Nonet at SemEval-2023 Task 6: Methodologies for Legal Evaluation

Shubham Kumar Nigam[1]   **Aniket Deroy**[2]   **Noel Shallum**[3]
Ayush Kumar Mishra[1]   **Anup Roy**[1]   **Shubham Kumar Mishra**[1]
Arnab Bhattacharya[1]   **Saptarshi Ghosh**[2]   **Kripabandhu Ghosh**[4]

[1] IIT Kanpur, India   [2] IIT Kharagpur, India
[3] Symbiosis Law School Pune, India   [4] IISER Kolkata, India
{shubhamkumarnigam,roydanik18,noelshallum,
saptarshi.ghosh,kripa.ghosh}@gmail.com
{ayushkm20,puna20,skmishra20,arnabb}@iitk.ac.in

Abstract

This paper describes our submission to the SemEval-2023 for Task 6 on LegalEval: Understanding Legal Texts. Our submission concentrated on three subtasks: Legal Named Entity Recognition (L-NER) for Task-B, Legal Judgment Prediction (LJP) for Task-C1, and Court Judgment Prediction with Explanation (CJPE) for Task-C2. We conducted various experiments on these subtasks and presented the results in detail, including data statistics and methodology. It is worth noting that legal tasks, such as those tackled in this research, have been gaining importance due to the increasing need to automate legal analysis and support. Our team obtained competitive rankings of 15th, 11th, and 1st in Task-B, Task-C1, and Task-C2, respectively, as reported on the leaderboard.

1 Introduction

The SemEval Task-6 (Modi et al., 2023) aims to automate several tasks to streamline the Indian legal process, which is often slow and delayed due to the country's large population and a shortage of judicial resources. Additionally, people in India are not always fully aware of the country's laws. To make the legal process more accessible to the general public, the SemEval task addresses crucial problems that are specific to the Indian judicial context. One task is the "Legal Named Entity Recognition (L-NER)" system, which identifies named entities in the legal text. Legal judgments contain intriguing entities like the names of the petitioner, respondent, judge, lawyer, date, organization, GPE, statute, provision, precedent, case number, witness, and other persons, which are typically not recognized by conventional entity recognition systems. Therefore, developing systems that are tailored to the legal domain is crucial. In task B, participants were tasked with identifying the legal entities present in legal judgments. A court judgment is divided into two parts: the preamble, which includes the names of

the parties, the court, lawyers, and other details, and the decision (judgment), which follows the preamble. The organizers separately provided the preamble and judgment text datasets.

Apart from the Legal NER system, the SemEval Task-6 also includes another important task C, that is crucial for automating the Indian legal process. The subtask, referred to as task C1, is the Legal Judgment Prediction (LJP) task. This task aims to determine whether the legal judgment favors the appellant or the defendant, which is modeled as a binary classification problem. Given a large number of pending legal cases in India, automating the process of predicting legal judgments can significantly reduce the burden on the judicial system. Moreover, the task of finding an explanation for the legal decision is equally important. For this, we locate a span from the legal judgment that is highly correlated with the reasoning behind the legal decision. Subtask C2 focuses on finding a suitable explanation for the binary classification task in subtask C1, thereby providing an additional layer of transparency to the legal process.

To accomplish these tasks, we experiment with several models and techniques. For the Legal Named Entity Recognition system, we try out a Spacy-based model and a fine-tuned BERT model to detect the legal named entities. For the Legal Judgment Prediction task, we pass the last 512 tokens of the legal judgment through transformer models, as well as try out hierarchical transformer models on the entire dataset to train the models for the task of judgment prediction. Lastly, for the Court Judgment Prediction with Explanation task, we check various span lengths taken from the end of the document. The datasets provided to the participants as a component of the SemEval task were in the English language.

Our main contributions can be summarized as: (1) Our contributions in this paper include participating in three subtasks: Legal Named Entity

Recognition (L-NER), Legal Judgment Prediction (LJP), and Court Judgment Prediction with Explanation (CJP).

(2) In the L-NER task, we explored three models: BERT-CRF, modified spaCy pipeline, and transformer embeddings.

(3) For the LJP task, we experimented with various hierarchical transformer models and utilized pretrained transformer models on the last 512 tokens of judgments.

