And Who Is To Judge: On Automatic Judgment Analysis

Abstract. The present paper addresses the task of Russian legal judgments analysis. We implement a pipeline including text preprocessing, metadata extraction, sentence segmentation, parts detection, and extracting juridical cliches (juridisms).

1. Introduction

Recently, journalists have discovered plagiarism in some court decisions texts [11]. The extent of the problem is unknown because it is impossible to process such a great number of texts manually. In order to detect plagiarism automatically, texts need to be processed in a certain way. In this work, we explore the task of automatic court decisions texts processing.

Despite the rapid development of the legal tech, automatic processing of courts decisions is still an underdeveloped area, especially for the Russian language. At the same time, the application of a tool to deal with texts of judicial sentences is extensive: from text summarization and annotation to finding similar criminal cases and plagiarism detection.

In addition to the systematic problems appearing during text processing we have to pay attention to the specifics of the task. Firstly, texts of court sentences contain a lot of data that create additional noise and obstruct further work with the text. Such data include names, dates, criminal codes. Thereby, one of the challenges is to learn how to extract this type of data. Secondly, some parts like citations from regulatory acts may be repeated from one document to another. On the other hand, another parts like judge's opinion, testimony and proofs can not be the same in different criminal cases. Thus, only certain parts need to be compared.

One reason for the difficulty of the work in legal field is the complexity of the domain. Most of the state-of-the-art approaches in NLP are focused on processing texts in general and not paying attention to its structure and features. This leads to the difficulty of using it on the texts of specific domain. What is more, the lack of annotated data does not allow us to use machine learning approaches. A variety of difficulties are involved in getting annotated data: first of all, because it needs legal experts opinion. Therefore, it seems to us optimal to use classical rule-based methods in this task.

Below we present an approach to deal with Russian court decision texts. Suggested pipeline consists of 5 major parts: text preprocessing, metadata extraction, sentence segmentation, parts detection, and searching for specific juridical cliches (which further will be called juridisms).

1.1. Related work

There are many studies on automatic processing of legal documents [9]. At the same time, approaches to solve emerging problems vary greatly depending on the specific type of documents, language, and judicial system of a country. Since our area of interest lies in the field of the court decisions texts, we give a brief overview of articles focused on the

processing of such documents. In this area researchers carry out a variety of tasks: from the information retrieval [4] to text summarization [5] and similarity detection [8].

Strong performance in the field of information retrieval from legal texts is shown by the ontology-based SAURON system for Portuguese language [4]. However, the similar system development for Russian language from scratch demands considerable time and the help of law experts, so this would be excessive for regardingly small information retrieval tasks.

In [1] the task of predicting US Supreme Court decisions using Circuit Court opinions is considered. Authors take into account the observation that US lawyers and judges often cite past cases that could influence the outcome of the current one. In this way, the purpose of their study is opposite to ours, as for the Russian court system repeating the outcomes of previous cases is somewhat inadmissible. The model predicts the case outcome using classifier which takes advantage of case title, citations, case type etc. Also, documents are divided into parts (Abstract, Opinion, Conclusion) as we do in our study, but the difference is that [1] use database of "high-quality expertly-coded" data, and we do not have much annotated data at hand.

The goal of our research is to process and structure the Russian legal data in such a way that it can be used for textual comparison in the future.

2. Main part

2.1. Data

We have about 1 million criminal cases from 2355 Russian district courts from 2002 to 2013. 650 cases of them were annotated, although the annotation was not quite consistent. Each juridical decision is a file in XML format. We consider only criminal cases, but initial corpus included other types of cases as well. So, the first task was to filter out these cases. With that done, our final corpus consists of 1 million forensic decisions on criminal cases.

Each document starts with structured information about the current case, e.g. court name, judge's name, case category etc. The main body can be divided into 3 parts. The first one (before words *суд установил/court stated*) describes case participants (e.g. judge, advocate, prisoner at the bar etc.). The last one (after words *суд приговорил/court sentenced*) describes case outcome, and the second part comprises the essential information, i.e. case description, testimony, proofs etc.

Our work mostly concerns the second part of a document, but we also use the first one in order to extract participants' names.

