Extraction of Legal References from Court Decisions

Dávid Varga[1], Martin Gojdič[1], Zoltán Szoplák[1], Peter Gurský[1], Šimon Horvát[1], Stanislav Krajči[1] and Ľubomír Antoni[1]

[1]Institute of Computer Science, Faculty of Science, Pavol Jozef Šafárik University in Košice, Jesenná 5, 040 01 Košice, Slovakia

Abstract

The natural language processing of court decisions have become an interesting research issue worldwide. We present rule-based methods for automated extraction of legal references from Slovak court decisions. We focus on extracting references to laws and previous court decisions by using regular expressions and Levenshtein similarity measure. We created a dictionary of law names and their aliases, in order to be able to extract not only full names of laws from references, but also their generally used aliases. We annotated a set of court decisions for the evaluation of our proposed methods.

Keywords

extraction, court decisions, references, legal acts, hyperlinks

1. Introduction

The Ministry of Justice of the Slovak Republic has already published more than four million court decisions in a semi-structured form on its website. A decision document in the collection contains the raw text of the original court decision and some additional attributes, such as the name of the court, the type of the decision, or the area of law. One of the attributes contains a list of extracted references to laws present in the court decision. Unfortunately, it is prevalent that these lists are incorrect, and they:

- do not contain all references present in the decision;
- often are empty;
- contain references to laws which do not exist;
- use wrong references which are not present in the text of the court decision.

We are unsure how these references are extracted, but they are used for searching on the Ministry's website [1]. This means that incorrect court decisions are often present in search results.

The structure of the decision documents in the collection contains references to other court decisions as another attribute. However, this attribute is empty in all court decisions.

Our long-term goal is to create a system for searching and analyzing court decisions. In this system, when displaying court decisions, we would like references to legislation and other court decisions to be clickable and thus allow interactive browsing, similar to the Canadian CanLII [2] system. Currently, extracted references to laws in the Slovak court decisions dataset are presented as a list of references without any connection to reference positions in the text; therefore, without position information, it is impossible to use the references as anchors in the text.

In this article, we focus on extracting references to laws and other court decisions from this dataset. Due to the many and often occurring flaws, we will consider this dataset as unlabeled. At first sight, the references used by judges have very variable forms, but from our observations, we discovered some often re-occurring patterns which rule-based systems could handle. Therefore, we decided to extract these references using a rule-based system and create baseline methods and results with which we will compare ourselves in the future. Our approach does not only recognize the presence of mentioned references in the text but also extracts the specific laws or court decisions the judge is citing.

The aims of this paper are:

- extraction of references to exact cited laws and court decisions;
- evaluation of results with a manually annotated dataset.

This paper is organized into five sections, continuing from this point with Section 2, where we present state of the art of extracting references from legal texts. In Section 3, we describe all our rule-based methods, starting with constructing our dictionary containing commonly

ITAT 2023 Information Technologies – Applications and Theory 2023, September 22–26, 2023, Tatranské matliare, Slovakia

david.varga@student.upjs.sk (D. Varga); martin.gojdic@student.upjs.sk (M. Gojdič); zoltan.szoplak@student.upjs.sk (Z. Szoplák); peter.gursky@upjs.sk (P. Gurský); simon.horvat@upjs.sk (Š. Horvát); stanislav.krajci@upjs.sk (S. Krajči); lubomir.antoni@upjs.sk (L. Antoni)

0000-0002-3176-8106 (D. Varga); 0009-0006-2371-2205 (M. Gojdič); 0000-0003-1823-0536 (Z. Szoplák); 0000-0002-4744-7390 (P. Gurský); 0000-0002-3191-8469 (Š. Horvát); 0000-0001-5612-3534 (S. Krajči); 0000-0002-7526-8146 (L. Antoni)

used abbreviations and aliases of laws and then describing the extraction methods. In Section 4, we evaluate our methods on our golden annotated dataset, and we end this paper with a conclusion in Section 5.

2. Related work

Extracting references from legal documents has already been studied in many works. Works stretch through different legal systems, languages, and types of legal documents. This chapter will focus on the most recent works extracting references from legal documents.

