DL4DS Legal Contract Dataset

Ethan Chang, Heng Chang, Josh Yip

April, 2024

Abstract

This project aims to develop a high-quality, labeled dataset of legal contract clauses to support AI-driven contract analysis. We introduce a hybrid clause segmentation pipeline that merges regex-based heuristics, semantic similarity using sentence embeddings, and supervised boundary classification to accurately detect clause boundaries in unstructured legal text. Building on this foundation, we assign contract-type-specific clause labels and explore intra-contract relationships by analyzing semantic and structural connections between clauses within the same document. Our work enhances the granularity and contextual understanding of contract language, enabling more robust clause classification, summarization, and fairness analysis.

Introduction

Legal contracts are complex documents filled with dense, technical language that can be difficult to analyze and understand without expert knowledge. With the growing demand for AI-assisted legal tools, there is a critical need for structured, high-quality datasets that support tasks like clause segmentation, classification, and summarization. Existing legal NLP datasets, such as CUAD and ContractNLI, provide useful foundations, but they often lack fine-grained clause-level annotations and domain-specific context.

Our project addresses this gap by building a dataset and system capable of accurately identifying and labeling contract clauses. We implement a hybrid clause segmentation pipeline that combines regex-based patterns, semantic similarity modeling, and supervised classification to detect clause boundaries with high precision. After segmentation, we apply contract-type-specific clause categorization and explore intra-contract clause relationships to model how clauses interrelate within the same legal context.

This work contributes a reproducible framework for fine-grained contract analysis, with applications in contract review, compliance checking, and legal summarization. Our code and documentation are available at: https://github.com/joshyipp/542-LegalContract-AI.git.

Related Work

Contract analysis has advanced with new datasets and methods for clause segmentation and labeling. One prominent dataset is the *Contract Understanding Atticus Dataset (CUAD)* curated by the Atticus Project. CUAD v1 contains 510 commercial contracts annotated by lawyers with over 13,000 clause labels spanning 41 clause types [?]. Its task is to identify salient contract passages that matter in legal review. As Hendrycks et al. note, CUAD's expert-annotated corpus "consists of over 13,000 annotations" and enables benchmarking of models on legal clause identification.

Stanford's *ContractNLI* dataset offers a different angle, framing contract analysis as document-level natural language inference [?]. ContractNLI annotates 607 non-disclosure agreements (NDAs) with entailment, contradiction, or neutral labels for 17 fixed hypotheses. It is described as "the first dataset to utilize NLI for contracts" and, as of 2021, the largest annotated contract corpus. Both CUAD and ContractNLI provide valuable benchmarks, but they differ in scope: CUAD's clause categories apply broadly to deal contracts, whereas ContractNLI focuses narrowly on NDAs and inference. Other specialized corpora exist (e.g., contracts labeled for rights and obligations), but none has combined clause-level segmentation with contract-type-specific categorization to the extent of our project.

Before the era of large datasets, rule-based approaches often drove clause segmentation. Practical tools—such as e-discovery software like Relativity—exploit document structure: for example, Relativity's contracts module splits a document into sections by detecting section headings and outline cues [?]. Likewise, custom pipelines frequently apply regular expressions to capture known markers such as "Table of Contents" or numbered section titles. For example, in one insurance-contract processing system, regex is first used to locate and clean the table of contents, and then section titles are matched in the contract body to create paragraph-level segments [?]. As shown in that study, this approach yields distinct "paragraph 1, paragraph 2. . . " units.

These methods successfully segment contracts into logical units, but they rely on rigid patterns (e.g., headings, keywords, punctuation) and may fail when contracts exhibit non-standard formatting. In research literature, similar ideas appear: Chalkidis and Androutsopoulos [?] essentially carved contracts into fixed "extraction zones" for each clause type. In short, prior work often uses regex or layout rules to hypothesize clause boundaries before any learning occurs.

More recent work has explored machine learning and semantic methods for clause extraction. Many approaches assume segmented text and focus on classification. For example, transformers fine-tuned on CUAD can predict clause labels, though Hendrycks et al. note that model performance remains far from perfect. Xu et al.'s *ConReader* (2022) explicitly goes beyond local context by modeling intra-contract relations [?]. They identify three key clause-level links in contracts—long-range context links, term–definition pairs, and similarity between same-type clauses—and incorporate them to improve clause-level extraction. Similarly, Borchmann et al. [?] defined a "contract discovery" task where a model retrieves clauses similar to given examples; their findings showed that off-the-shelf encoders struggle with few-shot clause retrieval.

