LeGen: End-to-end Legal Information Extraction using Generative Models

Anonymous ACL submission

Abstract

Despite the rapid growth in access to digital devices, the new users of the devices, especially in developing countries like India, are not able to access information on their rights and entitlements, jobs and livelihood, healthcare, education, etc. as the information is in the form of very long, complex sentences and heavy in legal parlance. Open information extraction techniques can be used to convert unstructured legal text into triples of the form ⟨`subject, relation, object`⟩ in a domain-independent manner. However, the legal text is long and complex which calls for extracting structure beyond triples, also called complex information extraction. This paper proposes a generative approach to perform complex information extraction from legal statements. We achieve this by encoding legal statements as trees to capture their complex structure and semantics. This end-to-end modelling reduces the propagation of errors across complicated pipelines. We experimented with multiple generative architectures to conclude that our proposed approach reports up to 14.7 % gain on an Indian Legal benchmark and is competitive on open information extraction benchmarks.

1 Introduction

The proliferation of smartphones and computing devices is evident, with a reported 71% smartphone penetration in 2023 (Sun, 2023; Gupta et al., 2022). Despite this, the Next Billion Users, new adopters of digital technology, struggle to utilize these 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).

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 unable 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 2 triples - i) ⟨`it, will be treated, as a strike`⟩, ii) ⟨`over 50 percent of a company's workers, take concerted, casual leave`⟩. The model fails to identify that a `condition` connects the two extractions. 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 interpret complex legal information.

The problem of extracting structure beyond triples is handled by a relatively new area of research known as complex information extraction (Mahouachi and Suchanek, 2020). However, most of these techniques (Niklaus et al., 2019; Prasojo et al., 2018) involve multiple-step pipelines for identifying clauses and relationships between them that propagate errors. They also lack language understanding and generalization capabilities. There are numerous applications for complex information extraction, including i)generating awareness among the general population, particularly those with limited comprehension of legal language, especially following the repercussions of COVID-19 in India. This extraction could be helpful in various downstream tasks like Question Answering System with voice support. ii) Could also be used by legal professionals for court judgment prediction explanation if the legal information is stored in a knowledge base in the form of discourse trees.

This paper proposes $LeGen$, an end-to-end gen-

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 2) The nominee has to prove the identity to claim the amount 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

erative approach for complex information extraction from legal sentences. 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. We have used T5 and BART to understand Legal text rather than advanced large language models like Open AI and Llama as they are computationally extensive and proprietary-owned. So, we trained on smaller models, which are privacy-friendly. By encoding legal sentences as a discourse tree (Niklaus et al., 2019), (Section 4.1), BART and T5 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. Our salient contributions are:

1. We employ open-domain information extraction techniques on Indian legal sentences to enhance their accessibility to the general public. We propose utilizing techniques for extracting complex information from legal statements.

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

3. We release a new benchmark for legal information extraction, curated from Indian Law statements

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

5. We show $LeGen$'s flexibility by training it as an OIE task, and conclude that it is competitive on an OIE benchmark.

Our paper is organized as follows. In Section 2, we discuss work related to legal, complex, and open information extraction. We formally describe the problem in Section 3 and introduce $LeGen$ in Section 4. We discuss our experiments and results in Section 5 and 6 and discuss future work in Section 7. The limitations of our approach are described in Section 8. Additional details and experiments are listed in the Appendix (Section A).

2 Related Work

2.1 Legal Information Extraction

Legal Information Extraction has evolved rapidly, requiring NLP techniques to aid legal professionals (Chalkidis et al., 2017; Leivaditi et al., 2020; Cardellino et al., 2017). In (Zadgaonkar and Agrawal, 2021), authors review open information techniques to extract structured triples from legal statements. This still suffers from the issues pointed out in Section 1. In (Mistica et al., 2020), authors classify sentences into three labels: facts, reasoning, and conclusion while we focus on extracting information (discourse trees) from individual legal sentences.

