# LeGen: Complex Information Extraction from Legal sentences using Generative Models

**Chaitra C R[1], Sankalp Kulkarni[2], Sai Rama Akash Varma Sagi[2], Shashank Pandey[2], Rohit Yalavarthy[1], Dipanjan Chakraborty[1], Prajna Upadhyay[1]**
[1]BITS Pilani, Hyderabad
[2]Work done when affiliated to BITS Pilani Hyderabad
p20210024@hyderabad.bits-pilani.ac.in, sankalpkulkarni@gmail.com, ss24bc@fsu.edu
s.pandey@nyu.edu, {f20212294,dipanjan,prajna.u}@hyderabad.bits-pilani.ac.in

## Abstract

Constructing legal knowledge graphs from unstructured legal texts is a complex challenge due to the intricate nature of legal language. While open information extraction (OIE) techniques can convert text into triples of the form ⟨`subject, relation, object`⟩, they often fall short of capturing the nuanced relationships within lengthy legal sentences, necessitating more sophisticated approaches known as complex information extraction. This paper proposes $LeGen$ – an end-to-end approach leveraging pre-trained large language models (GPT-4o, T5, BART) to perform complex information extraction from legal sentences. $LeGen$ learns and represents the discourse structure of legal sentences, capturing both their complexity and semantics. It minimizes error propagation typical in multi-step pipelines and achieves up to a 32.2% gain on the Indian Legal benchmark. Additionally, it demonstrates competitive performance on open information extraction benchmarks. A promising application of the resulting legal knowledge graphs is in developing question-answering systems for government schemes, tailored to the Next Billion Users who struggle with the complexity of legal language. Our code and data are available at https://github.com/prajnaupadhyay/LegalIE.

## 1 Introduction

The Next Billion Users, new adopters of digital technology, struggle to utilize digital devices effectively for accessing critical information such as rights, employment opportunities, health, and education (Google, 2023). This is partly due to the predominantly textual nature of available information, particularly in legal contexts, characterized by intricate and lengthy sentence structures (Abdallah et al., 2023). Processing and acting upon such information impose significant cognitive burdens on these users, who often lack the necessary education and skills to comprehend it (Joshi, 2013).

| Sentence | Clauses | Relations | Relations among Clauses |
|---|---|---|---|
| If balance amount in the account of a deceased is higher than 150,000 then the nominee or legal heir has to prove the identity to claim the amount | 1) Balance amount in the account of a deceased is higher than 150,000 then<br>2) The nominee has to prove the identity to claim the amount<br>3) Legal heir has to prove the identity to claim the amount | CONDITION, DISJUNCTION | $R_{CONDITION}$(Balance amount in the account of a deceased is higher than 150,000 then, $R_{DISJUNCTION}$(The nominee has to prove the identity to claim the amount, Legal heir has to prove the identity to claim the amount)) |

Table 1: Examples of clauses and relations `CAUSE`, `CONDITION`, `CONTRAST`, and `DISJUNCTION` among clauses

NLP techniques can assist in structuring and organizing legal data to enable automatic search and retrieval (Dale, 2019; Zhong et al., 2020). Open information extraction (OIE) techniques (Kolluru et al., 2020; Stanovsky et al., 2018; Etzioni et al., 2011) can be used to extract structured information such as triples of the form ⟨`subject, relation, object`⟩ from a sentence in a domain-independent manner. However, legal text poses unique challenges - Legal sentences and documents are lengthy with complex inter-clausal relationships between them (Chalkidis et al., 2020). Existing OIE techniques are not equipped to return the best results on legal sentences. For instance, the output of OpenIE6 (Kolluru et al., 2020) on *If over 50 percent of a company's workers take concerted casual leave, it will be treated as a strike* are two triples - $i$) ⟨`it, will be treated, as a strike`⟩, $ii$) ⟨`over 50 percent of a company's workers, take concerted, casual leave`⟩. The model cannot identify complex relationships between the two extractions, such as `condition`. Apart from `condition`, clauses can have relations such as `contrast` or `disjunction`, etc (Table 1) among them. Identifying such relations is important to design systems that empower users to interpret complex legal information.

The problem of extracting structure beyond triples is handled by a relatively new area of re-

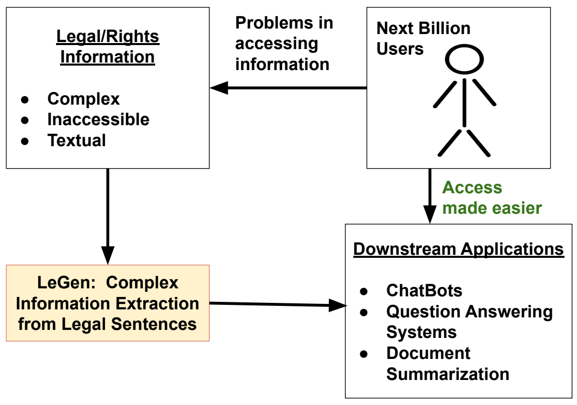


Figure 1: Next Billion Users often face challenges accessing legal text as it is complex and textual. LeGen can help these users understand the legal text better through downstream applications.

search known as complex information extraction (Mahouachi and Suchanek, 2020). Complex information extraction from legal sentences can support many downstream tasks, such as the automatic curation of legal knowledge bases (Correia et al., 2022) and analysis of court proceedings (Zadgaonkar and Agrawal, 2021). Existing techniques for complex information extraction (Niklaus et al., 2019; Prasojo et al., 2018) involve multiple-step pipelines for identifying clauses and relationships from sentences that propagate errors. They also lack language understanding and generalization skills.

This paper proposes $LeGen$, an end-to-end generative approach for complex information extraction from legal sentences (Figure 1). Generative architectures, such as T5 (Raffel et al., 2020), BART (Lewis et al., 2019), or GPT (Radford et al., 2018) have been very successful in understanding text and generalization. These architectures capture both the structure and semantics of a complex sentence more accurately. Such end-to-end modelling reduces the propagation of errors across multiple steps. In this work, we demonstrate how the discourse tree structure (Niklaus et al., 2019) (Section 3.1) of a legal sentence can be learnt using large language models such as BART, T5 and GPT. Our salient contributions are:

**1.** We propose $LeGen$, an end-to-end generative approach that learns accurate tree-based representations to encode the complex structure of any legal statement.

**2.** We report substantial gain over Graphene (Niklaus et al., 2019), a state-of-the-art complex information extraction technique on the Indian Legal benchmark.

**3.** We release the discourse tree structures for legal text curated from Indian Law statements.

**4.** We show $LeGen$'s flexibility by training it as a coordinate boundary detection task and conclude that it is competitive (Kolluru et al., 2020).

**5.** We propose new metrics for measuring the quality of discourse trees.

Our paper is organized as follows. We formally describe the problem in Section 2 and introduce $LeGen$ in Section 3. We discuss our experiments and results in Section 4 and 5. In Section 6, we discuss work related to legal, complex, and open information extraction and in Section 7, we discuss future work. The limitations of our approach are described in Section 9. Additional details and experiments are listed in the Appendix (Section A).

## 2 Problem Definition

We denote the sentences (example in Table 1) by $\mathcal{S}$. Our goal is to identify from $\mathcal{S}$:

**1.** A set $C$ of all clauses in $\mathcal{S}$. A clause refers to an indivisible, atomic sentence in $\mathcal{S}$. $C = \{$"`Balance amount in the account of a deceased is higher than 150,000 then`", "`The nominee has to prove the identity to claim the amount`", "`Legal heir has to prove the identity to claim the amount`"$\}$ for the example in Table 1.

**2.** A set $COMP$ of complex sentences that are obtained either by $i)$ combining N clauses *which are subsets of clauses,* $C$, using an N-ary relation, or, $ii)$ by combining subsets of $C$ and $COMP$ using N-ary relation.

**3.** A set $R$ of N-ary relations that relate $N$ clauses or complex sentences and generate a new complex sentence. In other words, $R_{r_i}$: $\{C \cup COMP\}^N \longrightarrow COMP$, where $R_{r_i} \in R$. For $\mathcal{S}$, $R = \{R_{\texttt{condition}}, R_{\texttt{disjunction}}\}$. The output of $R_{\texttt{condition}}$("`Balance amount in the account of a deceased is higher than 150,000 then`", $R_{\texttt{disjunction}}$"`The nominee has to prove the identity to claim the amount`","`Legal heir has to prove the identity to claim the amount`" )) is $\mathcal{S}$.

