LLM Project Course IV

Project Name: Legal Assistant

Oskar Tillmann Winter, Lukas Tesch, Michael Hecher

Inhaltsverzeichnis

1. Introduction

1.1. Task

Our task is to develop an LLM that supports legal experts in identifying relevant laws for specific cases. The idea is straightforward: the user provides a case description, and the model outputs the specific legal articles or paragraphs that may apply. It is important to note that the model will not predict case outcomes or provide legal conclusions. Such tasks would require specialized legal expertise that we do not have and cannot train the model for. Our focus is strictly on information retrieval, not legal interpretation.

To build this system, we first need to supply the model with the necessary legal texts. For this project, we will focus on German law as our foundation. In addition, we need datasets containing real cases along with the laws applied in each decision. Although such data exists online, it typically requires web scraping, cleaning, and preparation before it can be used. This preprocessing step will form the first phase of our project.

1.2. Data

When searching for suitable datasets, we quickly realized that there were almost no publicly available, pre-labeled datasets that matched our specific use case. However, there is a large amount of openly accessible legal text online. For this reason, we decided to build our own dataset by scraping case data from the Open Legal Data platform.

Using Python, we retrieved the raw case files, cleaned the HTML content, and converted each document into plain text. In total, we collected around 1,200 cases. From these, we constructed two separate datasets:

- A large self-supervised training dataset consisting of 1,000 cleaned case texts. This corpus allows our model to learn the structure, terminology, and linguistic patterns of German legal writing without requiring labels.
- A dataset consisting of 200 legal cases, where for each case, we extracted all statutory citations using our custom regular expression pipeline and then manually verified the results to ensure labeling accuracy. This annotated dataset serves two purposes: (1) it is used for evaluating the system's ability to retrieve the correct legal provisions, and (2) it forms the supervised component of our training procedure. It is important to acknowledge that using the same dataset for both supervised fine-tuning and evaluation may introduce bias, as the model is partially evaluated on data it has already seen.

Together, these datasets provide a solid foundation for building and validating our legal language model.

```python
def fetch_cases_clean(n):
    cases = []
    url = "https://de.openlegaldata.io/api/cases/?offset=0"

    while url and len(cases) < n:
        resp = requests.get(url).json()
        cases.extend(resp["results"])
        url = resp.get("next")

    cases = cases[:n]

    cleaned_cases = []

    for case in cases:
        raw_html = case.get("content", "")
        soup = BeautifulSoup(raw_html, "html.parser")
        plain_text = soup.get_text(separator="\n", strip=True)

        cleaned_cases.append({
            "id": case.get("id"),
            "court": case.get("court", {}).get("name"),
            "date": case.get("date"),
            "text": plain_text
        })

    return cleaned_cases

# Fetch 1200 cases total
cases_all = fetch_cases_clean(1200)
train_cases = cases_all[:1000]
test_cases  = cases_all[1000:1200]
print("fertig")
```

We also explored fetching full legal books (law texts, commentaries, and statutory compilations) and integrating them into our model. However, this turned out to be a significantly larger task than expected. The length, complexity, and hierarchical structure of legal books would require a far more advanced LLM architecture with extended context windows and more sophisticated preprocessing. This idea remains valuable, but it is better suited for future work once the foundational model is in place.

In this project, we focused on a more manageable and targeted approach: extracting individual court cases from the Open Legal Data platform and then identifying the legal citations within them. Using our custom regular expression pattern, we were able to capture most common forms of German legal references and attach them to the corresponding case. This provided us with a reliable dataset for both self-supervised pretraining and supervised evaluation.

```python
soup = BeautifulSoup(raw_html, "html.parser")
text = soup.get_text(separator=" ")

pattern = r"""
(§{1,2}              # § or §§
\s*
[\d,\s\-–abcfIVX]+   # numbers, letters, ranges, multiple sections
\s*
([A-ZÄÖÜ]{2,10})     # law abbreviation (ZPO, BGB, StGB, GG, etc.)
)
"""

matches = re.findall(pattern, text, flags=re.VERBOSE)

# Clean
citations = [m[0].strip() for m in matches]
print(citations)
```

```
['§ 242 BGB', '§ 203 BGB', '§§ 91, 709 ZPO']
```

This updated pattern successfully captured all relevant legal citations.

