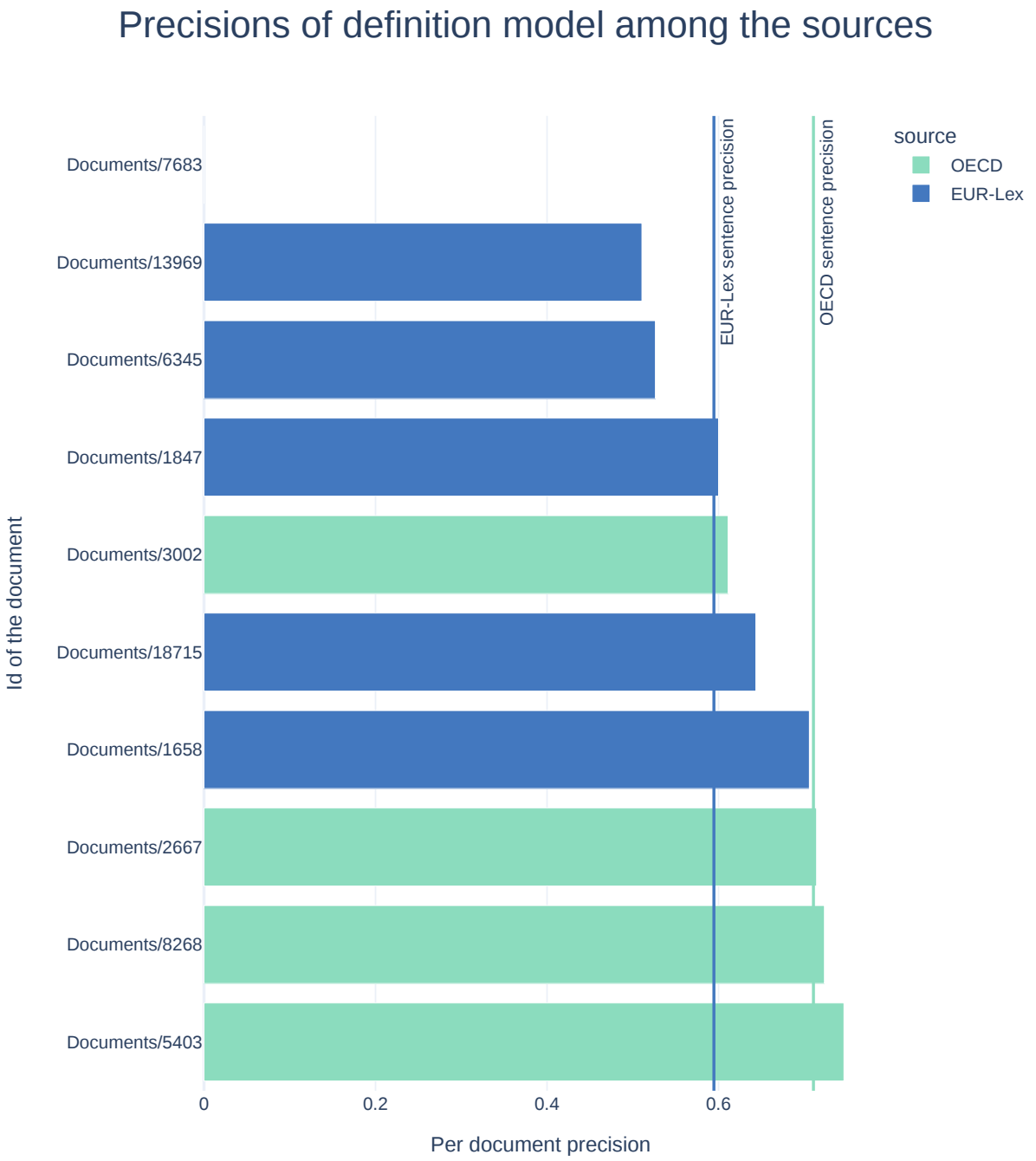

Figure 7.1: Precisions for each document from EUR-Lex and OECD. The model performs better on OECD documents. The mean precision for sentences (not documents) from OECD and EUR-Lex are denoted by vertical lines. The overall sentence precision for OECD was 0.71 and 0.59 for EUR-Lex

The average precision of the model was 0.66. Nevertheless, it performed significantly better when dealing with texts from OECD than EUR-Lex. The specific legalese of EUR-Lex documents made the model confuse some statements with definitions. It shows that the model is sensible

for the origin of the document and the language and grammar forms that it uses.

7.4. Keyword Issues Model

Issues (or topics) in AI documents help localize themes and specific, well-defined fields. Such issues are, for example, responsibility, fairness, and transparency. Those issues are popular themes mentioned in AI publications and regulations. The documents, regulations, and sometimes articles can be a few dozen pages long. Therefore, there is a need for a tool that would quickly mark sentences with specific topics. For example, fairness is a topic that has been gaining popularity in recent years Mehrabi et al. [2021]. It would be helpful to localize and identify this topic in some regulatory documents automatically.

7.4.1. Implementation

The keyword issue model searches for a specific keyword in the sentence. The set of issues was described by the authors of Jobin et al. [2019]. They found and described the most popular and influential ones mentioned in principles and guidelines made by private companies, public sector organizations, and research institutions. We adopted those issues to our use. The topics are defined with the corresponding keywords that are their synonyms or are frequently used with them. The sentences are firstly lemmatized, and the lemmatized issue keywords are searched for[4]. If the keyword appears in the sentence, then it is marked as one containing that specific issue. The topics and their lemmatized keywords are presented in Table 7.2.