In 2015, a software framework called xLLx [3] was created for court decisions from the Netherlands. This software uses rule-based methods for extracting references from court decisions, specifically a parsing expression grammar (from now on PEG). They claim that a large rule-based system has advantages over regular expressions due to the better maintainability of PEG. In this work, authors describe the recognition of aliases of full names of laws within a court decision. These aliases are called 'local aliases' and are declared after a reference with a full law name. In our work, we also focus on local aliases and found them very helpful in identifying the law that the reference points to.

Authors in [4] describe a NER tool, which uses entities from ontologies like LKIF [5] to describe legal concepts. Using Word2Vec [6] as embeddings for English texts of court decisions from the European Court of Human Rights and applying Support Vector Machine [7] classifier and simple neural network with one hidden layer. They reached relatively high accuracy. However, final F-score measures were mostly below 50%, and the discovered named entities did not reference the exact articles of laws.

Work [8] presents a newly created corpus of 350 annotated court decisions from the Czech Republic. The most specific annotated type was the court decision type. This means that annotated sub-types were the id of the court decision, the name of the court, and the date on which the decision was issued. Another annotated type was a literature reference, precisely title, author, and other possible information, such as the literature's place, year, and publisher. However, they intentionally omitted references to laws because they found them irrelevant to their broader inquiry.

On the other hand, the president of Lexum company, Ivan Mokanov, which is one of the founding members of the Free Access to Law Movement [9] presented in his web post [10] titled 'Good Old Hyperlinks,' why hyperlinks to specific law act and court decisions are still relevant. Lexum provides all services for CanLII [2] website, where citations are used for grouping similar documents, creating citators, visualizing presentations of legal content, or even making predictive tools. We also believe that extracting references from court decisions to other legal documents will be more helpful than using them only as clickable hyperlinks in our future court decisions browser.

One of the most recent works [11] on legal reference extraction has been done on a German data set, where the task was to identify references to law or other court decisions within court decisions. They could not extract specific documents that references pointed to; however, this work can be helpful for us in the future. The authors' highest-performing algorithm used BERT and had an F1-score of around 98%. Our work presents rule-based methods to extract specific references to legal acts and other court decisions. Still, later in this work, we will present our results where we have obtained a problem with false positives. We see the positive results of mentioned BERT-based model, and it is our inspiration to use a large language model to identify the reference location first and lower the number of false positive examples.

3. Methods

This chapter will focus on our implemented algorithms, which extract references to laws and other court decisions. However, we will first describe a dictionary where keys are IDs of laws and values are abbreviations and their short forms, which we call aliases. This dictionary is used to identify a specific law in a reference in the legal acts extraction method.

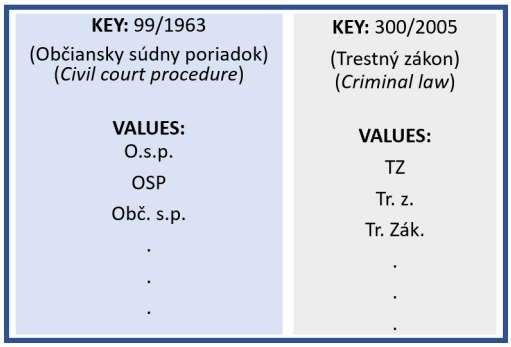

Figure 1: Example of two keys and their multiple values from the dictionary of law IDs and aliases.

3.1. Dictionary of laws and their aliases

In the Slovak Republic, there are currently more than 2000 laws that we have to consider and can be cited in court decisions. The full names and their IDs are well-known and obtainable from published laws. It is relatively easy to identify the reference to a particular law if the law's

full name or law ID is present in a text. We extracted the full names of laws and their IDs from HTML documents of laws from Slov-Lex [12] website, an official website of the Government of the Slovak Republic that publishes laws online.

The problem in identifying law from a reference starts when judges use aliases of law names in their decisions. Until now, there has not been a database or a dictionary of frequently used aliases for law names, so we decided to create our dictionary of law names and aliases used in court decisions. Figure 1 shows an example of two laws, with law IDs of Civil court procedure and Criminal law.

The dictionary has been created by extracting local aliases from Slovak court decisions similarly as described in [3]. In our dataset, local alias declarations were present after some references to laws. Typical reference to law starts with the '§' sign followed by the article number and section number or letters. These local alias declarations are usually contained in parentheses where the text 'ďalej len' (meaning 'from now on') and the text of the alias are present. The critical observation is that there is almost always a full name of the law before the local alias declaration, which we use to match extracted alias with the correct law ID. We show two examples of these declarations of local aliases for two different laws in Figure 2, where full names of laws are highlighted in pink, local alias declarations are highlighted in lime color, and aliases are written in bold format.