Three properties that should be satisfied by $C$, $COMP$ and $R$ are:

**Correct:** Every $c \in C$, $c' \in COMP$ and $r \in R$ should convey the same meaning as expressed in $\mathcal{S}$

**Non-redundant:** $C$, $R$, and $COMP$ should not contain repeated information

**Complete:** All information conveyed in the sentence should be expressed by $C$, $R$, and $COMP$

## 3 LeGen

We propose $LeGen$, an end-to-end generative model to perform complex information extraction from legal sentences. $LeGen$ is based on the idea of discourse trees, which are defined in the next subsection. We model it as a generation task, that outputs discourse trees for a sentence.

### 3.1 Discourse Tree

The Discourse Tree (Cetto et al., 2018; Niklaus et al., 2019) originates from Rhetorical Structure Theory (RST) (Mann and Thompson, 1988), which identifies hierarchical text structures and rhetorical relations between text parts. These relations are categorized as coordination and subordination.

Coordinating sentences join independent clauses with coordinating conjunctions like `and`, `or`, and `but`. Subordination sentences combine main clauses with dependent clauses, using subordinating conjunctions like `while`, `because`, `if`, etc.

The Discourse Tree follows a top-down approach, breaking text into smaller parts, unlike the bottom-up approach of RST. Simplified sentences can vary and may require adjustments based on specific structures. Figure 2 (left) illustrates a Discourse Tree example, with leaf nodes representing clauses and non-leaf nodes representing complex sentences formed by combining clauses using relation labels. Relations in a discourse tree fall into co-ordinations and sub-ordinations categories.

Our goal is to learn accurate discourse trees for legal sentences (Section 3.2). We can model two types of discourse structures:

**Discourse Trees for Identifying Subordinations and Coordinations** In this case, we learn both subordination and coordination from the sentence. The sentence is parsed into multiple clauses, also referred to as EDUs (Elementary Discourse Units), by identifying logical connectives such as subordinates and coordinates. Both the clauses and the relationships between them are identified and structured as a linear discourse tree. We refer to this task as `Task 1` henceforth.

**Discourse Trees for Coordination Boundary Detection** The problem of coordinate boundary detection (Saha et al., 2018) can be expressed as a special case of learning discourse tree where all the non-leaf nodes represent the same relation, i.e. `COORDINATION`. We investigate this approach to learn discourse trees for the problem of coordinate boundary detection. We refer to this task as `Task 2`.

We build separate models for identifying coordinates and subordinates due to the distinct nature of each task. The coordination task focuses on recognizing coordination boundaries and forming independent clauses, involving only one type of relationship. In contrast, the subordination task involves identifying multiple inter-clausal relationships. Combining these tasks could increase the problem's complexity (Evans, 2011).

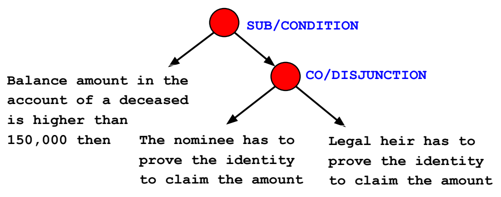
### 3.2 Generating Discourse Trees

Any existing rule-based approach can be used to generate the discourse trees for sentences. Currently, Graphene (Niklaus et al., 2019) generates discourse trees with good precision and recall. Graphene uses a set of 39 hand-crafted rules to identify 19 relations (Cetto et al., 2018). However, on analyzing these rules, we observed redundancies and inconsistencies. $i$) For instance, it is very difficult to distinguish between `BACKGROUND`, `ELABORATION`, or `EXPLANATION` relations. $ii$) the rules proposed for identifying `TEMPORAL_BEFORE` and `TEMPORAL_AFTER` relations from the text are not accurate. $iii$) Does not identify the date and named entities correctly. To address $i$) and $ii$), we merged `BACKGROUND`, `ELABORATION`, and `EXPLANATION` into `ELABORATION`. We converted `TEMPORAL_BEFORE` and `TEMPORAL_AFTER` into a single `TEMPORAL` relation. We did not address $iii$), but we show in Section 5 that $LeGen$ is robust to these issues. The final 10 relation set used in the training are `SPATIAL`, `LIST`, `ATTRIBUTION`, `CONTRAST`, `DISJUNCTION`, `CAUSE`, `CONDITION`, `ELABORATION`, `TEMPORAL` and `PURPOSE`. The above relations are explained in detail with the example in the Appendix (Section A).

### 3.3 Encoding of Discourse Tree

Figure 2 demonstrates the conversion of a discourse tree into a sequence encoding, simplifying complex information extraction. We treat this process as a generation task, where the input is the legal sentence and the output is the tree encoding. Our method converts original input sentences, including clauses and relationships, into explicit discourse

***If balance amount in the account of a deceased is higher than ₹150,000 then the nominee or legal heir has to prove the identity to claim the amount.***


SUB/CONDITION
Balance amount in the account of a deceased is higher than 150,000 then
CO/DISJUNCTION
The nominee has to prove the identity to claim the amount
Legal heir has to prove the identity to claim the amount

```
SUB/CONDITION('Balance amount in the
account of a deceased is higher than
150,000 then .', CO/DISJUNCTION('The
nominee has to prove the identity to
claim the amount .','Legal heir has to
prove the identity to claim the amount
.'))
```

Figure 2: Discourse tree for an example law sentence (on the left). Corresponding linear encoding of the Discourse tree (on the right). `SUB` and `CO` refer to subordination and coordination, respectively.

trees. We encode the discourse tree by doing a pre-order traversal of the tree. Algorithm 1 discusses our steps.

**Algorithm 1:** Generating encoding $\mathcal{E}$ for a Discourse Tree $T$.

```
Input: Discourse Tree $\mathcal{T}$ with root $root$
Output: Encoding, $\mathcal{E}$
Append '$root.label($' to $\mathcal{E}$
foreach $child$ of $root$ in $\mathcal{T}$ do
    if $child$ is a leaf then
        Append '$child.label,$' to $\mathcal{E}$
    end
    else
        Generate encoding $\mathcal{E}'$ of Discourse SubTree
          with $child$ as root
        Append $\mathcal{E}'$ to $\mathcal{E}$
    end
end
Append ')' to $\mathcal{E}$
return $\mathcal{E}$
```

### 3.4 Learning Discourse Tree with LLMs

The sequence generated using Algorithm 1 can be learnt by fine-tuning smaller LLMs such as T5 or BART or prompting larger LLMs such as GPT in a few-shot or zero-shot manner. We have prompted larger models (GPT) only as smaller models like T5 and BART lack the flexibility or capacity to interpret the complex prompts (Raffel et al., 2020). We propose using the following approaches for prompting GPT-4o:

**Few shot Learning:** We provided the model with a few examples, each illustrating different types of relationships and the clauses that might be present in the sentence. The prompts are in Section A.8 and A.9.

**Zero-Shot Learning:** In the absence of any examples, we provided explicit steps to construct a discourse tree to ensure a consistent output format, aligning these steps with the chain-of-thought (CoT) process (Feng et al., 2024). We tried two kinds of zero-shot learning.

1. Unrestricted: We did not supply any examples or specify any particular types of relationships. The model was expected to infer the relationships based on the presence of subordinates and coordinates within the sentences. This is illustrated in Section A.7.

2. Restricted: We provided the model with a predefined set of relationships as outlined in Section A.1, and the prompts used for the same can be found in Section A.6.

There are no restricted and unrestricted relations in `Task 2` as it has only one kind of relation, `COORDINATION`, and the prompts are in Section A.10.

### 3.5 Custom Loss Function for Handling Hallucinations

Any generative model is prone to hallucinations (Ji et al., 2023). Handling them is crucial in the context of generating trees for an accurate understanding of legal sentences. A common form of hallucination observed is repetition, i.e. more than 1 leaf node in the tree contains the same sentence. This form of hallucination is difficult to be penalized using regular cross entropy loss function since in most of the cases, all leaf node sentences only differ by a few words, so when the model generates the same sentences for multiple leaf nodes, regular loss would still be low. So, we propose a custom loss function to punish the model for this kind of output.

$$Custom\_Loss = Reg\_Loss \times \left(1 + \lambda \left(1 - \frac{u(T)}{n(T)}\right)\right)$$

where $T$ denotes the discourse tree, $Reg\_Loss$ refers to regular cross-entropy loss, $n(T)$ denotes the number of leaf nodes in $T$, $u(T)$ denotes the number of unique leaf nodes, and $\lambda$ is a hyperparameter which can take any real value greater than zero. If $n(t) = u(T)$, $Reg\_Loss = Custom\_Loss$. The loss increases linearly parameterized by $\lambda$ as $u(t) << n(t)$.