Our Files

training_cases.csv

This file contains our initial training dataset. It includes 1,000 legal cases, each with its corresponding court, ID, and date. The text column holds the full case description including the facts, the outcome, and the laws applied in each decision.

	id	court	date	text
0	325566	Landgericht Köln	2029-11-13	Tenor\nI. Die Beklagte wird verurteilt, der Kl...
1	323770	Sozialgericht Düsseldorf	2027-04-06	Tenor\nDie Klage wird abgewiesen. Außergericht...
2	343880	Landesarbeitsgericht Düsseldorf	2022-12-15	Tenor\n1. Auf die Berufung des Klägers wird da...
3	343848	Amtsgericht Essen	2022-10-14	Tenor\nDer Vollstreckungsbescheid des Amtsgeri...
4	346952	Oberlandesgericht München	2022-10-13	Tenor\nI. Die Berufung der Beklagten gegen das...

test_cases.csv

The second file contains our test dataset, which we use to evaluate the model. In this dataset, we extracted all laws and specific paragraphs referenced in each case. This allows us to assess how accurately the model identifies the relevant legal provisions.

	id	court	date	text	citations
0	345731	Verwaltungsgericht Göttingen	2022-06-17	Tatbestand\n1\nDer Kläger begehrt die Flüchtli...	['§ 60 Abs. 5 und 7 Satz 1 AufenthG', '§ 102 A...
1	345695	Oberverwaltungsgericht Nordrhein-Westfalen	2022-06-17	Tenor\nDer angefochtene Beschluss wird geänder...	['§ 81 BauO', '§ 81 Abs. 1 BauO', '§ 81 Abs. 2...
2	345694	Oberverwaltungsgericht Nordrhein-Westfalen	2022-06-17	Tenor\nDie Beschwerde wird zurückgewiesen.\nDe...	['§ 146 Abs. 4 Satz 6 VwGO', '§§ 80 Abs. 5, 80...
3	345693	Oberverwaltungsgericht Nordrhein-Westfalen	2022-06-17	Tenor\nDer Antrag des Klägers auf Zulassung de...	['§ 78 Abs. 3 Nr. 1 AsylG', '§ 78 Abs. 3 Nr. 1...
4	345692	Oberverwaltungsgericht Nordrhein-Westfalen	2022-06-17	Tenor\nDer Antrag der Kläger auf Zulassung der...	['§ 78 Abs. 3 Nr. 1 AsylG', '§ 78 Abs. 3 Nr. 1...

2. Models

In specific we used these three models: “mpnet”, “legal-bert”, “legal-sbert”.

2.1. MPNet

MPNet combines the strengths of two major pre-training strategies: BERT’s Masked Language Modeling (MLM) and XLNet’s Permuted Language Modeling (PLM). MLM masks tokens and predicts them independently, which prevents the model from capturing dependencies between masked words. PLM, in contrast, explicitly models dependencies among predicted tokens but lacks full positional information during pre-training, creating a mismatch with fine-tuning.

MPNet resolves these limitations by splitting the input sequence into non-predicted and predicted token sets. It then applies two-stream self-attention, introduced in XLNet, to model autoregressive dependencies in the predicted tokens. In addition, MPNet introduces position compensation, giving the model access to the full positional information of the entire sentence and removing the positional discrepancy present in XLNet. For the non-predicted part of the sequence, MPNet maintains bidirectional encoding, preserving BERT's ability to use full contextual information.

As demonstrated in the paper by Song et al. (2020), MPNet outperforms both MLM-based and PLM-based models and achieves state-of-the-art performance across a wide range of NLP tasks.

