”ALEXANDRU-IOAN CUZA” UNIVERSITY OF IAȘI

FACULTY OF COMPUTER SCIENCE

MASTER’S THESIS

Approaches Towards a Romanian Chatbot for Understanding Legislation

Cristian-Mihai Roșu

July, 2023

Coordinator

Conf. Dr. Diana Trandabăț

”ALEXANDRU-IOAN CUZA” UNIVERSITY OF IAȘI

FACULTY OF COMPUTER SCIENCE

Approaches Towards a Romanian Chatbot for Understanding Legislation

Cristian-Mihai Roșu

July, 2023

Coordinator

Conf. Dr. Diana Trandabăț

Contents

Introduction

0.1 Presentation

In a world where information is overflowing and abundant, it has become a skill for a person to be able to search for and filter sources as well as information itself in order to learn or even find out facts for immediate use. This is especially true when looking towards institutions which specialize in processing and distributing information for the general public such as news outlets and even schools.

Among those, however, arguably the most important one and, paradoxically, the most bothersome one is the State itself or, better said, the state institutions which create laws and regulations for the population. This is information that affects each individual citizen and thus it is paramount that it is publicly available but also easily available and understood.

The paradox manifests such that, although this information is available to anyone for free, the regular citizen is in a perpetual uphill battle against the State when it comes to the rights available to him and how, most of the time, they are either missing pieces of information and context or misunderstanding the text of the law completely, in the worst case.

As far as Romania is concerned, there are a multitude of official websites[1] and other sources[2] which make searching for legislation easier but not less demanding. It is a given that a person should know what they are looking for beforehand, but they should also be able to understand their result. Usually, the results are paragraphs extracted verbatim from the legislation itself and so they are hard to understand as the vocabulary is mostly comprised of specialty terms and phrases.

The recent popularity of friendly interfaces such as ChatGPT[3], and chatbots in general, has made it clear that having access to a sea of information is not as important as (1) being able to find the right information and (2) being able to understand it with little difficulty - in other

[1] `https://monitoruloficial.ro/e-monitor/`, `https://legislatie.just.ro/`

[2] `https://www.cdep.ro/pls/legis/legis_pck.frame`, `https://www.lege-online.ro/`

[3] `https://openai.com/blog/chatgpt`

words, presentation.

0.2 Motivation

The purpose of technology has always been to innovate and to make life easier and more comfortable, as such it is no surprise that the process of learning should be affected by it as well.

It goes without saying that having an expert break down a complex concept into easier-to-digest pieces of information is one of the most tried-and-true methods of teaching. Consequently, it is still the cornerstone of our schooling systems. Considering that, the apparent sudden mainstream rise of tools like ChatGPT should not come as much of a surprise.

The field of Natural Language Processing (NLP) has been researching and developing the systems necessary for such a tool to exist for a long time even before this apparent innovation. The concept of a ”chatbot” or ”chatterbot” has existed well before the 2000s in the form of any program capable of reproducing a conversation [1]. These evolved into templating systems, essentially matching a certain sentence or question to an already-prepared templated answer and later they flourished with the rapid development of NLP techniques and Machine Learning (ML) methods such as Deep Learning. Not to mention, virtual assistant applications like the Google Assistant[4], Siri[5] or Alexa[6] have also been around for a few years, although, as their name implies, these tools would only assist a user with simple tasks (setting alarms, calendar events, factual questions) and would redirect a user's more complex questions to a source instead of interpreting its contents directly.

ChatGPT is the latest addition to OpenAI's[7] GPT (Generative Pre-trained Transformer) models and it has seen overwhelming success exactly because of its ability to explain difficult concepts in a step-by-step manner combined with the ability to maintain conversation and keep track of its history. This tool is by no means perfect but it is a definite step forward in making people's lives comfortable and more assured when it comes to searching for and understanding information.

Considering the above, this paper aims at creating a friendly interface, similar to the ones described above, in the context of helping regular Romanian citizens better understand the country's legislation.

[4] https://assistant.google.com/

[5] https://www.apple.com/siri/

[6] https://developer.amazon.com/en-GB/alexa

[7] https://openai.com/

0.3 General objectives and contributions

The main objective of this project is to create a simple chatbot interface capable of querying and explaining the legal text of Romanian laws and using the result in a Question Answering (QA) manner based on a custom-made knowledge base of legislation.

In the paper, the proposed system is described along with the creation process of the custom dataset and how it compares to others in the same domain.

Contributions:

- Single-page web application user interface
- Two unique custom data sets for the Romanian language:
    1. one compiling the Romanian codes of law into a single easily-parsable structure; in the context of this paper, the codes of law considered are: The Constitution, The Civil Code, The Civil Procedure Code, The Penal Code, The Penal Procedure Code, The Fiscal Code, The Fiscal Procedure Code, The Labor Code and The Administrative Code.
    2. one representing a corpus of the most common words in the Romanian language together with their frequency relative to a collection of the best articles[8][9] from Romanian Wikipedia[10].

All relevant code can be found at `https://github.com/Nenma/ro-legal-bot`.

0.4 Structure of the paper

The structure of this paper consists of 3 main chapters, each described in more detail in the following:

Chapter 1: Related work provides an overview of similar papers and describes the procedures relevant to the task at hand while highlighting the original contributions.

Chapter 2: Dataset and Methods presents the project in detail starting with the creation of the custom data set for legislation, as well as the challenges in creating it, together with the outline of the system architecture in its entirety and the user flow.

[8] `https://en.wikipedia.org/wiki/Wikipedia:Featured_articles`
[9] `https://en.wikipedia.org/wiki/Wikipedia:Good_articles`
[10] `https://ro.wikipedia.org/wiki/Pagina_principal%C4%83`

Chapter 3: Evaluation and Results evaluates the responses of the proposed system against other chatbots and compares the results.

Conclusions makes a short summary of the project overall and reiterates the outcome of the evaluation while also proposing ideas for future work.

Chapter 1

Related work

1.1 Romanian Chatbots

Despite the undeniable popularity of chatbots, or human-simulation conversational programs in general, there have been few attempts to establish one dedicated to the Romanian language and specialized on all of its particularities. As it is, the most robust and complex dialogue systems for Romanian are more often than not also proficient in many other languages, and offer a language-specific interface and experience through translation methods.

It is the case that technology innovations may be harder to adopt in our society, speculation partly confirmed by the sizeable number of papers studying the normal citizen's opinions and reaction to chatbot technology.

With day-to-day life as focus, [11] chooses 10 online stores in Romania to showcase the experience of a normal client and concludes that regardless of the high potential when correctly applied, most shopping assistants exhibit poor awareness and display low-quality content and products which actually deters the continuation of a user journey on the platform, to the point of dissatisfaction. The authors of [2], meanwhile, examine the apparent slow acceptance rate of it in the banking industry, looking towards analysing the customer feedback for reasons as to why, chief among which are usually heightened privacy concerns and underappreciated usefulness. In a similar vein, [8] questions the ethical implications of allowing such technology to be used along psychotherapeutic sessions and processes. Looking at the more physical reality, [12] conducts a survey for drivers asking for their position next to the rapid emergence of autonomous, self-driving vehicles and noting unsurprising results: the average person would not fully trust a system that, in the end, has its success and failure bound to the amount of personal and, potentially, private information that it has access to.

Even still, recent progressive approaches and research looks towards improving current systems by creating task-specific chatbots that provide a service. Among more humanitarian efforts, [10] confidently pave the way to bettering the national emergency system by proposing a couple of solutions for human-to-computer critical communications based on Interactive Voice Response (IVR) and the Botpress chatbot platform[1]. They judged the solutions with real-world data and decided that telephony IVR best fits the national safety requirements and specifications, being capable of emotional state recognition during emergency situations.

The system proposed in this paper, RoLegalBot, aims at fulfilling a similar space of purpose for a normal citizen, as far as keeping informed on legislation is concerned.

1.2 Natural Language Processing (NLP)

Although the methods of text summarization and text simplification have been successfully applied in the past, they have been employed more often than not for news articles and scientific papers. Despite the size of the Romanian legislation and the volume of law modifications, decrees and ordinances that are put out everyday, these texts have not been previously considered for such a task. Even so, that is not necessarily the case for states apart from Romania and so below we look over a selection of papers that tackle the issue for similar texts.

1.2.1 Text summarization of legal text

In [9] the authors undertake the task of creating a corpus specially made for automatic summarization for US legislation. They highlight the uniqueness of the custom data set they built for the specific task and train and test it using a set of extractive summarization approaches.

It is important to note that apart from extractive, there is also abstractive summarization. The difference between the two is that extractive summarization, as the name implies, consists of pulling subtexts from the original text verbatim and concatenating them to create one coherent summary while for the abstractive approach the aim is to create a system that is capable of deeper understanding of meaning and structure behind the original text such that is can generate a new text based on it, serving as a summary.

They point out that existing pretrained abstractive models produce ungrammatical results and the size of their data set does not allow for fine-tuning or retraining. They compare their approaches to existing unsupervised baselines with positive results and conclude that good sum-

[1] https://botpress.com/

maries share common language elements. Moreover, even though the summarization method was trained on US bills and corresponding summaries, it can work effectively when transferred to California bills as well, without employing additional human written summaries.

To clarify, the evaluation metric used is ROUGE (Recall-Oriented Understudy for Gisting Evaluation) which is, in fact, a set of metrics for evaluating both automatic summaries and machine translations. The main idea is that it uses a reference summary or translation - usually human-made - to compare against the automatic result.

It is important, then, to consider a proper structure when building out the data set and consider the language particularities of summaries in general when implementing the approach, including contextual features.

The authors of [17] challenge both approaches to summarization using European legislation documents. They set a baseline model to compare against using an extractive method based on weighted frequency tokens of sentences. This was done due to ease of implementation as well as time efficiency since the method does not require training and is based just on the contents of the document. As for the methods used in the experiment, although they consist of a nice selection of both extractive and abstractive, they all have one common trait: they are all based on encoder-decoder transformer models.