## 4 Experiments

### 4.1 Datasets

#### 4.1.1 Training

We trained $LeGen$ using 17k sentences from Penn Tree Bank (Marcus et al., 1993) dataset. We have used the same dataset for both `Task 1` and `Task 2` because we wanted to test the transfer learning capability of our approach on the legal domain. We performed our experiments on 32x2 cores AMD EPYC 7532, 1 TB of memory, and 8x A100 SXM4 80GB GPU systems. We trained the models using BART-base (139 M), BART-small (70.5 M), T5-base (246 M), and T5-small (77M) architectures. BART trained faster (2 hours on small and 2.5 hours on base). T5 took considerably longer time (3 hours for small and 4 hours for base). We train it separately for `Task 1` and `Task 2`.

For `Task 1`, we encoded every sentence into a discourse tree structure as described in Section 3.2. We trained BART (Lewis et al., 2019) and T5 (Abdallah et al., 2023) models for 30 epochs using cross-entropy loss with a learning rate of $e^{-}5$. Results are averaged over 3 seeds (Section 5). With GPT-4o models, we experimented with three kinds of prompting as outlined in Section 3.4. We selected 11 examples for few-shot learning, corresponding to the 10 identified types of relationships in the text, plus an additional example for cases where the sentence cannot be split (NONE) (Section A.8). For zero-shot learning, we applied Chain of Thought-style prompting with both restricted and unrestricted sets of relationships.

For `Task 2`, we kept the same hyperparameters that we used for the `Task1` and obtained the best results for batch size 3. Results are averaged over 3 seeds (Section 5). For both of them, we also trained the model with a custom loss function, setting $\lambda = 1$. With GPT-4o models, we provided the model with 11 examples for few-shot learning, with the prompt in Section A.9. These examples included sentences across hierarchical levels (0-5), showing how input sentences can be split into independent ones. For zero-shot learning, we provided steps to build a hierarchical representation of coordinating sentences, with prompts in Section A.10 . We only performed restricted prompting because it involves only one type of relation. The GPT-4o model parameters were set to: *temperature = 1* and *top_p = 1*

#### 4.1.2 Test

**1) ILDC Dataset (Used for Task 1).** ILDC is a Indian Legal Dataset (Malik et al., 2021) comprising the transcripts of 35k Indian Supreme Court Cases. We sampled 50 sentences from this corpus. The dataset is fairly noisy with multiple spelling and structural inconsistencies.

**2) Indian Legal Dataset (Used for Task 1).** ILDC corpus is noisy, so we looked for cleaner legal sentences to test our model. We constructed a new dataset of 107 sentences from Wiki on Labour Law [1]. We used the Petscan tool to collect sentences belonging to the 'Labour Law' category from Wiki. These sentences contained multiple references, requiring pre-processing to remove mentions of other articles. The sentences were also presented as itemized lists, which had to be merged into single sentences. To understand the data, two authors of the paper spent time constructing the discourse tree structure for each sentence from scratch. We observed that there were multiple correct tree representations for one sentence, as evident from the example in Section A.3. The problem becomes more complex for trees with greater height.

**3) Penn Tree Bank (Used for Task 2).** Penn Tree Bank (Marcus et al., 1993) consists of 985 sentences from articles in the Wall Street Journal. It is annotated with coordinate boundaries (`and`, `or`, `but`, `comma-separated list`) and the text spans it connects. This test set was used to evaluate $LeGen$'s flexibility in identifying co-ordinations.

### 4.2 Metrics

#### 4.2.1 Metrics for Task 1

Various metrics have been proposed in the literature to evaluate discourse trees (Vadlapudi et al., 2009; Yuan et al., 2021). A key disadvantage of these metrics is that they either focus on surface-level relations, or they completely ignore the relations (Mitocariu et al., 2013), without adequately addressing multiple discourse relations such as `ELABORATION`, `CAUSE`, or `RESULT`. The metric proposed in (Yuan et al., 2021) specifically focuses on dependency

[1]https://en.wikipedia.org/wiki/Indian_labour_law

distance and the complexity of constructing the discourse tree but does not account for inter-clausal relations. Additionally, discourse trees are often evaluated based on their performance in downstream tasks, such as question answering (Pyatkin et al., 2020; Sovrano et al., 2024) or machine translation (Yuan et al., 2021). We also noted that a single sentence could have multiple correct tree representations, particularly evident for taller trees as illustrated in Section A.3 (Appendix). Given these issues, we used human judgment to evaluate the trees based on $i$) structure of the tree and $ii$) content of the tree, i.e., the relation labels. We propose two metrics.

**Tree Structure Evaluation (TSE).** We employed a strict evaluation technique, i.e. it was marked as correct only if all the 3 requirements cited in Section 2 were satisfied – $i$) Every node in the tree was correctly split. $ii$) Tree does not contain multiple nodes with the same information, $iii$) All information in the sentence was conveyed in the tree. **TSE** reports the percentage of sentences that generated correct trees.

**Tree Content Evaluation (TCE).** To assess tree content, annotators were tasked with labeling each relation as correct or incorrect, informed about the relations present in the test set. A relation was marked incorrect if it was expressed differently or if it connected incorrect clauses. Inaccuracies in relations resulted in penalties applied to the entire tree structure post-clause verification.

#### 4.2.2 Metrics for Task 2

We employed a **mapping-based approach** proposed in CalmIE (Saha et al., 2018) to compare the clauses generated by our technique with the gold set. For every conjunctive sentence, we evaluated it by matching its collection of system-generated clauses with the reference set. This involved establishing the most optimal one-to-one correspondence between the clauses in both sets. Subsequently, precision was determined for each mapping by calculating the ratio of shared words to the total words in the generated sentence, while recall was calculated as the ratio of shared words to the total words in the reference sentence.

Let $G = \{G_1, G_2, G_3 \ldots\}$ be gold/reference clauses each represented as a bag of words model, i.e. $G_i = \{G_i^{a1}, G_i^{a2}, G_i^{a3} \ldots\}$ where each $G_i^{aj}$ denotes a token in a clause. Similarly let $T = \{T_1, T_2, T_3 \ldots\}$ be clauses generated by a model where $T_i = \{T_i^{a1}, T_i^{a2}, T_i^{a3} \ldots\}$. CalmIE performs matching in a greedy fashion, however, this type of matching is not optimal and might change based on the order in which greedy matching is performed. So, we performed matching to get the global maximum. This problem of finding the global optimum from a distance or similarity matrix can be treated as a linear sum assignment problem (Crouse, 2016). We matched clauses from Gold Set $G$ and Predicted Set $T$ to maximise the F1 score. The F1 score was computed using precision and recall metrics. All equations are presented in the Appendix in Section A.2 of appendix A.

### 4.3 Baselines

**Graphene Default.** We used the default Graphene (Niklaus et al., 2019) as the competing technique for `Task 1`. We observed that although it can split long complex sentences, it is unable to identify the relations correctly.

**Graphene.** We used modified Graphene (Refer Section 3.2) as the competing technique for `Task 1`.

**OpenIE6_Coordinate-Boundary_Detection.** We used the Coordination Boundary Detection Model released with OpenIE6 as our baseline for `Task 2`.

## 5 Results

### 5.1 Task 1

Table 2 presents the results for TSE and TCE scores and the number of clauses and relations generated in the discourse trees using three different techniques. The results demonstrate that the generative approach to discourse tree creation significantly outperforms Graphene on both datasets—the Indian Legal Dataset and ILDC. The GPT-4o model performs the best, achieving a TSE score of 82% on the Indian Legal Dataset and 90% on the ILDC Dataset. T5 and BART-Base hallucinates more and the reason for its underperformance is the generation of terms not present in the original sentence. Graphene Default performs worse than modified Graphene. While it splits clauses correctly, it's TCE is much lower because of our observations reported in Section 3.2.