[4]The code for this module can be found at mars/keyword_topic_model.py

Issue	Keyword Lemmas
Transparency	transparency, understandability, disclosure, explicability, transparent, xai, explainable, interpretable, explainability, interpretability, iml
Justice, fairness, and equity	fairness, consistency, diversity, inclusion, plurality, reversibility, accessibility, bias, discrimination, discriminate, bias, fair, justice, equity, equality, mistreatment, unfairness, unfair
Non-maleficence	maleficence, maleficent, security, safety, harm, protection, precaution, prevention, integrity, harmful
Responsibility and accountability	responsibility, responsible, accountability, liability
Privacy	privacy, private, confidentiality, privateness
Beneficence	beneficence, benefaction, benefit, peace
Freedom and autonomy	freedom, autonomy, consent, choice, liberty, empowerment
Trust	trust, trustworthiness, trustworthy
Sustainability	sustainability, sustainable, continual, environment, energy
Dignity	decency, dignity, honor, moral, morality, decent
Solidarity	solidarity, cohesion, agreement, consensus

Table 7.2: Issues and corresponding lemmatized keywords.

7.5. Evaluation of Keyword Issues Model

The performance of the keyword model is vastly dependent on the document. Its drawback is that it will have precision equal to 0 if there is no or limited mention of AI in the document. Therefore we cannot honestly evaluate the model as we did in the case of the definition extraction model because the score would have a vast proxy on the content. That is why we decided to use an example document. We used European Commission [2021] which is a proposal of the European Commission on rules regarding AI.

In this 108 page long document, there were in total 336 issues. Like in the case of definitions, we manually annotated if the issue found was true or false positive, and then we calculated

precision for each issue. The logic behind the annotation was that the keyword must adhere to AI system (e.g., trusted AI, trust in AI, responsible AI) or something caused by the AI systems (e.g., harms caused by AI, AI does not limit freedoms). Examples of annotations are presented in Appendix B. The table below 7.3 shows the number of specific issues, amount of correct predictions out of all predictions, their precision, and those statistics for all of the issues.

Issues	correct / total found	precision
Transparency	25/30	0.83
Justice, fairness, and equity	23/41	0.56
Non-maleficence	53/100	0.53
Responsibility and accountability	3/40	0.08
Privacy	11/26	0.42
Beneficence	7/15	0.47
Freedom and autonomy	13/25	0.52
Trust	32/37	0.86
Sustainability	8/10	0.8
Dignity	5/5	1
Solidarity	0/7	0
TOTAL	181/336	0.54

Table 7.3: The issues were manually annotated to the found amount of correctly labeled sentences. Then the precision was calculated for each issue

The performance of the keyword issues model is far from perfect. The total precision was 0.54, although some topics were identified with significantly better metric scores. Along the annotation process, we identified some problems that were associated with the selection of the keywords. In *Justice, fairness, and equity* majority of false positives were connected with the usage of the word "consistency". One of the worst metrics was achieved in the case of *Responsibility and accountability*. The reason for this was that the document frequently mentioned the responsibility of some parties and stakeholders, not the responsible design of AI itself. This shows that there is significant room for improvement, with better choice of keywords to making more sophisticated models (for example, checking if AI is the subject of a sentence, making transformer models)

8. Tests

Three tests were implemented and conducted: usability tests, unit tests, and integration tests. Below we are describing it in detail.

8.1. Usability test

We wanted to understand how unbiased users would interact with projects' UI and identify hidden bugs in the website. In order to test usability, we arranged a meeting with a data science student from Warsaw University of Technology. We choose the final year student because he has sufficient knowledge about machine learning and no bias, as he has not seen the project before. The participant was provided with a link and PDF document - AI Now Report 2018 Whittaker [2018] and credentials to UI and asked by the facilitator to perform the following tasks:

1. Uploading given document
2. Choosing this document
3. Discovering and examining definitions
4. Changing the document
5. Exploring issues analysis
6. Changing the model of issue analysis
7. Choosing a different number of definitions
8. Viewing segment above and below definition

No time limit was given to participants, and they used their computer to complete the tasks, as the facilitator observed and got feedback presented in table 8.1. As suggested by the user, we made some changes to the UI, including the separation of segments. We did not hide issues not present in the document, as the list provides information on what is in the document and what is not in it. Other suggestions were marked as low priority.

Feedback	Difficulty in implementation	Priority
It takes a bit of time to process uploaded documents, and the participant suggested displaying information with a warning about time	2	1
There is no way of telling if a highlighted text contains only one definition or more	4	5
Issues that aren't present in document are option to choose in dropdown	3	3
There is no way to hide segments and know where the original segment started or ended	4	5
No warning before deleting document is given to user	4	1

8.2. Unit and integration tests

Unit tests are implemented in tests directory. Tests were created when new modules were added to keep continuity in our project. This allowed us to avoid errors when modules were modified.

To make our tests independent from the environment and health of our infrastructure, we decided to integrate them into Continuous Integration and Docker Hub Docker Inc [2021b]. The whole process operates on GitHub Actions git and is triggered by commits. Each time a new commit arrives, tests are run, and if they are successful, Docker Images are updated on Docker Hub. Each service has its own Docker Image[1].

1. Pull repository

2. Pull models and data from storage

3. Create containers

[1]Each image can be inspected under https://hub.docker.com/u/mi2mair

4. Start database container
5. Run tests inside containers
6. Push images to the repository

This procedure guarantees that code works on the deployment because the same images are used to test and deploy.