(4) For the subtask of Explanation for Prediction, we proposed an intuitive approach of keyword-based matching technique for court judgment decision identification and extracted the court judgment explanation using the span lengths from the ending portions of legal judgments.

We released the codes and datasets for all subtasks via GitHub[1].

2 Background

We have participated in task-B named Legal Named Entity Recognition (L-NER), task-C1 called Legal Judgment Prediction (LJP), and task-C2, called Court Judgment Prediction with Explanation (CJPE).

Legal Named Entity Recognition: The task of Legal Named Entity Recognition (Legal NER) (Skylaki et al., 2020) involves identifying named entities in structured legal text. These entities include petitioner, respondent, court, statute, provision, precedent, and more. However, conventional Named Entity Recognizers, like SPACY, do not recognize these entity types, highlighting the need for domain-specific legal NER systems. Additionally, the Indian legal system has its unique processes and terminologies, making it necessary to develop a specific legal NER for Indian court decision texts. Unfortunately, there are no publicly available annotated datasets for the task of L-NER on Indian court cases.

To address this gap, a recent study (Kalamkar et al., 2022) has developed a Legal NER model specifically for Indian legal data. This work highlights the need for an annotated dataset to train the Legal NER model and proposes a methodology for creating such a dataset. By training their model on the created dataset, they achieved state-of-the-art results on the task of Legal NER on Indian legal data.

Legal Judgment Prediction: Legal judgment prediction is an important task in the field of legal informatics. A recent work by Malik et al. (Malik et al., 2021) focuses on predicting the outcome of Indian Supreme Court cases using state-of-the-art transformer models. The task is modeled as a binary classification problem and applied to the entire legal case except for the final judgment. The authors use the manually annotated explanations for every legal case judgment in the Indian Legal Document Corpus (ILDC) dataset as gold standards for the court judgment explanation task.

Another work that addresses the task of judgment prediction is the Chinese AI and Law Challenge dataset (Xiao et al., 2018). This dataset is the first large-scale Chinese legal dataset with 2.6 million cases, which makes it an excellent resource for judgment prediction on Chinese data. The human annotation in this work is rich, and the authors compared simple text classification approaches, such as FastText, TFIDF+SVM, CNN, etc., on the facts of the legal judgment.

The Hindi Legal Document Corpus (Kapoor et al., 2022) is a corpus of 900K legal documents annotated in Hindi, designed for the task of bail prediction. The work experiments with a range of state-of-the-art models and proposes a multi-task learning framework where the main task is bail prediction, and the auxiliary task is summarization.

Finally, there is a work on Legal judgment prediction (Chalkidis et al., 2019), which provides a dataset of 11k legal judgments. The work compares different models, including BiGRU-attn, HAN, LWAN, BERT, and Hier-BERT, on the facts of the legal case.

Explainability Methods: The method of Integrated Gradients is an explainability method (Sundararajan et al., 2017) for deep learning models where we have to load a trained deep learning model into this method, and then the method of Integrated Gradients computes the gradient of the output given by the model to its input features. The model gives attribution scores to every sentence in a document. The method helps in understanding and extracting the features which contribute to a model decision. Also, there are standard explainability techniques like LIME and SHAP for machine learning models. SHAP (Lundberg and Lee, 2017) is a predominant explainability technique that focuses on the individual impact of every feature on the model's final prediction. LIME (Ribeiro et al.,

---

2016) tries to understand how the perturbations in the model input affect the final output prediction. Explainable AI (Polley et al., 2021) is a recent and vibrant research area gathering a lot of focus. The work presents a recent explainable system called SIMFIC 2.0 which is an enhanced version of a recent explainable system. The idea here is to define the notion of similarity in fiction books. The system uses handcrafted interpretable features for fiction books and then provides a global explanation for fitting a linear regression and local explanation based on similarity features. Explainable Information retrieval (Anand et al., 2022) is an emerging research area that tries to improve upon the trustworthiness of the information retrieval methods.

3 System Overview

In this section, we provide an overview of the methodologies used for the tasks we participated in, which include Legal Named Entity Recognition (L-NER), Legal Judgment Prediction (LJP), and Court Judgment Prediction with Explanation.