Graphene also underperforms in sentences where domain-specific named entities such as statutes, laws, or case names are present, e.g. *Shops and Establishment Act 1960* or *The Factories Act 1948* (Indian Legal Dataset of Table 3).

| **Dataset** | **Models** | **TSE** | **TCE** | **#(Relations, Clauses)** |
|---|---|---|---|---|
| ILDC | Graphene Default | 0.54 | 0.74 | (174,125) |
| | Graphene | 0.54 | 0.77 | (174,125) |
| | T5 | 0.56 | **1** | (137,88) |
| | T5 Custom Loss | 0.56 | 1 | (137,88) |
| | BART | 0.48 | 1 | (111,62) |
| | BART Custom Loss | 0.48 | 0.83 | (127,76) |
| | GPT-4o (11 Shot) | **0.90** | <u>0.89</u> | (100,50) |
| | GPT-4o(Zero shot CoT U) | <u>0.70</u> | 0.88 | (152,96) |
| | GPT-4o (Zero shot CoT R) | 0.64 | <u>0.90</u> | (156,92) |
| Indian Legal Dataset | Graphene Default | 0.62 | 0.54 | (247, 347) |
| | Graphene | 0.62 | 0.92 | (247, 347) |
| | T5 | 0.71 | <u>0.96</u> | (191, 349) |
| | T5 Custom Loss | 0.56 | **1** | (404,238) |
| | BART | 0.70 | 0.92 | (183, 281) |
| | BART Custom Loss | 0.61 | 0.95 | (289,185) |
| | GPT-4o (11 Shot) | **0.82** | 0.87 | (236,134) |
| | GPT-4o (Zero shot CoT U) | 0.76 | 0.79 | (319,248) |
| | GPT-4o (Zero shot CoT R) | <u>0.79</u> | 0.90 | (317,187) |

Table 2: Results for Task 1: TSE and TCE results of Graphene, GPT-4o, T5, and BART with regular and custom loss function on 2 datasets averaged over 3 seeds. The best values are in bold. The second best is underlined. U stands for Unrestricted and R stands for Restricted.

Graphene also cannot identify non distributive coordination like 'between' and splits sentences on them. All these issues are handled very well by generative models even though they were trained on Graphene's output. The error analysis of the T5 and BART models is presented in Section A.4.

GPT-4o models perform auto-correction of words as observed in the ILDC dataset in Table 3, which is a further improvement on T5 and BART. The input sentence of the ILDC data set has many words which are misspelt, like `companytained`, `companydition` and `companytract`, which was auto-corrected by the GPT-4o model to `contained`,`condition` and `contract`.

While evaluating for **TCE**, we took into consideration the fact that there could be multiple ways of representing sentences with different relations. There are situations where models can split the sentences but are unable to identify the relations, and BART has made spelling mistakes in identifying the relation. Although such scenarios were rare in T5, we came across them in Graphene and BART.

**Inter-annotator Agreement.** We sampled 50% of the sentences annotated by Annotator 1 and asked Annotator 2 to evaluate them. We obtained a Cohen's Kappa agreement value of 0.73 for TSE and 0.71 for TCE, indicating substantial agreement (Blackman and Koval, 2000).

## 5.2 Task 2

Table 4 shows our results. We obtained competent results from the T5-base against OpenIE6_Coordinate-Boundary_Detection. The slight drop in the performance of T5-Base could be attributed to ambiguous labels in the Penn Tree Bank dataset. For instance, one split in the gold for "*He retired as senior vice president, finance and administration, and chief financial officer of the company Oct. 1*" is "*He retired as senior vice president, finance Oct. 1*", while T5 generates "*He retired as senior vice president, finance, of the company Oct. 1*". T5 generates a better split but it gets penalised because this is not captured in gold.

BART did not perform well as it hallucinated while generating the output where it used words that are not in the input. BART was also unable to split all elements of comma-separated lists. The same problem was observed for T5-small which improved with T5-base.

The results obtained for `Task 2` with few-shot and zero-shot learning did not match those achieved with T5 and BART. The GPT-4o model did not perform well mainly because of two reasons: difficulty in correctly splitting sentences into multiple hierarchical levels and the loss of contextual information. The model could correctly identify the conjunctions in the sentence but failed to form independent clauses. We are working towards fixing this issue with GPT models by refining the prompts to include context information and also improving hierarchical levels wherever needed.

## 5.3 Effect of Custom Loss Function

On `Task 2`, using the custom loss function improved the results for T5-small, T5-Base, and BART-Base (Table 4, example in Appendix, Figure **??**). BART hallucinates by inventing new relations in the discourse tree which is not handled by our custom loss function. This could be the reason for low performance of BART-small with custom loss.

On `Task 1`, using the custom loss function gave mixed results. Results are shown in Table 2. On the ILDC corpus, it didn't lead to any improvement for TSE while TCE reduced for BART. This is similar to what we observed for BART on `Task 2`. On the Indian Legal Dataset, enforcing custom loss made the model split a sentence into more number of clauses, however, this does not necessarily mean it is a correct splitting. This led to a reduction in the TSE scores. The total number of relations gen-

| Dataset | Input | Clauses generated by Graphene | Clauses generated by T5 BASE | Clauses generated by GPT-4o model |
|---|---|---|---|---|
| **Indian Legal Dataset** | The Factories Act 1948 and the Shops and Establishment Act 1960 mandate 15 working days of fully paid vacation leave each year to each employee with an additional 7 fully paid sick days. | 1) This was with an additional 7 fully paid<br>2) This was to each employee<br>3) The Factories leave each year sick days<br>4) Act 1948 mandate 15 working days of fully paid vacation The Factories<br>5) The Shops and Establishment Act 1960 mandate 15 working days of fully paid vacation The Factories | 1) This was to each employee with an additional 7 fully paid sick days<br>2) The Factories Act 1948 mandate 15 working days of fully paid vacation leave each year<br>3) The Shops and Establishment Act 1960 mandate 15 working days of fully paid vacation leave each year. | 1) The Factories Act 1948 and the Shops and Establishment Act 1960 mandate 15 working days of fully paid vacation leave each year to each employee.<br>2) This was in addition to 7 fully paid sick days. |
| **ILDC** | Learned companynsel appearing on behalf of the appellant has argued that the arbitration clause companytained in the tender companydition amounts to a companytract of adhesion, and since there is unfair bargaining strength between respondent No.2 and the appellant, this clause ought to be struck down following the judgment in Central Inland Water Transport Corpn.v. Brojo Nath Ganguly, 1986 3 SCC 156. | 1) This was what learned companynsel appearing on behalf of the appellant has argued .<br>2) The arbitration clause companytained in the tender companydition amounts to a companytract of adhesion.<br>3) Central Inland Water Transport Corpn.v. Brojo Nath Ganguly was 1986 3 SCC 156.<br>4) There is unfair bargaining strength between respondent No. 2.<br>5) There is the appellant.<br>6) This clause ought to be struck down following the judgment in Central Inland Water Transport Corpn.v . Brojo Nath Ganguly . | 1) Brojo Nath Ganguly was the judgment in Central Inland Water Transport Corpn.v . Brojo Nath Ganguly , 1986 3 SCC 156.<br>2) This was what learned companynsel appearing on behalf of the appellant has argued.<br>3) The arbitration clause companytained in the tender companydition amounts to a companytract of adhesion , and since there is unfair bargaining strength between respondent No.2 and the appellant , this clause ought to be struck down following Brojo Nath Ganguly , 1986 3 SCC 156. | 1) Learned counsel appearing on behalf of the appellant has argued.<br>2) That the arbitration clause contained in the tender condition amounts to a contract of adhesion.<br>3) And since there is unfair bargaining strength between respondent No.2 and the appellant.<br>4) This clause ought to be struck down following the judgment in Central Inland Water Transport Corpn.v. Brojo Nath Ganguly, 1986 3 SCC 156. |

Table 3: Examples showing the superiority of generative architectures in identifying correct clauses. Their strength also lies in the accurate detection of named entities and auto-correction.

erated by both BART and T5 reduced which may have led to an increase in TCE scores. Overall, we can conclude that subordination is a more complex task than coordination which needs more nuanced handling of hallucinations.

| Models | Mapping Based Approach | | | |
|---|---|---|---|---|
| | Loss function | Precision | Recall | F1 Score |
| **OpenIE6_Coordinate -Boundary_Detection** | Regular | **0.9803** | **0.9845** | **0.9816** |
| **T5 Small** | Custom | 0.9671 | 0.9538 | 0.9578 |
| | Regular | 0.9647 | 0.9544 | 0.9571 |
| **T5 Base** | Custom | <u>0.9756</u> | 0.974 | <u>0.9739</u> |
| | Regular | 0.9747 | <u>0.973</u> | 0.9726 |
| **BART Small** | Custom | 0.8273 | 0.7334 | 0.7672 |
| | Regular | 0.8215 | 0.7391 | 0.7682 |
| **BART Base** | Custom | 0.8418 | 0.7613 | 0.7903 |
| | Regular | 0.8369 | 0.7574 | 0.7903 |
| **GPT-4o** (11 Shot) | - | 0.4124 | 0.2816 | 0.3198 |
| **GPT-4o** (Zero Shot CoT) | - | 0.6024 | 0.3823 | 0.4503 |

Table 4: Results on `Task 2`: Mapping-based approach is used to calculate precision, recall and f1 score using cross-entropy loss function and custom loss function. The best values are in bold. The second best is underlined.