2.2. Legal-BERT

Legal-BERT is specifically designed for the legal domain, addressing limitations of generic BERT models when applied to highly specialized legal language. Legal text differs substantially from general corpora in vocabulary, syntax, and semantics, often functioning as its own "sublanguage." As shown in the paper by Chalkidis et al. (2020), models pre-trained on general text underperform on legal tasks, while domain-adapted models significantly improve accuracy and robustness. Furthermore was Legal-BERT trained on a large and diverse body of legal documents, such as EU and UK legislation, ECJ and ECHR cases, U.S. case law, and U.S. contracts.

2.3. Legal-SBERT

Building directly on this foundation, Legal-SBERT extends the benefits of Legal-BERT into the domain of sentence-level representation. While Legal-BERT improves token-level and document-level encoding, many legal NLP applications, such as similarity search, clause matching, contract review, and precedent retrieval, require high-quality sentence embeddings. Legal-SBERT uses the Sentence-BERT architecture combined with a Legal-BERT encoder to produce embeddings that reflect legal meaning more accurately than standard SBERT models. Experimental results show that replacing BERT with Legal-BERT within SBERT leads to meaningful performance gains on demanding legal tasks, confirming that domain-specific pretraining remains valuable even at the sentence-embedding level (Chauhan, 2023).

3. Code

3.1. Preparation

To build our three models, we first had to configure and select the model architectures. All hyperparameters, folder paths, and general settings were centralized in one configuration block to ensure consistency and easy adjustments. We then created a dictionary to store and manage our three models. In addition, we enabled caching and defined an output directory to store embeddings, checkpoints, and performance metrics. We also selected the relevant legal texts, which we retrieved directly from URLs. This preparation step was crucial, as it allowed us to define all parameters, model choices, output directories, and data sources in a structured and reproducible way.

The next step was to load our prepared datasets. After loading, we performed validation checks to ensure the data was complete and correctly formatted. This included verifying that all required columns were present and contained valid entries. Once validated, we stored the cleaned datasets in new variables for further processing.

3.2. Improved Unsupervised Pre-training

Next to support our multiple transformer architecture, we implemented a felxible model loading function. This function enable the switching between the models without modifying the training pipeline. The first training stage perfroms unsupervised pre training on a sample subset of the legal cases. This steps is included to adapt the model to the linguistic characteristics of German legal texts before supervised fine-tuning. This is implemented by each case duplicating into a pair and then using the Multiple Negatives Ranking Loss to map semantically similar texts to nearby regions in the embedding space

3.3. Robust Citation Parsing

To standardize heterogeneous citation formats in the dataset, we implemented a robust parsing function that normalizes all citation fields into a list of strings. The function handles missing values, native Python lists, JSON-encoded lists, and Python list. If structured decoding fails, the parser falls back to splitting by common delimiters or treating the input as a single citation. This ensures reliable and consistent citation labels for supervised training and evaluation.

3.4. Supervised Fine-Tuning

After the unsupervised learning phase, the model underwent supervised fine-tuning to learn the explicit and task-specific relationship between case texts and the laws they reference. In this stage, the model is provided with real, historically grounded examples: each case in the dataset contains one or more cited legal norms, which function as positive supervision signals. Before training, the raw citation fields are normalized to obtain a clean and standardized list of statute identifiers. Each case–citation pair is then transformed into an InputExample with a positive label and used to continue training with the Multiple Negatives Ranking Loss. This process further shapes the embedding space so that relevant legal provisions are positioned close to their corresponding case texts, thereby improving the model's ability to retrieve accurate statutes for unseen cases.