3.1 Legal Named Entity Recognition (L-NER)

3.1.1 Using SpaCy Model

For L-NER, we propose an innovative approach that leverages the strengths of pre-trained transformer models and domain-specific embeddings to build custom NER models in the legal domain. Specifically, our model architecture involves fine-tuning a RoBERTa (Liu et al., 2019a) transformer with Spacy[2] and incorporating external embeddings from Law2Vec[3] (200 dimensions) to enhance the corpus's contextual understanding of legal terms.

Our pipeline involves data preparation, pipeline configuration, model fine-tuning, and evaluation. To begin with, we use the dataset provided by the organizers and create Doc objects that contain the text. We also incorporate external embeddings from Law2Vec to enhance the corpus's contextual understanding of legal terms. The custom Spacy's model training procedure is shown in Figure 1[4], which involves an iterative process of comparing the model's predictions against reference annotations to estimate the gradient of the loss and using it to calculate the gradient of the weights through back-propagation. The gradients indicate

how the weight values should be changed so that the model's predictions become more similar to the reference labels over time.

Figure 1: Spacy Custom NER Training Procedure

The Spacy pipeline configuration includes the RoBERTa transformer model, the Law2Vec embeddings, and additional NER components using the transformer, ner, and vectors components provided by Spacy. We fine-tune the BERT transformer model with the Adam.V1 optimization algorithm, a learning rate of $2e^{-5}$, and 100 epochs, and evaluate the performance of our model on a separate test set. SpaCy Model description in detail is here[5]. SpaCy is powerful but somewhat opaque and hard to modify. Overall, our work demonstrates the effectiveness of our model architecture for fine-tuning BERT transformers with Spacy 3 for Legal NER and highlights the potential of incorporating external embeddings from Law2Vec for building high-performance NER models in the legal domain.

3.1.2 Fine-Tuned BERT Model

We perform the task of Named Entity Recognition for legal documents using a pre-trained legal BERT base uncased model (Chalkidis et al., 2020). The model architecture comprises a BERT model with a token-level classifier, followed by a Linear-Chain CRF.

For an input sequence of n tokens, BERT outputs an encoded token sequence with hidden dimension H. The classification model takes each token's encoded representation to the tag space, i.e., $\mathbb{R}^H$ -> $\mathbb{R}^K$, where K is the number of tags in the dataset. And the output score maps to $\mathbb{R}^{n*K}$ of the classification model which is then fed to the CRF layer, which is used to predict in BIO format.

When we tested on BERT-CRF, our model could have performed better in this way because the sentence length was too much and the number of entities significantly less. Even in some of the training sentences, there were no entities in the sentence. We tried to make it more robust by adding part of the speech tag in the embedding layer of the BERT model. The observation behind providing further

---

Embedding is that most of the named entities are nouns, so we want our span to be more exact.

In the classical BERT model (Devlin et al., 2019), we have input embeddings as the sum of the token embeddings, the segment embedding, and the position embedding. After getting input embedding, it passes to Bert Layer Norm, then the dropout layer. After that, we return to the Embedding.

Actually, we cannot directly append a POS tag with the token because the pre-trained tokenizer only knows the token, not the token + POS tags. A better way is to create an additional input to the model(besides input_ids and token_type_ids) called pos_tags_ids, for which we add an additional embedding layer(nn. Embedding) 2. In that way, we sum the Token embeddings, token types, and POS tags. The max number of a pos tag is the total unique number of POS tags, also called the "vocabulary size" of the embedding layer. The hidden size is the size of the embedding vector that we want to learn for each pos tag(which is 768 by default for BERT-base). In order to improve the performance of the BERT-CRF model for named entity recognition (NER), we have generated part-of-speech (POS) tags as an additional feature for the provided dataset. These tags are used in conjunction with the tokens during training to make NER tag predictions. The POS tags are generated for each token in the sentences using a pre-trained spaCy (Honnibal and Montani, 2017) model, which has demonstrated an accuracy of 97%[6] for this specific task.