## 6 Related Work

### 6.1 Legal Information Extraction

Legal Information Extraction has advanced with NLP techniques aiding legal professionals (Chalkidis et al., 2017; Leivaditi et al., 2020; Cardellino et al., 2017). Although open information extraction methods attempt structured triple extraction from legal statements (Zadgaonkar and Agrawal, 2021), challenges remain (see Section 1). Core tasks include NER, document summarization, and judgment prediction (LJP), facilitated by systems like Eunomos (Boella et al., 2016; Abood and Feltenberger, 2018; Nguyen et al., 2018) and enhanced by legal ontologies like YAGO (Cardellino et al., 2017).

Early judgment prediction relied on rule-based models (HYPO, CATO) (Rissland and Ashley, 1987; Aleven and Ashley, 1995) and ML approaches like SVMs (Aletras et al., 2016), which achieved 79% accuracy. Recent studies continued this with neural models (Medvedeva et al., 2020;

Chalkidis et al., 2019a) and semi-supervised techniques (Branting et al., 2021). Adapted pre-trained models such as LegalBERT (Chalkidis et al., 2020) and datasets in multiple languages have enriched the field (Chalkidis et al., 2021, 2019b).

In the Indian context, works like InLegalBERT (Paul et al., 2023) and corpora like ILDC (Malik et al., 2021) advance NLP on Indian legal data. Legal tasks in ILDC include NER, rhetorical role prediction (Kalamkar et al., 2022), and court judgment prediction (Modi et al., 2023). Other Indian resources include HLDC for bail prediction (Kapoor et al., 2022), LJP (Cui et al., 2023) and NLP benchmarks (Kalamkar et al., 2021).

### 6.2 Open Information Extraction

Open Information Extraction uses an independent paradigm to extract the information as a triple, ⟨`subject, relation, object`⟩. (Yates et al., 2007) introduced the concept of Open Information Extraction and proposed Text Runner. Following this, many rule-based systems were developed, like REVERB (Etzioni et al., 2011) and OpenIE5 (Saha et al., 2018). RNNOIE (Stanovsky et al., 2018) which uses a neural-based approach to open information extraction and is trained by the data extracted from non-neural systems. The state-of-the-art in Open Information Extraction, OpenIE6 (Kolluru et al., 2020) uses iterative grid labeling with BERT architecture to generate triples from input.

### 6.3 Complex Information Extraction

Several OIE systems address complex sentence extraction (Mahouachi and Suchanek, 2020), including OLLIE (Schmitz et al., 2012), MinIE (Gashteovski et al., 2017), and Graphene (Cetto et al., 2018). Methods vary from rule-based (ClausIE, MinIE) to syntactical (StuffIE (Prasojo et al., 2018)) and structured approaches (Graphene (Niklaus et al., 2019)). Our work uniquely explores generative neural architectures for complex information extraction.

### 6.4 Discourse Tree and its Applications

Discourse trees (DT) originated from Rhetorical Structure Theory (RST), which organizes text through relations within parts to create hierarchical structures. DTs can be generated by various methods, including data-driven approaches assessing topicality (Schilder, 2002), learning-based techniques for sentence and cross-sentence relations (Soricut and Marcu, 2003; Baldridge and Lascarides, 2005), shift-reduce parsing (Ji and Eisenstein, 2014), and handcrafted rules (Cetto et al., 2018). Recent advances leverage NLP and computer vision for constructing DTs from heterogeneous data (Schneider et al., 2023). Different DT representations, like SDRT and EDU theory, approach discourse as either structured graphs or without assuming tree structures. DTs are applied in tasks such as Question Answering (Sovrano et al., 2024), answer indexing (Galitsky and Ilvovsky, 2019), and summarization (Yoshida et al., 2014; Pu and Demberg, 2024). Our work, however, focuses on constructing DTs at the sentence level rather than the document level as in previous studies.

## 7 Conclusion

We developed an end-to-end generative model for legal information extraction that improves legal sentence comprehension. Using T5, BART, and GPT-4o models, we learned sentence discourse trees, which outperformed Graphene on an Indian Legal and ILDC Dataset and achieved competitive performance in coordinate boundary detection. In the future, we will use the discourse trees generated from Indian Law to populate a legal knowledge graph which can be used to develop a question-answering system to support low-literate users.

## 8 Acknowledgements

We would like to thank the HPC facility at BITS Pilani Hyderabad Campus for extending their support to conduct experiments. The last 2 authors are supported by OpenAI Researcher Access and API Award and BITS Pilani Sparkle Grant. The last author is also supported by BITS Pilani New Faculty Seed Grant in executing this project.

## 9 Limitations

∗ Our dataset could be biased due to unequal training instances for each kind of relation.
∗ GPT models generate a varied number of clauses and relations for the same input sentence. This randomness of GPT models is propagated to our models as well. Due to this, our models generate a varied number of clauses and relations for the same input sentence.
∗ Due to the presence of multiple correct discourse trees for subordination tasks, it is difficult to create a benchmark to automatically evaluate the models. They require expensive human annotations.

## References

Abdelrahman Abdallah, Bhawna Piryani, and Adam Jatowt. 2023. Exploring the state of the art in legal qa systems. *arXiv preprint arXiv:2304.06623*.

Aaron Abood and Dave Feltenberger. 2018. Automated patent landscaping. *Artificial Intelligence and Law*, 26(2):103–125.

Nikolaos Aletras, Dimitrios Tsarapatsanis, Daniel Preoţiuc-Pietro, and Vasileios Lampos. 2016. Predicting judicial decisions of the european court of human rights: A natural language processing perspective. *PeerJ computer science*, 2:e93.

Vincent Aleven and Kevin D Ashley. 1995. Doing things with factors. In *Proceedings of the 5th international conference on artificial intelligence and law*, pages 31–41.

Jason Baldridge and Alex Lascarides. 2005. Probabilistic head-driven parsing for discourse structure. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 96–103.

Nicole J-M Blackman and John J Koval. 2000. Interval estimation for cohen's kappa as a measure of agreement. *Statistics in medicine*, 19(5):723–741.

Guido Boella, Luigi Di Caro, Llio Humphreys, Livio Robaldo, Piercarlo Rossi, and Leendert van der Torre. 2016. Eunomos, a legal document and knowledge management system for the web to provide relevant, reliable and up-to-date information on the law. *Artificial Intelligence and Law*, 24:245–283.

L Karl Branting, Craig Pfeifer, Bradford Brown, Lisa Ferro, John Aberdeen, Brandy Weiss, Mark Pfaff, and Bill Liao. 2021. Scalable and explainable legal prediction. *Artificial Intelligence and Law*, 29:213–238.

Cristian Cardellino, Milagro Teruel, Laura Alonso Alemany, and Serena Villata. 2017. Legal nerc with ontologies, wikipedia and curriculum learning. In *15th European Chapter of the Association for Computational Linguistics (EACL 2017)*, pages 254–259.

Matthias Cetto, Christina Niklaus, André Freitas, and Siegfried Handschuh. 2018. Graphene: Semantically-linked propositions in open information extraction. *arXiv preprint arXiv:1807.11276*.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019a. Neural legal judgment prediction in english. *arXiv preprint arXiv:1906.02059*.

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

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. Legal-bert: The muppets straight out of law school. *arXiv preprint arXiv:2010.02559*.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. 2019b. Large-scale multi-label text classification on eu legislation. *arXiv preprint arXiv:1906.02192*.

Ilias Chalkidis, Manos Fergadiotis, Dimitrios Tsarapatsanis, Nikolaos Aletras, Ion Androutsopoulos, and Prodromos Malakasiotis. 2021. Paragraph-level rationale extraction through regularization: A case study on european court of human rights cases. *arXiv preprint arXiv:2103.13084*.