In all these cases, however, the input clause spans are often assumed to be known or pre-segmented. Few prior works simultaneously address both identifying clause boundaries and labeling them with fine-grained, contract-specific categories.

Compared to these existing works, our approach integrates and extends multiple paradigms. Like rule-based systems, we use regex and structural signals to propose initial clause boundaries. Unlike purely hand-crafted methods, we also apply semantic features (e.g., contextual embeddings) and supervised learning to refine boundaries and classify clauses. In contrast to CUAD or ContractNLI's one-size-fits-all label sets, our clause labels are tailored to specific contract types (e.g., NDA vs. service agreement). We also explicitly model relationships between clauses within the same contract—for example, linking a definition clause to the term it defines—drawing inspiration from ConReader's term–definition and same-type clause relations.

In sum, while prior datasets and systems provide critical building blocks (e.g., clause corpora, regex heuristics, transformer models), our work uniquely merges regex, semantic, and supervised techniques into a unified pipeline that segments contracts, assigns contract-specific clause labels, and models intra-document clause structure.

Methodology

LLM-Assisted Legal Contract Classification with LegalBERT Fine-Tuning

To enable scalable classification of legal contracts by type, we developed an LLM-assisted workflow that leverages GPT-4 for high-quality labeling and fine-tunes a lightweight model, LegalBERT, for efficient downstream classification.

Although large language models such as GPT-4o have demonstrated strong performance in understanding and classifying complex legal texts, their high inference cost presents practical barriers to scalable deployment. At the same time, training smaller, more efficient models like BERT requires large labeled datasets, which are difficult and expensive to obtain in the legal domain. To resolve this tension, we adopt an LLM-assisted labeling approach: although using LLMs for inference is costly, we use them once to label a high-quality training set, enabling the downstream training of a lighter predictive model suitable for deployment.

We first constructed a taxonomy of common legal contract types, such as Sale of Goods, Real Estate Contracts, Employment Contracts, and Licensing Agreements. Each category

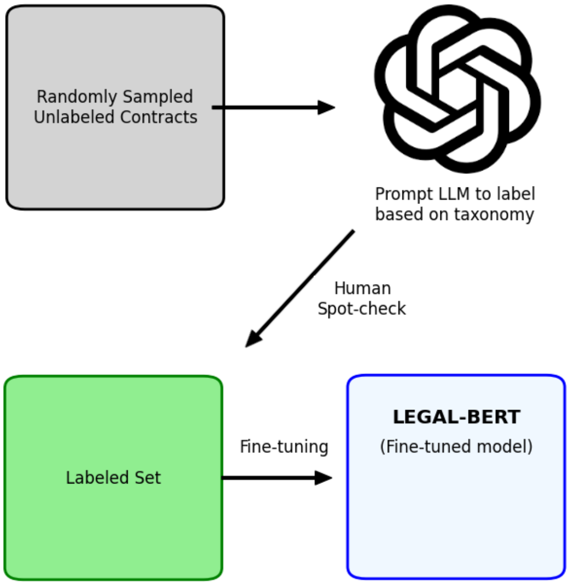
Randomly Sampled Unlabeled Contracts
Prompt LLM to label based on taxonomy
Human Spot-check
Labeled Set
Fine-tuning
LEGAL-BERT
(Fine-tuned model)

Figure 1: Overview of the LLM-assisted labeling and LegalBERT training pipeline.

included clear inclusion and exclusion criteria. From a large pool of unlabeled contracts, we randomly sampled a subset (we used 5000 samples) and used GPT-4 to label them according to the taxonomy. The LLM output included predicted labels, confidence scores, and rationales. A sample of these outputs was manually reviewed to ensure accuracy.

The resulting labeled dataset was then used to fine-tune a predictive model. Prior to supervised training, we applied domain-adaptive pretraining (DAPT) by continuing masked language modeling (MLM) on the contract corpus using LegalBERT. This pretraining step used the standard cross-entropy loss over randomly masked tokens with a masking probability of 15%. For the downstream classification task, we fine-tuned the model using weighted categorical cross-entropy, where class weights were assigned based on label frequencies to account for class imbalance, and used gradient accumulation to accommodate long sequences.

This effectively addresses the lack of labeled datasets in the legal domain by leveraging LLMs for scalable, high-quality annotation and enabling downstream training of specialized, efficient models.