An additional complexity of BERT-like models is that they rely on subword tokens, rather than words. This means that a word like "playing" might be tokenized into ["play", "##ing"]. This means that we will also have to provide POS tags at the token level. Similar to how each token is turned into an integer (input_ids), we will also have to turn each POS tag into a corresponding integer (pos_tag_ids) in order to provide it to the model. So we would actually need to keep a dictionary that maps each POS tag to a corresponding integer.
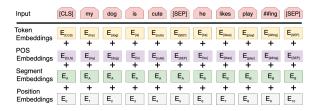

Figure 2: Modified BERT Embedding Layer

3.1.3 Fusion of Model

As discussed above, two models were trained to perform legal name entity recognition task. These models are SPACY and BERT-CRF. The output of both models is an array of tuples consisting of the entity's starting index, ending index, and label. To determine the final prediction, if the tuples in the output of both models intersect (start, end), then the intervals are merged, keeping the label the same for the final prediction. For instance, if the prediction from the SPACY model is (43,53, ORG) and that of the BERT-CRF model is (45,60, ORG), the final output will be (43,60, ORG). Otherwise, the union of outputs is taken from both models. This approach improved the prediction accuracy by 2% compared to using only the SPACY model for this task.
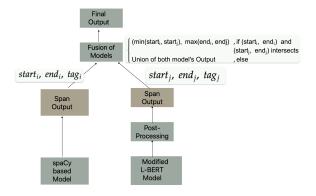

Figure 3: Fusion of Models

3.2 Court judgment Prediction with Explanation (CJPE)

Task C was divided into two sub-tasks:

1. Legal Judgment Prediction (LJP): given a legal judgment document, the task involves automatically predicting the outcome (binary: accepted or denied).

2. Explanation for Prediction: explanations are in the form of relevant text spans in the document that contributes to the decision.

For better understanding, we make the schematic diagram 4 for court judgment prediction and explanation. In that diagram, first, we preprocess the documents where we remove the meta information of case documents like judge name(s), court name, petitioner name(s), defendant name(s), hyperlinks, etc. (if it is present). Then pass, it to transformer-based models to classify the outcome. We pass

either the part of case documents that contain the relevant information or chunked the case file with an overlap of 100 token window size. Then the second subtask is to give or highlight the spans of text in the document, which helps or contributes to the decision.

3.2.1 Legal Judgment Prediction (LJP)

We used the pre-trained transformers which are trained on general corpus like XLNet (Yang et al., 2019) and Roberta (Liu et al., 2019b) as well as trained on legal corpus LegalBERT (Chalkidis et al., 2020), InlegalBERT, IncaseLawBERT (Paul et al., 2022) and we also train the BERT large on Indian judgment cases. Since the transformers have restrictions that they can not accommodate more than 512 tokens, so we gave only the last 510 tokens (two special tokens are reserved for CLS and SEP) as (Malik et al., 2021) mentioned in the paper that in general most relevant information is present at the end of the documents. Another way to accommodate the long document is that we took inspiration from (Chalkidis et al., 2020) and we also tried Hierarchical Transformer model architecture. We divided each document into chunks using a moving window approach where each chunk was of length 512 tokens, and there was an overlap of 100 tokens. We obtained the [CLS] representation of these chunks, which were then used as input to sequential models (BiGRU + attention). But we are not getting better accuracy on hierarchical models compared to the transformer-based models. The possible reason could be provided data is not sufficient for passing the embedding information to the sequential models.

3.2.2 Explanation for Prediction

We perform the task of Court judgment prediction by following pattern matching. We check for keywords that represent whether a court judgment is in favor of the appellant or against the appellant. We check for keywords namely dispose of, disposed of, accept, allow, allowed, accepted, and upheld in the entire court judgment which are keywords that help in detecting decisions in favor of the appellant. We check for keywords namely dismiss, dismissed, discard, discarded, reject, and rejected in the court judgment which are keywords that help in detecting decisions against the appelant.

There are research works (Polsley et al., 2016) on extractive legal document summarization where the legal domain experts say that towards the end

of a legal judgment, there are sentences that tend to summarize the entire judgment.

Current research works on abstractive summarization of legal judgements (Salaün et al., 2022) also suggests that the legal judgment is reversed and then fed into abstractive encoder-decoder architectures like BART which can take inputs up to a fixed input length of 1024 tokens so that the important information present at the end of a judgment is not missed.