A transformer model is basically a neural network (NN) that learns context and meaning by tracking relationships in sequential data. Among the selection, all but one are using the BERT (Bidirectional Encoder Representations from Transformers) model, the exception achieving the overall best results as per the paper. This one is based on the T5 (Text-To-Text Transfer Transformer) model. These are two of the most widely used transformer models, and for good reason, since they achieve solid results over a wide range of NLP tasks, including summarization. The similarity between them lies in the shared use of bidirectional models for masking language, although each employs a slightly different version of this. The main idea is that for any given word or token, its representation is derived from both the left and the right context of it. The efficiency of T5 is similarly evaluated using the ROUGE metrics in the paper.

[18] approach the issue of extractive summarization of Bulgarian legislation by taking advantage of the inner structure of the legal documents themselves. They likewise mention the lack of pre-trained models and resources in general for this language compared to English and are thorough in presenting their data collection step. This is a key detail of the paper as the data resources mentioned extend to Romanian legislation as well.

They present their own solution and compare it successfully against the same baseline model as in the previously mentioned paper and against a specialized BERT model trained for

Bulgarian, Czech, Polish and Russian named SlavicBERT, using the ROUGE metrics.

1.2.2 Text simplification of legal text

In [7] the aim of the authors is to explore the potential of existing state-of-the-art text simplification methods as well as the challenges of using existing automatic evaluation metrics for the legal domain in general using sentences from the LEDGAR dataset (proposed and created in [14]). This dataset consists of contracts of the US Securities and Exchange Commission (SEC).

They employ a range of both custom supervised and custom unsupervised methods of text simplification tackling three types: lexical, sentence splitting and end-to-end. The task of simplifying text is a great undertaking and so it is natural that it can be performed at different levels. Lexical simplification usually entails replacing words that are deemed too complex with equivalent, simpler ones. Here, the authors define a word as being too complex if its frequency is low enough in respect to a dataset of Wikipedia text with a vocabulary size of approximately half a million tokens. Syntactic simplification generally tackles the complexity of the sentence structure rather than the individual word meaning, thus resorting to different ways to split it into shorter ones. End-to-end approaches encapsulate both styles and ideally split and rephrase the original sentence while maintaining fluency.

The various model outputs are evaluated based on meaning preservation, syntactic simplicity, fluency, hallucination and readability measures. It bears importance to mention that in this context hallucination refers to the new content found in a computed result that is not present in its input. It follows that the authors measure this metric using the percentage of entities in the output not found in the input; Readability is checked using Flesch Kincaid (FK), SMOG and the Automatic Readability Index (ARI) to work out the reading age of the resulting simplified text; Syntactic simplicity is computed using the average depth of the sentences' dependency parse trees; Fluency is estimated using perplexity score from GPT-2; For meaning preservation they use a selection of custom defined metrics, each of which are described succinctly in the paper and are worth the readers' peruse.

However, these automatic evaluation methods are also put side-by-side with manual, human evaluation of two legal experts. There is quite a discrepancy observed between the automated approached and the manual text simplifications, but surprisingly there is also quite a difference between the experts themselves. The authors themselves point out that further investigation is needed to correctly understand the comparisons made by the two.

They conclude by indicating that text simplification for legal texts is still in its nascent stages and would benefit from having a dedicated legal lexicon. Words used for replacement sometimes do not fit the legal context while splitting and end-to-end methods mostly achieve satisfying results at the cost of either fluency or meaning preservation. This paper brings to light many of state-of-the-art approaches for general text simplification while highlighting the shortcomings that are made visible after bringing them to the legal domain.

Works such as [5] and [4], although each introducing their own solution for text simplification of legal text (Philippine Senate and House bills and US Supreme Court legal cases, respectively), they both approach the issue in a similar fashion: a lexical simplification step followed by a syntactic simplification step. The custom systems are derived from state-of-the-art methods as the ones mentioned in [7] and use similar evaluation metrics.

Chapter 2

System presentation

2.1 Architecture

The proposed solution is RoLegalBot, a single-page web application serving a simple and intuitive UI, similar to a chat room in appearance. Using this interface, a user would be able to send messages and queries concerning the Romanian Codes of law and receive appropriate response from the chatbot, with the added benefit of having annotated sections of the more complex responses with the purpose of making everything as clear as possible for the user.

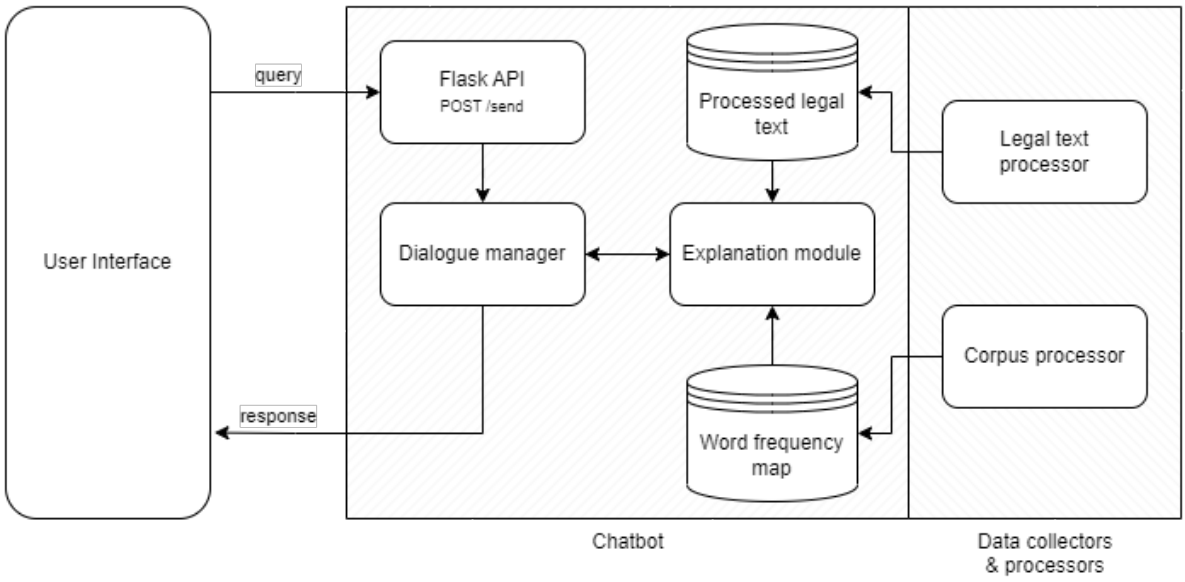

Figure 2.1: **Chatbot system architecture**

Figure 2.1 presents an overview of the system with a focus on the back-end modules, as well as the general flow of user interaction. As shown, the back-end is split into two independent, but nevertheless tied together, sub-systems.

On one hand, there is the chatbot itself, at the heart of which lies the Dialogue Manager responsible with parsing the user query and choosing the appropriate handling method for generating the response before sending it back to the UI, relying on two separate data sources in order to do it.

On the other hand, there are the Data Collectors & Processors that crawl the suitable web sources, processing the text data in order to create what will end being the knowledge base of the chatbot. These are further split into two processors: one for handling the legal text and one for the general corpus meant as reference for comparison against the more specialized legal one. These two are described in more detail in section 2.4 Dataset.

2.2 User interface

The single-page web application (Figure 2.2) is rendered using a single HTML document that gets dynamically updated using a Javascript script every time a new Q&A exchange takes place. The interface is served at the root endpoint of a Python Flask API.

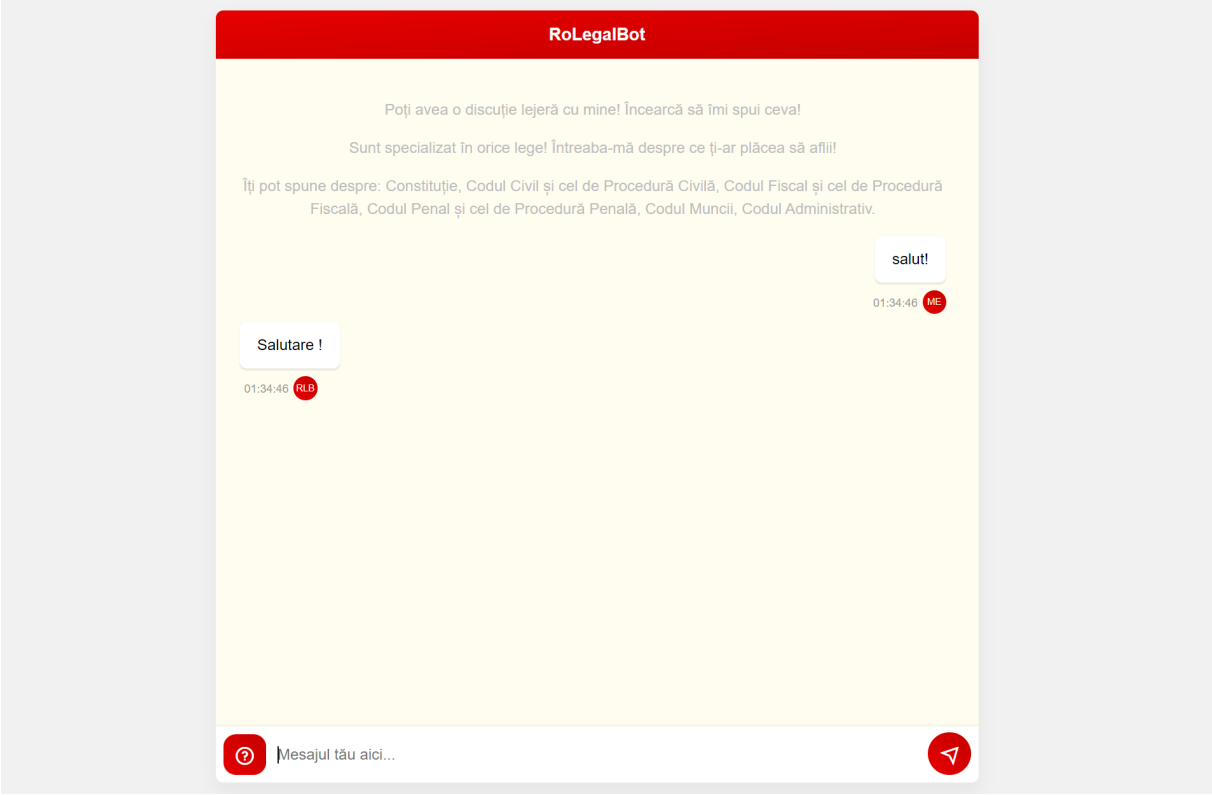

Figure 2.2: **Web application UI**

Additionally, the API exposes a `/send` POST endpoint (Figure 2.3) which is asynchronously called each time a new query is received from the user, representing the body of