Numerous systems, including Eunomos (Boella et al., 2016; Abood and Feltenberger, 2018; Nguyen et al., 2018), have been developed to simplify and streamline legal tasks, employing a variety of machine learning techniques and recurrent neural network architectures. Examples of tasks covered in legal information extraction include named entity recognition, document summarization, document structure extraction, or judgement prediction and explanation. Dozzier pioneered legal NER using rule-based methods (Dozier et al., 2010). (Cardellino et al., 2017) enhanced NER task with a legal ontology mapped to YAGO. Recent advancements, like pre-trained language models and prompt-based learning, outperformed rule-based systems for NER (Liu et al., 2023).

In court judgment prediction, systems like

HYPO (Rissland and Ashley, 1987) and CATO (Aleven and Ashley, 1995) provided arguments without definitive evaluations. Rule-based systems, as discussed by (Sergot et al., 1986), offered outcomes and reasoning. IBP (Bruninghaus and Ashley, 2003) integrated CATO-like techniques for outcome prediction. Early ML approaches, like those by (Pannu, 1995), utilized neural networks and genetic algorithms. (Aletras et al., 2016) achieved 79% accuracy on ECHR decisions with SVMs. Subsequent studies explored ML in this domain (Medvedeva et al., 2020; Chalkidis et al., 2019a; SAYS and Judgement, 2020; Kaur and Bozic, 2019; Medvedeva et al., 2023). (Branting et al., 2021) introduced semi-supervised case annotation to explain AI-predicted judgments.

Pre-training models for legal domain adaptation has also been a popular direction of research. Researchers introduced LegalBERT (Chalkidis et al., 2020), which is BERT (Kenton and Toutanova, 2019) pre-trained on 12 GB of diverse English legal text from legislation, court cases and contracts. It was evaluated on three legal datasets (EURLEX57, ECHR Cases, and CONTRACTS NER). Several datasets are made available in various languages for various legal NLP tasks (Chalkidis et al., 2021, 2019b,a; Yao et al., 2022; Zheng et al., 2021).

In the Indian context, Paul et al (Paul et al., 2023) retrain two existing legal pre-trained Language Models, namely LegalBERT and CaseLawBert (BASELINE), on Indian Legal data, renaming them InLegalBert and InCaseLawBert evaluating their model on both Indian and Non-Indian datasets using the perplexity score metric. (Malik et al., 2021) introduce a large corpus, named ILDC, which consists of 35k Indian Supreme Court cases in the English language annotated with original court decisions. The SemEval task (Modi et al., 2023) introduced three problems to be tackled on the ILDC corpus (Malik et al., 2021). $-\ i)$ legal named entity recognition (Kalamkar et al., 2022a) performs named entity recognition on the ILDC corpus, $ii)$ rhetorical role prediction structures legal transcripts into rhetorical roles (Kalamkar et al., 2022b) and $iii)$ court case judgment prediction proposes using AI-based techniques to automate course case judgments. Based on ILDC, Malik et al., propose the task of Court Judgement Prediction and Explanation, where an automated system predicts and explains the outcomes of legal cases (Malik et al., 2021).

Kapoor et al. (Kapoor et al., 2022) present the Hindi Legal Documents Corpus (HLDC), containing over 900k Hindi legal documents, for downstream applications. They demonstrate a bail prediction use case, experimenting with Doc2Vec, IndicBert, and a Multi-Task Learning (MTL) approach. Kalamkar et al. (Kalamkar et al., 2021), in their research work, highlight the need for an NLP benchmark on Indian Legal text as it is entirely different from other countries' legal text. Cui et al. (Cui et al., 2023), survey LJP tasks, evaluating 31 datasets and SOTA models over multiple tasks.

2.2 Open Information Extraction

Open Information Extraction uses an independent paradigm to extract the information as a triple, $\langle$`subject, relation, object`$\rangle$. (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). Moving from rule-based system, we have 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 sentences. It combines the results from the three models (coordination model, OIE model, and Allennlp models) to generate triples from input sentences.