Malik et al. (2021) states that the most important information corresponding to the court judgment is present towards the end portions of the legal document. The most important syntactic and semantic information occurs toward the end of a case. The largest occlusion and attention scores are assigned to the chunks present at the end of the document. For explaining the judgment of a court, we need to explain the reason behind the court judgment, which is given by the judges and generally present towards the end of a court judgment. The beginning portions of a legal document do not contain information that gives you a suitable explanation for explaining the court judgment decision. The problem description says that the explanations should be in form of relevant text spans in the document that contributes to the decision. So we take up a simple and intuitive approach to address the problem of finding a suitable explanation for the court judgment. So we choose continuous text spans of different span lengths from the end of the legal judgment. So we capture the last 550 words, last 520 words, last 512 words, last 500 words, last 450 words, last 400 words, last 350 words, last 300 words, and last 250 words of every court judgment document to act as an appropriate explanation of a court judgment.

For the purpose of breaking down every court judgment text into words, we used the split() function available in Python.

Though there are various sophisticated explainability approaches like the Method of Integrated Gradients which can rank the sentences present in the court judgment on the basis of attribution scores. These methods tend to work well if the model trained on the legal judgment prediction task has got high f1-scores. If the legal judgment prediction model trained on legal judgments is not giving good f1-scores on the test set, then the sophisticated explainability methods(like the Method of Integrated Gradients) might not work well because
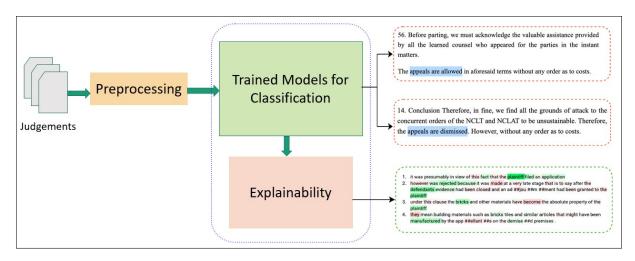
Figure 4: Court Judgment Prediction and Explanation (Task C) schematic diagram

the legal judgment prediction model itself is not very accurate. The pre-trained XLNET-large model trained on the last 512 tokens on the train set has given a low f1-score of 0.5287 on the test set(for subtask C1). So we can understand that the pre-trained XLNET large model trained for subtask-C1 is not very accurate. So we decided to follow a simple yet intuitive approach for the task of Court judgment prediction with explanation rather than using a sophisticated explainability approach(like the Method of Integrated Gradients). Any other standard explainability approach will also not work well because the best-performing model XLNET large(on subtask C1) has a low f1 score of 0.5287 on the test set and hence using this model any standard explainability approach will not be able to give a good explanation for the prediction task.

4 Experimental Setup

4.1 Legal-Named Entity Recognition (L-NER)

4.1.1 Dataset Description of L-NER

The Dataset provided comprised two sections. The first one was the Preamble, which contained the name of the parties, court, lawyers, etc., and the second one was the judgment, which contained the name of the court, case number, precedent, provision, witness, etc. There were 13 tags in the Judgment section and 5 in the Preamble section. The distribution of tags for the two sections is shown in Figure 5 and Figure 6 respectively, and the distribution of sentences for the Train, Dev, and Test Data split is shown in the Table 1.
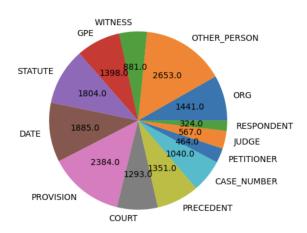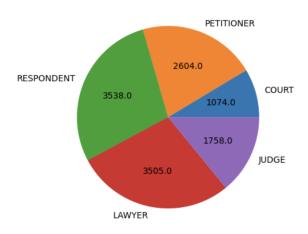

Figure 5: Judgement Tag Distribution

Figure 6: Preamble Tag Distribution

	Train	Dev	Test
PREAMBLE	1560	125	441
JUDGMENT	9435	949	4060

Table 1: Train, Dev and Test Data Split for L-NER

4.1.2 Dataset Preprocessing for L-NER

Different preprocessing layers are involved in L-NER, which consists of preprocessing before running the model, which requires cleaning the dataset in which extra spaces are removed (between words and start and end of sentences), similarly removing extra symbols, i.e., repetition of non-alphanumeric characters. We observed that there were multiple newline characters, tabs, and extra spaces in the dataset because of the structure of the legal document of India. After the preprocessing, there occurred differences in spans of entities in the original text and preprocessed text, so we used regex to find new start and end indexes for entities in preprocessed text for the SPACY model, and for the BERT model, we used BIO format for training over preprocessed text. At the time of prediction, for the BERT model, we converted the BIO format output back to the JSON format, which had the entities with their labels and starting and ending indexes in the sentence.