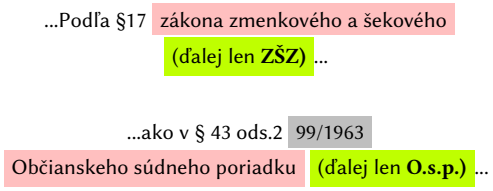

Figure 2: Examples of references to laws with full names in pink, declarations of local aliases in lime, law ID in gray, and aliases in bold format.

The dictionary of law aliases that is used in the algorithm for extraction of law references, has been created by extraction from all court decisions using the following steps.

STEP 1: Reference recognition. The first step of the extraction was searching the '§' signs and taking a text 200 characters long after the '§' sign to narrow down the search for local alias declaration. Then we search for occurrences of '(ďalej len' in each found text, following the '§' sign.

STEP 2: Alias extraction. The alias was extracted from text between '(ďalej len' and the closing parenthesis ')'.

STEP 3: Law identification. First, we search for the law ID between the alias declaration and the '§' sign. If the law ID was not found, we tried to find one of the full names of laws from our database in the mentioned text of a reference.

We measure similarities between law names from the database and the text of a reference with the Levenshtein distance [13]. To obtain better results, we lemmatized both full names of laws from our database and the text from the reference because the Slovak language uses different inflected forms of the exact words. The lemmatization was done by a word form dictionary called Tvaroslovník [14], a database of word forms with their lemmatized form.

Article numbers, sections, and letters, which specify the exact parts of laws, also created a problem with matching the full names of laws. The solution was that only a chain of words from the reference that Tvaroslovnik could lemmatize was used in the Levenshtein distance comparison.

Then, the dictionary was manually cleaned from unwanted results, e.g., when an abbreviation of some other term than a law name was extracted. After processing all court decisions (cca 4 million) from our dataset, the final dictionary contained 209 aliases that matched 111 laws.

3.2. Identifying law from a reference

As we described earlier, we created the dictionary for the method that identifies a specific law from a given text. This method has a short text at the input, and the output contains information about extracted law. We show the example of the output extraction object in Figure 3, where the judge used the alias 'O.s.p.' for referencing Civil court procedure.

```
{
    isAliasDeclaration=False
    isAliasUsed=True
    lawId=99/1963
    lawName='Občiansky súdny poriadok'
    previousLawUsed=False
    textInDecision='O.s.p.'
    start=13
    end=22
}
```

Figure 3: Extraction result of law name from single reference.

We describe the algorithm of identifying the law from a single part of text following '§' sign in the next steps, where from each step, if we extract a result, we compare it to the previous best result. More specifically, we compare the offset (field *start* in the Figure 3) of the result within

a given text, and when it is closer to the '§' sign, i.e., the offset is smaller, then we take that result as the best current result. This set of steps runs iteratively for each found '§' sign.

STEP 1: Local alias. If the set of local aliases is not empty (i.e. the set was filled with local aliases in previous iterations), then the presence of some already extracted local alias from the current court decision is verified. If we do not find a local alias, we repeat this verification on the lemmatized text.

STEP 2: Law ID and alias declaration. Law ID is extracted by regular expression `\d{1,3}/\d{4}`. If law ID is present, we check whether there is a local alias declaration. If the local alias declaration is present, we extract it in the same way described in the previous subsection 3.1 and insert it into a set of local aliases for the current decision.

STEP 3: Dictionary of aliases. In this step, the dictionary of laws and aliases is used to check for the exact match of any alias from the dictionary. If we do not find an alias from the dictionary, verification proceeds again on the lemmatized text.

STEP 4: Full name of the law. The full names of laws are compared to lemmatized text as described in the previous subsection 3.1.

STEP 5: Previous law. This step deals with a situation when we do not find matches in previous steps, i.e., the text following the '§' sign. We found out from our observations that most of these happen because no law is specified after the '§' sign of a current reference. Also, that judge cites the articles of law mentioned in a previous reference. That is why we maintain the most recently extracted law from the current court decision in the previous context and assign this law.

As seen in Figure 3 we extract the start and end offset within the text of the current examined reference for future use in creating clickable links.