2.3 Complex Information Extraction

Many OIE systems have been developed which cater to identifying triples in a complex sentence (Mahouachi and Suchanek, 2020) like OLLIE (Schmitz et al., 2012), MinIE (Gashteovski et al., 2017), ClausIE (Del Corro and Gemulla, 2013), StuffIE (Prasojo et al., 2018) and Graphene (Cetto et al., 2018).

ClausIE, MinIE, and OLLIE use a linguistic-based approach to information extraction. OLLIE open information system uses a set of pre-defined templates and rules to identify the relation present in the sentence. MinIE also uses a linguistic approach to extract information with a difference that enhances the output by adding other semantic information like polarity, modality, attribution, and quantities. StuffIE (Prasojo et al., 2018), another open information system that aims to extract complex information which is referred to as facets in this work, uses syntactical dependency to tag facets or relations in the sentence. Graphene (Niklaus et al., 2019) uses 39 handcrafted rules to construct a discourse tree and then obtain the triples from the sub-sentences of the input sentences. These techniques are either rule-based or use a pipeline

of techniques to extract the structure of a complex sentence. To the best of our knowledge, ours is the first attempt at using generative neural architectures to model complex information extraction.

3 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 $COMP$

4 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.

4.1 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 coordinations and subordinations.

Coordinating sentences join independent clauses with coordinating conjunctions like 'and', 'or', and 'but', enhancing sentence complexity. Subordination sentences combine main clauses with dependent clauses, providing additional information or context 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 1 (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.

4.2 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 didn't address iii), but we show in Section 6 that $LeGen$ is robust to these issues. The final list of relations that were kept is in the Appendix (Section A).

4.3 Encoding of Discourse Tree

Figure 1 demonstrates the conversion of a discourse tree into a sequence encoding, simplifying complex information extraction. We treat this process as a language translation task, where the output language is the tree encoding. Teacher forcing, employed during training, influences the generated text based on input pairs from two languages. The encoder processes text in one language, while the decoder predicts the next token for each position in the other language. Our method converts origi-

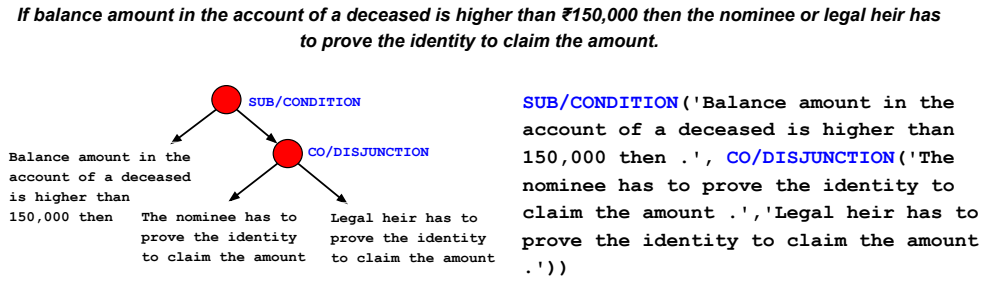

Figure 1: Discourse tree for an example law sentence (on the left). Corresponding linear encoding of the Discourse tree (on the right). SUB and CO refer to subordination and coordination, respectively.

nal input sentences, including clauses and relationships, into explicit discourse trees. We encode the discourse tree by doing a pre-order traversal of the tree. Algorithm 1 discusses our steps.

4.4 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.

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

where T denotes the discourse tree, $Reg\ Loss$ refers to regular cross entropy loss, $n(T)$ denotes number of leaf nodes in T, $u(T)$ denotes number of unique leaf nodes, and λ 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 λ as $u(t) << n(t)$.