3.5. Enhanced HTML Scraping with Caching

To build the statutory corpus required for the retrieval task, we developed a scraping and caching pipeline that extracts individual legal provisions directly from the official "Gesetze-im-Internet" webpages. The system retrieves the full HTML of each statute and identifies section markers through a combination of structural HTML and regular-expression pattern matching for citation formats like "§ x", "Art. x", or "Artikel x". The associated text is cleaned, consolidated, and normalized to produce a consistent representation of each provision. Because the formatting of different laws varies significantly, the pipeline integrates a secondary text-based extraction method that segments the raw HTML by section markers. Extracted provisions are cached locally to ensure reproducibility and reduce network overhead, and a small fallback corpus is used when scraping fails due to irregular formatting or connection issues. This process

produces a unified, machine-readable set of statutory sections suitable for embedding and downstream retrieval.

3.6. Build FAISS Index

To support efficient legal retrieval, all laws are embedded using the fine-tuned sentence transformer and indexed with FAISS.

FAISS (Facebook AI Similarity Search) is a high-performance library for large-scale vector similarity search. It enables efficient nearest-neighbor retrieval in high-dimensional embedding spaces and scales to millions of vectors, outperforming traditional database search methods.

Given a case text, the system encodes the query and returns the top-k most similar provisions, with citation matching facilitated by normalization and fuzzy comparison to account for formatting variability. Retrieval performance is evaluated on held-out cases using standard information-retrieval metrics such as Recall, Precision, F1, Hit Rate, MRR (Mean Reciprocal Rank), and MAP (Mean Average Precision), providing a comprehensive assessment of how accurately the system identifies the relevant statutory sections.

Mean Reciprocal Rank (MRR) is a ranking metric that evaluates how quickly a retrieval system returns the first relevant result for a query. For each query, the reciprocal rank is computed as the inverse of the position of the first correct item (e.g., 1 for rank 1, ½ for rank 2, ¼ for rank 4). The MRR is then obtained by averaging these reciprocal ranks across all queries. A higher MRR indicates that the system tends to rank relevant results near the top of the list.

Mean Average Precision (MAP) is a ranking metric that evaluates both how many relevant items a system retrieves and how highly these items are ranked. For each query, the metric first computes the Average Precision (AP), which is the average of the precision values at all ranks where relevant items appear within the top-K results. The MAP score is then obtained by taking the arithmetic mean of these AP values across all queries or users. MAP ranges from 0 to 1, with higher values indicating that the system consistently places relevant items near the top of the ranked list.

3.7. Main Pipeline with Multi-Model Comparison

The final stage of the project integrates all components into a complete training, indexing, and evaluation pipeline. This pipeline supports both single-model training and multi-model comparison. For each selected transformer architecture, the system first performs unsupervised domain adaptation on raw case texts, followed by supervised fine-tuning on case–citation pairs. After training, all laws are embedded and indexed with FAISS, enabling efficient semantic retrieval. A unified retrieval module is then created for inference and evaluation.

The system evaluates each model on the same held-out dataset using a comprehensive set of ranking metrics. These metrics allow direct comparison of retrieval effectiveness across different base models. The framework automatically summarizes performance, identifies the best-performing architecture for each metric, and returns the highest-scoring model for downstream use. The pipeline also provides an example retrieval output to illustrate how the system identifies relevant statutory provisions for a given case text.

4. Model Comparison

4.1. Metrics

MPNet

The baseline MPNet model shows very limited retrieval capability on the legal test set. The model achieves a Recall@5 of only 0.0075, meaning it retrieves fewer than 1% of all relevant statutes within the top five results. Precision@5 is similarly low (0.0667), indicating that only a small fraction of retrieved items are actually relevant. The resulting F1@5 score of 0.0135 reflects this severe imbalance.

The Hit Rate@5 of 0.15 reveals that the model returns at least one correct citation for only 10 out of 66 test queries, while the Mean Reciprocal Rank (0.0687) and MAP (0.0687) confirm that even when correct citations are retrieved, they tend to appear at low ranks and with low precision.

After supervised fine-tuning on our labeled dataset, MPNet shows substantial improvements across almost all metrics. Recall@5 increases from 0.0075 to 0.0307, and Precision@5 rises from 0.0667 to 0.2727, representing significant gains in both the number and correctness of retrieved citations. The Hit Rate more than doubles (from 0.15 to 0.32), indicating that the model now finds at least one correct law for about one-third of the cases. Ranking-based metrics, such as MRR and MAP, also improve markedly, suggesting that the model not only identifies more relevant statutes but places them higher in the returned list.