3.3. Parsing details from law reference

In this subsection, we describe parsing article numbers, section numbers, and letters from a given reference to create a hyperlink to the referenced law. We have to count with different abbreviations, multiple article numbers, typos, ranges of letters, and section numbers.

In Figure 4, we show a complex reference translated to English where we highlighted the input for our parsing algorithm in pink. As we can see, only a part before law

§ 172, § 234 section 1 and § 171b letter c) to e) Cr. proc.

Figure 4: An example of a more complex reference, the input for the method which parses details, is highlighted in pink.

name is used as the input for this algorithm, which is a part before the starting offset of the law name. This single reference points to these laws:

1. § 172 of the Criminal procedure;
2. § 234 section 1 of the Criminal procedure;
3. § 171b letter c) of the Criminal procedure;
4. § 171b letter d) of the Criminal procedure;
5. § 171b letter e) of the Criminal procedure.

To avoid repeating ourselves, after the first step of the algorithm, we will only show the parsing of details on the last part of reference – '§ 171b letter c) to e)', and the results from this part will be laws 3, 4, and 5. However, the algorithm outputs all five laws from the reference.

STEP 1: Separate by §. In the first step, we separate the highlighted part by '§' signs.

§ 172, § 234 section 1 and § 171b letter c) to e) Cr. proc.

STEP 2: Article number. For each part of the reference, separated in the previous step, we extract the article number and separate it from the rest of the part. We achieve this separation using the following regular expression `(\d+[a-z]{0,2})((.|\s)*)`. The first parentheses of the regular expression extract the article number, which we highlighted in gray. In contrast, the second parentheses extract the rest of the text, which we highlighted in lime.

§ 172, § 234 section 1 and § 171b letter c) to e) Cr. proc.

STEP 3: Tokenizing article details. We tokenize article details, which we highlighted in lime. We use a regular expression for tokenization, which extracts only alphanumeric parts of tokens.

§ 172, § 234 section 1 and § 171b letter c) to e) Cr. proc.

STEP 4: Parsing article tokens. We process and categorize each token from the previous step. These categories are:

- the letter or section word (lime);
- specific letter or number of the section (yellow);

§ 172, § 234 section 1 and § 171b letter c) to e) Cr. proc.

- declaration of a range (cyan).

To identify whether one of the previously mentioned tokens is a letter word or a section word was done by comparing Levenshtein distances to words 'písme' (part of word 'písmeno', meaning letter) and 'odse' (part of word 'odsek', meaning section).

We discovered the use of 'písme' and 'odse' by analyzing every word in extracted references longer than two characters from a set of 320,000 court decisions. Judges use mostly abbreviations 'pís.', 'písm.' or the full word 'písmeno' to specify upcoming letters. Part of the word 'písme' gives us the smallest average Levenshtein distance to these abbreviations and complete words. Part of the word 'odse' was discovered analogically.

3.4. References to other court decisions

This subsection explains how we extracted references to other court decisions. We must note that we have focused only on references to other Slovak court decisions. In the future, we plan to extract references to judgments from the European Court of Human Rights and the Court of Justice of the European Union too.

A reference to a court decision mainly consists of three parts. In most cases, the judge specifies the name of the court on which referenced court decision was made. After that, the date of the procedure is stated, after which a file number, or what we call a docket number, is specified. We show an example of a reference to the court decision that we translated to English in Figure 5.

… Regional court Bratislava from day 17.10.2011 d. num. 7C/169/2010-59 …

Figure 5: Example of a reference to other court decision. The name of the court is highlighted in yellow, the date of the proceeding in lime, and the docket number in pink.

We need to extract all three parts of the reference to pinpoint the court decision the judge refers to because docket numbers are not unique. A case can be processed on the same court under the same docket number in more than one court decision, and to distinguish them, we need to extract the date as well. Also, different court cases can be processed under the same docket number. That is why we need to extract the name of the court as well. We explain this extraction in the following steps.

STEP 1: Docket numbers. We extract docket numbers with the use of a regular expression. Then, we take 90 characters long text before the docket number for each docket number and 90 characters long text after the docket number. Docket numbers are some sort of anchors around which we extract the date and the name of the court.

STEP 2: Court name. First, we try to find the full name or abbreviation of special courts in the text, e.g., the supreme court. If the name of a special court is absent, we proceed with identification, whether a county or regional court is present. Then we search only for city names with a court of the identified type. To measure the similarity between texts and court names, we use the *find_near_matches()* method from the fuzzysearch [15] library.