5 Experiments

5.1 Datasets

5.1.1 Training

We trained $LeGen$ using 17k sentences from Penn Tree Bank (Marcus et al., 1993) dataset. We perform our experiments on 32x2 cores AMD EPYC 7532, 1 TB of memory, and 8x A100 SXM4 80GB GPU systems. We train 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 2 tasks. For both of them, we also trained the model with custom loss function, setting $\lambda = 1$.

Task 1: Identifying Sub-ordinations and Co-ordinations. We encoded every sentence into a discourse tree structure as described in Section 4. 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 6).

Task 2: Identifying Co-ordinations. In order to test $LeGen$'s flexibility, we also separately trained it as a coordinate boundary detection task (Saha et al., 2018). The purpose of this study was to test the competency of generative models in splitting sentences over state-of-the-art non-generative techniques like OpenIE6. We converted the OpenIE6 coordinate boundary labels into a discourse tree. The non-leaf nodes in this tree represented only the coordination relation. We kept the same hyperparameters that we used for the subordination task and obtained the best results for batch size 3. Results are averaged over 3 seeds (Section 6).

5.1.2 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 'Labour Law' category from

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

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.4. 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 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 containing 985 sentences was used to evaluate $LeGen$'s flexibility in identifying co-ordinations.

5.2 Metrics

5.2.1 Metrics for Task 1

While discourse trees have been used to improve downstream tasks such as text classification (Ferracane et al., 2019) or open information extraction (Niklaus et al., 2019), we are unaware of any metric used to evaluate them directly. It was noted that a single sentence could have multiple correct tree representations, particularly evident for taller trees as illustrated in Section A.4 (Appendix). So, 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 used 2 annotators to compute these 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 3 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.

Usability Evaluation. We conducted user evaluations with 8 PhD scholars to determine whether hierarchically separating sentences helps them understand legal text better than simply reading the legal sentence. 5 out of 8 students were from Computer Science (CS) and 3 were from non-CS backgrounds. We built a user interface using Flutter and a custom wrapper to visualize the tree. Users could enter the sentence and it would return its tree representation (screenshot shown in Figure 3 in Appendix (Section A.7)). We presented to them 8-10 sentences of reasonable complexity (from both the datasets) and their discourse trees. They were asked questions related to the ease and time needed to interpret legal sentence with/without the tree visualization. More details about the questions asked are in Appendix (Section A.7 of A).

5.2.2 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.3 of appendix A.

5.3 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 as the competing technique for Task 1.

OpenIE6. We used the Coordination Boundary Detection Model released with OpenIE6 as our baseline for Task 2.

6 Results

6.1 Task 1

Table 2 shows the **TSE**, **TCE**, and the number of clauses and relations generated in the discourse trees by each of these 3 techniques. It is clear that the generative approach for discourse tree creation outperforms Graphene. T5-Base performs the best and beats Graphene by 9 pts with a TSE score of 71%. 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 4.2. Graphene also underperforms on 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* (Table 3). Graphene also cannot identify nondistributive 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. 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. The results in Table 2 indicate that T5 outperforms Graphene, suggesting LeGen's potential for enhanced understanding of laws and legal transcripts.

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).

Results of User Study. 6 out of 8 users reported it was not easy to read legal text without a hierarchical representation. When it came to using the visualization tool, 7 out of 8 users felt it easier to use the visualization rather than reading the predictions produced. 6 out of 8 users felt the hierarchical representation helped them simplify long complex sentences and reduce interpretation time while the remaining did not report any substantial gain in understanding through the tool. 7 out of 8 users stated they would highly recommend new users to check the hierarchical representation rather than reading the encoding to understand the legal text. From this study, we can conclude that our tree based representation of legal sentences is useful towards their interpretation by non-legal professionals.

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)
Indian Legal Dataset	Graphene Default	0.62	0.54	(247, 347)
	Graphene	0.62	0.92	(247, 347)
	T5	**0.71**	0.96	(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)

Table 2: TSE and TCE results of Graphene, 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.