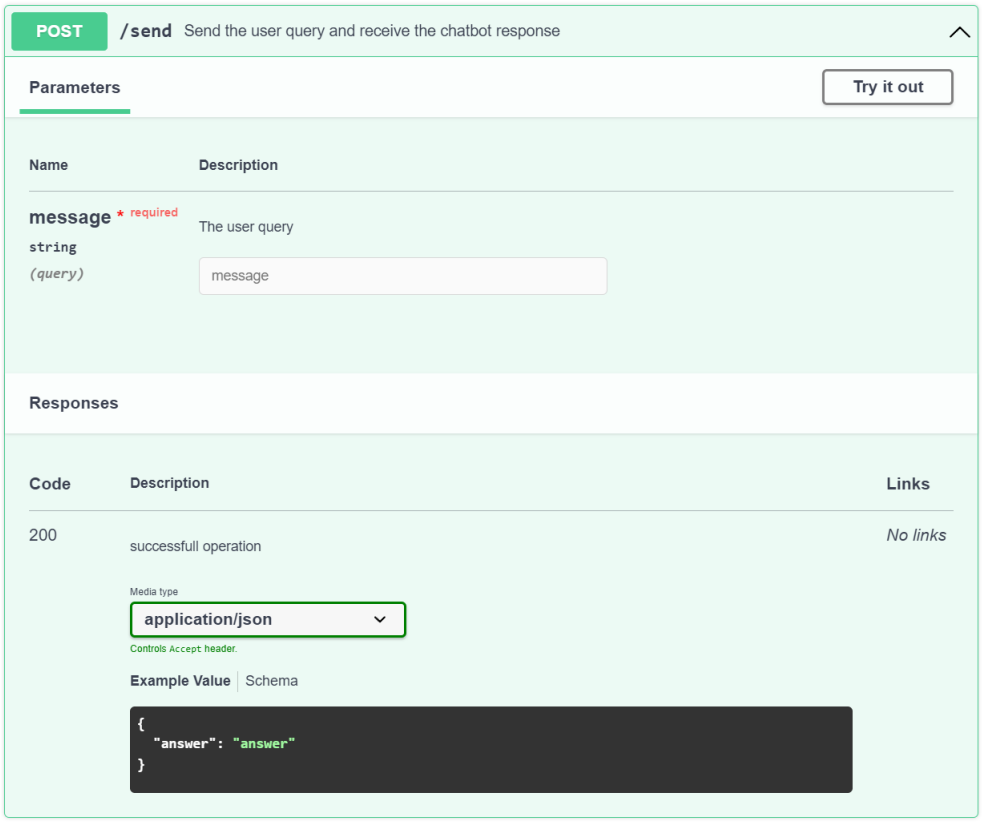

Figure 2.3: **Flask API `/send` POST endpoint OpenAPI specification**

the POST request.

The interactive elements of the UI consist of two buttons and one text area, all located in the bottom area of the chat box. The left *help* button overlays a message instructing the user on the Codes of law that it is capable of gathering information from and answering questions about, while the right *send* button forwards the contents of the text area to the back-end modules.

2.3 Chatbot system

Using the categorization criteria presented in [1], RoLegalBot is a closed-domain, interpersonal, conversational and rule-based chatbot whose functionality is publicly available and open-source on a GitHub[1] repository. As its purpose is strictly facilitating legal text searching and understanding, it is more efficient for the knowledge domain to be restricted as this greatly improves the time spent looking for candidate solutions and thus the answer generation time. It exhibits limited personality traits since its conversational capabilities are based on AIML's (Artificial Intelligence Markup Language) [16] pattern recognition and pattern matching, but thanks to that, the language it employs is simple and friendly. The knowledge base it uses is

[1] `https://github.com/`

```
<aiml version="1.0.1" encoding="UTF-8">
   <category>
      <pattern>CE EŞTI</pattern>
      <template>
         Sunt doar un micuţ bot menit să te ajute cu
         înţelegerea cu uşurinţă a legislaţiei din România!
      </template>
   </category>
   <!-- ... -->
   <category>
      <pattern>CÂTE ARTICOLE SUNT ÎN *</pattern>
      <template>
         <think>
            <set name="legal_code_size">
               <star index="1" />
            </set>
         </think>
      </template>
   </category>
   <category>
      <pattern>CÂTE ARTICOLE ARE *</pattern>
      <template>
         <srai>CÂTE ARTICOLE SUNT ÎN <star index="1" /></srai>
      </template
   </category>
</aiml>
```

Figure 2.4: **Snippet of RoLegalBot's AIML XML**

text data retrieved from specific web sources and it works as an indexed data base from which it searches for and selects the text that best matches the user query.

As an overview, AIML is an XML-based markup language specifically designed with the purpose of creating programs capable of simulating human conversation. It is tag-based, meaning all of the file's contents are tag elements which make up the rules that define the conversational potential of the chatbot. Based on this fact, it follows that the more rules, and so more tags, are added, the more the chatbot is able to respond to, thus appearing more intelligent and even more human.

As shown in Figure 2.4, the basic unit of AIML is the `category` tag which encapsulates one single conversation scenario.

Within this tag, we define what the expected user query is, here labeled `pattern`, and what the corresponding answer should be, here labeled `template`. The pattern text is case insensitive and only accepts letters and numbers, as well as a selection of wildcard symbols, notably the asterisk (*) and the caret (^), used to capture a selection of one or more words and zero or more words of the input, respectively. In the case of RoLegalBot, we would always need to capture at least one word and as such the `star` tag is prevalent throughout the scenarios.

This information retention potential is fully realised once paired with AIML's predicate

functionality. This is achieved using the `set` tag and the `name` attribute, resulting in what is essentially a locally stored variable. We can access this information in a separate scenario by name using the `get` tag if needed, but more importantly we can also access it by making use of Python's `python-aiml` module within the back-end component.

Flexibility in handling different input queries that should be answered in the same way is attained by employing the `srai` tag which in effect works as a redirection to a "root" pattern. The more form variance covered in a pattern, the more formulations of the same query can be resolved in an identical manner.

The Dialogue Manager is, then, intrinsically linked to a `Kernel` object instance that is built upon RoLegalBot's AIML file, its user query handling behaviours mirroring the conversation scenarios described in the markup. This decision handling happens every time the `/send` POST endpoint gets called.

Concretely, RoLegalBot handles 4 separate conversation scenarios or, more accurately, information requests, each with a considerable amount of sentence variation. The scenarios cover queries in the form of:

1. What articles are of interest to me concerning X topic?
2. How many total articles are there in Code of law X?
3. What is article X of Code of law Y?
4. Define X for me

The first two types of questions are meant to act as starting points for the conversation chain. Intuitively, the user would be inclined to ask about any topic relevant to the legal field and therefore the first question type it the one with the most form variations, to account for as many ways of conveying the query as possible. The structure of the response here is essentially a list of all the Codes of law in which the search term appears at least once, each further enumerating the article indexes in which the term is used (Figure 2.5). In case the search term does not occur not even once in the entire collection of legal texts, an appropriate message is sent instead prompting the user to verify spelling or try another form of the word.