In order to make sure that our code works on a lower level without bugs, we developed 21 unit tests, using package pytest Krekel et al. [2004]. We wanted to ensure that our project works not only as a whole, tested in integration tests, but also that functions on the lower level are clear and efficient. We tested six parts of the code:

1. Database connection
2. Definitions model
3. Key topic model
4. Scraper
5. Parser
6. Sentence embeddings

All of tests were successfully, as can be seen in Figure 8.1.

```
| ============================= test session starts ==============================
| platform linux -- Python 3.8.12, pytest-6.2.5, py-1.11.0, pluggy-1.0.0
| rootdir: /mair
| plugins: cov-3.0.0, anyio-3.3.4
| collected 21 items
|
| tests/test_db.py ...                                                    [ 14%]
| tests/test_definition_scoring.py ..                                     [ 23%]
| tests/test_key_topic_model.py .                                         [ 28%]
| tests/test_language_recognition.py ......                               [ 57%]
| tests/test_parser.py .                                                  [ 61%]
| tests/test_scraper.py .                                                 [ 66%]
| tests/test_search.py ...                                                [ 80%]
| tests/test_sentence_embeddings.py ....                                  [100%]
|
| ================== 21 passed in 61.87s (0:01:01) ==================
```

Figure 8.1: Project tests output

9. Deployment of project

This platform can be deployed using production-ready images in OCI standard The Linux Foundation [2022]. Despite the fact we provided configuration file and instruction only for Docker Compose Docker Inc [2021a], running it on Kubernetes or using Docker CLI Docker Inc [2020] is also possible.

9.1. Docker Compose deployment

1. Download project code from GitHub source code
2. Remove `.example` suffix from `.env.example` file
3. Set variable `DOCKER_PORT_PREFIX` to set digits except last of port numbers to be used
4. Remove `.example` suffix from files in `secrets/` directory and fill passwords there
5. Run `docker-compose pull` to download images
6. Run `docker-compose up -d` to start services:
    - Arango Database - available at port `${DOCKER_PORT_PREFIX}0`
    - Airflow frontend - available at port `${DOCKER_PORT_PREFIX}3`
    - WebDAV - available at port `${DOCKER_PORT_PREFIX}2`
    - Webserver - available at port `${DOCKER_PORT_PREFIX}5`

10. User Interface

The central part of our thesis is the user interface (UI). Users can upload their documents there store and analyze them. It is hosted on a web server and can be accessed via a web page from the standard browser. Below we describe the technical aspects and show the interface through figures.

10.1. Web server

The whole user interface and API are served using Flask Ronacher [2022] framework, based on Web Server Gateway Interface. There are a few reasons for this decision:

1. Supports modularity using Blueprint mechanism
2. Easily integrates static web pages with Python-based REST API
3. Well-established user base and documentation
4. Builtin parsers and validators

10.2. Web page

The user interface is available as a web page served via the web server described in the previous section. The design pattern Model-View-ViewModel Kouraklis [2016] was chosen with Vue framework You [2022] implementing ViewModel part and composing components into Single Page App Li and Zhang [2021].

10.3. Interface

The interface has three different sections, as can be seen in figure 10.1. Via UI, users may upload documents, check the processing status, and analyze the documents covered in the sections

below.

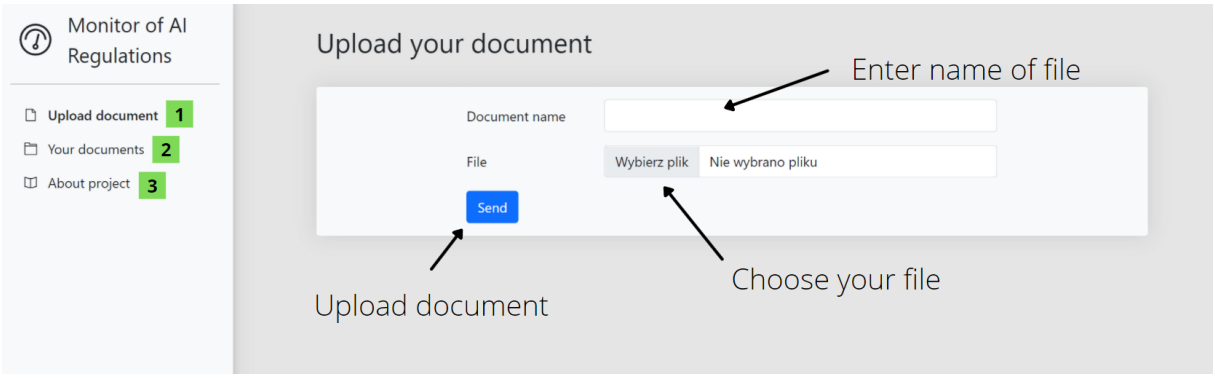

Figure 10.1: Main frontend view. Green squares represent sections. The first (current) is called "Upload document", and it is a section of uploading a document. Arrows point to sections where a user may choose a file (legal document regarding AI), enter its name, and upload it. It will trigger Redis Queue, and it will be visible in the "Your documents" section. The second square points to the "Your documents" section, where users may store and pick documents. The last section, the third square, is "About project" where our names and affiliation might be found.

10.3.1. Document uploading

The interface allows users to upload new documents to catch up with the dynamic growth of artificial intelligence legislature. The form is quick and easy because it requires only the document name and file in PDF or HTML. A user who submits this form is redirected to the waiting page, where the status of each processing phase is visible. Their definitions and key issues are found. The shot of both interfaces is shown by figures 10.2 and 10.3. At the end of the processing stage, the user gets the report described in the next section. Users can also access their documents in other sections, as shown in 10.4.