2.2. Methodology

As was mentioned before, legal text processing pipeline includes five main steps: text preprocessing, metadata extraction, sentence segmentation, parts detection, juridisms extraction. All these parts are described below.

2.2.1. Text preprocessing

To handle XML format we used Beautiful Soup package. Each text was divided into 3 parts:

- case participants
- case description
- case outcome

Our next step was to remove all the digits and dates from the main (second) part of a document which was gained by regular expressions. Another task was to get rid of rather frequent patterns which indicated references to legal documents, e.g. *п. «в» ч. 2 ст. 158 УК РФ* (*paragraph «в» of part 2 of article 158 of Criminal Code of Russian Federation*). We wanted to remove them because while searching for plagiarism, such patterns would create noise and thus complicate the task. We replaced these patterns by special tags using regular expressions.

2.2.2. Metadata extraction

Another step of judgment preprocessing was the metadata extraction of metadata. We extracted the following data about the case:

- the court name
- case result
- related criminal code articles
- participants' names (judge, prosecutor, advocate, secretary, accused).

Several of given categories were wrapped in tags in the beginning of the document, so they can be easily extracted with XML parsing. For the rest of the categories (the names of prosecutor, advocate, secretary and accused) we developed a rule-based system. It combines named entity recognition library natasha and several simple rules based on regular expressions. We also had some experiments with DeepPavlov NER. Collected metadata is stored as a pandas Data Frame. Metadata collection output let us remove the excessive information from the case: replace the names of the case participants in the main parts of the case with corresponding metadata tags.

2.2.3. Sentence segmentation

In order to divide the text into semantic parts, it should be divided into sentences beforehand. If there are punctuation marks in the text, a letter case is observed and few errors, finding the boundaries of sentences for most cases will be quite simple. In ordinary texts, most sentences end with a period, question or exclamation marks followed by some indentation and another capital letter. However, given the specificity of data, we could not resort to the rule-based solutions. Texts of court decisions are distinguished by a large number of abbreviations. For example, such texts often contain quotations from regulatory acts, references to the criminal

code (*ч.1 ст. 228 УК РФ /part 1 of article 228 of the Criminal Code of the Russian Federation*), abbreviations of various words that are inconsistent and differ from document to document (*18 час. 15 мин. до 19 час. 37 м./18 o'clock 15 minutes to 19 hours 37 minutes, 20 мл, медицинский шприц 10 мл./20 ml, medical syringe 10 ml.*), and a great number of name abbreviations. All this brings additional difficulties in finding sentence boundaries. Thus, it was impossible to prescribe the rules for dividing a text into sentences, since it would be impossible to take into account all the options for possible reductions. Therefore, to solve this problem, we used the unsupervised multilingual sentence boundary detection system [7]. It uses collocational information for the detection of abbreviations, initials, and ordinal numbers. We trained this statistical model on our data. The training set containing 9280 texts of sentences from different courts was selected randomly. Then, some abbreviations of names and patronymic names that the model did not take into account were added to the list of exceptions.

2.2.4. Parts detection

A typical legal judgment is a complex text document with its own structure. The components are as follows:

- factual circumstances of the case (which further will be called *Fabula*)
- testimony (which further will be called *Witness*)
- physical evidence description and expert evidence (which further will be called *Proof*)
- judge's reasoning (which further will be called *Meditation*)

Parts detection was solved as a sentence multiclass classification task: we split each document into sentences as described above, then fit a classifier to predict a part label (*Fabula*, *Witness*, *Proof* or *Meditation*) for each sentence. There were two ways of sentence vectorization for the classification:

1. calculating tf-idf regarding each sentence as a document;
2. calculating sentence embedding as a mean of its word embeddings (word representations were obtained using FastText [2] trained on our data).

As a classifier, logistic regression was used. After labeling, sentences with similar labels were concatenated into one part using some simple heuristics.

2.2.5. Juridisms extraction

While studying legal documents, we noted that there is a large number of very frequent language cliches — juridisms — which cannot be viewed as plagiarism, as all the judges use them, and for this very reason these expressions complicate searching for the real plagiarism (e.g. *суд учитывает характер и степень общественной опасности / court takes into consideration nature and extent of social danger*). So, one of the research goals was to

compose as full list of juridisms as possible, and then replace each occurrence of a juridism in a document by special tag.