Despite these improvements, the absolute performance remains low and the model continues to miss the vast majority of citations. This underscores the limitations of fine-tuning with a small supervised dataset and highlights that substantially more training data, better citation extraction, and an expanded statute corpus are required for robust legal retrieval. Nonetheless, the improvements demonstrate that the model is responsive to supervision and capable of learning domain-specific legal associations.

Metric	Baseline	Fine-Tuned	Improvement (%)
Recall@5	0.0075	0.0307	309
Precision@5	0.0667	0.2727	309
F1@5	0.0135	0.0553	309
MRR	0.0687	0.2652	286
MAP	0.0687	0.2467	259

Legal-BERT

The baseline Legal-BERT model shows slightly better initial performance than MPNet, but overall, its capabilities in the legal retrieval task remain weak. The Recall@5 of 0.0120 is roughly twice as high as MPNet's, yet still below 2% of all relevant statutes. Precision@5 reaches 0.1061, meaning that only around 10% of the retrieved items are actually correct. The resulting F1@5 score of 0.0215 further illustrates that the model captures only very limited relationships between case text and statutory provisions.

Ranking-based metrics are similarly low: both MRR and MAP score at 0.0876, indicating that correct citations - when they appear at all - tend to be located at lower ranks. The Hit Rate@5, identical to MPNet at 0.1515, reveals that Legal-BERT returns at least one correct citation in only about one out of ten test cases.

After supervised fine-tuning on the constructed case-to-statute dataset, Legal-BERT improves notably. Recall@5 increases to 0.0212 (+77%), Precision@5 to 0.1879 (+77%), and F1@5 to 0.0381 (+77%). The Hit Rate also rises from 0.15 to 0.27, meaning that in roughly every fourth case the model now retrieves at least one relevant statute among the top five results. MRR and MAP improve moderately as well (around +56%).

Compared to MPNet, Legal-BERT benefits less from fine-tuning, suggesting that MPNet's pretrained structure aligns more naturally with the linguistic patterns found in legal texts. Nonetheless, Legal-BERT clearly responds to supervision and improves in both recall and ranking.

Metric	Baseline	Fine-Tuned	Improvement (%)
Recall@5	0.0120	0.0212	77
Precision@5	0.1061	0.1879	77
F1@5	0.0215	0.0381	77
MRR	0.0876	0.1371	56
MAP	0.0876	0.1376	57

Legal-SBERT

Did not managed to evaluate this model in time as the computation took too long.

4.2. Error Evaluation

MPNet

We evaluated our system using the test subset, selecting several representative cases and comparing the model's predicted statutes with the ground-truth citations. The results show that the model fails to retrieve most of the correct legal provisions. However, this does not indicate a fundamentally flawed model architecture or retrieval pipeline. Instead, the errors primarily stem from limitations in data coverage and citation parsing.

First, many of the citations appearing in these test cases were never included in the training data or statute corpus. Because the model has not been exposed to these provisions, it is unable to retrieve them. Second, our citation-extraction pattern is imperfect: some extracted strings are incomplete, malformed, or not valid statute references at all (e.g., "§ 78 Abs. 3 Nrn"), while others refer to contract clauses or insurance conditions (e.g., "§ 14 AVB") that do not exist in statutory law. Since these cannot be mapped to real sections in our database, they are inevitably marked as missing.

Additionally, our current regular-expression patterns are sometimes too rigid and may fail to normalize citations with unusual formatting or embedded commentary references ("Rn", "Teil", "Satz"). Future iterations should separate statute extraction from commentary or contract references, and ideally rely on more advanced NLP-based citation normalization instead of handcrafted patterns.