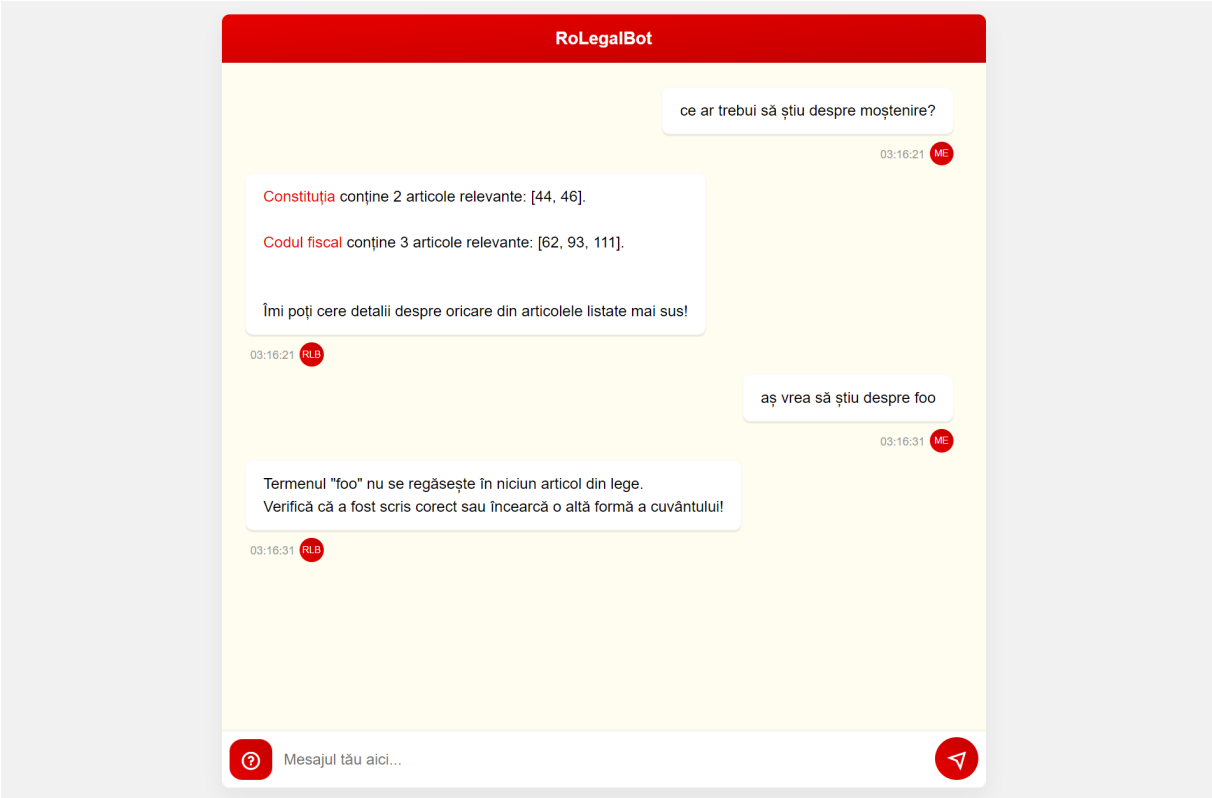

Figure 2.5: **Type 1 query answer example**

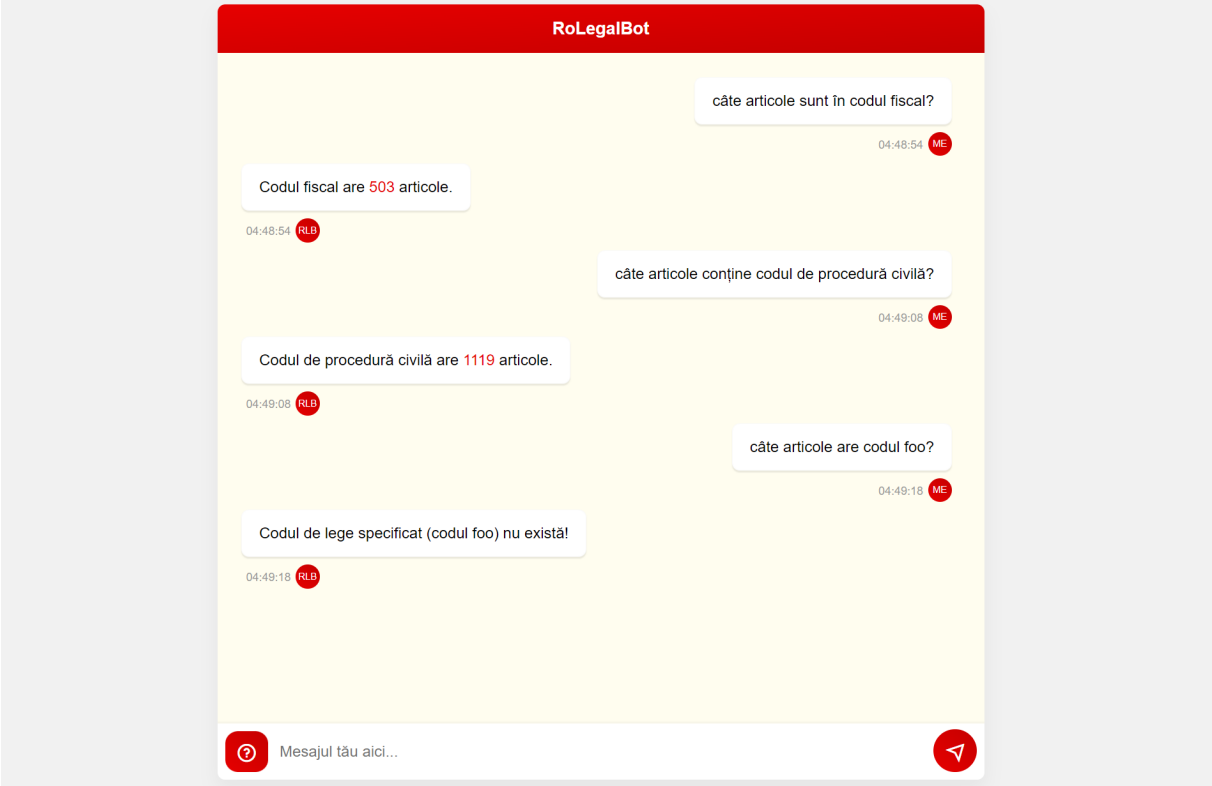

Figure 2.6: **Type 2 query answer example**

The second type of question is meant to mend the issue of inquiring about randomly selected article numbers. The response is a simple sentence mentioning the total number of articles that the specified Code of law contains (Figure 2.6). Apart from establishing limits, this information is useful in pointing out that although each Code is as important as the next, the amount of information in it can vastly differ between them. Asking about a non-existent Code of law prompts the sending of a suitable message.

Queries of type 3 are the cornerstone of RoLegalBot as these will be answered with the actual legal text corresponding to the article number requested, if valid. A fitting message is sent in the event of an article index number exceeding the Code's limit. The answer is put together with the article title at the beginning followed by its contents (Figure 2.7). Although the article text is sent in its entirety, the words within which are labeled as complex and/or rare will be highlighted and offer their first DEX (The Romanian Explanatory Dictionary[2]) definition as a tool-tip upon being hovered over.

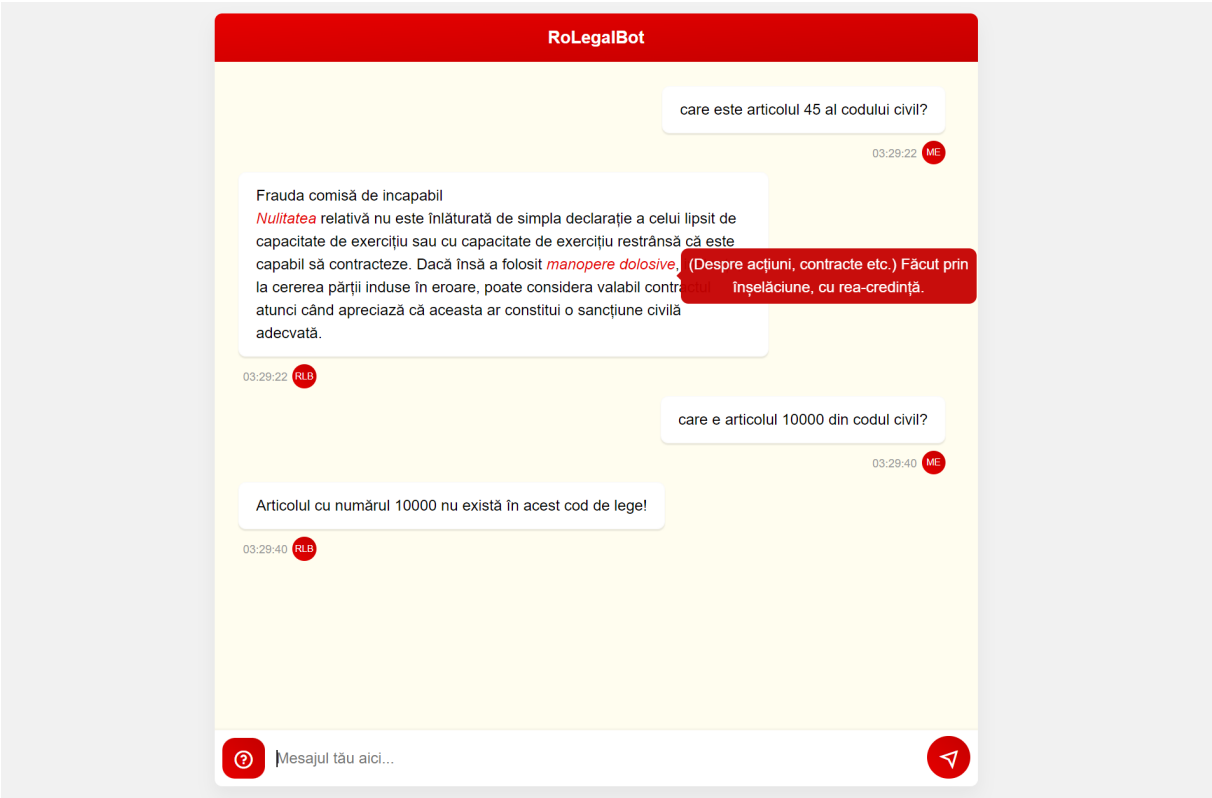

Figure 2.7: **Type 3 query answer example**

[2] https://dexonline.ro/

The last type of query is meant to serve as a fact-checker, as well as a last resort of sorts in case there are still potential words that are unfamiliar yet not highlighted. The answer to this query is crawled directly from its DEX entry in the same fashion as above (Figure 2.8). In the unlikely event that an entry is not found, an apt message is sent instead.

