Project Atticus Graph: LLM-Based Knowledge Graph Extraction from Legal Contracts

Aditya Gollamudi
Department of Computer Science and Engineering
Texas A&M University
College Station, TX 77843
adigo@tamu.edu

Abstract

Manual review of legal contracts is time consuming, error prone, and dosent capture the intricate relationships between parties, clauses, and obligations. This paper presents a system that uses Large Language Models (LLMs) to automatically extract structured knowledge graphs from legal contracts. We compare LLM based extraction against traditional regex based NLP methods on the CUAD (Contract Understanding Atticus Dataset). To narrow down the scope for research purposes, we will be focusing on 9 Intellectual Property (IP) clause types across 17 dedicated IP agreement contracts. Our results show that the LLM based approach gets a Macro F1 of 0.734 compared to 0.392 for traditional methods. This is a 87% relative improvement. The LLM achieves 91% precision and 85% recall and its winning on 7 of 9 clause types. So while traditional methods cannot extract relationships well, our LLM approach successfully extracts 90 IP relationships like who grants/assigns IP to whom, enabling graph based reasoning and queries in Neo4j. This could have huge impacts for Graph Rag systems in law.

1 Introduction

Legal contracts are super common in business, however their analysis is still mainly manual. Therefore, contract review has a lot of challenges:

- **Volume**: Organizations manage thousands of contracts with complex interdependencies
- **Complexity**: Legal language have super nuanced phrasing that varies across jurisdictions and industries
- **Evolution**: As new contract types and clauses happen, manual rule systems need constant updates
- **Relationships**: Contracts contain rich relationships between parties that are difficult to query systematically

Traditional Natural Language Processing approaches like using regular expressions and keyword matching struggle with the variety in legal language. For example a clause granting a "perpetual, irrevocable license" could be paraphrased as "license that survives termination" or "permanent rights". Regex based systems could easily miss these patterns. And requires us to manually code these cases to check for these different cases.

Research Question Can Large Language Models outperform traditional NLP methods in extracting structured knowledge from legal contracts?

Contributions This paper makes the following contributions:

1. A **knowledge graph extraction pipeline** that transforms unstructured legal text into queryable graph representations
2. **Comparing and evaluating** LLM based vs. simple traditional nlp extraction on the CUAD dataset
3. **Relationship extraction** capabilities that enable graph-based reasoning (WHO $\rightarrow$ WHAT $\rightarrow$ WHOM)
4. **Evidence** that LLMs achieve 87% relative improvement in Macro F1 over regex patterns

2 Related Work

Legal NLP The legal domain has seen increasing NLP applications, from document classification to named entity recognition. The Contract Understanding Atticus Dataset (CUAD) (Hendrycks et al., 2021) is a significant contribution. It has 510 commercial contracts with over 13,000 expert

annotations across 41 clause types. This dataset help in evaluation of contract analysis methods. Prior work on legal NER has focused on entities like party names and dates, but clause level extraction remains challenging because of the semantic complexity of legal language.

Knowledge Graphs Knowledge graph construction from unstructured text has been explored already (Ji et al., 2022), with applications in biomedical, scientific, and general domains. However, legal domain applications remain limited. Most of the previous papers focuses on case law and judicial decisions rather than commercial contracts. The challenge in the legal domain is capturing not just entities but the complex relationships between parties, obligations, and rights that ate hidden in contractual language.

LLMs for Information Extraction Large Language Models have shown good capability for structured information extraction (Wei et al., 2023). Zero shot and few shot prompts can enable extraction without task specific data. Recent papers evaluated LLMs against human lawyers for contract review tasks (Martin et al., 2024). However, legal contract analysis still have some challenges. Main ones are: domain specific terminology that differs from general language, complex nested clause structures, cross references between sections, and the need to understand relationships that aren't explicitly mentioned. Our work addresses these challenges through carefully designed prompts that encode legal domain knowledge.

Legal Language Models Domain specific language models have been promising for legal NLP tasks. LEGAL BERT (Chalkidis et al., 2020) demonstrated that pre training on legal corpora improves performance on legal NER and clause classification. The LexGLUE benchmark (Chalkidis et al., 2022) established standardized evaluation across multiple legal NLP tasks, where as ContractNLI (Koreeda and Manning, 2021) focused specifically on document level inference for contracts. The LEDGAR corpus (Tuggener et al., 2020) provides contract provision classification from SEC filings at scale. Earlier work on contract element extraction (Chalkidis et al., 2017) also setup important methods for the field.