Fernando A Correia, Alexandre AA Almeida, José Luiz Nunes, Kaline G Santos, Ivar A Hartmann, Felipe A Silva, and Hélio Lopes. 2022. Fine-grained legal entity annotation: A case study on the brazilian supreme court. *Information Processing & Management*, 59(1):102794.

David F Crouse. 2016. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696.

Junyun Cui, Xiaoyu Shen, and Shaochun Wen. 2023. A survey on legal judgment prediction: Datasets, metrics, models and challenges. *IEEE Access*.

Robert Dale. 2019. Law and word order: Nlp in legal tech. *Natural Language Engineering*, 25(1):211–217.

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, et al. 2011. Open information extraction: The second generation. In *Twenty-Second International Joint Conference on Artificial Intelligence*. Citeseer.

Richard J Evans. 2011. Comparing methods for the syntactic simplification of sentences in information extraction. *Literary and linguistic computing*, 26(4):371–388.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. 2024. Towards revealing the mystery behind chain of thought: a theoretical perspective. *Advances in Neural Information Processing Systems*, 36.

Boris Galitsky and Dmitry Ilvovsky. 2019. Two discourse tree-based approaches to indexing answers. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 367–372.

Kiril Gashteovski, Rainer Gemulla, and Luciano del Corro. 2017. Minie: minimizing facts in open information extraction. Association for Computational Linguistics.

Google. 2023. Next billion users. https://blog.google/technology/next-billion-users/.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 13–24.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Anirudha Joshi. 2013. Technology adoption by'emergent'users: the user-usage model. In *Proceedings of the 11th Asia Pacific conference on computer human interaction*, pages 28–38.

Prathamesh Kalamkar, Astha Agarwal, Aman Tiwari, Smita Gupta, Saurabh Karn, and Vivek Raghavan. 2022. Named entity recognition in indian court judgments. *Preprint*, arXiv:2211.03442.

Prathamesh Kalamkar, Janani Venugopalan, and Vivek Raghavan. 2021. Benchmarks for indian legal nlp: a survey. In *JSAI International Symposium on Artificial Intelligence*, pages 33–48. Springer.

Arnav Kapoor, Mudit Dhawan, Anmol Goel, TH Arjun, Akshala Bhatnagar, Vibhu Agrawal, Amul Agrawal, Arnab Bhattacharya, Ponnurangam Kumaraguru, and Ashutosh Modi. 2022. Hldc: Hindi legal documents corpus. *arXiv preprint arXiv:2204.00806*.

Keshav Kolluru, Vaibhav Adlakha, Samarth Aggarwal, Soumen Chakrabarti, et al. 2020. Openie6: Iterative grid labeling and coordination analysis for open information extraction. *arXiv preprint arXiv:2010.03147*.

Spyretta Leivaditi, Julien Rossi, and Evangelos Kanoulas. 2020. A benchmark for lease contract review. *arXiv preprint arXiv:2010.10386*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Mechket Emna Mahouachi and Fabian M Suchanek. 2020. Extracting complex information from natural language text: A survey. In *CIKM (Workshops)*.

Vijit Malik, Rishabh Sanjay, Shubham Kumar Nigam, Kripa Ghosh, Shouvik Kumar Guha, Arnab Bhattacharya, and Ashutosh Modi. 2021. Ildc for cjpe: Indian legal documents corpus for court judgment prediction and explanation. *arXiv preprint arXiv:2105.13562*.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank.

Masha Medvedeva, Michel Vols, and Martijn Wieling. 2020. Using machine learning to predict decisions of the european court of human rights. *Artificial Intelligence and Law*, 28:237–266.

Elena Mitocariu, Daniel Alexandru Anechitei, and Dan Cristea. 2013. Comparing discourse tree structures. In *Computational Linguistics and Intelligent Text Processing: 14th International Conference, CICLing 2013, Samos, Greece, March 24-30, 2013, Proceedings, Part I 14*, pages 513–522. Springer.

Ashutosh Modi, Prathamesh Kalamkar, Saurabh Karn, Aman Tiwari, Abhinav Joshi, Sai Kiran Tanikella, Shouvik Kumar Guha, Sachin Malhan, and Vivek Raghavan. 2023. Semeval 2023 task 6: Legaleval - understanding legal texts. *Preprint*, arXiv:2304.09548.

Truong-Son Nguyen, Le-Minh Nguyen, Satoshi Tojo, Ken Satoh, and Akira Shimazu. 2018. Recurrent neural network-based models for recognizing requisite and effectuation parts in legal texts. *Artificial Intelligence and Law*, 26:169–199.

Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh. 2019. Transforming complex sentences into a semantic hierarchy. *arXiv preprint arXiv:1906.01038*.

Shounak Paul, Arpan Mandal, Pawan Goyal, and Saptarshi Ghosh. 2023. Pre-trained language models for the legal domain: a case study on indian law. In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*, pages 187–196.

Radityo Eko Prasojo, Mouna Kacimi, and Werner Nutt. 2018. Stuffie: Semantic tagging of unlabeled facets using fine-grained information extraction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 467–476.

Dongqi Pu and Vera Demberg. 2024. Rst-lora: A discourse-aware low-rank adaptation for long document abstractive summarization. *arXiv preprint arXiv:2405.00657*.

Valentina Pyatkin, Ayal Klein, Reut Tsarfaty, and Ido Dagan. 2020. Qadiscourse–discourse relations as qa pairs: Representation, crowdsourcing and baselines. *arXiv preprint arXiv:2010.02815*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Edwina L Rissland and Kevin D Ashley. 1987. A case-based system for trade secrets law. In *Proceedings of the 1st international conference on Artificial intelligence and law*, pages 60–66.

Swarnadeep Saha et al. 2018. Open information extraction from conjunctive sentences. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2288–2299.

Frank Schilder. 2002. Robust discourse parsing via discourse markers, topicality and position. *Natural Language Engineering*, 8(2-3):235–255.

Michael Schmitz, Stephen Soderland, Robert Bart, Oren Etzioni, et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 523–534.

Florian Schneider, Tim Fischer, Fynn Petersen-Frey, Isabel Eiser, Gertraud Koch, and Chris Biemann. 2023. The D-WISE tool suite: Multi-modal machine-learning-powered tools supporting and enhancing digital discourse analysis. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 328–335, Toronto, Canada. Association for Computational Linguistics.

Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 228–235.

Francesco Sovrano, Monica Palmirani, Salvatore Sapienza, and Vittoria Pistone. 2024. Discolqa: zero-shot discourse-based legal question answering on european legislation. *Artificial Intelligence and Law*, pages 1–37.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 885–895.

Ravikiran Vadlapudi, Poornima Malepati, and Suman Yelati. 2009. Hierarchical discourse parsing based on similarity metrics. In *Proceedings of the Student Research Workshop*, pages 89–93.

Alexander Yates, Michele Banko, Matthew Broadhead, Michael J Cafarella, Oren Etzioni, and Stephen Soderland. 2007. Textrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 25–26.

Yasuhisa Yoshida, Jun Suzuki, Tsutomu Hirao, and Masaaki Nagata. 2014. Dependency-based discourse parser for single-document summarization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1834–1839.

Jingting Yuan, Qiuhan Lin, and John SY Lee. 2021. Discourse tree structure and dependency distance in efl writing. In *20th International Workshop on Treebanks and Linguistic Theories (TLT 2021)*, pages 105–115. Association for Computational Linguistics (ACL).

Ashwini V Zadgaonkar and Avinash J Agrawal. 2021. An overview of information extraction techniques for legal document analysis and processing. *International Journal of Electrical & Computer Engineering (2088-8708)*, 11(6).

Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2020. How does nlp benefit legal system: A summary of legal artificial intelligence. *arXiv preprint arXiv:2004.12158*.

## A Appendix

### A.1 Graphene Relations used for LeGen training

1. **SPATIAL**: This relation is used to denote the place of occurrence of an event.

   E.g., The Interstate Migrant Workmen Act 's purpose was to protect workers whose services were requisitioned outside their native states in India .

   `SUB/ELABORATION('The Inter-state Migrant Workmen Act 's purpose was to protect workers .', **SUB/SPATIAL**('This is in India .','Workers 's services are requisitioned outside their native states .'))`

2. **ATTRIBUTION**: This relation is used when a statement is being made by some person or institution.

   Eg: But some militant SCI TV junk-holders say that 's not enough .