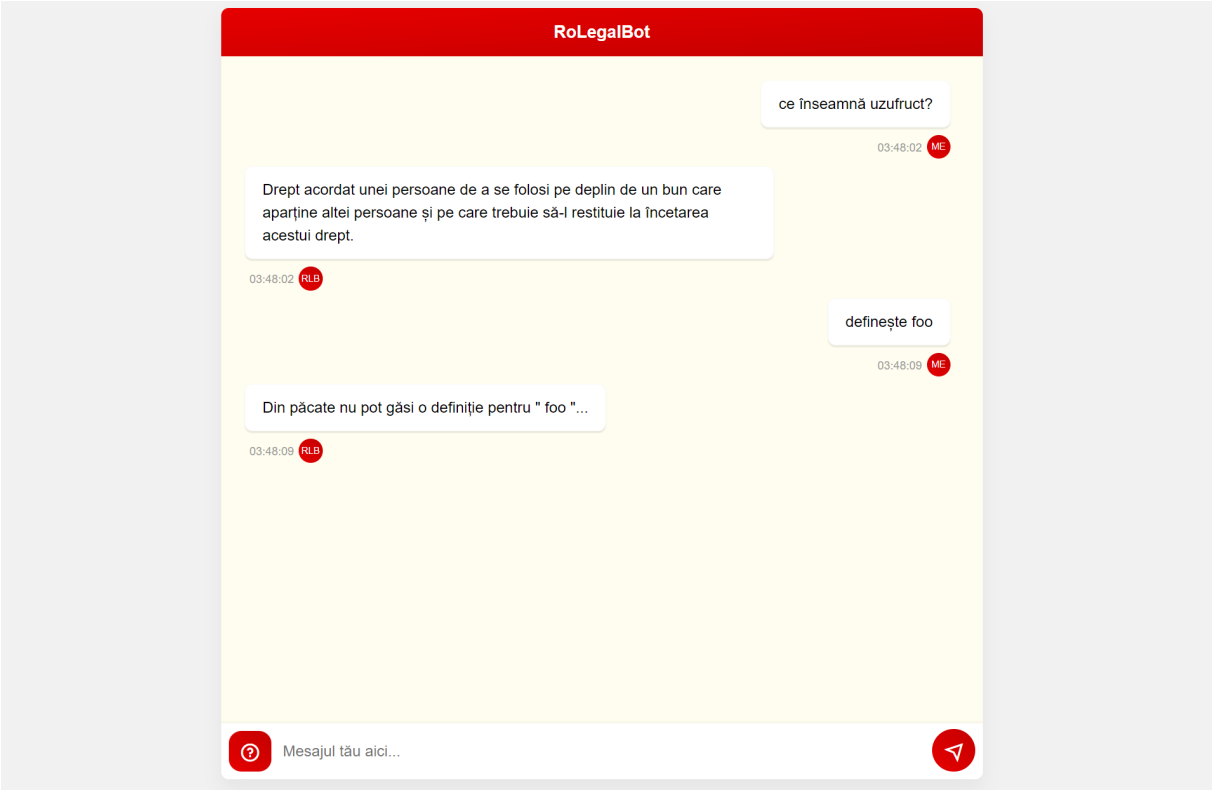

Figure 2.8: **Type 4 query answer example**

Finally, any other input or query that does not match any of the expected patterns will be met with one of a selection of random messages from the chatbot acting confused or asking for reformulation (Figure 2.9).

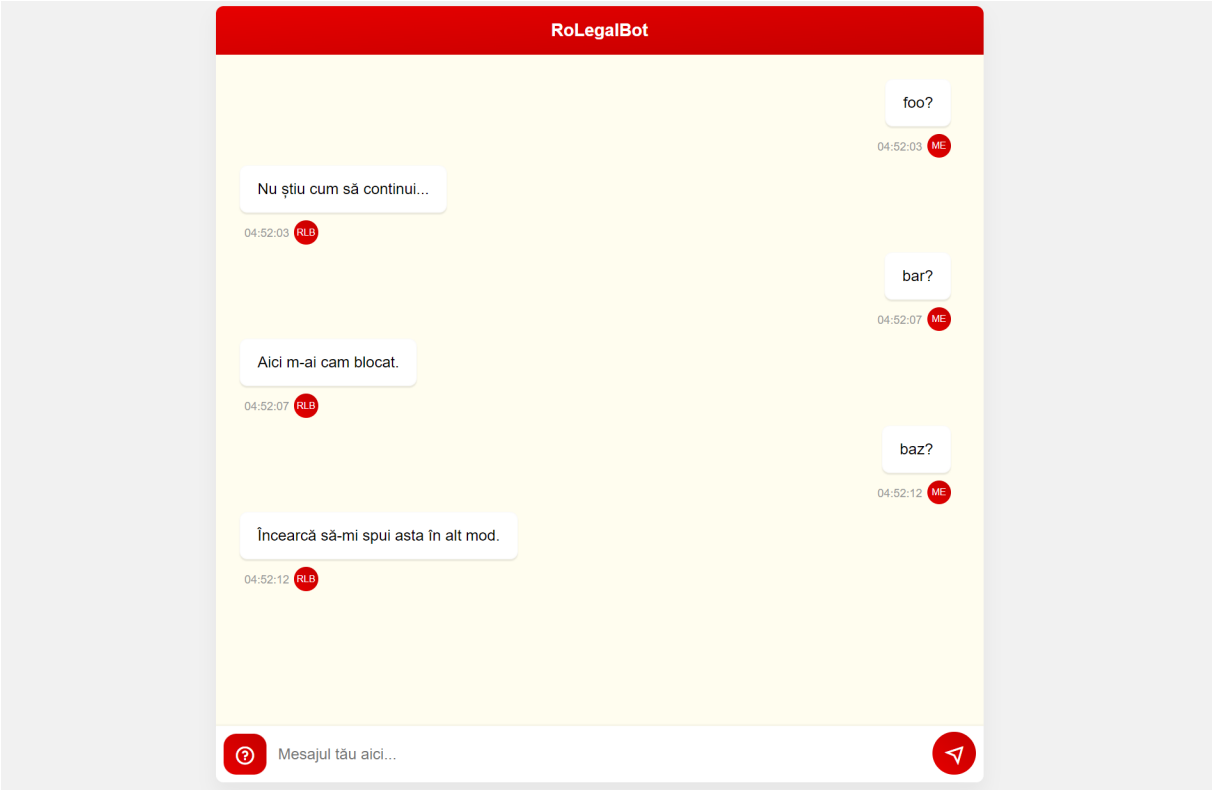

Figure 2.9: **Default messages in response to non-templated input**

2.4 Dataset collection and processing

2.4.1 Legal text

As mentioned at the beginning of this paper, the Romanian State provides and maintains a few official sources on web site platforms which keep track of modifications, decrees, ordinances and/or addendums to the legislation, in addition to its base form.

The most important of these is the Official Monitor of Romania. As of the writing of this paper, the web site offers electronic text versions of all public legal documents from January 1st, 2000, all the way to the present, and are actively working on extending the time interval to the past, down to the year 1990, in full. It is an invaluable public resource and it stands only to benefit every citizen. That being said, it is considerably difficult to navigate the results of a specific date. The result consists of a list of numbered *parts*, each listing links to external PDF files containing the relevant legal content, files whose names are number-coded. This formatting decision is not easily explained or intuitive which, in combination with the file contents themselves, creates a fairly overwhelming experience.

To counterbalance this, we opt to use the base forms of the core Codes of law provided by The Romanian Legislative Portal. In the context of this paper, the core Codes of law are: (1)

The Constitution, (2) **The Civil Code**, (3) **The Civil Procedure Code**, (4) **The Penal Code**, (5) **The Penal Procedure Code**, (6) **The Fiscal Code**, (7) **The Fiscal Procedure Code**, (8) **The Labor Code** and (9) **The Administrative Code**.

The reasoning behind this is that every modification proposed and accepted for the legislation is to be eventually consolidated into the article text of these core laws. Although the modifications are in effect immediately after being voted, they are still considered separate from the core legislation and are for now categorised just as additions. We go forward with the assumption that the base form of the core Codes is sufficient in understanding the role of the article and as such we reference the URLs of the Codes' raw text for crawling.

The general structure of the Codes is as follows:

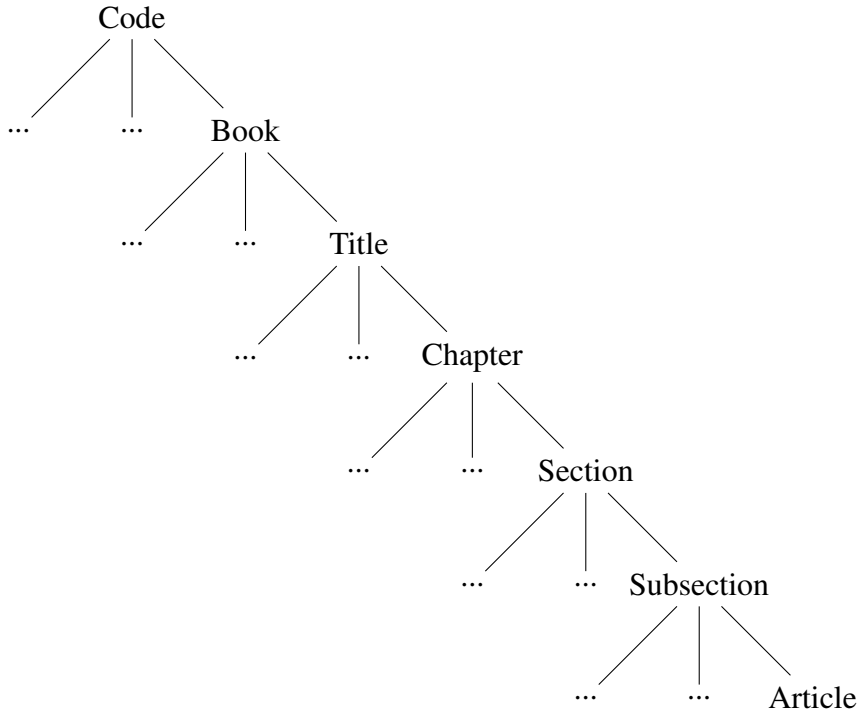

Figure 2.10: **General tree structure of a Code of law from the Legislative Portal**

It is worth mentioning that not all Codes share the exact same hierarchical structure, and instead Figure 2.10 presents what the union of each individual hierarchy looks like.

In the previous section, we briefly mentioned that despite their equal importance, the Codes differ in total number of articles. This is just as true for actual word content. Figure 2.11 plots these two statics side-by-side using bar-graphs, emphasising the scale of text data that needs to be parsed and searched through. Of note, while the left bar-graph depicts the number of article texts each Code contains, the right bar-graph charts the number of unique word occurrences excluding Romanian stop-words.