Clause Type	Supp.	Description
IP Ownership Assignment	16	Transfer of IP rights
License Grant	14	Permission to use IP
Affiliate License (Licensee)	12	Affiliates may use
Non-Transferable License	11	Cannot assign license
Irrevocable/Perpetual License	10	Cannot be revoked
Affiliate License (Licensor)	7	Affiliates may license
Joint IP Ownership	1	Shared IP ownership
Unlimited/All-You-Can-Eat	1	No usage limits
Source Code Escrow	0	Code with escrow agent

Table 1: IP Clause Types and the number of contracts containing each clause type.

3 Dataset

We use the **Contract Understanding Atticus Dataset (CUAD)** (Hendrycks et al., 2021), containing 510 commercial legal contracts with over 13,000 expert annotations across 41 clause types.

Focus: IP Clauses We focus on 9 Intellectual Property clause types (Table 1) due to their high business impact in technology and licensing agreements, clear binary presence absence determination, and well defined annotations in CUAD.

Test Set We evaluate on 17 dedicated IP Agreement contracts from CUAD, This focused test set ensures consistent evaluation across both methods.

4 Methods

Our pipeline consists of two extraction approaches evaluated against CUAD ground truth annotations.

4.1 LLM-Based Extraction

We use **GPT-OSS-120b** which is an open weight model by OpenAI. It is accessed through the OpenRouter API with structured prompting. This model provides a good balance between capability and cost, enabling processing of lengthy legal documents while maintaining extraction quality.

Prompt Design The prompt follows a structured template with four key techniques:

1. **Role specification**: The model is told to act as an "expert legal contract analyzer specializing in IP agreements"

2. **Clause definitions**: Each of the 9 IP clause types is defined with 3 to 5 example phrases that might indicate presence. For example: "grants a license", "hereby licenses", "permission to use"

3. **Output schema**: A JSON schema that forces structured output with required fields for each clause type

4. **Relationship extraction**: Instructions specifically ask for extraction of WHO-WHAT-WHOM triples.

Output Format The LLM returns structured a JSON containing the following: parties with names, roles of licenser or licensee, and entity types. And also IP clauses with boolean existence flags, evidence quotes from the text, and confidence scores between 0 and 1. I also has the IP relationships as subject-predicate-object triples with related IP type.

Error Handling We implement retry logic with exponential backoff for API failures and JSON parsing errors. So that bad responses trigger prompting again with explicit format reminders.

4.2 Traditional NLP Extraction

For the baseline we use regex pattern matching along with spaCy NER for entity extraction. We developed hand crafted patterns for each clause type based on common legal phrasing.

Pattern Examples For each clause type, we crafted a couple regex patterns. For example:

- **License Grant**: `grants?\s+.*license`, `hereby\s+licenses`
- **IP Assignment**: `assigns?\s+.*intellectual\s+property`, `transfer.*ownership`
- **Irrevocable**: `irrevocable`, `perpetual.*license`, `survives?.*termination`

Limitations Traditional methods have a lot of weaknesses:

1. **Paraphrase blindness**: Cannot recognize semantically equivalent expressions like "permanent rights" vs "perpetual license"

2. **False positives**: Partial pattern matches in unrelated contexts for example "unlimited liability" matching with "Unlimited License"

3. **No relationship extraction**: cannot identify WHO grants WHAT to WHOM

4. **Maintenance trouble**: Each new phrasing variant needs manual pattern updates

4.3 Knowledge Graph Schema

We define a schema optimized for legal contract analysis with three entity types and five relationship types.

Entity Types

- **Contract**: Properties include name, contract_type, effective_date, and file_path
- **Party**: Properties include name, role (licensor/licensee/assignor/assignee), and entity_type (corporation/individual/LLC)
- **Clause**: Properties include clause_type, exists (boolean), evidence_text, and confidence_score

Relationship Types

- `HAS_PARTY`: Links Contract to Party nodes
- `HAS_CLAUSE`: Links Contract to Clause nodes
- `GRANTS_LICENSE_TO`: IP licensing relationships
- `ASSIGNS_IP_TO`: IP ownership transfer
- `SHARES_IP_WITH`: Joint IP ownership

Graph Database Extracted knowledge is stored in Neo4j. This is a native graph database. This enables Cypher queries for complex traversals, such as finding all downstream licensees of a particular IP holder or identifying contracts with similar clause structures.

5 Evaluation Methodology

Task Binary classification. We need to check: does clause type c exist in contract d?

Ground Truth CUAD provides Yes or No annotations for each clause type per contract, created by legal experts.