Input	Clauses generated by Graphene	Clauses generated by T5 BASE
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 2) This was to each employee 3) The Factories leave each year sick days 4) Act 1948 mandate 15 working days of fully paid vacation The Factories 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 2) The Factories Act 1948 mandate 15 working days of fully paid vacation leave each year 3) The Shops and Establishment Act 1960 mandate 15 working days of fully paid vacation leave each year.

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

6.2 Task 2

Table 4 shows our results. We obtained competent results from the T5-base against OpenIE6. 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

Model		OpenIE	T5 Small		T5 base		BART Small		BART Base	
	Loss Function	Regular	Custom	Regular	Custom	Regular	Custom	Regular	Custom	Regular
Mapping based approach	Precision	**0.9803**	0.9671	0.9647	0.9756	0.9747	0.8273	0.8215	0.8418	0.8369
	Recall	**0.9845**	0.9538	0.9544	0.974	0.973	0.7334	0.7391	0.7613	0.7574
	F1-score	**0.9816**	0.9578	0.9571	0.9739	0.9726	0.7672	0.7682	0.7903	0.7859

Table 4: Mapping-based approach is used to calculate precision, recall and f1 score using cross-entropy loss function and custom loss function

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.

6.3 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 2). 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 the 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 generated 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.

6.4 Ablation study

6.4.1 Models trained with a subset of data

To understand the effect of sentences with varying heights of discourse trees in the training set, we trained models with different partitions of training set. We denote `Level_n` as the group of sentences with height n. We then created 3 partitions – `P1`, `P2`, and `P3`. `P1` consisted of 50% of `Level_0` sentences (selected randomly) and all `Level_1` sentences, `P2` consisted of the remaining 50% of `Level_0` and `Level_2` and above sentences, and `P3` consisted of all `Level_0` and `Level_1` sentences. We trained the models with these 3 partitions. The results of this experiment are presented in Table 5. We observed that the models were not able to generalize with `P1` or `P2` but the performance improved substantially with `P3`. This indicates that even in the absence of trees with greater height (`Level_2` and above) in the training set, the model can generalize well.

6.4.2 Models trained with different types of fine tuning

We fine-tuned our models T5 and BART base by freezing the decoder, freezing the encoder and standard fine-tuning where both encoder and decoder are fine-tuned. All the models are fine-tuned for 30 epochs and batch size 3. The results of this experiment are in the Table 5. The model performs the best with both the decoder and encoder. We also observed that the time taken to fine-tune did not reduce with fine-tuning either encoder and decoder and to obtain competitive results with just encoder and decoder is computationally intensive and time-consuming.

Task	Mapping Based	Metric	T5 Base	BART Base
Partitioned Dataset	P1	Precision	0.5897	**0.5953**
		Recall	0.4414	**0.4467**
		F1 Score	0.4931	**0.4984**
	P2	Precision	0.5512	**0.5363**
		Recall	**0.4437**	0.4352
		F1 Score	**0.4814**	0.4706
	P3	Precision	**0.9551**	0.8494
		Recall	**0.9642**	0.7648
		F1 Score	**0.9567**	0.7946
Type of Fine tunning	Freeze Decoder	Precision	**0.9658**	0.9041
		Recall	**0.9574**	0.6769
		F1 Score	**0.959**	0.7522
	Freeze Encoder	Precision	**0.9447**	0.7179
		Recall	**0.9306**	0.6279
		F1 Score	**0.9343**	0.66
	Standard Fine Tuning	Precision	**0.9733**	0.8324
		Recall	**0.9795**	0.7655
		F1 score	**0.9762**	0.7899

Table 5: Ablation study: Mapping based on scores on T5 and BART over two subsets of data and different types of fine-tuning

7 Conclusion

We proposed an end-to-end generative legal information extraction technique modelled as complex information extraction that can improve the understanding of long and complex legal sentences.We learned sentence discourse trees using T5 and BART models. We outperformed Graphene, a state-of-the-art complex information extraction technique on an Indian Legal Benchmark, and achieved competitive results on the task of the coordinate boundary detection technique. We plan to extend the generative-based complex information extraction for rhetorical role prediction and extend support for Indian languages.