4.1.3 Hyperparameters of L-NER

Table 2 shows the hyperparameters for both the models which gave best result for this task.

Hyperparameter	SPACY	L_Bert-Crf
Learning Rate	$5e^{-4}$	$3e^{-5}$
Batch Size	128	32
Number of Epochs	50	50
Activation Function	ReLU	ReLU
Dropout Rate	0.1	0.25
Optimizer	Adam	AdamW
Loss Function	Cross-Entropy	Cross-Entropy

Table 2: Hyperparameters for the Models for L-NER

4.2 Legal Judgment Prediction (LJP)

4.2.1 Dataset for LJP

The provided dataset for task-C1, Legal judgment Prediction (LJP) consisted of two sets of ILDC$_{single}$, and ILDC$_{multi}$. The ILDC$_{single}$ and ILDC$_{multi}$ contain 4982 and 5082 train documents, correspondingly. For both sets, organizers provided 994 and 1500 common validation and test doc-

Corpus	Train	Dev	Test
ILDC_single	4982	994	1500
ILDC_multi	5082	994	1500

Table 3: LJP statistics

uments correspondingly. Data split is shown in Table 3.

4.2.2 Hyperparameters of LJP

If the document is longer than 10000 tokens, then we truncate the document. In transformers, we kept the batch size 16, epochs = 5, learning rate 2e-6, and the remaining parameters set to default.

For hierarchical transformers we make a chunk size of 500 tokens and an overlapping of 100 tokens.

4.3 Dataset for Explanation for Prediction

The dataset for task-C2, court judgment prediction with explanation consisted of 50 legal court judgments. The average no of words in the dataset is 2315.12 words. The average no of sentences in the dataset is 207.98. We tried to detect the rhetorical roles in the dataset of 50 court judgments.

Now we are trying to analyze the dataset in terms of rhetorical roles to get a better understanding of the nature of the dataset.

Rhetorical role	Fraction of rhetorical role
Facts	0.3858
Argument	0.0425
Ruling by present court	0.2284
Ruling by lower court	0.0124
Precedent	0.1270
Statute	0.1975
Ratio of the decision	0.0061

Table 4: Fraction of Rhetorical roles in every document in the dataset of 50 court judgments

Table 4 shows the fraction of Rhetorical roles (Bhattacharya et al., 2019) in every document in the dataset of 50 court judgments. The work (Bhattacharya et al., 2019) of rhetorical role detection uses a hierarchical BiLSTM-CRF model to address the problem of rhetorical role detection. The hierarchical BiLSTM layers extract necessary features from the sentence and the CRF layer helps to design the sequential presence of sentences belonging to different rhetorical roles. We have run the rhetorical role extractor just to understand and analyze the nature of the dataset. But we have not used sentences belonging to specific rhetorical roles to select a text span from the court judgment

as an explanation for the court judgment. Basically, the rhetorical role extractor has just been used to give us a better understanding of the nature of the dataset.

5 Results and Analysis

5.1 Legal Named Entity Recognition (L-NER)

In Task-B of the competition, we attained a ranking of 15th place, as demonstrated in Table 5. Our model achieved an F1-Score of 0.5532 on the test dataset. Our model was trained using 80% of the available training data, with the remaining 20% utilized as validation data. To evaluate the precision, recall, and F1 score of our model, the organizer's validation data was utilized as the test dataset. The best-performing model was chosen for submission. The classification report over the validation data serves as an indication of our model's performance on the test dataset. The weighted average F1-Score on the validation data, used as the test dataset, was 85.50 and 83.74 for the preamble and judgment, respectively, as illustrated in Tables 6 and 7.

There is a significant difference between the F1-Score achieved on the test dataset compared to what was displayed in the classification report for the validation dataset. Unfortunately, as the labeled test dataset has not been released, we are unable to provide a comprehensive analysis of why the model did not perform as well as it did on the validated dataset used as the test dataset.

