D21 – Model for predicting the verdicts (EN)

Project funded by the European Union's Next Generation EU through the Slovak Republic's Recovery and Resilience Plan under project no. 09I05-03-V02-00049.

Financované Európskou úniou NextGenerationEU
PLÁN OBNOVY
ÚRAD PODPREDSEDU VLÁDY SLOVENSKEJ REPUBLIKY PRE PLÁN OBNOVY A ZNALOSTNÚ EKONOMIKU
VAIA VÝSKUMNÁ AKTIVAČNÁ AUTORITA
VÝSKUMNÁ AGENTÚRA

INTRODUCTION

The project **"Automation of Legal Text Analysis based on Machine** Learning" (hereinafter "ALTAML") represents a key initiative aimed at integrating innovative approaches in the field of data processing and subsequent analysis, with a focus on the field of law, which also includes information and communication technology law. The aim of this project is to develop and validate effective methods for the automated analysis of legal texts using machine learning techniques. This involves, in particular, the development of tools that facilitate the processing and analysis of large volumes of data in the form of various legal documents, as well as the extraction of relevant information (attributes) from such documents, including the identification of key terms, references to legislation and other attributes.

The aim of the ALTAML project is therefore to contribute to a more efficient approach to legal information and to the acceleration of legal processes, thereby ensuring a higher degree of legal certainty and improving the accessibility of legal texts, the results of their analysis and relevant legal information for both professionals and the general public.

As part of work package KPB4, the output 'Model for predicting the verdicts' was developed. The proposed system represents a hybrid legal analysis pipeline for the automated classification of the outcome of criminal proceedings from Slovak court decisions and the subsequent extraction of imposed sentences. The solution combines deterministic rules based on legal provisions, semantic vector representations of documents, and a large language model deployed locally via the Ollama tool.

The system utilises a multi-layer Tier 0–Tier 2 architecture, in which each layer ensures a step-by-step decision-making process with an increasing degree of semantic processing. The primary objective of the system is to identify whether the defendant was acquitted or convicted, and in the event of a conviction, to automatically extract the sentences imposed, including their type, severity, conditions and details of sentence enforcement.

To improve decision-making accuracy, the system utilises a Retrieval-Augmented Generation (RAG) approach based on retrieving historically similar cases using embeddings. Each decision is represented by a vector representation created using the BGE-M3 embedding model, with the similarity of cases evaluated via the cosine similarity between the embeddings of individual documents. The most relevant historical cases are then used as few-shot examples for a large language model.

The final decision on sentence segmentation and extraction is made by the locally deployed LLaMA 3.3 language model, which is guided by strictly defined prompts formulated in Slovak. The model's outputs are deterministically restricted to JSON

structures, thereby ensuring the consistency and machine-processability of the results. A more concise description can be found below.

A hybrid system for predicting the outcome of criminal proceedings and extracting sentences from Slovak court decisions

The system is a software module implemented in Python, designed to automatically analyse Slovak criminal court rulings. The pipeline combines rule-based legal reasoning, semantic search for similar cases, and a large language model to achieve accurate classification of rulings and extraction of the sentences imposed.

System functionality

The main functionality of the system consists of the following steps:

1. Preparation of input data

The system works with pre-processed structured decisions, which contain:

- a statement of facts,
- identified criminal offences,
- mitigating and aggravating circumstances,
- identified evidence,
- referenced legislation,
- metadata on the decision.

The input texts are aggregated into a unified contextual representation, which is then used for both embeddings and language model inference.

2. Semantic representations of documents

Each decision is represented by an embedding generated using the BGE-M3 model. The embedding includes:

- a factual description,
- identified criminal offences,
- mitigating circumstances,
- aggravating circumstances,
- identified evidence.

Embeddings are stored in a local database in JSON format, thereby minimising the need for repeated calculations during further processing.

3. Multi-layer decision classification

3.1. Tier 0 – Detection of cessation of action

The first layer of the system identifies cases that have been discontinued or suspended on the basis of explicit references to specific sections in the decision. Such cases are automatically excluded from further processing.

3.2. Tier 1 – Rule-based classification of exemptions

The second tier utilises deterministic legal rules based on identified statutory provisions.

The system analyses the occurrence of:

- sections indicating acquittal,
- paragraphs indicating conviction,
- dominant provisions imposing a conviction.

Based on these rules, the system decides:

- acquittal = true,
- acquittal = false,
- or an indeterminate result, which proceeds to the next layer.

3.3. Tier 2 – Hybrid decision-making using RAG and LLM

If the rule layer is unable to make a reliable decision, the system activates a hybrid mode based on Retrieval-Augmented Generation (RAG).

For the current case:

- the decision embedding is calculated,
- the most similar historical cases are retrieved,
- few-shot examples with the highest cosine similarity are selected.

These cases are then fed into the prompt of a large language model, which makes the final decision on acquittal or conviction.

4. Extraction of imposed sentences

If the decision is classified as a conviction, the system performs automatic extraction of the imposed sentences.

The language model identifies:

- the type of sentence,
- the length of the sentence,
- the conditions of the sentence,
- the category of the sentence,
- additional details of enforcement.

5. Retrieval-Augmented Generation (RAG)

A key component of the system is the mechanism for searching for similar cases.

For each analysed document:

- an embedding is calculated,
- it is compared with the embeddings of historical decisions,
- the most relevant cases are selected using cosine similarity.

These cases are subsequently used as few-shot examples in the inference of a large language model, thereby increasing the stability and legal consistency of the outputs.

6. Prompt engineering and deterministic outputs

The system uses strictly defined Slovak prompts designed for:

- classification of exemptions,
- sentence extraction,
- validation of the output structure.

The model is explicitly restricted:

- to output in JSON format,
- to the use of defined penalty types,
- to the consistent use of legal terminology.

Inference is performed at zero temperature (temperature = 0.0), thereby minimising the variability of the generated responses.

System output

The system output is a JSON structure which, for each decision identifier, contains:

- information regarding acquittal or conviction,
- a list of previous convictions,
- the severity and conditions of the sentence,
- information on skipped cases.

Results are saved continuously during processing, ensuring the pipeline's resilience to calculation interruptions.

Requirements and technologies

The system is implemented in Python and utilises the following technologies:

- a local LLM (LLaMA 3.3) for legal inference,
- the BGE-M3 embedding model for semantic representations,
- cosine similarity for searching for similar cases,
- Retrieval-Augmented Generation (RAG),
- a JSON embedding database for caching representations,
- Ollama API for local model inference.

32 GB of RAM or more is recommended. GPU acceleration (CUDA) significantly speeds up both embedding calculations and language model inference when processing large legal corpora.

Source files

The project's source code and data file are maintained in a repository and are available at: https://gitlab.science.upjs.sk/altaml/verdict-prediction