Considering the size of the data and the task at hand, the usefulness of Python dictionaries

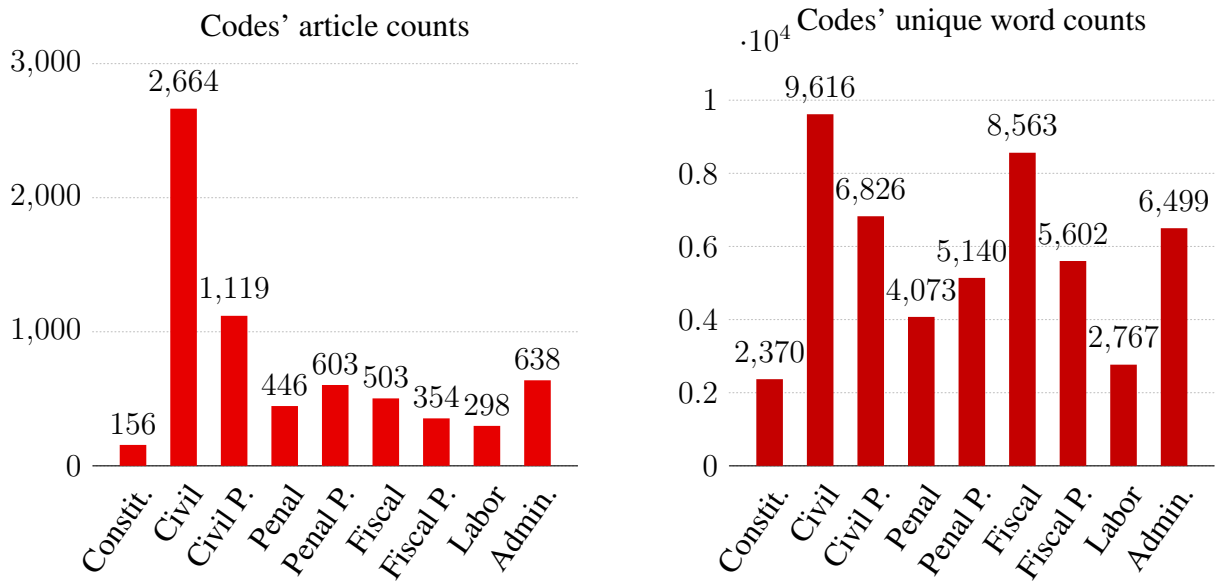

Figure 2.11: **Codes' statistics**

becomes evident due to their lookup speed efficiency. With this in mind, we create a list of dictionaries mimicking the tree structure observed above by using regular expression pattern matching on the entire legislative text collection. By following the hierarchy downwards, we start with a *Code* object that has a list property comprised of *Book* objects, which in turn have a list property comprised of *Title* objects and so on and so forth the structure deepens until we reach the lists of *Subsection* objects each with their own array of *Article* objects.

These article objects are the ones containing the actual legal text, along with the corresponding number relative to the Code they belong to, while the encapsulating objects mostly contain metadata such as component index and title name, but with one important exception: an additional two-integer array representing the article index range within that containing object (Figure 2.12). This index range is what enables the efficient article index search within this complex nested structure, on top of the native dictionary hashing.

By traversing this custom structure, the Dialogue Manager is capable of answering all of the query types described in section 2.3, exception being the type 4 external definition retrieval operation.

```
[
   {
      "name": "CODE_NAME",
      "code_number: 1,
      "range": [i_0, i_n1],
      "books": [
         {
            "name": "BOOK_NAME",
            "number": 1,
            "range": [j_0, j_n2],
            "titles": [
               {
                  "name": "TITLE_NAME",
                  "number": 1,
                  "range": [k_0, k_n3],
                  "chapters": [
                     {
                        "name": "CHAPTER_NAME",
                        "number": 1,
                        "range": [l_0, l_n4],
                        "sections": [
                           {
                              "name": "SECTION_NAME",
                              "number": 1,
                              "range": [o_0, o_n5],
                              "subsections": [
                                 {
                                    "name": "SUBSECTION_NAME",
                                    "number": 1,
                                    "range": [p_0, p_n6],
                                    "articles": [
                                       {
                                          "name": "ARTICLE_NAME",
                                          "number": 1,
                                          "text": "ARTICLE1_TEXT"
                                       },
                                       # ...
                                    ]
                                 },
                                 # ...
                              ]
                           },
                           # ...
                        ]
                     },
                     # ...
                  ]
               },
               # ...
            ],
         },
         # ...
      ],
   },
   # ...
]
```

Figure 2.12: **Processed legal text**

2.4.2 Common text

In this subsection, we'll go into detail about the Explanation module and how the complexity of a word is measured.

We approach determining a word's complexity in a similar fashion to [7], by establishing a rarity threshold based on the word's frequency within Romanian Wikipedia. As of writing this paper, Romanian Wikipedia brandishes approximately 440.000 articles[3] and it should more than suffice for obtaining common and/or frequently occurring words as the language used in writing the articles is meant to accessible to as large a demographic as possible. Despite that, a significant amount of those articles lack consistency, being very short and succinct.

An initial attempt at creating a list of Romanian Wikipedia articles made use of Wikipedia's `https://ro.wikipedia.org/wiki/Special:Random` page link which randomly redirects the user to any of the existent articles on the domain. Although it seemed appropriate for generating a non-biased list of articles, after crawling the first couple thousands pages it quickly became apparent that they very much lacked vocabulary and size, as the resulting frequency map was comparably short considering the number of articles parsed by that point.

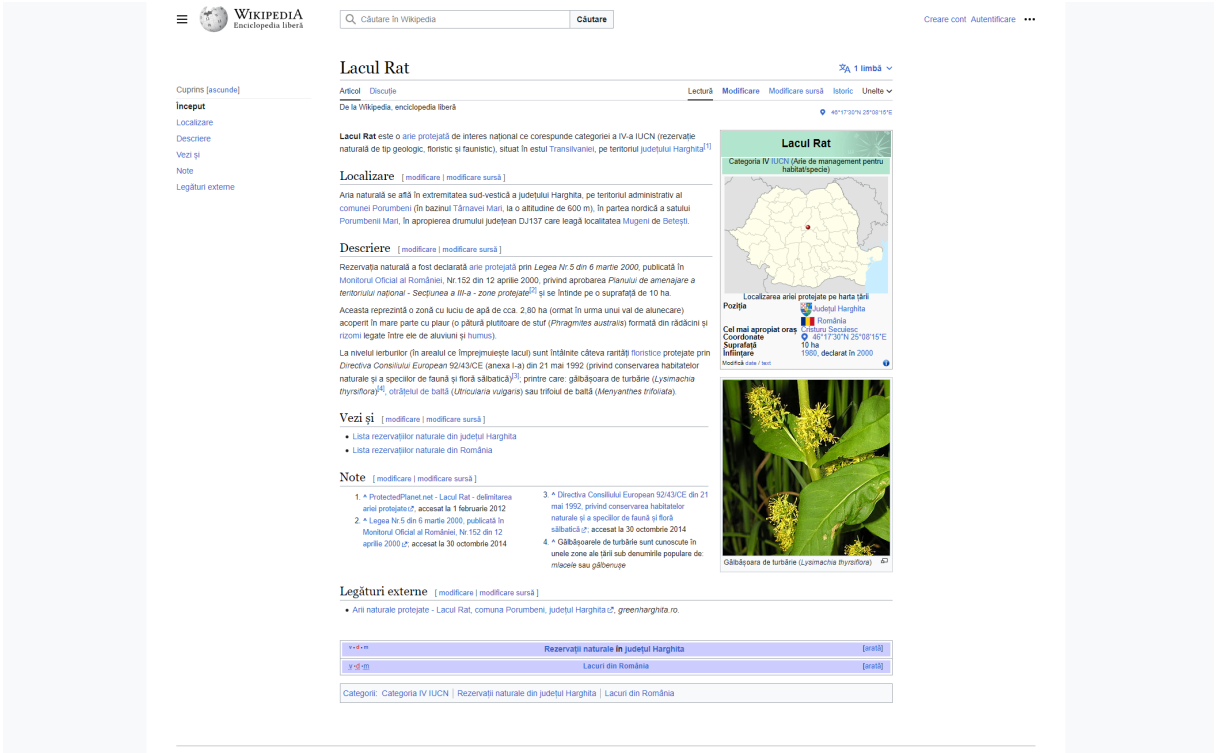

Figure 2.13: **Randomly generated Wikipedia page example 1**

Figure 2.13 presents a more fortunate example where we have 4 paragraphs of text that can be processed, while Figure 2.14 presents what is sadly not a rare occurrence, where there is

[3] `https://ro.wikipedia.org/wiki/Wikipedia`

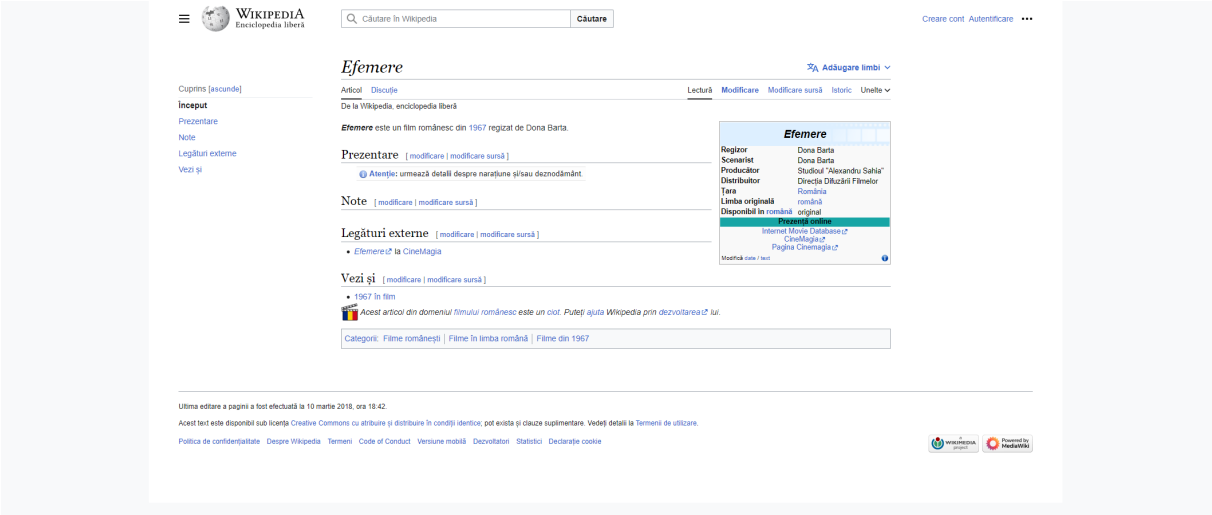

Figure 2.14: **Randomly generated Wikipedia page example 2**

only a single paragraph with less than a few sentences.