   `SUB/ATTRIBUTION('This is what some militant SCI TV junk-holders say .',"s not enough .')`

3. **CONTRAST**: This relation is indicated by the words 'although' , 'but' , 'but now', 'despite' , 'even though' , 'even when', 'except when' , 'however', 'instead' , 'rather', 'still' ,

‘though’ , ‘thus’, ‘until recently’, ‘while’ and ‘yet’.

Eg: This can have its purposes at times , but there ’s no reason to cloud the importance and allure of Western concepts of freedom and justice .

`CO/CONTRAST(SUB/ELABORATION(‘This is at times .’,‘This can have its purposes .’ ), ‘There ’s no reason to cloud the importance and allure of Western concepts of freedom and justice .’)`

Eg2: No one has worked out the players ’ average age , but most appear to be in their late 30s .
`CO/CONTRAST(‘No one has worked out the players ’ average age .’,‘ most appear to be in their late 30s . ’)`

4. **LIST**: This is used to indicate conjunctions ( ’and’ or comma seperated words) between the sentences

   Eg: He believes in what he plays , **and** he plays superbly .
   `CO/LIST(‘He believes in what he plays .’,‘He plays superbly .’)`

5. **DISJUNCTION**: This is used to show the presence of ’OR’ in the sentences.

   Eg: The carpet division had 1988 sales of $ 368.3 million, or almost 14 % of Armstrong ’s $ 2.68 billion total revenue .

   `CO/DISJUNCTION(‘The carpet division had 1988 sales of $ 368.3 million .’,‘The carpet division had 1988 sales of almost 14 % of Armstrong ’s $ 2.68 billion total revenue .’)`

6. **CAUSE**: Indicates the presence of the word - ‘because’ or ‘since’.

   Eg: Jaguar ’s own defenses against a hostile bid are weakened , analysts add , because fewer than 3 % of its shares are owned by employees and management .

   `SUB/CAUSE(‘Jaguar ’s own defenses against a hostile bid are weakened , analysts add .’,‘Fewer than 3 % of its shares are owned by employees and management .’)`

7. **CONDITION**: When multiple sentences are connected by phrase ’if’ ‘in case’,‘unless’ and ’until’, CONDITION relationship phrase is used to denote the connection between the sentences.

   Eg: Unless he closes the gap , Republicans risk losing not only the governorship but also the assembly next month .

   `SUB/CONDITION(‘He closes the gap .’,‘Republicans risk losing not only the governorship but also the assembly next month .’)`

8. **ELABORATION**: Identified by the presence of words such as ‘more provocatively’,‘even before’ ,‘ for example’,‘recently’,‘ so’ ,‘so far’ ,‘ where’ ,‘whereby’ and ‘whether’ .

   REGEX:

   ```
   `since(\\W(.*?\\W)?)now"
   ```

   Eg: Not one thing in the house is **where** it is supposed to be, but the structure is fine.

   `CO/CONTRAST(SUB/ELABORATION(‘Not one thing in the house is .’,‘It is supposed to be .’ ), ‘The structure is fine .’)`

9. **TEMPORAL**: Denotes the time or date of occurrence of the event.

   Eg: These days he hustles to house-painting jobs in his Chevy pickup before and after training with the Tropics .

   `SUB/TEMPORAL(‘These days he hustles to house-painting jobs in his Chevy pickup before and after .’,‘These days he is training with the Tropics .’)`

10. **PURPOSE**: This kind of relation is identified by the presence of words such as ‘for’ or ‘to’.

    Eg: But we can think of many reasons to stay out for the foreseeable future and well beyond .

    `SUB/PURPOSE(‘But we can think of many reasons .’,‘This is to stay out for the foreseeable future and well beyond .’)`

## A.2 Precision, Recall, and F1 score computation

$$p = precision(G_i, T_j) = \frac{|G_i \cap T_i|}{|T_i|} \quad (1)$$

$$r = recall(G_i, T_j) = \frac{|G_i \cap T_i|}{|G_i|} \quad (2)$$

$$f1(G_i, T_j) = \frac{2pr}{p + r} \quad (3)$$

Let $m(.)$ be matching function such that $G_i$ matches with $T_{m(i)}$ and conversely $G_{m(j)}$ matches with $T_j$. If $|G| \neq |T|$, then only $k = min(|G|, |T|)$ matches are possible. Thus in such cases, $m(i)$ will not return valid value for all $i$ and $precision(G_i, T_{m(i)})$ and $recall(G_i, T_{m(i)})$ will be zero.

$$\begin{aligned} p_{example} &= precision(G, T) \\ &= \frac{1}{|T|} \sum_{i=1}^{|T|} precision(G_{m(i)}, T_i) \end{aligned} \quad (4)$$

$$\begin{aligned} r_{example} &= recall(G, T) \\ &= \frac{1}{|G|} \sum_{i=1}^{|G|} precision(G_i, T_{m(i)}) \end{aligned} \quad (5)$$

$$f1_{example} = (G, T) = \frac{2 p_{example} r_{example}}{p_{example} + r_{example}} \quad (6)$$

Please note that (4) to (6) represent scores for only one example in the test set.

## A.3 Multiple correct trees for same sentence

Eg: The Code on Wages Bill was introduced in the Lok Sabha on 10 August 2017 by the Minister of State for Labour and Employment ( Independent Charge), Santosh Gangwar.

Tree1: `SUB/ELABORATION('This was by the Minister of State for Labour and Employment ( Independent Charge ), Santosh Gangwar', SUB/TEMPORAL('The Code on Wages Bill was introduced in the Lok Sabha', 'This was on 10th August 2017'))`

Tree2: `SUB/TEMPORAL( 'This was on 10th August 2017', SUB/ELABORATION('This was by the Minister of State for Labour and Employment ( Independent Charge ), Santosh Gangwar', 'The Code on Wages Bill was introduced in the Lok Sabha', 'This was on 10th August 2017'))`

## A.4 Error Analysis

We manually analyzed the outcomes of subordination as predicted by the T5 Base and BART Base models. The primary causes of errors are identified as follows:

1. Clauses not correctly identified by model: We observed that the T5 model failed to correctly identify clauses 16% of the time, and the BART model, experiencing similar challenges, had a 17% failure rate. Moreover, BART occasionally not only failed to recognize clauses but also exhibited hallucinations during these instances.

2. Wrong Relation or relation not identified at all: We observed that the T5 model fails to identify the correct relation, defaulting to `ELABORATION`, 0.018% of the time. We found one example in T5 where the model exhibited hallucination as well as generated wrong clauses. Similarly, BART also struggles to identify the relation in 0.04% of cases and tends to exhibit more instances of hallucination compared to the T5 model.

3. Both Clauses and Relation are wrong: T5 encountered challenges in identifying both relations and clauses in 0.018% of cases, whereas BART faced failures 0.03% of the time and demonstrated a higher frequency of hallucinations.

4. Not split the sentences: T5 and BART experienced difficulty in sentence splitting in 0.07% of instances.

5. Model repeats the original input sentence in the split and Hallucination: T5 encountered challenges in both sentence splitting and hallucination 0.06% times, whereas BART exhibited a higher rate of hallucination and failed to split 0.14% of the time.

6. Grammatical error: We found minimal grammatical errors in the hierarchical sentence structure, such as bracket mismatches and misspellings. T5 made a grammatical mistake only once, while BART made two grammatical errors.

In summary, we noticed that BART exhibited a higher frequency of hallucinations compared to T5. This occurred particularly when BART struggled to identify both clauses and relations within the input sentence.