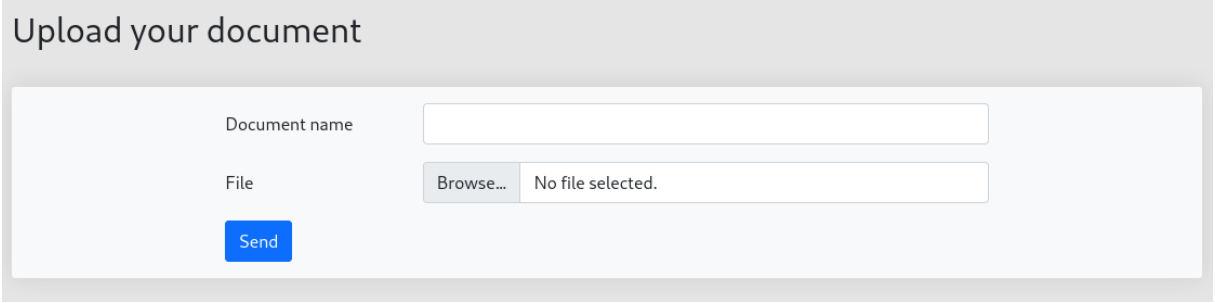

Figure 10.2: Document upload form

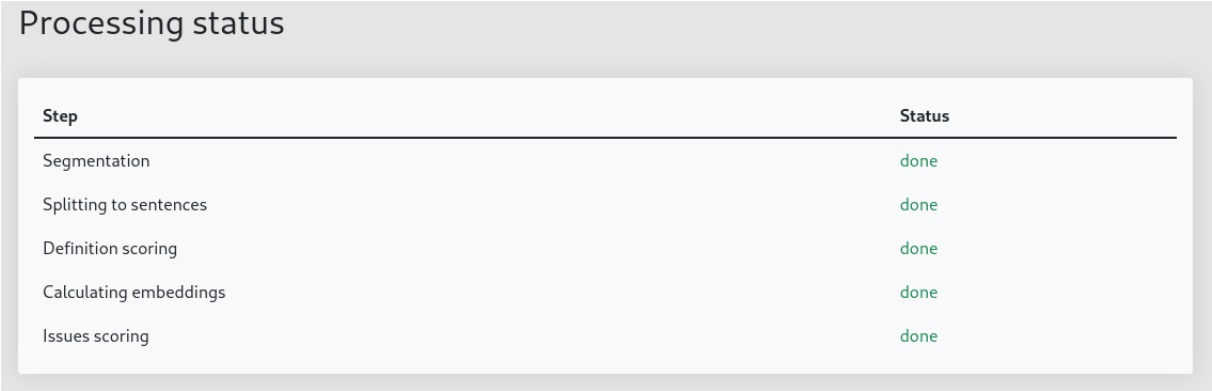

Figure 10.3: Status of Redis task in web interface

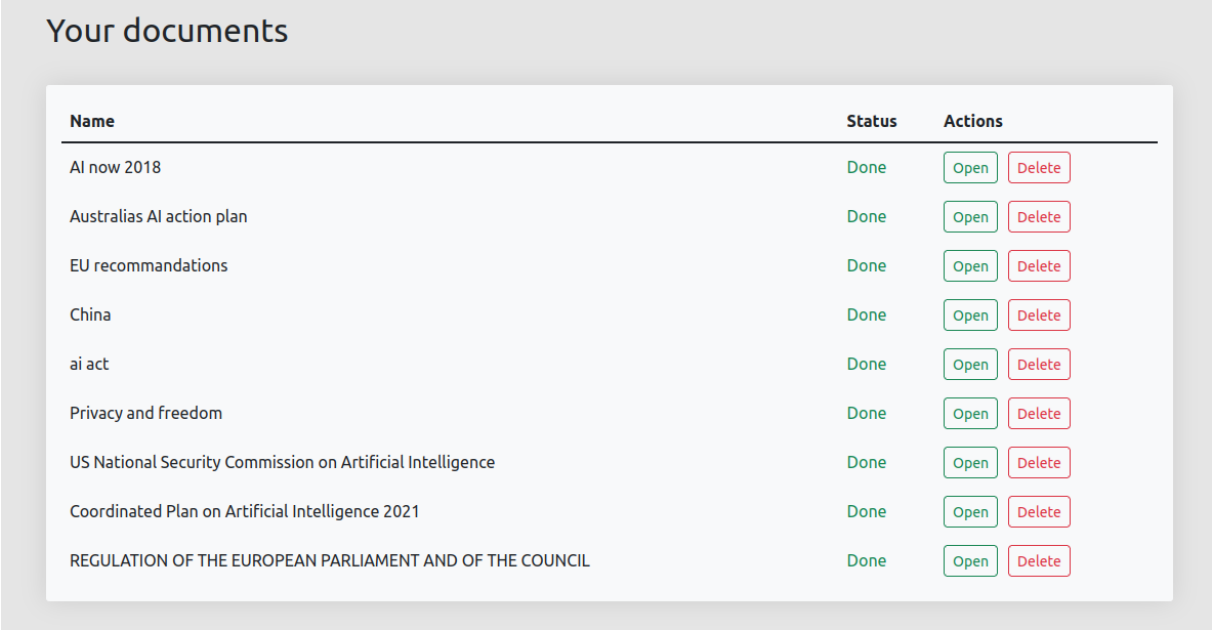

Figure 10.4: Document list

10.3.2. Rashomon effect

The design of the document viewer exposes and counteracts the Rashomon effect Anderson [2016]. This effect occurs when different narratives present the same story from different perspectives and tell contradictory details. To avoid being misled by document analysis, the researcher has to look at as many perspectives on this document as possible. The purpose of viewer design was to help achieve this objective with minimal involvement from the user. Document viewer implements tabs that quickly switch between various analysis aspects and used methods. Aspects include definitions and issues scoring via different machine learning models. What is essential for future work, the list of aspects and methods is easily extendable.

