Justice Delayed is Justice Denied: Enabling Legal Artificial Intelligence via Bail Prediction on Hindi Case Documents

Thesis submitted in partial fulfillment of the requirements for the degree of

_Master of Science_
_in_
_Computer Science and Engineering by Research_

by

Arnav Kapoor
20171040
arnav.kapoor@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2022
Copyright © Arnav Kapoor, 2022
All Rights Reserved
CERTIFICATE

It is certified that the work contained in this thesis, titled ‘Justice Delayed is Justice Denied: Enabling Legal Artificial Intelligence via Bail Prediction on Hindi Case Documents’ by Arnav Kapoor, has been carried out under my supervision and is not submitted elsewhere for a degree.

_________________________  ____________________________
Date                                      Advisor: Prof. Ponnurangam Kumaraguru
To Family
Acknowledgments

The dedication of this Thesis is split seven ways.

To PK, for being more than a professor for being a guide, a mentor and a friend.

To Mummy, To Papa and To Tishya for everything

To Friends for the wonderful memories and the amazing college life.

To Precog for the constant support and encouragement

To you the reader, hopefully you find something interesting and of value in this Thesis.
Abstract

Many populous countries including India are burdened with a considerable backlog of legal cases. Development of automated systems that could process legal documents and augment and help legal practitioners can mitigate this. However, there is a dearth of high-quality corpora that is needed to develop such data-driven systems. The problem gets even more pronounced in the case of low resource languages such as Hindi. Additionally one of the most common and time sensitive cases handled by the courts are bail cases.

In this Thesis, we first introduce the *Hindi Legal Documents Corpus (HLDC)*, a corpus of more than 900K legal documents in Hindi. Documents are cleaned and structured to enable the development of downstream applications. We then introduce and tackle the task of bail prediction. We select the bail cases from our HLDC corpus and further extract the facts and arguments and judge’s summary. We experiment with a battery of models and propose a Multi-Task Learning (MTL) based model for the same. Our MTL model uses summarization as an auxiliary task along with bail prediction as the main task. The intermediate summarisation step is a novel introduction which serves dual purposes. First, it reduces the document size without compromising on the information. Since many transformer models have constraint on the input length, sending a summarised version of the documents allows us to overcome this barrier. Second, it builds towards explainable legal NLP systems as it allows us to identify salient sentences.

This Thesis lays the foundation for research in Legal NLP for Hindi court documents. The multitude of legal NLP tasks and challenges are indicative of the need for further research in this area.
Contents