Composing the list of juridisms. The simplest thing to achieve this was to extract a number of most frequent n-grams from our documents. However, we can not just get the list of n-grams of a random court, as in this case we maybe would extract some plagiarism. Besides, it is often the case that some local courts which do not have many cases to consider, work mostly with particular sort of cases, for instance, with thefts or with murders, so the vocabulary there is quite limited. And we should make sure that our list of juridisms cover as many case types as possible, so the algorithm is as follows. Firstly, we take the court with maximum number of documents (more than 6000 texts), lemmatize each text using morphological analyser pymorphy2, and extract 5000 most frequent n-grams from 4 to 20 words using NLTK. At the next step, we take 20 random courts, and repeat these steps for each of them. Thus, we have 21 lists of most frequent n-grams for corresponding courts. Then, we intersect each of the 20 lists for 'small' courts with the list of ngrams of the 'big' court, and put together 20 resulting lists. The algorithm was applied to top-5 biggest courts in our corpus, and at each iteration there was a new set of 20 'small' courts. All in all, to get the list of juridisms we handled documents of 105 courts. The resulting list of juridisms was filtered by expert, as sometimes there were phrases that do not relate to legal field as such, and got there by chance. The final list contains about 500 phrases.

Searching for juridisms. After splitting the text into parts we lemmatized each part and searched for juridisms using Boyer - Moore - Horspool algorithm [6]. Each juridism was replaced by tag JUR.

2.3. Experiments

In this section we give some experimental results and its analysis. Note that there is no evaluation of text processing as it is essentially impossible to evaluate. Besides, we did not evaluate juridisms extraction per se.

2.3.1. Metadata extraction evaluation

To evaluate the quality of metadata extraction, we manually marked up 100 random documents and then compared this marking with extracted information. We used the following information retrieval evaluation metrics from [3]:

$$\textit{Soft Precision} = (\textit{Correct} + 0.5 * \textit{Partial}) / (\textit{Correct} + \textit{Spurious} + \textit{Partial} + \textit{Incorrect})$$
$$\textit{Soft Recall} = (\textit{Correct} + 0.5 * \textit{Partial}) / (\textit{Correct} + \textit{Missing} + \textit{Partial} + \textit{Incorrect})$$
$$\textit{Strict Precision} = (\textit{Correct}) / (\textit{Correct} + \textit{Spurious} + \textit{Partial} + \textit{Incorrect})$$
$$\textit{Strict Recall} = (\textit{Correct}) / (\textit{Correct} + \textit{Missing} + \textit{Partial} + \textit{Incorrect})$$
$$\textit{Soft F1} = 2 * ((\textit{Soft Precision} * \textit{Soft Recall}) / (\textit{Soft Precision} + \textit{Soft Recall}))$$
$$\textit{Strict F1} = 2 * ((\textit{Strict Precision} * \textit{Strict Recall}) / (\textit{Strict Precision} + \textit{Strict Recall}))$$

where *Correct* is the amount of the correct answers, *Partial* is the amount of the partially correct answers (wrong grammatical form/one name is missing/only one name from two is correct), *Incorrect* is the amount of wrong answers, *Spurious* is the amount of situations where the name is not present in gold standard (not found) but is present in retrieved names, *Missing* is the amount of situations where the name is present in gold standard but is not present in retrieved names (not found).

We got the following results:

Rules and regular expressions		
entity	Strict F1	Soft F1
advocate	0.92	0.93
prosecutor	0.91	0.92
secretary	0.98	0.98

Table 1. Advocate, prosecutor and secretary entities extraction evaluation

Accused entity extraction		
method	Strict F1	Soft F1
rules + regular expressions	0.76	0.80
rules + regular expressions + natasha	**0.81**	**0.85**
rules + regular expressions + ner_rus (DeepPavlov)	0.57	0.71

Table 2. Accused entity extraction evaluation

2.3.2. Parts detection evaluation

To evaluate the quality of parts detection extraction, we used 650 annotated documents from the corpus.