Overall, the poor evaluation results primarily reflect limitations in training depth, corpus completeness, and citation parsing and not a failure of the underlying semantic retrieval method. With better preprocessing, a more comprehensive statute database, and more supervised training data, the model's performance would be expected to improve substantially.

Excerpt of error evaluation:

```
**Sample Cases with MISSING Citations (Top 5):**
  - Case ID 95: ['MISSING: § 144 Abs. 1 Nr. 1 BauGB', 'MISSING: § 80 Abs. 5 Satz 1 VwGO', 'MISSING: § 80 Abs. 2 Satz 1 Nr. 4 VwG
O', 'MISSING: § 80 Abs. 3 Satz 1 VwGO', 'MISSING: § 80 Abs. 5 Satz 1 VwGO', 'MISSING: § 80 Abs. 2 Satz 1 Nr. 4 VwGO', 'MISSING: §
113 Abs. 1 Satz 1 VwGO', 'MISSING: § 155 Abs. 1 Satz 1 VwGO', 'MISSING: § 52 Abs', 'MISSING: § 53 Abs. 2 Nr. 2 GKG']
    - True Citations: ['§ 144 Abs. 1 Nr. 1 BauGB', '§ 80 Abs. 5 Satz 1 VwGO', '§ 80 Abs. 2 Satz 1 Nr. 4 VwGO', '§ 80 Abs. 3 Satz 1
VwGO', '§ 80 Abs. 5 Satz 1 VwGO', '§ 80 Abs. 2 Satz 1 Nr. 4 VwGO', '§ 113 Abs. 1 Satz 1 VwGO', '§ 155 Abs. 1 Satz 1 VwGO', '§ 52 A
bs', '§ 53 Abs. 2 Nr. 2 GKG'] | Retrieved Citations: ['§ 6 GG', '§ 164b BauGB', '§ 9a BauGB', '§ 355 StGB', '§ 29a AsylG']
  - Case ID 15: ['MISSING: § 124a Abs. 4 Satz 4 und Abs. 5 Satz 2 nur', 'MISSING: § 124 Abs. 2 VwGO', 'MISSING: § 124a Abs. 4 Satz
4 VwGO', 'MISSING: § 124 Abs. 2 Nr. 5 VwGO', 'MISSING: § 124 Abs. 2 Nr. 5 VwGO', 'MISSING: §§ 105 VwGO', 'MISSING: § 124 VwGO', 'M
ISSING: § 105 Rn', 'MISSING: §§ 119, 120 VwGO', 'MISSING: § 124 Abs. 2 Nr. 5 VwGO', 'MISSING: § 124 Rn', 'MISSING: § 124 VwGO', 'M
ISSING: § 105 Rn', 'MISSING: § 124 Abs. 2 Nr. 1 VwGO', 'MISSING: § 124 Abs. 2 Nr. 1 VwGO', 'MISSING: § 154 Abs. 2 VwGO', 'MISSING:
§§ 47 Abs. 1 und', 'MISSING: §§ 68 Abs. 1 Satz 5, 66 Abs. 3 Satz 3 GKG', 'MISSING: § 152 Abs. 1 VwGO', 'MISSING: § 124a Abs. 5 Sat
z 4 VwGO']
    - True Citations: ['§ 124a Abs. 4 Satz 4 und Abs. 5 Satz 2 nur', '§ 124 Abs. 2 VwGO', '§ 124a Abs. 4 Satz 4 VwGO', '§ 124 Abs.
2 Nr. 5 VwGO', '§ 124 Abs. 2 Nr. 5 VwGO', '§§ 105 VwGO', '§ 124 VwGO', '§ 105 Rn', '§§ 119, 120 VwGO', '§ 124 Abs. 2 Nr. 5 VwGO',
'§ 124 Rn', '§ 124 VwGO', '§ 105 Rn', '§ 124 Abs. 2 Nr. 1 VwGO', '§ 124 Abs. 2 Nr. 1 VwGO', '§ 154 Abs. 2 VwGO', '§§ 47 Abs. 1 un
d', '§§ 68 Abs. 1 Satz 5, 66 Abs. 3 Satz 3 GKG', '§ 152 Abs. 1 VwGO', '§ 124a Abs. 5 Satz 4 VwGO'] | Retrieved Citations: ['§ 589
ZPO', '§ 296 ZPO', '§ 771 ZPO', '§ 511 ZPO', '§ 505d BGB']
  - Case ID 30: ['MISSING: § 78 Abs. 3, Abs. 4 Satz 4 AsylG', 'MISSING: § 78 Abs. 3 Nrn', 'MISSING: § 78 Abs. 3 Nr. 1 AsylG', 'MIS
SING: § 78 Abs. 3 Nr. 3 AsylG', 'MISSING: § 138 Nr. 3 VwGO', 'MISSING: § 78 Abs. 3 Nr. 1 AsylG', 'MISSING: § 78 Abs. 3 Nr. 1 Asyl
G', 'MISSING: § 132 Abs. 2 Nr. 1 VwGO', 'MISSING: § 51 Abs. 1 VwVfG', 'MISSING: § 78 Abs. 3 Nr. 3 AsylG', 'MISSING: § 138 Nr. 3 Vw
GO', 'MISSING: § 108 Abs. 2 VwGO', 'MISSING: § 78 Abs. 3 Nr. 3 AsylG', 'MISSING: § 138 Nr. 3 VwGO', 'MISSING: § 154 Abs. 2 VwGO',
'MISSING: § 83b AsylG', 'MISSING: § 80 AsylG']
    - True Citations: ['§ 78 Abs. 3, Abs. 4 Satz 4 AsylG', '§ 78 Abs. 3 Nrn', '§ 78 Abs. 3 Nr. 1 AsylG', '§ 78 Abs. 3 Nr. 3 Asyl
G', '§ 138 Nr. 3 VwGO', '§ 78 Abs. 3 Nr. 1 AsylG', '§ 78 Abs. 3 Nr. 1 AsylG', '§ 132 Abs. 2 Nr. 1 VwGO', '§ 51 Abs. 1 VwVfG', '§ 7
8 Abs. 3 Nr. 3 AsylG', '§ 138 Nr. 3 VwGO', '§ 108 Abs. 2 VwGO', '§ 78 Abs. 3 Nr. 3 AsylG', '§ 138 Nr. 3 VwGO', '§ 154 Abs. 2 VwG
O', '§ 83b AsylG', '§ 80 AsylG'] | Retrieved Citations: ['§ 296 ZPO', '§ 697 ZPO', '§ 331 ZPO', '§ 690 ZPO', '§ 74 AsylG']
```