Rank	User	Team Name	F1
1	Pinal-Patel	ResearchTeam_HCN	0.912
2	bluesky	-	0.9099
3	DeepAI	-	0.9099
15	**ShubhamKumarNigam**	**Nonet**	**0.5532**

Table 5: Scores and leader-board ranks for subtask B

Entity type	Precision	Recall	F1-Score
COURT	83.15	97.37	89.70
PETITIONER	65.50	90.34	75.94
LAWYER	88.74	90.13	89.43
RESPONDENT	78.95	79.71	79.33
JUDGE	93.53	89.04	91.23
Weighted Average			85.50

Table 6: L-NER per Entity Score for PREAMBLE on Validation dataset

5.2 Legal Judgment Prediction (LJP)

For task C-1 metric used for evaluation is the standard Macro F1 score. We ranked 11th with 4 entries in the LJP task and got a 0.5287 Macro F1

Entity type	Precision	Recall	F1-Score
STATUTE	90.61	90.61	90.61
PRECEDENT	68.97	76.92	72.73
JUDGE	72.73	100.00	84.21
GPE	77.91	74.16	75.64
OTHER PERSON	86.72	88.68	87.69
DATE	82.85	98.51	90.00
PROVISION	88.39	89.59	88.99
CASE NUMBERS	75.59	84.96	80.00
COURT	91.72	89.60	90.64
ORG	67.69	59.86	63.54
PETITIONER	50.00	77.78	60.87
WITNESS	90.00	93.10	91.53
RESPONDENT	100.00	80.00	88.89
Weighted Average			83.74

Table 7: L-NER per Entity Score for JUDGEMENT on Validation dataset

score using a pre-trained XLNet large model. The result of our best model is in table 8 along with the top 3 teams that participated in this task. In the training phase, we tried different pre-trained transformers which are trained on the general corpus and trained on the legal corpus. We also tried hierarchical transformers but we get the best result by using the XLNet model.

Rank	User	Team Name	F1
1	bluesky	-	0.7485
2	irit_iris	irit_iris	0.7228
3	uottawa.NLP23	uottawa.nlp23	0.6782
11	**ShubhamKumarNigam**	**Nonet**	**0.5287**

Table 8: Scores and leader-board ranks for subtask C-1

In Table 9, we present the results of the Legal Judgment Prediction (LJP) task. We experimented with various pre-trained transformers, including those trained on general and legal corpora. Among these models, the best performance was achieved with XLNet_large. We also explored hierarchical transformers; however, we did not observe improved accuracy compared to transformer-based models. This could be due to the lack of sufficient data for effectively passing embedding information to sequential models.

Models	Precision	Recall	F1-Score
InLegalBERTLarge	0.6825	0.6821	0.6823
LegalBert	0.6299	0.6288	0.6294
XLNet_large	**0.7703**	**0.7596**	**0.7649**
Roberta_large	0.7580	0.7354	0.7465
InLegalBERT	0.7468	0.7364	0.7416
InCaseLaw	0.7162	0.7072	0.7117

Table 9: Judgment prediction results on Validation dataset

5.3 Explanation for Prediction

For task C-2 metric used for evaluation is the standard ROUGE-2 score.

Rank	User	Team Name	F1	ROUGE-2
1	**ShubhamKumarNigam**	**Nonet**	**0.5417**	**0.0473**
2	bluesky	-	0.4797	0.047
3	nicolay-r	nclu_team	0.4789	0.0465

Table 10: Scores and leader-board ranks for subtask C-2.

We ranked in the first position in the Court judgment prediction with explanation subtask. We show the leaderboard of task C2: Court judgment prediction with an explanation for the set of 50 court judgments in table 10. The macro-f1 score for the court judgment prediction task is 0.5417.

Table 11 shows the results of Court judgment prediction subtask and explanation subtask for 50 court judgments. The highest rouge-2 score for the explanation subtask is obtained for the last 300 words for all court judgments.

Explanation span length	Rouge-2 Score	Percentage of Text covered
Last 250 words	0.0458	16.05
Last 300 words	**0.0473**	19.26
Last 350 words	0.0468	22.47
Last 400 words	0.0462	25.69

Table 11: Results of task C2: Explanation for Prediction for the set of 50 court judgments. The best result is indicated in bold. Also, the percentage of text covered in the entire court judgment(measured in terms of no. of words) for different span lengths for all 50 court judgments is mentioned.