Table 1: Classification report on 300 testing samples

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Agency Agreements	0.3333	0.3333	0.3333	3
Confidentiality (NDA) Agreements	0.0000	0.0000	0.0000	1
Construction Contracts	1.0000	1.0000	1.0000	4
Employment Contracts	0.9429	0.7857	0.8571	42
Franchise Agreements	1.0000	1.0000	1.0000	1
Guarantee Contracts	0.8462	0.9167	0.8800	12
Indemnity Contracts	1.0000	0.8462	0.9167	13
Insurance Contracts	1.0000	0.8462	0.9167	13
Lease Agreements	0.8095	1.0000	0.8947	17
Licensing Agreements	0.6981	0.8605	0.7738	43
Loan Agreements	0.8462	0.9167	0.8800	24
Partnership Agreements	0.8500	0.7727	0.8095	22
Real Estate Contracts	1.0000	0.7333	0.8462	15
Sale of Goods	0.8649	0.9412	0.9014	34
Service Contracts	0.7027	0.7027	0.7027	37
Settlement Agreements	0.9091	0.8000	0.8511	25
Accuracy			**0.8133**	300
Macro avg	0.8002	0.7881	0.7902	300
Weighted avg	0.8189	0.8133	0.8113	300

Clause Segmentation using Regex + Semantic + Supervised Bounds Detection

To extract meaningful clause units from raw contract text, we designed a hybrid segmentation pipeline that combines rule-based, semantic, and supervised methods for clause boundary detection. This multi-pronged approach addresses the limitations of any single technique and is tailored to the variability of real-world contract formatting.

Regex-Based Boundary Detection

Our first boundary detection step uses a suite of regular expressions to identify structural cues within contract text. These include:

- Numbered section headers (e.g., `1. Definitions`, `2.1 Termination`)
- Uppercase headers (e.g., `CONFIDENTIALITY`, `TERMINATION`)
- Legalistic keywords such as `WHEREAS` or `NOW, THEREFORE`

These patterns are matched line-by-line to mark likely clause start points. Although effective for well-structured documents, regex alone is brittle when formatting is inconsistent or headings are missing.

Semantic Boundary Detection

To detect shifts in meaning beyond surface patterns, we apply semantic similarity using sentence embeddings. Specifically, we use the `all-MiniLM-L6-v2` model from `SentenceTransformers` to encode each sentence in the contract into a 384-dimensional embedding. We then compute cosine similarity between adjacent sentences. A significant drop in similarity (below a threshold of 0.45) signals a potential boundary:

$$\text{similarity}(s_i, s_{i+1}) = \frac{s_i \cdot s_{i+1}}{\|s_i\|\|s_{i+1}\|} \tag{1}$$

Sentences following sharp semantic shifts are marked as new clause beginnings. This method complements regex by identifying topic transitions not reflected in formatting.

Supervised Boundary Classification

To further refine clause segmentation, we train a supervised classifier using a labeled dataset of clause context windows. Each instance consists of a text snippet and a binary label indicating whether a clause boundary follows. We use a pipeline with TF-IDF vectorization and logistic regression, achieving strong performance on cross-validation. This classifier captures linguistic signals missed by regex or embeddings alone, such as transitions marked by subtle discourse cues.

Boundary Merging and Clause Construction

We combine boundary signals from all three sources—regex, semantic, and supervised—by taking the union of predicted boundary indices. The contract is then segmented into chunks based on these merged boundaries. Each resulting clause is stored with metadata including:

- Clause text
- Detected heading (if present)
- Clause number (e.g., `2.1`)
- Sentence count and position

Subclauses (e.g., `(a)`, `(b)`) are heuristically merged with their parent clauses to preserve continuity. We also apply post-filtering to discard noise (e.g., page numbers, SEC URLs) using hand-crafted rules.

Outcome

This hybrid segmentation method enables us to extract high-quality clause units from contracts of varying format and complexity. These clause units form the basis for downstream labeling, summarization, and inter-clause relationship modeling in our broader pipeline.

Datasets

Describe the datasets that you used to train and evaluate your models. Or if you are doing a dataset project, describe the dataset you created. If it is a theoretical or algorithmic project, describe any datasets that your theory or algorithm may be applicable to.

Dataset text here.

Evaluation Results

Describe your evaluation results. What metrics will did you use? What baseline from an existing solutions can you compare to?

Evalutation text here.

Conclusion

Summarize your project, results and contributions. Describe any future work that you or someone else would do if they continued the project.

Conclusion text here.

References

[1] Author1. *Title1*. Publisher, Year.

[2] Author2. *Title2*. Publisher, Year.