Our supervised fine-tuning led to a modest but measurable improvement in retrieval quality. While the zero-shot model often returned completely irrelevant statutes, the fine-tuned model more frequently selected provisions from the correct legal domain (e.g., VwGO for administrative cases, AsylG for asylum matters). However, the improvement remains limited: the system still fails to retrieve most of the exact citations contained in the test data. As before this is not primarily due to model failure, but due to structural challenges in the data. Therefore, the observed errors reflect limitations in dataset quality and corpus completeness rather than deficiencies in the retrieval architecture itself.

Legal-BERT

Across all inspected test cases, the baseline Legal-BERT model still fails to retrieve the majority of the true legal citations. Although its retrieved results appear slightly more domain-

appropriate compared to MPNet (e.g., retrieving more VwGO and BauGB sections), the model continues to miss nearly all of the expected citations in the evaluation subset. The underlying issues remain the same: the model has not been trained on enough domain-specific supervision, several true citations do not exist in the scraped statute corpus (e.g., incomplete references like "§ 8b Teil" or commentary references like "Rn") and our citation-matching pattern cannot reliably normalize such noisy references. As a result, although the retrieved statutes look marginally closer to the target domain, no meaningful performance improvement is visible yet, confirming that deeper fine-tuning, broader statute coverage, and improved citation extraction are required.