8 Limitations

- Our dataset could be biased as it does not contain an equal distribution of training instances for each kind of relation.
- Additionally, our study's limitation lies in the varying numbers of clauses and relations generated for the same input sentence.
- Generative models are prone to hallucinations
- Due to the presence of multiple correct discourse trees for subordination task, 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.

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.

Stefanie Bruninghaus and Kevin D Ashley. 2003. Predicting outcomes of case based legal arguments. In *Proceedings of the 9th international conference on Artificial intelligence and law*, pages 233–242.

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*.

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.

Luciano Del Corro and Rainer Gemulla. 2013. Clausie: clause-based open information extraction. In *Proceedings of the 22nd international conference on World Wide Web*, pages 355–366.

Christopher Dozier, Ravikumar Kondadadi, Marc Light, Arun Vachher, Sriharsha Veeramachaneni, and Ramdev Wudali. 2010. *Named entity recognition and resolution in legal text*. Springer.

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.

Elisa Ferracane, Greg Durrett, Junyi Jessy Li, and Katrin Erk. 2019. Evaluating discourse in structured text representations. *CoRR*, abs/1906.01472.

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/.

Meghna Gupta, Devansh Mehta, Anandita Punj, and Indrani Medhi Thies. 2022. Sophistication with limitation: Understanding smartphone usage by emergent users in india. In *ACM SIGCAS/SIGCHI Conference on Computing and Sustainable Societies (COMPASS)*, pages 386–400.

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. 2022a. Named entity recognition in indian court judgments.

Prathamesh Kalamkar, Aman Tiwari, Astha Agarwal, Saurabh Karn, Smita Gupta, Vivek Raghavan, and Ashutosh Modi. 2022b. Corpus for automatic structuring of legal documents. *arXiv preprint arXiv:2201.13125*.

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*.

Arshdeep Kaur and Bojan Bozic. 2019. Convolutional neural network-based automatic prediction of judgments of the european court of human rights. In *AICS*, pages 458–469.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2.

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*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

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.

Masha Medvedeva, Martijn Wieling, and Michel Vols. 2023. Rethinking the field of automatic prediction of court decisions. *Artificial Intelligence and Law*, 31(1):195–212.

Meladel Mistica, Geordie Z Zhang, Hui Chia, Kabir Manandhar Shrestha, Rohit Kumar Gupta, Saket Khandelwal, Jeannie Paterson, Timothy Baldwin, and Daniel Beck. 2020. Information extraction from legal documents: A study in the context of common law court judgements. In *Proceedings of the The 18th Annual Workshop of the Australasian Language Technology Association*, pages 98–103.

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.

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*.

Anandeep S Pannu. 1995. Using genetic algorithms to inductively reason with cases in the legal domain. In *Proceedings of the 5th international conference on Artificial intelligence and law*, pages 175–184.

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.

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.

JURI SAYS and An Automatic Judgement. 2020. Prediction system for the european court of human rights. In *Legal Knowledge and Information Systems: JURIX 2020: The Thirty-third Annual Conference, Brno, Czech Republic, December 9-11, 2020*, volume 334, page 277. IOS Press.

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.

Marek J. Sergot, Fariba Sadri, Robert A. Kowalski, Frank Kriwaczek, Peter Hammond, and H Terese Cory. 1986. The british nationality act as a logic program. *Communications of the ACM*, 29(5):370–386.

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.

Shangliao Sun. 2023. Smartphone penetration rate in india from 2009 to 2023, with estimates until 2040. https://www.statista.com/statistics/1229799/india-smartphone-penetration-rate/#:~:text=In%202023%2C%20the%20penetration%20rate,end%20of%202020%20was%20Xiaomi.