The following metrics were used for evaluation:

$$Precision = (TP) / (TP + FP)$$
$$Recall = (TP) / (TP + FN)$$
$$F1 = 2 * ((Precision * Soft\ Recall) / (Precision + Recall))$$

where *TP* is the intersection of unique sentences in gold standard and predicted corpora, *FP* is the difference in unique sentences number of gold standard and predicted corpora, *FN* is the difference in unique sentences number of predicted and gold standard corpora.

We got the following results:

F1 for sentence classification		
part	tf-idf	mean of FastText word embeddings
Fabula	**0.56**	0.22
Witness	**0.75**	0.65
Proof	**0.46**	0.22
Meditation	**0.76**	0.51
overall	**0.68**	0.52

Table 3. Sentence classification for part detection evaluation

2.4. Difficulties analysis

We faced with problems concerning both data peculiarities and language factors. Firstly, the dataset is large (about 1 million documents) and it does not have proper marking. This fact makes it difficult to get the whole picture of the data.

Even though legal documents are quite well-structured, texts we analyze do not always share the identical pattern, e.g:

- separators in the text are not always the same
- there are several types of "hiding" the names of the case participants (for example, in different cases *Петров Дмитрий Иванович* can be represented as *ИДИ, Петров ФИО 18* or *ФИО 18*), and sometimes different forms can indicate the same person (*ФИ04* and *ФИ05* for one individual within the document)
- the structure of some judgment parts can be different for different courts

However, the amount of this stylistic varieties is limited, and it should be possible to improve our system so it can handle almost all of them.

Some of the mistakes within the task of name extraction are related to the specific features of the Russian language morphology: for example, genitive masculine and nominative feminine forms sometimes cannot be differentiated without context (e.g. *Иванова А.П.*), which complicates data preprocessing. This problem can possibly be solved with the use of morphological parsers, but this step significantly slows down the preprocessing of the above-mentioned large corpus.

3. Conclusions and future work

We developed the legal texts processing system which should improve the following plagiarism detection. Given that our system removes excessive information about the case (case participant and document names, digits) and juridisms, the plagiarism detection system can focus on the most informative part of the case. Moreover, parts detection accelerates the process of similarities search. In addition to the system we also created the frequency list of the most popular juridisms.

In the future we plan to improve our system. Firstly, we are going to better current performance for metadata extraction and parts detection. Secondly, we want to enrich our juridism collection with the new type of legal cliches: quotations from the most popular regulatory acts. Besides, web application with RESTful API for Russian legal texts processing is already being made.

References

1. Ash E. et al. Precedent vs. Politics? Case Similarity Predicts Supreme Court Decisions Better Than Ideology. - 2017.
2. Bojanowski P. et al. Enriching word vectors with subword information //Transactions of the Association for Computational Linguistics. - 2017. - T. 5. - pp. 135-146.
3. Chinchor N. MUC-4 evaluation metrics //Proceedings of the 4th conference on Message understanding. - Association for Computational Linguistics, 1992. - pp. 22-29.
4. de Araujo D. et al. Exploring the inference role in automatic information extraction from texts //Proceedings of the Joint Workshop on NLP&LOD and SWAIE: Semantic Web, Linked Open Data and Information Extraction. - 2013. - pp. 33-40.
5. Farzindar A., Lapalme G. Letsum, an automatic legal text summarizing system //Legal knowledge and information systems, JURIX. - 2004. - pp. 11-18.
6. Horspool R. N. Practical fast searching in strings //Software: Practice and Experience. - 1980. - T. 10. - №. 6. - pp. 501-506.
7. Kiss T., Strunk J. Unsupervised multilingual sentence boundary detection //Computational Linguistics. - 2006. - T. 32. - №. 4. - pp. 485-525.
8. Kumar S. et al. Finding similar legal judgements under common law system //International Workshop on Databases in Networked Information Systems. - Springer, Berlin, Heidelberg, 2013. - pp. 103-116.
9. Robaldo L. et al. D2.1 Collection of state-of-the-art NLP tools for processing of legal texts,. - 2017.
10. https://www.novayagazeta.ru/articles/2019/03/20/79929-sudi-derzhat-nas-za-bolvanku