STEP 3: Date. There can be many dates present in court decisions, but in most cases, the dates between the names of the courts and the dockets are the ones that belong to the references. Therefore, we search for a date primarily between the court name and the docket number. The second most common way of stating the reference date is not far after the docket number in the text.

From our observations, we have noticed that court names were always present before the docket number, and therefore, we searched for them only in this part of the text. We also have a collection of the most commonly used abbreviations of court names, e.g., 'NSSR' (short for Najvyšší súd Slovenskej republiky) for the Slovak Supreme Court.

To extract the date, we currently use our date extractor, which uses the regular expression `(\d{1,2}).(\d{1,2}).(\d{4})` supporting the most widely used date format in court decisions. All extracted parts contain the offset of the court decision text, which we use to create clickable hyperlinks in our court decision browser.

4. Evaluation

We evaluated our methods on two datasets, each containing 20 manually annotated court decisions. The first dataset is annotated with references to laws, and the second is annotated with references to other court decisions. We should also mention that our datasets do not contain offsets of the references, and each court decision contains a set of distinct references. In the future, we plan to make a larger annotated dataset, even with offsets and all occurrences.

4.1. Results for law references

In the dataset for testing extraction of references to laws, we created an attribute called **annotated_laws**, which contained references to laws present in the court decision. We used the same format as the format used in the original dataset from the Ministry of Justice, which contains the law's ID, article number, section number, and a letter.

We show an example of **annotated_laws** attribute in Figure 6 for one of the court decisions from our annotated dataset.

annotated_laws: [
/SK/ZZ/1963/99/#paragraf-172.odsek-1
/SK/ZZ/1963/99/#paragraf-174.odsek-1
/SK/ZZ/1963/99/#paragraf-175
/SK/ZZ/2005/300/#paragraf-74.odsek-1.pismeno-b
]

Figure 6: Example of the set of extracted law references for one court decision.

We show the final results of our method, which extracts the references to laws and compare them with the annotations from Slov-Lex dataset Table 1. There were altogether 280 annotated references, on average 14 per court decision. We have managed to extract 249 of them, giving us a sensitivity of about 88.93%.

	our method	Slov-Lex
exact matches	**249**	**34**
partial matches	**7**	**43**
false positives	**12**	**3**
not found	**24**	**203**
annotated references	**280**	

Table 1
Results of extracting references to laws.

We have also managed to extract six incomplete references, where we have managed to identify the correct law and article number. Still, sometimes we miss the extraction of a letter or a section. There was also one case where the annotated reference contained a sentence as a part of the article, which our algorithm cannot extract. These seven incomplete references make up 2.5% of annotated references.

Another row of the 1 contains the number of false positives, which added up to 12. Those are the references we extracted but are not present in the court decision. In most cases, we did not identify the correct law, and they make up about 4.4% of all extracted references by our algorithm, giving us a precision of 95.4%.

The last row contains 24 references that we did not manage to find, and they make up about 8.6% of actual references that we missed, which gives us an F1-score of 92.05%. The reasons we did not manage to extract these 24 references were very varied. For example, our algorithm did not extract article number ranges from a reference '§ 243i to 243k'.

The Slov-Lex dataset has been annotated with only 34 exact matches, 43 partial matches, and missed completely 203 references. However, there were only three false positive annotations, which is better than our method.

Table 2 presents results of extracting references to other court decisions. Out of 34 annotated references, we managed to extract 24 of them, giving us a sensitivity of 70.59%. This result is worse than the result of extracting references to laws, but on the other hand, we got only one false positive reference, which gives us a precision of 96%.

On the other hand, we obtained seven partial matches, which make up 20.59% of annotated references. We consider a partial match a match that does not contain one of the three extracted attributes of references. In 5 cases, we could not extract the date of the reference, and in the rest, we did not extract the name of the court.

We could not extract 3 out of 34 annotated references, which is 8.8%. We could not extract these three references because we did not manage to find a docket number in the first place, and then we did not even search for a court name and a date. Overall, for this task, we achieved an F1-score of 81.36%.

annotated references	**34**
exact matches	**24**
partial matches	**7**
false positives	**1**
not found	**3**

Table 2
Results of extracting references to other court decisions.