Feng Yao, Chaojun Xiao, Xiaozhi Wang, Zhiyuan Liu, Lei Hou, Cunchao Tu, Juanzi Li, Yun Liu, Weixing Shen, and Maosong Sun. 2022. Leven: A large-scale chinese legal event detection dataset. *arXiv preprint arXiv:2203.08556*.

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.

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).

Lucia Zheng, Neel Guha, Brandon R Anderson, Peter Henderson, and Daniel E Ho. 2021. When does pretraining help? assessing self-supervised learning for law and the casehold dataset of 53,000+ legal holdings. In *Proceedings of the eighteenth international conference on artificial intelligence and law*, pages 159–168.

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 .

   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 .')
   ```

   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 f 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 Algorithm to linearize discourse tree into an encoding

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 $\mathcal{E}$

A.3 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.

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

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

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

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

A.4 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.5 Level-wise scores

We also evaluated the performance of our model against sentences with different levels of complexity. Conjunctive sentences are likely to have multiple conjunctions and thus produce complicated coordination tree structures with greater height. We evaluated models for sentences with different coordination tree heights in the gold set (Table 6). The model will generate `NONE` as output for these sentences. We see a similar trend with OpenIE6 slightly outperforming the generative approach. One reason for this is the presence of ambiguous labels in the test set for hierarchies with multiple levels. On such sentences, even though T5 generates a better split, it is still penalised. BART does well in identifying sentences that should not be split, however, it hallucinates when sentences become more complex.

A.6 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.

Level	Mapping Based Approach	OpenIE	T5-base	T5-small	BART-base	BART-small	Count (Train, Test)
Level 0	Precision	**0.9796**	0.9632	0.9182	0.9755	0.9714	(2426,163)
	Recall	**0.9816**	0.9632	0.9182	0.9755	0.9714	
	F1 Score	**0.9816**	0.9632	0.9182	0.9755	0.9714	
Level 1	Precision	**0.9856**	0.9800	0.9789	0.8240	0.8126	(12958,716)
	Recall	**0.9866**	0.9773	0.9669	0.7418	0.7287	
	F1 Score	**0.9856**	0.9781	0.9717	0.7720	0.7580	
Level 2	Precision	0.9465	**0.9518**	0.9428	0.7287	0.6789	(1716,98)