## A.5 Relation count in Indian Legal Dataset

Table 5 shows relation distribution in the test dataset and the accuracy of prediction by T5.

| Relation | Count | T5 BASE ACCURACY OF RELATION PREDICTION |
|---|---|---|
| SPATIAL | 10 | 0.2 |
| ATTRIBUTION | 18 | 0.44 |
| ELABORATION | 446 | 0.18 |
| TEMPORAL | 3 | 0.67 |
| CONTRAST | 23 | 0.69 |
| LIST | 112 | 0.3 |
| DISJUNCTION | 26 | 0.15 |
| CAUSE | 5 | 0.08 |
| CONDITION | 18 | 0.72 |
| PURPOSE | 18 | 0.27 |

Table 5: Relation distribution in Indian Legal Test data

### A.6 Zero Shot Restricted CoT Prompt for Subordination Task

*To construct the discourse Tree, follow the below steps*:

*Step 1: Identify the subordinating phrases like for, while, however, because, as, etc., in the input sentence and then divide it into two sentences (clauses) by identifying the relation between them. If there are no subordinating phrases, identify coordinating phrases and create two independent clauses by identifying the relation between them. Make sure clauses are complete by adding terms like 'This was' at the beginning of the clauses for incomplete clauses. Relation includes ELABORATION, SPATIAL, CONTRAST, CONDITION, SPATIAL, ATTRIBUTION, DISJUNCTION, LIST, CAUSE, TEMPORAL and PURPOSE. Use only the above relations.*

*Step 2: For each of the clauses identified in Step 1, identify a subordinating phrase in each of the clauses and repeat Step 1.*

*Step 3: If there are no subordinating phrases in clauses identified in step 1, identify coordinating phrases like and, or,  and but and repeat step 1*

*Step 4: Repeat steps 1 to 3 till all the subordinating and coordinating phrases in the individual clauses are identified.*

*Step 5: If there are no subordinating or coordinating phrases in the input sentence, then the output will be the same as the input sentence. Else, output the discourse tree in the format*: *The Discourse Tree: "Relation('Clause1', Relation ('Clause.', 'Clause.') 'Clause2' ...)"*

### A.7 Zero Shot Unrestricted CoT Prompt for Subordination Task

*To construct the discourse Tree, follow the below steps:*

*Step 1: Identify the subordinating phrases like 'for', 'while', however, because, as, etc., in the input sentence and then divide it into two sentences (clauses) by identifying the relation between them. If there are no subordinating phrases, identify coordinating phrases and create two independent clauses by identifying the relation between them. Make sure clauses are complete by adding terms like 'This was' at the beginning of the clauses for incomplete clauses.*

*Step 2: For each of the clauses identified in Step 1, identify a subordinating phrase in each of the clauses and repeat Step 1.*

*Step 3: If there are no subordinating phrases in clauses identified in step 1, identify coordinating phrases like and, or,  and but and repeat step 1*

*Step 4: Repeat steps 1 to 3 till all the subordinating and coordinating phrases in the individual clauses are identified.*

*Step 5: If there are no subordinating or coordinating phrases in the input sentence, then the output will be the same as the input sentence. Else, output the discourse tree in the format: The Discourse Tree: "Relation ('Clause1', Relation ('Clause', 'Clause') 'Clause2' ...)"*

### A.8 11 Shot Prompt used for Subordination Task

*Following are a few examples of legal input sentences under 'Input' that have been converted into discourse trees, which are shown under 'Output'. using the following examples as a format, convert new legal sentences into trees. Create a discourse tree from the provided sentence without introducing new words or explanations. A discourse tree identifies hierarchical text structures and rhetorical relations between text parts. These relations are categorized as coordination and subordination. below are some examples of how the discourse tree should be generated -*

1. ***SPATIAL**: This relation is used to denote the place of occurrence of an event .*
   *Eg: The Interstate Migrant Workmen Act 's purpose was to protect workers whose services are requisitioned outside their native states in India .*

SUB/ELABORATION('The Inter-state Migrant Workmen Act 's purpose was to protect workers .', **SUB/SPATIAL**('This is in India .','Workers 's services are requisitioned outside their native states .'))

2. ***ATTRIBUTION**: This relation is used when a statement is being made by some person or institution.*
   *Eg: But some militant SCI TV junk-holders say that 's not enough .*
   SUB/ATTRIBUTION('This is what some militant SCI TV junk-holders say .',"s not enough .')

3. ***CONTRAST**: This relation is indicated by the words 'although' , 'but' ,'but now', 'despite' , 'even though' , 'even when', 'except when' , 'however', 'instead' , 'rather", 'still' , 'though' , 'thus', 'until recently', 'while' and 'yet'.*
   *Eg: This can have its purposes at times , but there 's no reason to cloud the importance and allure of Western concepts of freedom and justice.*
   CO/CONTRAST(SUB/ELABORATION('This is at times .','This can have its purposes .' ), 'There 's no reason to cloud the importance and allure of Western concepts of freedom and justice .')

4. ***LIST** : This is used to indicate conjunctions ( 'and' or comma seperated words) between the sentences*
   *Eg: He believes in what he plays , **and** he plays superbly .*
   CO/LIST('He believes in what he plays .','He plays superbly .')

5. ***DISJUNCTION**: This is used to show the presence of 'OR' in the sentences.*
   *Eg: The carpet division had 1988 sales of $ 368.3 million , or almost 14 % of Armstrong 's $ 2.68 billion total revenue .*
   CO/DISJUNCTION('The carpet division had 1988 sales of $ 368.3 million .','The carpet division had 1988 sales of almost 14 % of Armstrong 's $ 2.68 billion total revenue .')

6. ***CAUSE**: Indicates the presence of the word - 'because' or 'since'.*
   *Eg: Jaguar 's own defenses against a hostile bid are weakened , analysts add , because fewer than 3 % of its shares are owned by employees and management .*
   SUB/CAUSE('Jaguar 's own defenses against a hostile bid are weakened , analysts add .','Fewer than 3 % of its shares are owned by employees and management .')

7. ***CONDITION**: When multiple sentences are connected by phrase 'if' 'in case','unless' and 'until', CONDITION relationship phrase is used to denote the connection between the sentences.*
   *Eg: Unless he closes the gap , Republicans risk losing not only the governorship but also the assembly next month .*
   SUB/CONDITION('He closes the gap .','Republicans risk losing not only the governorship but also the assembly next month .')

8. ***ELABORATION**: Identified by the presence of words such as 'more provocatively','even before' ,'for example','recently' ,'so' ,'so far' ,' where' ,'whereby' and 'whether' .*

   REGEX:

   ```
   `since(\\W(.*?\\W)?)now"
   ```

   *Eg: Not one thing in the house is **where** it is supposed to be , but the structure is fine .*
   CO/CONTRAST(SUB/ELABORATION('Not one thing in the house is .','It is supposed to be .' ), 'The structure is fine .')

9. ***TEMPORAL** : Denotes the time or date of occurrence of the event.*
   *Eg: These days he hustles to house-painting jobs in his Chevy pickup before and after training with the Tropics .*
   SUB/TEMPORAL('These days he hustles to house-painting jobs in his Chevy pickup before and after .','These days he is training with the Tropics .')

10. ***PURPOSE**: This kind of relation is identified by the presence of words such as 'for' or 'to'.*
    *Eg: But we can think of many reasons to stay*

*out for the foreseeable future and well beyond .*
`SUB/PURPOSE('But we can think of many reasons .','This is to stay out for the foreseeable future and well beyond .')`

11. ***NONE**: This kind of relation is given if the sentence does not contain any subordinates or coordinates.*
*Eg: Or was it because Ms. Collins had gone?*
`NONE`

### A.9 11 Shot Prompt used for Co-ordination Task

*Coordination is a frequently occurring syntactic structure along with several phrases, known as conjuncts. The task of coordination disambiguation is identifying the boundaries of each conjunct with a single coordinator word as one coordinate structure instance. Given a coordinator word (e.g., 'and', 'or' or 'but'), a system must return each conjunct span if the word actually plays the role of a coordinator; otherwise, NONE is output for the absence of coordination. Following this, 11 examples are provided.*

### A.10 Zero Shot CoT Prompt for Coordination Task

*Coordinating sentences join independent clauses with coordinating conjunctions like 'and', 'or', and 'but', enhancing sentence complexity. Your task is to form independent clauses by identifying the coordinating phrases. To construct the hierarchical tree, follow the below steps:*

*Step 1: Identify the coordinating phrase like and, or and but. Sometimes a sentence can have comma as well to distinguish between different words. Consider that as well while forming independent sentences.*
*Step 2: Join all the dependent phrases of the coordinating phrase to make an independent sentence. Independant phrases should contains subject, object and a verb.*
*Step 3: Loop over all the clauses from step 2, and if there are still coordinating phrases present, repeat steps 1 and step 2 till all the coordinating phrases are identified in the input sentence. The clauses should be completely independent.*
*Step 4: Repeat steps 1 to 3 till all the subordinating and coordinating phrases in the individual clauses are identified.*
*Step 5: Do not print the results of intermediate steps; print only the final output. If there are no coordinating phrases in the input sentence, the output will be NONE. Else, the output of the hierarchical tree in the format : "COORDINATION('Clause1', Clause2' , COORDINATION('clause', 'clause2').......)"*