After supervised training, Legal-BERT demonstrates that the model continues to miss the majority of relevant statutory citations in the test cases. Although the retrieved provisions are sometimes closer to the correct legal domain than in earlier experiments, the model still fails to identify the specific sections referenced in the ground truth. This indicates that, in its un fine-tuned form, Legal-BERT does not yet possess sufficient domain alignment to perform precise statute-level retrieval. The errors are largely attributable to the same structural issues observed previously.

Legal-SBERT

Did not managed to evaluate this model in time as the computation took too long.

5. Conclusion and Group Work

5.1. Interpretation of Results and Conclusions

In this project, we developed a legal retrieval system designed to identify statutory provisions relevant to German court cases. Despite strong our efforts the overall retrieval accuracy remained realtively low across all tested models (MPNet, Legal-BERT, Legal-SBERT).

The experiments demonstrate that the core retrieval architecture works as intended: supervised fine-tuning consistently improves all metrics, and the system becomes more capable of returning domain-appropriate statutes. However, the model still fails to retrieve most ground-truth citations. This outcome is not primarily due to model limitations, but rather structural issues in data quality, incomplete statutory coverage, and noisy or malformed citations extracted from the case texts.

Overall, the project successfully establishes a functioning legal retrieval pipeline and provides clear evidence that the approach is viable but requires substantially more training data, broader legal corpora, and more reliable preprocessing to achieve accurate, real-world performance.

Several improvements would significantly enhance retrieval quality:

- Expand the statutory corpus: Many cited provisions in the test cases do not exist in the scraped corpus. Including all major German law books (VwGO, BauGB, AsylG, VAG, VVG, etc.) is essential.

- Improve citation parsing: The current regex captures noisy or incomplete references. A hybrid, NLP-assisted citation normalizer would reduce errors and map citations to canonical legal identifiers.
- Increase supervised data: The 200 labeled cases are insufficient for reliable learning. A larger and more diverse training set would dramatically improve recall and ranking.
- Address annotation bias: Future evaluations should use a test set that is not part of supervised fine-tuning to avoid optimistic performance estimates.
- Explore larger or domain-adapted models: Models with extended context windows or multilingual training (e.g., Legal-Longformer, GPT-style embeddings) may capture legal semantics more effectively.

5.2. Contribution of each member

All team members contributed equally to the final project, with complementary roles that supported the overall development. Lukas focused on the initial model experiments, which, although ultimately not used, provided important insights that informed the design of the final system. He also implemented the RAG component and assisted with model integration and coding tasks. Michael served as the primary developer, taking responsibility for the implementation of the models, the training pipeline and the evaluation framework. Oskar was responsible for data preparation, including the collection and processing of legal cases and statutory texts. He additionally coordinated the workflow, ensured that all project requirements were met, and authored the written report to balance his lighter coding workload. Lukas also contributed by assisting with parts of the report and reviewing it to ensure accuracy and consistency. Oskar also supported the team by refining documentation, improving markdown formatting, and identifying relevant research papers to guide the selection of suitable models.

6. Sources

Song, K., Tan, X., Qin, T., Lu, J., & Liu, T.-Y. (2020). *MPNet: Masked and permuted pre-training for language understanding*. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N., & Androutsopoulos, I. (2020). *LEGAL-BERT: The Muppets straight out of Law School*. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (pp. 2898–2904).

Chauhan, J. (2023). *Legal-SBERT: Creating a sentence transformer for the legal domain and generating data* (Stanford CS224N Custom Project).

Meta AI. (n.d.). *FAISS: Facebook AI Similarity Search*. https://ai.meta.com/tools/faiss/

Marqo AI. (n.d.). *What is MRR in machine learning?* https://www.marqo.ai/blog/what-is-mrr-in-machine-learning

Evidently AI. (n.d.). *Mean Average Precision (MAP)*. https://www.evidentlyai.com/ranking-metrics/mean-average-precision-map