10.3.3. Visual exploration

Legislature documents often have several dozen of pages. The key to fast reviewing is to show only interesting parts with their context. The report page reuses the same Vue component for every aspect of analysis. This component visualizes the best sentences of a given aspect with their context, as shown in figure 10.5 and 10.6. Users can view the full context of each sentence by clicking on it.

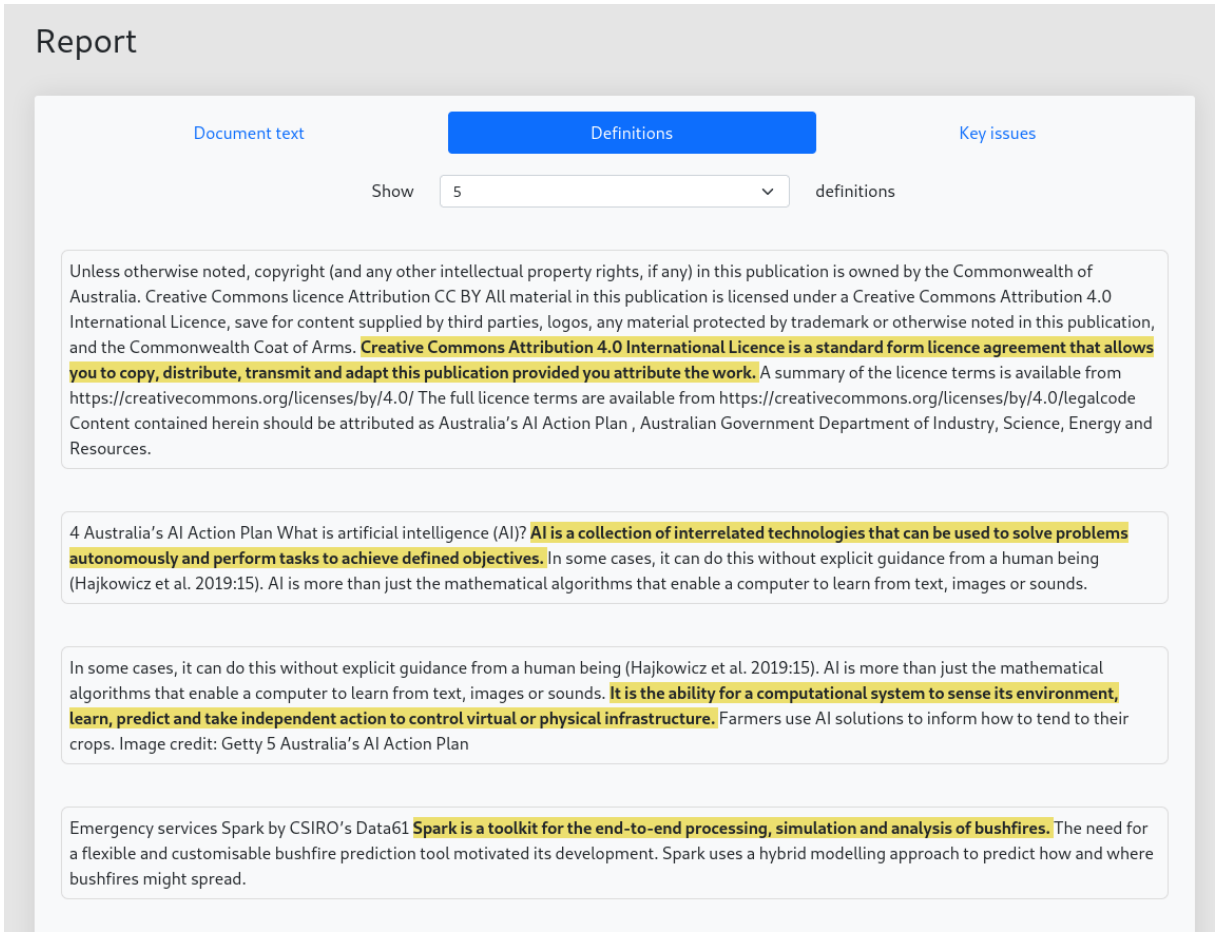

Figure 10.5: Document report - definitions

Users may explore documents in terms of definitions and terms of Key issues. In the definitions window (figure 10.5), definitions that have the most significant probability are shown. In the key issues window (in figure 10.6), a user may choose definitions from a drop-down menu.