Metrics We compute Precision, Recall, and F1:

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{1}$$

We report Macro F1 this is the average F1 across clause types, treating all clauses equally and also the Micro F1. This is the F1 computed on total TP/FP/FN, weighted by frequency.

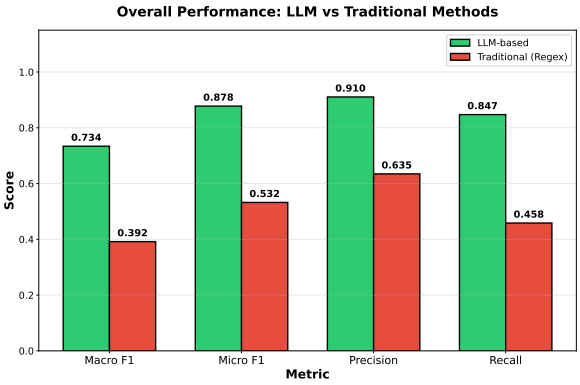

Figure 1: Performance comparison between LLM and Traditional methods across all metrics.

Method	Macro F1	Micro F1	Prec.	Rec.
LLM	**0.734**	**0.878**	**0.910**	**0.847**
Traditional	0.392	0.532	0.635	0.458
Improvement	*+87%*	*+65%*	*+43%*	*+85%*

Table 2: Performance Comparison of the 17 IP Agreement contracts and 9 clause types

6 Results

6.1 Overall Performance

Table 2 presents the results comparing LLM based and traditional extraction methods. And Figure 1 shows the performance gap between methods.

Key Findings

- The LLM achieves 1.87× better Macro F1
- The LLM wins on 7/9 clause types which is 77.8%
- we get upto 91% precision which means Highly accurate predictions
- We also gey 85% recall so it finds most existing clauses

6.2 Per-Clause Performance

Table 3 shows F1 scores by clause type. Figure 2 provides a visual comparison across all clause types.

Analysis

Affiliate License clauses: LLM achieves nearb perfect F1 while traditional methods completely fail. This demonstrates LLM's ability to understand semantic variations. License Grant: Both methods perform well, but LLM's shows better handling of paraphrased language. Joint IP & Source Code Escrow: Both methods fail due to extremely low support (1 and 0 samples).

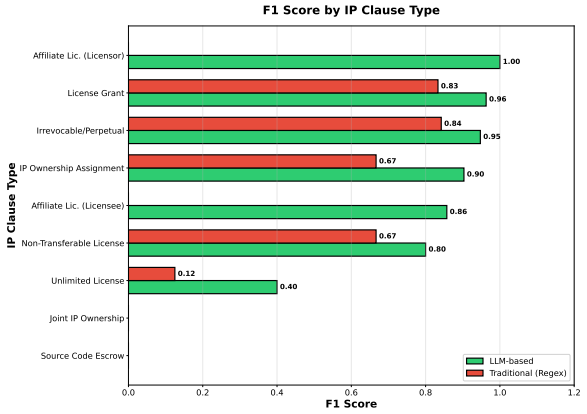

Figure 2: This shows per clause F1 score comparison. LLM significantly outperforms traditional methods on Affiliate License clauses.

Clause Type	LLM F1	Trad. F1	Δ
Affiliate Licenser	1.000	0.000	+1.000
Affiliate Licensee	0.857	0.000	+0.857
License Grant	0.963	0.833	+0.130
IP Ownership Assignment	0.903	0.667	+0.236
Irrevocable/Perpetual	0.947	0.842	+0.105
Non-Transferable License	0.800	0.667	+0.133
Unlimited/All-You-Can-Eat	0.400	0.125	+0.275
Joint IP Ownership	0.000	0.000	0.000
Source Code Escrow	0.000	0.000	0.000

Table 3: F1 Score by IP Clause Type. LLM wins on 7/9 clause types.

6.3 Relationship Extraction

A key difference is relationship extraction. Traditional regex methods cannot extract relationships at all. The LLM extracted 90 IP relationships across 17 contracts. Table 4 shows examples.