Also, Table 11 shows the percentage of text covered in the entire court judgment(measured in terms of no of words) for different span lengths for all 50 court judgments. The best results are obtained for the last 300 words, and the percentage of text covered is 19.26%.

6 Conclusion

In conclusion, we participated in three subtasks of the SemEval-2023 for Task 6 on LegalEval: Legal Named Entity Recognition (L-NER), Legal Judgment Prediction (LJP), and Court Judgment Prediction with the explanation. For L-NER, we utilized a combination of pre-trained transformer models, domain-specific embeddings, and a modified spaCy pipeline to build a high-performing NER model. In LJP, we experimented with various hierarchical transformer-based models and utilized the last 512 tokens of judgments through pre-trained transformer models. For the explanation task, we found that taking the last 300 words from the end of the legal judgment performed best in terms of the rouge-2 f1 score.

Furthermore, we compared the results of different models for each subtask to analyze their strengths and weaknesses. However, since the gold standard data corresponding to the test set for every subtask was not released by the organizers, we were unable to perform an error analysis on the test dataset. We will do this once the gold standard labels are available publicly.

Looking ahead, we plan to explore joint optimization frameworks that incorporate both rhetorical role information and sentence position in the document to create more accurate span detection models. Overall, our participation in the competition has allowed us to showcase the effectiveness of our proposed models and techniques for tasks in the legal domain.

References

Avishek Anand, Lijun Lyu, Maximilian Idahl, Yumeng Wang, Jonas Wallat, and Zijian Zhang. 2022. Explainable information retrieval: A survey. *arXiv preprint arXiv:2211.02405*.

Paheli Bhattacharya, Shounak Paul, Kripabandhu Ghosh, Saptarshi Ghosh, and Adam Wyner. 2019. Identification of rhetorical roles of sentences in Indian legal judgments. In *Proc. JURIX*.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural legal judgment prediction in English. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4317–4323, Florence, Italy. Association for Computational Linguistics.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

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

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*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019a. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

Vijit Malik, Rishabh Sanjay, Shubham Kumar Nigam, Kripabandhu Ghosh, Shouvik Kumar Guha, Arnab Bhattacharya, and Ashutosh Modi. 2021. ILDC for CJPE: Indian legal documents corpus for court judgment prediction and explanation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4046–4062, Online. Association for Computational Linguistics.

Ashutosh Modi, Prathamesh Kalamkar, Saurabh Karn, Aman Tiwari, Abhinav Joshi, Sai Kiran Tanikella, Shouvik Guha, Sachin Malhan, and Vivek Raghavan. 2023. SemEval-2023 Task 6: LegalEval: Understanding Legal Texts. In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, Toronto, Canada. Association for Computational Linguistics (ACL).

Shounak Paul, Arpan Mandal, Pawan Goyal, and Saptarshi Ghosh. 2022. Pre-training transformers on indian legal text.

Sayantan Polley, Rashmi Raju Koparde, Akshaya Bindu Gowri, Maneendra Perera, and Andreas Nuernberger. 2021. Towards trustworthiness in the context of explainable search. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 2580–2584, New York, NY, USA. Association for Computing Machinery.

Seth Polsley, Pooja Jhunjhunwala, and Ruihong Huang. 2016. CaseSummarizer: A system for automated summarization of legal texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 258–262, Osaka, Japan. The COLING 2016 Organizing Committee.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Olivier Salaün, Aurore Troussel, Sylvain Longhais, Hannes Westermann, Philippe Langlais, and Karim Benyekhlef. 2022. Conditional abstractive summarization of court decisions for laymen and insights from human evaluation. In *Legal Knowledge and Information Systems*, pages 123–132. IOS Press.

Stavroula Skylaki, Ali Oskooei, Omar Bari, Nadja Herger, and Zac Kriegman. 2020. Named entity recognition in the legal domain using a pointer generator network. *arXiv preprint arXiv:2012.09936*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Yansong Feng, Xianpei Han, Zhen Hu, Heng Wang, et al. 2018. Cail2018: A large-scale legal dataset for judgment prediction. *arXiv preprint arXiv:1807.02478.*

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.