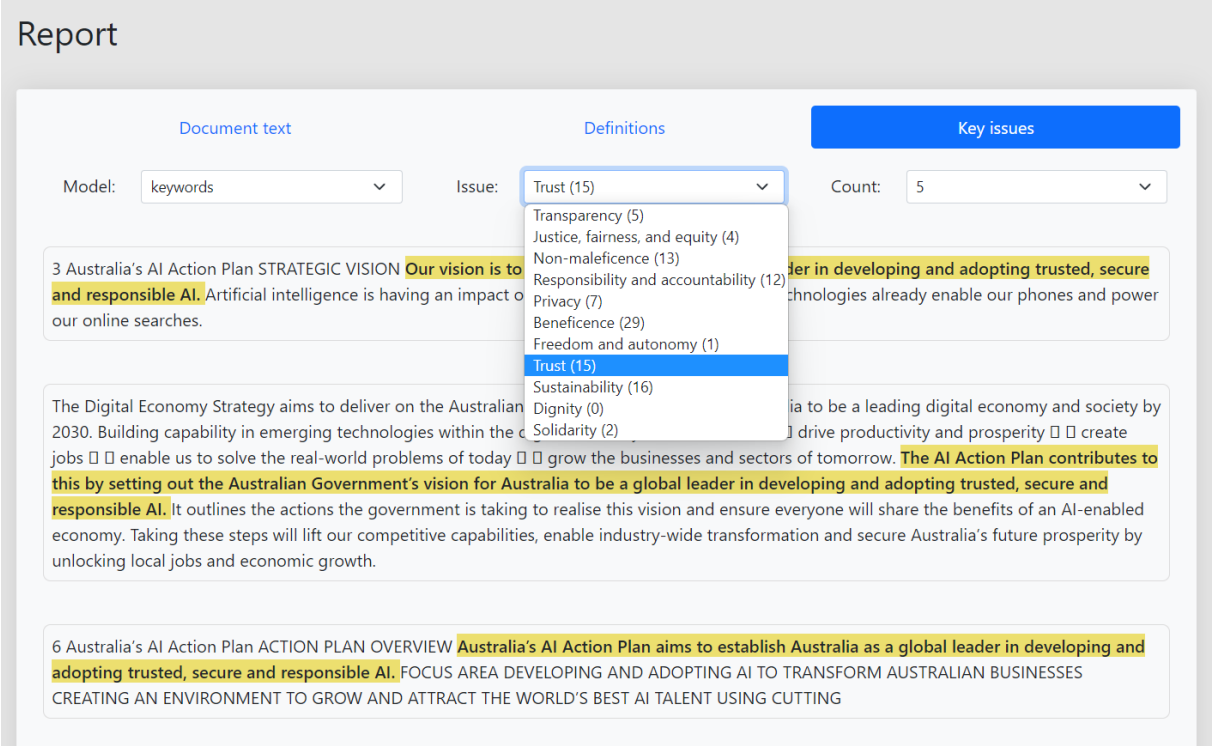

Figure 10.6: Document report - issues. Issues can be chosen from the drop down menu.

11. Conclusions

To conclude, we proposed a solution to the problem of how to obtain the AI regulations and policies and how to analyze them.

We described the database of AI policies and documents. We have shown how it works, how it is filled using scrapers, our processes, and synchronized.

We implemented an analytical user interface that can be accessed through a website. We described uploading the document, displaying it, viewing the topics and definitions.

We believe that this solution will aid the policymakers in document analysis, and the data scientists will benefit from the number of documents that can be further analyzed.

In the context of future work, it is worth noting that the database can be easily expanded by adding more scrapers and scheduling them in Airflow. Similarly, additional models can be added in User Interface and analysis modules. The models can also be improved. Especially the keyword issue model can be treated as a baseline and act as an initial benchmark to evaluate future models.

12. Division of work

This thesis is joint work of Piotr Piątyszek, Jakub Wiśniewski and Hanna Zdulska.

Chapter/Appendix	Contributor
Abstract, Introduction	JW
Chapter 1	PP
Chapter 2	HZ
Chapter 3	HZ
Chapter 4	HZ
Chapter 5	Joint work
Chapter 6	PP
Chapter 7	JW
Section 8.1	HZ
Section 8.2	PP
Chapter 9	PP
Chapter 10	PP
Chapter 11	JW
Appendix A	JW
Appendix B	JW

Table 12.1: Chapter's biggest contributors

A. Measuring the precision of definition extraction model

To show how we labelled predicted definitions we will randomly sample 10 true positives and 10 false positives out of nearly 800 manually labelled sentences. The most of the false positives do not give full picture (if at all) of what a subject is or does.

A.0.1. True Positives

- GANs consist of a generator neural network and a discriminator neural network that are pitted against one another in order to anonymize data.
- Section 4.1 "Delivery Modes" includes a detailed presentation of the common model proposed for delivery articulated along three axes (large shared infrastructure, upgrade and networking of MS capacities and use of capacities).
- 22 Term Definition Permissive License Software license which confers the right to redistribute, alter and create proprietary derivative works without restriction.
- In areas where no Union legislation exists, the principle of mutual recognition means that goods that are lawfully marketed in one Member State enjoy the right to free movement and can be sold in another Member State, unless the Member State concerned has grounds to oppose the marketing of the goods, provided that such a0restriction is non-discriminatory, justified by legitimate public interest objectives, as set out in Article036 of the Treaty on the Functioning of the European Union (TFEU) or recognised by the case-law of the Court of Justice, and proportionate to the aim pursued.
- The burden of proof is the central component of the Directive that triggers the right for compensation.
- In 2017, the agency issued a grant to a robotics laboratory at the University of Michigan to develop a prototype autonomous vehicle for rural delivery routes, the Autonomous Rural Delivery Vehicle (ARDV).
- Analytical processes are a tool to inform human decision-making and should never entirely

replace human oversight, although the extent of human oversight may depend on the significance of the decision and on other safeguards in place.

- Such technology can be in the form of translation memory software which inserts segments from translation memories (see above), where there is a likelihood that a segment has been previously translated.

- What is now CIRA was originally called the Accounting Quality Model (AQM) and was used to monitor inappropriate managerial discretion in the usage of accounting accruals.

- The PHRaE process runs alongside any development of a proposed use of client data by the Ministry, and prompts the project owner to detail and discuss the way in which their project will use personal information.

A.0.2. False Positives

- The draft provided for the liability of the supplier of services for direct damage caused (by his fault in the provision of the service) to the health and physical integrity of persons or their property.

- In addition, the Radio Equipment Directive, the Regulations on medical devices and on in vitro diagnostic, may include software.

- Horizon 2020 also covers, to a certain extent, funding for research on technologies that support teaching and learning.

- 12 Courts routinely consult AI-based risk-assessment tools, referring to automatically-generated risk scorecards in making sentencing decisions.

- Decisions about specific individuals are the responsibility of the relevant schools or education providers.