Relationship Analysis The 90 extracted relationships break down as: 72 GRANTS_LICENSE_TO, 15 ASSIGNS_IP_TO, and 3 SHARES_IP_WITH. These relationships enable powerful queries:

- **Graph traversal**: "Find all companies licensed by Honeywell"
- **Network analysis**: Identify licensing hubs and IP flow patterns
- **Contract reasoning**: Find similar licensing structures across contracts

7 Result Analysis

7.1 Why LLMs Outperform Traditional Methods

Semantic Understanding LLMs grasp meaning beyond surface patterns. When a contract

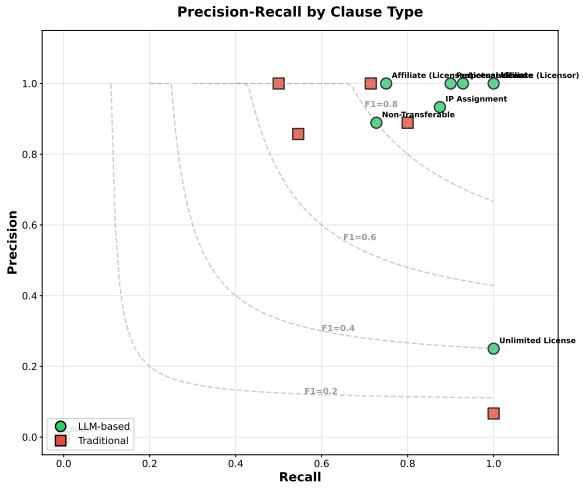

Figure 3: Precision-Recall comparison. LLM achieves both higher precision (91%) and recall (85%).

Licensor	Relation	Licensee
Nuance Comm.	GRANTS_LICENSE	Cerence Inc.
Honeywell Int'l	GRANTS_LICENSE	Garrett Motion
UTC	GRANTS_LICENSE	Otis Elevator
AWI Licensing	GRANTS_LICENSE	AHF Holding
B&W Company	GRANTS_LICENSE	B&W Enterprises

Table 4: Sample IP Relationships (5 of 90 extracted). Traditional methods extracted 0 relationships.

states "the license shall survive any termination," the LLM correctly identifies this as "Irrevocable/Perpetual," while regex patterns miss it.

Context Awareness LLMs consider surrounding context. The phrase "may not assign" might indicate Non Transferable in a license section but something different in an assignment section.

Generalization LLMs can handle paraphrased language without explicit patterns. Traditional methods require new rules for each variation.

7.2 Error Analysis

LLM Errors False Positives on Unlimited License: LLM over detected "unlimited" in 3 contracts where the term referred to geographical scope, not usage quantity. Missed Joint IP: The single positive sample used unusual phrasing ("co developed intellectual property").

Traditional Method Errors Complete failure on Affiliate clauses: Patterns searched for "licensor's affiliate" but contracts used "affiliates of Party Name". High false positive rate on Unlimited: Matched unlimited" in unrelated contexts for example "unlimited liability".

8 Discussion

Relationship Extraction Value The ability to extract WHO → WHAT → WHOM relationships transforms contract analysis. Traditional approaches answer binary questions like Does X contract grant a license?, while our approach allows us to use structured queries like Who grants which IP types to whom, under what terms, and with what restrictions?. This enables use cases like:

- **Due diligence**: Quickly map all IP rights in an acquisition target's contract portfolio
- **Compliance monitoring**: Track license chains and identify potential violations
- **Portfolio analytics**: Identify patterns in licensing structures across different industries

Practical Implications Our results say that LLMs can automate significant portions of contract review:

- **Cost reduction**: At $0.01-0.05 per contract this LLM analysis is orders of magnitude cheaper than manual review ($100-500/hour for legal professionals)
- **Speed**: Processing a contract takes seconds versus the hours for manual review
- **Consistency**: LLMs apply the same criteria uniformly and reduce reviewer variance
- **Scalability**: Can process thousands of contracts for M&A due diligence

When Traditional Methods Suffice Despite LLM doing good, traditional methods may be better when: privacy constraints prohibit sending contracts to external APIs, extremely high precision is needed and manual verification is feasible, or the clause patterns are standardized within an organization.

Limitations

1. **Test Set Size**: 17 IP contracts may not represent full diversity of legal language
2. **Cost**: LLM API calls are expensive at scale ($0.01–0.05 per contract)
3. **Rare Clauses**: Insufficient samples for Joint IP, Source Code Escrow

4. **JSON Parsing**: Occasional LLM format errors (handled with retry logic)
5. **Reproducibility**: LLM outputs may vary slightly between runs
6. **Evaluation**: Data set didn't allow for the construction of a proper gold standard graph to actually evaluate the graph better

9 Future Work

1. **Scale to full CUAD**: Evaluate on all 510 contracts and 41 clause types
2. **Fine-tuned models**: Train smaller models (Llama 3, Mistral) for cost efficiency
3. **Exact span extraction**: Extract character offsets for clause evidence
4. **Cross-contract reasoning**: Find similar clauses across contracts
5. **EDC Framework**: Implement Extract-Define-Canonicalize for better entity normalization