5. Conclusion

This paper presented rule-based methods for extracting references to other court decisions and laws. We achieved an F1-score of 92.05% for extracting references to laws and an F1-score of 81.36% for extracting references to other court decisions. We believe there is room for improvement, especially in lowering the number of false positives for extracting references to laws and extracting dates and court names for extracting references to other court decisions.

One of the challenges that emerged during the extraction of references to laws in court decisions is the identification of the correct version of the law. Slovak laws are continuously modified and updated, and it is common for a judge to refer to a version of the law that was not

up to date but was applicable when the act was committed. Therefore, incorporating domain-specific knowledge, such as legal systems or contextual understanding, could enhance the accuracy and comprehensiveness of reference extraction in the Slovak legal domain.

We consider this experiment to be a baseline for extracting references from Slovak court decisions, and we also published a golden annotated dataset[16]. By addressing these challenges and pursuing future research directions, we aim to establish a solid foundation for automated reference extraction in Slovak court decisions, ultimately facilitating a system for searching and analyzing court decisions.

Acknowledgments

The Slovak Research and Development Agency supported this work under contract No. APVV-21-0336 Analysis of court decisions by methods of artificial intelligence. Pavol Jozef Šafárik University in Košice supported this work with the internal project at vvgs-2023-2547 Legal text analysis using computer linguistics. This article was also supported by the Scientific Grant Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic under contract VEGA 1/0645/22 entitled by Proposal of novel methods in the field of Formal concept analysis and their application.

References

[1] Rozvoj elektronických služieb súdnictva (RESS) (2016). URL: https://obcan.justice.sk/.

[2] The Canadian Legal Information Institute (CanLII) (2001). URL: https://www.canlii.org/.

[3] M. van Opijnen, N. Verwer, J. Meijer, Beyond the experiment: the extendable legal link extractor, in: Workshop on Automated Detection, Extraction and Analysis of Semantic Information in Legal Texts, held in conjunction with the 2015 International Conference on Artificial Intelligence and Law (ICAIL), 2015.

[4] C. Cardellino, M. Teruel, L. A. Alemany, S. Villata, A low-cost, high-coverage legal named entity recognizer, classifier and linker, in: Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law, 2017, pp. 9–18.

[5] R. Hoekstra, J. Breuker, M. Di Bello, A. Boer, et al., The lkif core ontology of basic legal concepts., LOAIT 321 (2007) 43–63.

[6] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, 2013. arXiv:1301.3781.

[7] C. Cortes, V. Vapnik, Support-vector networks, Machine learning 20 (1995) 273–297.

[8] J. Harašta, J. Šavelka, F. Kasl, A. Kotková, P. Loutocký, J. Míšek, D. Procházková, H. Pullmannová, P. Semenišín, T. Šejnová, N. Šimková, M. Vosinek, L. Zavadilová, J. Zibner, Annotated corpus of czech case law for reference recognition tasks, in: P. Sojka, A. Horák, I. Kopeček, K. Pala (Eds.), Text, Speech, and Dialogue, Springer International Publishing, Cham, 2018, pp. 239–250.

[9] The Free Access to Law Movement (FALM), 2002. URL: http://falm.info/.

[10] I. Mokanov, Good Old Hyperlinks, 2017. URL: https://www.slaw.ca/2017/10/20/good-old-hyperlinks/.

[11] S. Peikert, C. Birle, J. Al Qundus, V. Le Duyen Sandra, A. Paschke, Extracting references from german legal texts ing named entity recognition (2022).

[12] Slov-Lex, 2022. URL: https://www.slov-lex.sk/vyhladavanie-pravnych-predpisov.

[13] V. I. Levenshtein, et al., Binary codes capable of correcting deletions, insertions, and reversals, in: Soviet physics doklady, volume 10, Soviet Union, 1966, pp. 707–710.

[14] S. Krajči, R. Novotný, Tvaroslovník–databáza tvarov slov slovenského jazyka, in: Proceedings of international conference ITAT 2012, SAIA, 2012, pp. 57–61.

[15] T. Einat, fuzzysearch, https://github.com/taleinat/fuzzysearch, 2022. GitHub repository.

[16] M. Gojdič, D. Varga, Slovak court decisions, annotated dataset, https://github.com/vargadavid304/slovak-court-decisions-dataset, 2023. GitHub repository.