- Agencies often hire third-party entities, which also advise private-sector clients on commenting campaigns, to build better comment analysis tools.

- Neural word embeddings are trained using a large amount of text that can originate from patent and nonpatent literature.

- Immigration New Zealand uses a range of operational algorithms to manage risk to New Zealand and ensure that the approximately 6.7 million travellers who pass through the border annually receive speedy, consistent, immigration decisions.

- As a private instrument, it leaves to the parties (i.e. to consumers) the burden to enforce its rules, i.e. to raise a claim in case of damage caused by a defective product.
- Sharing best practices, identifying synergies and aligning action where relevant will maximise the impact of investments in AI and help the EU as a whole to compete globally.

B. Measuring the precision of keyword topic model

We manually labelled each issue that was presented in the table 7.3. To show how this process looked we provide a table for issue *beneficence* - here we focused on the fact that the AI systems and technologies should benefit people who use them. Here most issues were connected with the subject of beneficence. We labelled sentence as positive when the AI use was beneficial to some parties. Below we show which of which of them we classified as false positive and true positive.

B.0.1. True Positives

- Artificial Intelligence (AI) is a fast evolving family of technologies that can bring a wide array of economic and societal benefits across the entire spectrum of industries and social activities.
- However, the same elements and techniques that power the socio-economic benefits of AI can also bring about new risks or negative consequences for individuals or the society.
- It is in the Union interest to preserve the EUs technological leadership and to ensure that Europeans can benefit from new technologies developed and functioning according to Union values, fundamental rights and principles.
- The proposal also responds to explicit requests from the European Parliament (EP) and the European Council, which have repeatedly expressed calls for legislative action to ensure a well-functioning internal market for artificial intelligence systems (AI systems) where both benefits and risks of AI are adequately addressed at Union level.
- The EP Resolution on a Framework of Ethical Aspects of Artificial Intelligence, Robotics and Related Technologies specifically recommends to the Commission to propose legislative action to harness the opportunities and benefits of AI, but also to ensure protection of ethical principles.
- (3) Artificial intelligence is a fast evolving family of technologies that can contribute to a wide array of economic and societal benefits across the entire spectrum of industries and social activities.

- Nonetheless, this Regulation should not hamper the development and use of innovative approaches in the public administration, which would stand to benefit from a wider use of compliant and safe AI systems, provided that those systems do not entail a high risk to legal and natural persons.

B.0.2. False Positives

- The costs incurred by operators are proportionate to the objectives achieved and the economic and reputational benefits that operators can expect from this proposal.
- To achieve that objective, rules regulating the placing on the market and putting into service of certain AI systems should be laid down, thus ensuring the smooth functioning of the internal market and allowing those systems to benefit from the principle of free movement of goods and services.
- (37) Another area in which the use of AI systems deserves special consideration is the access to and enjoyment of certain essential private and public services and benefits necessary for people to fully participate in society or to improve ones standard of living.
- Natural persons applying for or receiving public assistance benefits and services from public authorities are typically dependent on those benefits and services and in a vulnerable position in relation to the responsible authorities.
- If AI systems are used for determining whether such benefits and services should be denied, reduced, revoked or reclaimed by authorities, they may have a significant impact on persons livelihood and may infringe their fundamental rights, such as the right to social protection, non- discrimination, human dignity or an effective remedy.
- AI suppliers should benefit from a minimal but clear set of requirements, creating legal certainty and ensuring access to the entire single market.
- AI users should benefit from legal certainty that the high-risk AI systems they buy comply with European laws and values
- Consumers should benefit by reducing the risk of violations of their safety or fundamental rights.

Bibliography

Github continuous integration. URL https://docs.github.com/en/actions.

Forty-two countries adopt new OECD Principles on Artificial Intelligence, 2019. URL https://www.oecd.org/science/forty-two-countries-adopt-new-oecd-principles-on-artificial-intelligence.htm.

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

A. Airflow. Apache airflow, 2022. URL https://airflow.apache.org/docs/apache-airflow/stable/index.html.

R. Anderson. The rashomon effect and communication. *Canadian Journal of Communication*, 41(2), 2016.

S. Barocas, M. Hardt, and A. Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019. http://www.fairmlbook.org.

H. Bast and C. Korzen. A benchmark and evaluation for text extraction from pdf. In *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 1–10, 2017. doi: 10.1109/JCDL.2017.7991564.

C.-H. Chuan, W.-H. S. Tsai, and S. Y. Cho. Framing artificial intelligence in american newspapers. AIES '19, page 339344, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450363242. doi: 10.1145/3306618.3314285.

E. Dabbas. advertools - online marketing productivity and analysis tools, 2018. URL https://advertools.readthedocs.io/en/master/index.html.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2019.

Docker Inc. Use the docker command line, 2020. URL https://docs.docker.com/engine/reference/commandline/cli/.

Docker Inc. Overview of docker compose, 2021a. URL https://docs.docker.com/compose/.

Docker Inc. Overview of docker hub, 2021b. URL https://docs.docker.com/docker-hub/.

V. Driessen. Redis queue, 2022. URL https://python-rq.org/.

European Commission. Proposal for a regulation of the european parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts, 2021. URL https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52021PC0206.

J. Gesley, T. Ahmad, E. Soares, R. Levush, G. Guerra, J. Martin, K. Buchanan, L. Zhang, S. Umeda, A. Grigoryan, et al. Regulation of Artificial Intelligence in Selected Jurisdictions. page 138. URL https://digitalcommons.unl.edu/scholcom/177/.

M. Honnibal and M. Johnson. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1373–1378, Lisbon, Portugal, Sept. 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1162. URL https://aclanthology.org/D15-1162.