	Recall	**0.9737**	0.9685	0.9348	0.5790	0.4900	
	F1 Score	0.9564	**0.9567**	0.9365	0.6321	0.5611	
Level 3	Precision	0.9354	**0.9607**	0.9144	0.5454	0.6330	(153 ,6)
	Recall	**0.9914**	0.8823	0.8178	0.3574	0.3227	
	F1 Score	**0.9606**	0.9168	0.8536	0.4252	0.4155	
Level 4	Precision	0.7975	**0.9100**	0.8848	0.7666	0.6772	(26,2)
	Recall	**1.0000**	0.8950	0.8183	0.3480	0.3216	
	F1 Score	0.8814	**0.9008**	0.8416	0.4432	0.4334	

Table 6: Level-wise scores aggregated across 3 seeds. The best values are in bold. The second best is underlined.

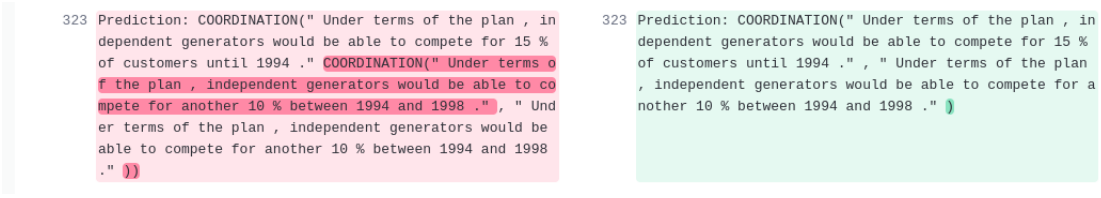

Figure 2: An example showing betterment of clauses on coordination dataset. Left one shows T5 with regular loss and right shows T5 prediction with custom loss

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.7 User Evaluation

To ascertain if hierarchical representation aids in simplifying legal text and reducing interpretation time, we conducted structured interviews with Ph.D. scholars from diverse departments. We e-mailed a Google Form along with a link to a visualization tool designed to create hierarchical representations visually. We had a total of five questions with varying options, similar to a Likert scale.

Here we provide a list of questions asked during a structured interview.

1. Please rate the interpretability of legal sentences without tree structure.
   (a) Very easy to understand
   (b) Easy to understand,
   (c) Neutral,
   (d) Difficult to understand
   (e) Very difficult to understand

2. Please rate the usability of the visualization.
   (a) Very easy to use

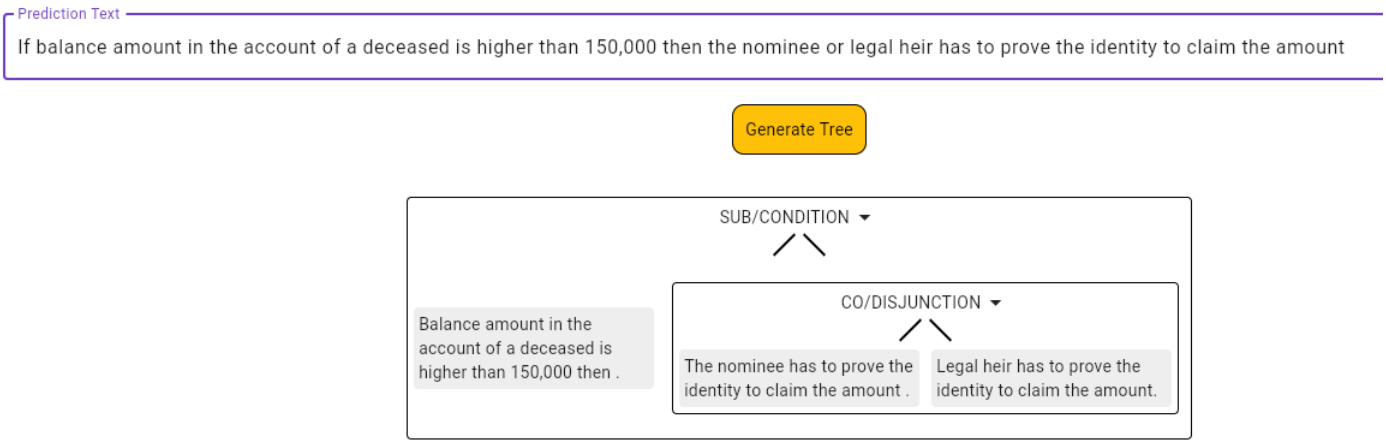

Figure 3: Visualization of Discourse Tree Structure generated through our tool.

(b) Easy to use
(c) Neutral
(d) Difficult to use
(e) Very difficult to use

3. Does the tree structure of long and complex legal statements simplify understanding?
   (a) Strongly Disagree
   (b) Disagree
   (c) Neutral
   (d) Agree
   (e) Strongly Agree
4. Does the tree structure of long and complex legal statements reduce interpretation time?
   (a) Strongly Disagree
   (b) Disagree
   (c) Neutral
   (d) Agree
   (e) Strongly Agree
5. How likely would you advise a new person to check visualisation first instead of a linear tree?
   (a) Very Unlikely
   (b) Unlikely
   (c) Neutral
   (d) Likely
   (e) Very Likely

A.8 Relation count in Indian Legal Dataset

Table 7 shows relation distribution in the test dataset and the accuracy of prediction by T5.

A.9 Improved result with custom loss

Figure 2 shows an instance where applying custom loss function improves 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 7: Relation distribution in Indian Legal Test data