The solution, however, still lied within Wikipedia, as they themselves affirm that among all existing Romanian articles, only a total of **693** are considered valuable articles[4]. More specifically, they are composed of **200** featured articles (”articole de calitate”) and **493** good articles (”articole bune”), these being distinctions awarded after peer review and fulfilling an assortment of criteria. This rounds up to around **0.17%** of articles, which seems like a very small percentage, but the contents of these few hundred of articles more than make up for it.

The first step in creating the word-frequency map is crawling each of the selected articles' URL and getting the text paragraphs. Next we clean the text by eliminating special characters and stop-words. At this point we initialize an instance of the `spacy` module's `ro_core_news_lg` pre-trained pipeline to filter out proper nouns out of the paragraphs and also convert the individual words to their lemmatized form. And by now we have the flow running for creating an intermediary word-frequency map with a lemmatized word's raw count within each of the articles, or the document base.

At this moment, we have all the information we need to compute the *Term Frequency - Inverse Document Frequency* (tf-idf) value of each word relative to each document. Tf-idf is a standardized measure for establishing the relative importance of a word inside a specific document; the higher the value, the more relevant the word in relation to that document. For our purposes, we will compute all of the tf-idf values of a word relative to the entire document base. These values will then be averaged and this value will become our final word frequency.

The formula used for determining the tf-idf value is shown in Figure 2.15 as the product

[4] `https://ro.wikipedia.org/wiki/Wikipedia:Articole_bune`

$$tfidf(t, d, D) = tf(t, d) * idf(t, D)$$

$$tf(t, d) = log(1 + f_{t,d})$$

$$idf(t, D) = log\left(\frac{N}{1 + |\{d \in D : t \in d\}|}\right)$$

Figure 2.15: **Tf-idf formula**

between the term frequency, here calculated logarithmically scaled with the raw count of the word t within document d as $f_{t,d}$, and the inverse document frequency which is a logarithmically scaled inverse fraction of the documents that contain the word t, obtained by dividing the total number of documents D by the number of documents in which the word appears plus one to avoid division-by-zero.

The entire mapping process is quite lengthy, but in the end we are left with a collection of around 64K unique lemmatized words and their frequency (Figure 2.16) in relation to the approximately 700 bulky articles from Romanian Wikipedia that we have selected.


```
719     "ianuarie": 0.4719435755579596,
720     "an": 0.19093732856148726,
721     "descris": 0.36611653200136446,
722     "melodie": 0.5318007860920462,
723     "foarte": 0.32263504193854453,
724     "dramatic": 0.2539220074424882,
725     "scris": 0.37376790217416045,
726     "inimă": 0.28336069178900025,
727     "debuta": 0.5206403631676597,
728     "loc": 0.18196073875868904,
729     "hot": 0.5136561586759761,
730     "obține": 0.3534663515514619,
731     "deveni": 0.2127478459998019,
732     "clasa": 0.40448114994869977,
733     "clasament": 0.6096902280141704,
734     "concret": 0.15566558023734045,
735     "hip": 0.1605874653927245,
736     "hop": 0.2691144937441713,
737     "recurrent": 0.01407143357622207,
738     "intenționa": 0.2032702553574383,
```


Figure 2.16: **Word-frequency map snippet**

With all that established, we can now easily define the Explanation module's utility: before the article text requested by the user gets sent to the UI, each word is lemmatized and searched for in the word-frequency map. If the word is not present in the map or if its frequency

is lower than a set threshold, the word's definition is pulled from the DEX and added to a special `span` HTML tag wrapped around the original word and it will only show up once the word is hovered over in the chatbot's speech bubble, as shown in the previous section in Figure 2.7.

Chapter 3

Evaluation and Results

3.1 Evaluation

A system such as the one proposed is not one that can be easily evaluated with objective means, especially considering its purpose is to offer information in a simple and friendly manner straight from a demonstrably trusty source. Manual testing has been a key factor in determining the word-frequency threshold value. Many attempts have lead to having it set to **0.05**, and it serves as a solid barrier of separation between common and uncommon words in a person's day-to-day vocabulary. As hoped, a good portion of the terms specific to the legal domain have been properly flagged as complex and/or uncommon, with few exceptions owing to the relatively small selection of domains covered by the 693 Wikipedia articles.

Figure 3.1: **Evaluation showcase 1 - Common word still marked as complex/rare**

As observed in Figure 3.1, ”calculării” (in the context, ”to calculate”) is a rather common word but due to it not being present enough in the article collection it is being highlighted and defined as any other complex word, which is unnecessary.

Of course, this same example also brings up the issue of some article texts being rather long and complex in sentence structure. Summarization of the text would risk losing important bits of meaning and that is a dangerous assumption to make about rules which should not be broken under any circumstance, while simplification of it can not be guaranteed as speciality terms usually do not have more than one meaning or potential replacements. Experiments have been run using RoWordNet [13] and its Python API [6] in order to find suitable alternatives to words marked as complex/rare, however, as mentioned, most of the specialty words being investigated had very limited graph connections, while, at the same time, words that slipped past the rarity check had far too many. All of this slowed down the answer preparation phase dramatically and so we opted for the option of crawling the definition out of the DEX.

It is also worth noting that the process of creating the word-frequency map was first attempted using RELATE's[1] TEPROLIN API[2] as this service handles a wide variety of NLP processing tasks specifically for Romanian, among which tokenization, lemmatization, pos-tagging and many others, all with great accuracy but at the cost of being very time-consuming and not open to parallelization. As such, even though we end up using `spacy`'s pipeline due to time concerns (the processing time of 1 bulky article dropped from 15 minutes to less than a minute) the quality of some lemmas is not ideal by comparison.

Better candidates for evaluation would be the 2 data structures created to support this system.

The dictionary mirroring the Codes of law selected, although conventionally simple, provides a machine-readable format of the base Romanian legislation that is easy and efficient to parse. The size of it is manageable enough that it does not need to occupy dedicated data base storage space and instead is very portable to JSON format. Evidently, large-scale corpora such as the ones presented in [15] and [3] easily outshine this file in both scale and scope, but that does not necessarily mean either of them would fair well as substitutes. The main difference is that our custom legislation tree is purpose-built for being parsed and consumed by RoLegalBot's systems. The MARCELL corpus, though, is meant to be as large as possible, as rich in contextual information as possible, such as sentence-splits, tokenizations, lemmatizations and morphological and syntactical annotations, and as internationally available as possible. Similarly,

[1] `https://relate.racai.ro/index.php`

[2] `http://relate.racai.ro/index.php?path=teprolin/doc_dev`

the EUR-Lex-Sum corpus is geared towards multilingual long-form legislative text formatted for summarization tasks.

On the other hand, the word-frequency map representing the commonness/rarity of words in the Romanian language is an area that has not been explored to our understanding, even though it is sure to benefit all language processing areas outside of the legal domain.

3.2 Results

Most the systems results and use cases have been showcased in previous sections, but one fascinating comparison that can be made is directly with ChatGPT itself. As hinted before, ChatGPT too is an example of system that boasts generally outstanding results even within the limitations of the Romanian language thanks to its amazing machine translation capabilities. However, when focusing its scope as precisely as RoLegalBot's domain of activity, some clear distinctions can be made, despite the retrieval-based nature of our system.

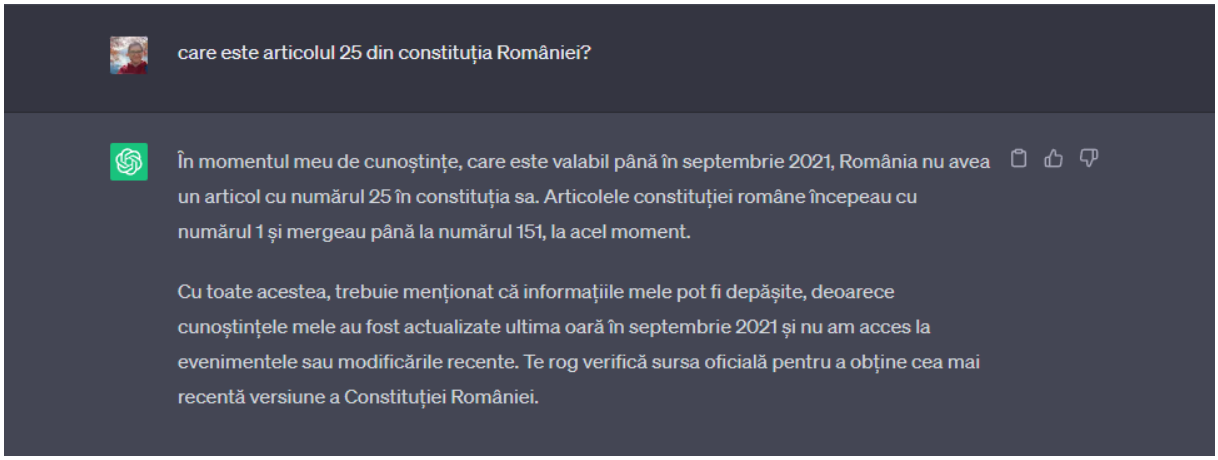

Figure 3.2: **ChatGPT Comparison - Factual article information retrieval 1**

In both scenarios, the chatbot is asked to retrieve factual information, in this case to get the text contents of the 25th article of the Constitution, in Romanian. On first occasion, ChatGPT claims it has no knowledge of this specifically-numbered article being in Romania's Constitution at least before September 2021, while on second occasion it does share the apparent contents of the article but they are demonstrably inaccurate. Attempting to ask the same question again but in English provides no better result.