10 Conclusion

This paper presents Project Atticus Graph, a system for extracting structured knowledge graphs from legal contracts using Large Language Models. Our comprehensive evaluation on 17 IP Agreement contracts from the CUAD dataset demonstrates great improvements over traditional regex based methods:

- 1.87× better Macro F1 (0.734 vs. 0.392) this is a 87% relative improvement
- 91% precision, 85% recall. high accuracy with good coverage
- 7/9 clause types won by LLM. consistent improvement across categories
- 90 relationships extracted (vs. 0 for traditional). Allowing graph based reasoning

The key differentiator is not just higher accuracy on clause detection, but the fundamental capability to extract structured relationships (WHO grants/assigns IP to WHOM). This transforms contract analysis from isolated document review to connected knowledge graph reasoning that can be paired with a graph rag systems opening new possibilities for legal analytics, due diligence automation, and compliance monitoring.

Acknowledgments

This work was completed as part of CSCE 489 - Special Topics in AI at Texas A&M University.

References

Ilias Chalkidis, Ion Androutsopoulos, and Achilleas Michos. 2017. Extracting contract elements. In *Proceedings of the 16th International Conference on Artificial Intelligence and Law (ICAIL)*, pages 19–28.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904.

Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Katz, and Nikolaos Aletras. 2022. LexGLUE: A benchmark dataset for legal language understanding in English. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4310–4330.

Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. 2021. CUAD: An expert-annotated NLP dataset for legal contract review. In *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514.

Yuta Koreeda and Christopher D. Manning. 2021. ContractNLI: A dataset for document-level natural language inference for contracts. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1907–1919.

Lauren Martin, Nick Whitehouse, Stephanie Yiu, Lizzie Catterson, and Rivindu Perera. 2024. Better call GPT, comparing large language models against lawyers. *arXiv preprint arXiv:2401.16212*.

Don Tuggener, Pius von Daniken, Thomas Peber, and Mark Cieliebak. 2020. LEDGAR: A large-scale multi-label corpus for text classification of legal provisions in contracts. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)*, pages 1235–1241.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, Yong Jiang, and Wenjuan Han. 2023. ChatIE: Zero-shot information extraction via chatting with ChatGPT. *arXiv preprint arXiv:2302.10205*.

A Detailed Results

Confusion Matrix Summary Across all 153 predictions (17 contracts $\times$ 9 clause types), the LLM achieved: 61 True Positives, 6 False Positives, 11 False Negatives, and 75 True Negatives. The traditional method achieved: 33 True Positives, 19 False Positives, 39 False Negatives, and 62 True Negatives.

Clause Type	Prec.	Rec.	F1	Supp.
IP Ownership Assignment	0.933	0.875	0.903	16
License Grant	1.000	0.929	0.963	14
Affiliate License (Licensee)	1.000	0.750	0.857	12
Non-Transferable License	0.889	0.727	0.800	11
Irrevocable/Perpetual License	1.000	0.900	0.947	10
Affiliate License (Licensor)	1.000	1.000	1.000	7
Joint IP Ownership	0.000	0.000	0.000	1
Unlimited/All-You-Can-Eat	0.250	1.000	0.400	1
Source Code Escrow	0.000	0.000	0.000	0

Table 5: LLM Detailed Results by Clause Type. High support clauses achieve consistently high F1 scores.

B Sample Prompt Structure

The LLM prompt follows this structure (abbreviated):

```
You are an expert legal contract
analyzer specializing in IP agreements.

Analyze this contract and extract:
1. All parties (name, role, type)
2. IP clause presence (9 types)
3. IP relationships (WHO-WHAT-WHOM)

IP Clause Definitions:
- License_Grant: "grants a license",
  "hereby licenses", "permission to use"
- IP_Ownership_Assignment: "assigns all
  right", "transfer of ownership"
[... other clause definitions ...]

Output JSON format:
{
  "parties": [...],
  "ip_clauses": {...},
  "ip_relationships": [...]
}

CONTRACT TEXT:
[Full contract text here]
```

C Neo4j Cypher Query Examples

Sample Neo4j queries enabled by the knowledge graph:

```
// This finds all licensees of a company
MATCH (licensor:Party)-[:GRANTS_LICENSE_TO]
      ->(licensee:Party)
WHERE licensor.name = "Honeywell"
RETURN licensee.name

// This find contracts with IP assignments
MATCH (c:Contract)-[:HAS_CLAUSE]->(cl:Clause)
WHERE cl.type = "IP_Ownership_Assignment"
  AND cl.exists = true
RETURN c.name, cl.evidence_text
```