A. Jobin, M. Ienca, and E. Vayena. The global landscape of ai ethics guidelines. *Nature Machine Intelligence*, 1(9):389399, Sep 2019. ISSN 2522-5839. doi: 10.1038/s42256-019-0088-2. URL http://dx.doi.org/10.1038/s42256-019-0088-2.

J. Kouraklis. *MVVM as Design Pattern*. 10 2016. ISBN 978-1-4842-2213-3. doi: 10.1007/978-1-4842-2214-0_1.

H. Krekel, B. Oliveira, R. Pfannschmidt, F. Bruynooghe, B. Laugher, and F. Bruhin. pytest 5.4, 2004. URL https://github.com/pytest-dev/pytest.

N. Li and B. Zhang. The research on single page application front-end development based on vue. In *Journal of Physics: Conference Series*, volume 1883, page 012030. IOP Publishing, 2021.

N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan. A survey on bias and fairness in machine learning. *ACM Comput. Surv.*, 54(6), jul 2021. ISSN 0360-0300. doi: 10.1145/3457607. URL https://doi.org/10.1145/3457607.

I. Montani, M. Honnibal, M. Honnibal, S. V. Landeghem, A. Boyd, H. Peters, P. O. McCann, M. Samsonov, J. Geovedi, J. O'Regan, G. Orosz, D. Altinok, S. L. Kristiansen, Roman, E. Bot, L. Fiedler, G. Howard, W. Phatthiyaphaibun, Y. Tamura, S. Bozek, murat, M. Amery, B. Böing, P. K. Tippa, L. U. Vogelsang, B. Vanroy, R. Balakrishnan, V. Mazaev, and GregDubbin. spaCy: Industrial-strength Natural Language Processing in Python. 2020. doi: 10.5281/zenodo.5764736.

R. Navigli, P. Velardi, J. M. Ruiz-Martínez, et al. An annotated dataset for extracting definitions and hypernyms from the web. In *LREC*. Citeseer, 2010.

J. Nivre and J. Nilsson. Pseudo-projective dependency parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 99–106, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219853. URL https://aclanthology.org/P05-1013.

P. Office. *Cellar : the semantic repository of the Publications Office*. Publications Office, 2018. doi: doi/10.2830/03072.

T. Panch, H. Mattie, and L. A. Celi. The inconvenient truth about AI in healthcare. *npj Digital Medicine*, 2(1):77, Aug. 2019. ISSN 2398-6352. doi: 10.1038/s41746-019-0155-4. URL https://doi.org/10.1038/s41746-019-0155-4.

D. Paul, G. Sanap, S. Shenoy, D. Kalyane, K. Kalia, and R. K. Tekade. Artificial intelligence in drug discovery and development. *Drug Discov Today*, 26(1):80–93, Jan. 2021. ISSN 1878-5832. doi: 10.1016/j.drudis.2020.10.010. URL https://pubmed.ncbi.nlm.nih.gov/33099022. Edition: 2020/10/21 Publisher: Elsevier Ltd.

A. Ronacher. Flask web development, one drop at a time, 2022. URL https://flask.palletsprojects.com/en/2.0.x/.

V. Sanh, L. Debut, J. Chaumond, and T. Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2020.

M. Y. Shaheen. AI in Healthcare: medical and socio-economic benefits and challenges. *ScienceOpen Preprints*, 2021. Publisher: ScienceOpen.

S. Spala, N. A. Miller, F. Dernoncourt, and C. Dockhorn. Semeval-2020 task 6: Definition extraction from free text with the deft corpus. *arXiv preprint arXiv:2008.13694*, 2020.

B. C. Stahl, J. Timmermans, and B. D. Mittelstadt. The ethics of computing: A survey of the computing-oriented literature. *ACM Comput. Surv.*, 48(4), feb 2016. ISSN 0360-0300. doi: 10.1145/2871196. URL https://doi.org/10.1145/2871196.

C. Strauch, U.-L. S. Sites, and W. Kriha. Nosql databases. *Lecture Notes, Stuttgart Media University*, 20:24, 2011.

The Linux Foundation, 2022. URL https://github.com/opencontainers/image-spec.

C. Urmson and W. R. Whittaker. Self-driving cars and the urban challenge. *IEEE Intelligent Systems*, 23(2):66–68, 2008. doi: 10.1109/MIS.2008.34.

A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446.

E. J. Whitehead and Y. Y. Goland. Webdav: A network protocol for remote collaborative authoring on the web. In *ECSCW*, 1999.

M. Whittaker. *AI Now Report 2018*. PhD thesis, New York University, 2018.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2020.

E. You. The progressive javascript framework, 2022. URL https://vuejs.org/.

H. Zhang, Y. Wang, Z. Zhang, F. Guan, H. Zhang, and Z. Guo. Artificial Intelligence, Social Media, and Suicide Prevention: Principle of Beneficence Besides Respect for Autonomy. *The American Journal of Bioethics*, 21(7):43–45, 2021. doi: 10.1080/15265161.2021.1928793. URL https://doi.org/10.1080/15265161.2021.1928793. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/15265161.2021.1928793.

List of symbols and abbreviations

ML	Machine Learning
NLP	Natural Language Processing
DAG	Directed Acyclic Graph
CELEX	Communitatis Europeae Lex (European Union law database)
AI	Artificial Intelligence
UI	User Interface
API	Application programming interface
OCI	Open Container Initiative
CLI	Command line interface
MVVM	Model-View-ViewModel
REST API	Representational state transfer application programming interface
NoSQL	Not only SQL
SQL	Structured Query Language

List of Figures

List of tables

List of appendices