Similar results occur when attempting to ask for the number of articles in a Code of law or for the laws relevant to a certain topic. Of course, there is no assumption that ChatGPT should always be correct, and to its credit it ends most of the messages with a kind reminder to check

Figure 3.3: **ChatGPT Comparison - Factual article information retrieval 2**

official sources, but it is indeed fascinating to see its limitations being met in the context of apparently simple information retrieval from what should be solid sources.

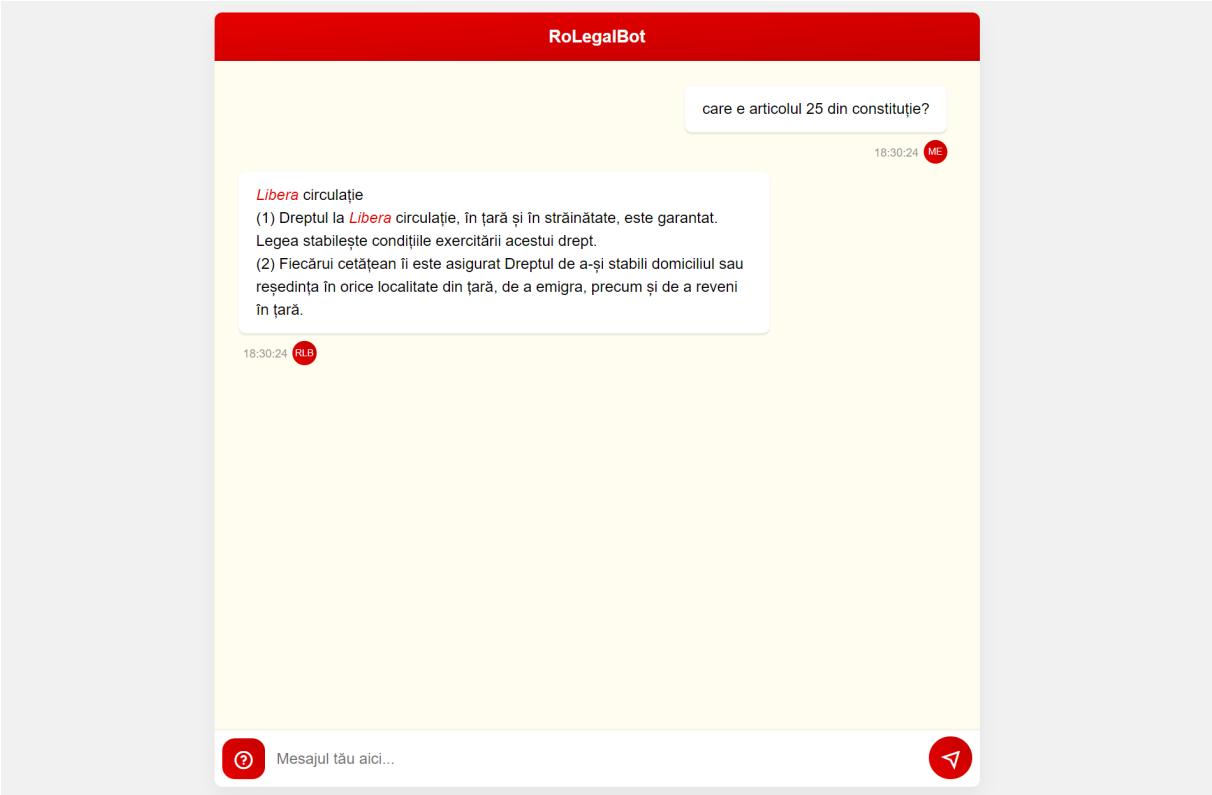

Figure 3.4: **RoLegalBot's reponse - Factual article information**

Conclusions

Chatbot systems are a testament to people's ambition to recreate what can essentially be described as potentially the most human aspect of ourselves. Having been researched and worked on for decades now, it is surprising that there are still domains of interest and tasks that could benefit even from the most basic of conversationalist partners.

RoLegalBot is proposed as a building block or a gateway towards more complex interactions and sub-tasks inside the legal domain. This is doubly true for propping up dialogue systems for the Romanian language in general.

The system presented is fairly simple in scope and as such one of the main things that would greatly improve its ability is an expansion of its data sources. The Codes of law considered are still the core of what legislation means in Romania, but it is a fact that the additions to this core are more numerous and convoluted than the existing texts. Similarly for the Wikipedia articles selected; although the domain provides a great format and quality data for information retrieval, its quantity leaves to be desired, but that is not something that could not be aided in any way, especially considering we employed a pipeline fine-tuned over news data. The other significant component holding this tool's potential back is the rigid AIML pattern matching behaviour. Although accessible to implement and understand, porting the query handling to a more mainstream and self-learned approach is sure to outgrow the current limits imposed upon itself by its rule-based nature.

Even so, its utility is difficult to question considering the layman's proficiency in navigating the official national legislation portals, besides understanding of the legal text itself. Apart from that, it has potential to become a great study tool and easy-access fact-checker.

Both of the data structures created as a result of this system bring value to the domain of natural language processing, but it is certain that the word-frequency map of common Romanian words is among the first of its kind. It is freely available on the GitHub repository provided in the beginning sections but it is a worthwhile endeavor to port it to other platforms as well and, better yet, expand upon it.

Bibliography

[1] E. Adamopoulou and L. Moussiades. An overview of chatbot technology. In *Artificial Intelligence Applications and Innovations: 16th IFIP WG 12.5 International Conference, AIAI 2020, Neos Marmaras, Greece, June 5–7, 2020, Proceedings, Part II 16*, pages 373–383. Springer, 2020.

[2] M.-A. Alt, I. Vizeli, and Z. Săplăcan. Banking with a chatbot–a study on technology acceptance. *Studia Universitatis Babes-Bolyai Oeconomica*, 66(1):13–35, 2021.

[3] D. Aumiller, A. Chouhan, and M. Gertz. Eur-lex-sum: A multi-and cross-lingual dataset for long-form summarization in the legal domain. *arXiv preprint arXiv:2210.13448*, 2022.

[4] M. Cemri, T. Çukur, and A. Koç. Unsupervised simplification of legal texts, 2022.

[5] M. Collantes, M. Hipe, J. L. Sorilla, L. Tolentino, and B. Samson. Simpatico: A text simplification system for senate and house bills. In *Proceedings of the 11th National Natural Language Processing Research Symposium*, pages 26–32, 2015.

[6] S. D. Dumitrescu, A. M. Avram, L. Morogan, and S.-A. Toma. Rowordnet–a python api for the romanian wordnet. In *2018 10th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pages 1–6. IEEE, 2018.

[7] A. Garimella, A. Sancheti, V. Aggarwal, A. Ganesh, N. Chhaya, and N. Kambhatla. Text simplification for legal domain: Insights and challenges. In *Proceedings of the Natural Legal Language Processing Workshop 2022*, pages 296–304, Abu Dhabi, United Arab Emirates (Hybrid), Dec. 2022. Association for Computational Linguistics.

[8] T.-D. Huțul and A. Karner-Huțuleac. Transhumanism in psychology: The attitude towards the use of chatbots in psychotherapy and ethical implications. *Journal of Intercultural Management and Ethics*, 5(4):41–49, 2022.

[9] A. Kornilova and V. Eidelman. Billsum: A corpus for automatic summarization of US legislation. *CoRR*, abs/1910.00523, 2019.

[10] B.-C. Mocanu, I.-D. Filip, R.-D. Ungureanu, C. Negru, M. Dascalu, S.-A. Toma, T.-C. Balan, I. Bica, and F. Pop. Odin ivr-interactive solution for emergency calls handling. *Applied Sciences*, 12(21):10844, 2022.

[11] E. Nichifor, A. Trifan, and E. M. Nechifor. Artificial intelligence in electronic commerce: Basic chatbots and the consumer journey. *Amfiteatru Economic*, 23(56):87–101, 2021.

[12] E. Sanda. Ai, intelligent robots and romanian drivers' attitudes towards autonomous cars. In *Proceedings of the International Conference on Business Excellence*, volume 16, pages 1483–1490, 2022.

[13] D. Tufiș and V. Barbu Mititelu. *The Lexical Ontology for Romanian*, volume 48 of *Text, Speech and Language Technology*, pages 491–504. Springer, 2014.

[14] D. Tuggener, P. von Däniken, T. Peetz, and M. Cieliebak. LEDGAR: A large-scale multi-label corpus for text classification of legal provisions in contracts. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1235–1241, Marseille, France, May 2020. European Language Resources Association.

[15] T. Váradi, S. Koeva, M. Yamalov, M. Tadić, B. Sass, B. Nitoń, M. Ogrodniczuk, P. Pezik, V. Barbu Mititelu, R. Ion, E. Irimia, M. Mitrofan, V. Păiș, D. Tufiș, R. Garabík, S. Krek, A. Repar, M. Rihtar, and J. Brank. The MARCELL legislative corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 3761–3768, Marseille, France, May 2020. European Language Resources Association.

[16] R. Wallace. The elements of aiml style. *Alice AI Foundation*, 139, 2003.

[17] V. Zmiycharov, M. Chechev, G. Lazarova, T. Tsonkov, and I. Koychev. A comparative study on abstractive and extractive approaches in summarization of european legislation documents. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1645–1651, 2021.

[18] V. Zmiycharov, G. Lazarova, T. Tsonkov, and I. Koychev. Structextsum–bulgarian legislation text extractive summarization by structure understanding. 2022.