Chapter | Page
--- | ---
1 | Introduction | 1
   | 1.1 Motivation | 1
   | 1.2 Legal Systems | 1
   | 1.3 Indian Legal System | 2
   | 1.4 District Courts in India | 2
   | 1.5 Legal Corpus | 2
   | 1.6 Legal AI | 3
   | 1.7 eCourts Project and Data | 3
   | 1.8 Legal Judgement Prediction | 4
      | 1.8.1 Bail Prediction | 4
   | 1.9 Thesis Contribution | 4
   | 1.10 Thesis Organisation | 5
2 | Related Work | 6
   | 2.1 Legal Corpus | 6
   | 2.2 Legal Judgement Prediction | 8
   | 2.3 Legal Summarisation | 8
3 | HLDC: Hindi Legal Document Corpus | 9
   | 3.1 HLDC Creation Pipeline | 9
   | 3.2 NER Removal | 11
   | 3.3 Legal Document Segmentation | 11
   | 3.4 Case Type Identification | 12
   | 3.5 Lexical Analysis | 13
   | 3.6 Challenges and Limitations | 14
4 | Bail Prediction | 15
   | 4.1 Bail Document Corpus | 15
   | 4.2 Bail Document Segmentation | 16
      | 4.2.1 Header | 17
      | 4.2.2 Case Result | 17
      | 4.2.3 Judge’s summary | 17
      | 4.2.4 Facts and Arguments | 18
      | 4.2.5 Extracting Bail Decision from Result | 19
      | 4.2.6 Extracting Bail Amount from Result | 19
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3</td>
<td>Sample Segmented Document</td>
<td>21</td>
</tr>
<tr>
<td>4.4</td>
<td>Segmentation Pipeline Verification</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>Models for Bail Prediction</td>
<td>25</td>
</tr>
<tr>
<td>5.1</td>
<td>Bail Prediction Task Formulation</td>
<td>25</td>
</tr>
<tr>
<td>5.2</td>
<td>Embedding Based Models</td>
<td>26</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Doc2Vec</td>
<td>26</td>
</tr>
<tr>
<td>5.2.2</td>
<td>IndicBERT</td>
<td>27</td>
</tr>
<tr>
<td>5.3</td>
<td>Summarization Based Models</td>
<td>27</td>
</tr>
<tr>
<td>5.4</td>
<td>Multi-Task Learning (MTL) Model</td>
<td>28</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Bail Prediction Task</td>
<td>29</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Salience Classification Task</td>
<td>29</td>
</tr>
<tr>
<td>5.5</td>
<td>Experiments and Results</td>
<td>30</td>
</tr>
<tr>
<td>5.5.1</td>
<td>Dataset Splits</td>
<td>30</td>
</tr>
<tr>
<td>5.5.2</td>
<td>Results</td>
<td>31</td>
</tr>
<tr>
<td>5.5.3</td>
<td>Error Analysis</td>
<td>32</td>
</tr>
<tr>
<td>5.6</td>
<td>Challenges and Limitations</td>
<td>33</td>
</tr>
<tr>
<td>6</td>
<td>Ethical Considerations</td>
<td>35</td>
</tr>
<tr>
<td>7</td>
<td>Conclusion and Future Works</td>
<td>36</td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>End-to-end HLDC (Hindi Legal Document Corpus) Creation Pipeline from extraction of raw documents from eCourts to final cleaned case documents.</td>
<td>11</td>
</tr>
<tr>
<td>3.2</td>
<td>Variation in number of case documents per district in the state of U.P. Prominent districts are marked.</td>
<td>12</td>
</tr>
<tr>
<td>3.3</td>
<td>Temporal variation of case type for Bail cases in district Kanpur nagar. We see that in 2020 Kanpur nagar switched from ‘Bail Appl’ to ‘Bail Application’ to represent bail cases.</td>
<td>13</td>
</tr>
<tr>
<td>4.1</td>
<td>Ratio of number of bail applications to total number of applications in U.P. Gautam Budh Nagar (0.60), Kanpur Nagar (0.57) and Bagpat (0.52) have the highest ratio. The ratio is less than 0.5 for rest of the districts. The ratio for Allahabad is 0.14 which is one of the lowest.</td>
<td>16</td>
</tr>
<tr>
<td>4.2</td>
<td>Bail Corpus Creation Pipeline to extract relevant documents and information for the Bail Prediction task.</td>
<td>16</td>
</tr>
<tr>
<td>5.1</td>
<td>Overview of our multi-task learning approach, with bail prediction as the primary task and salient sentence identification as the auxiliary task.</td>
<td>29</td>
</tr>
<tr>
<td>5.2</td>
<td>ROC curve for the proposed model. The total AUC (Area under curve) is 0.85.</td>
<td>33</td>
</tr>
<tr>
<td>5.3</td>
<td>Kernel Density Estimate (KDE) plots of our proposed bail prediction model. The majority of errors (incorrectly dismissed / granted) are borderline cases with model output score around 0.5.</td>
<td>34</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Case types in HLDC. Out of around 300 different case types, we only show the prominent ones. Majority of the case documents correspond to bail applications.</td>
<td>13</td>
</tr>
<tr>
<td>4.1</td>
<td>Phrases used to construct regular expression for extracting judge’s opinion. The list is just an indicative of the various phrases and variants used; the entire list can be found in code.</td>
<td>18</td>
</tr>
<tr>
<td>4.2</td>
<td>Mapping of Bail decision tokens to the binary label of bail granted or denied.</td>
<td>20</td>
</tr>
<tr>
<td>4.3</td>
<td>Mapping of the word tokens to the corresponding total bail amount.</td>
<td>20</td>
</tr>
<tr>
<td>4.4</td>
<td>A sample segmented document with header, facts and arguments, judge’s opinion and result section.</td>
<td>23</td>
</tr>
<tr>
<td>4.5</td>
<td>Evaluation results of bail document division and bail decision extraction pipeline.</td>
<td>24</td>
</tr>
<tr>
<td>5.1</td>
<td>Number of documents in the train, validation and test sets across the district wise and all district data split.</td>
<td>30</td>
</tr>
<tr>
<td>5.2</td>
<td>List of Hyper-Parameters used for the final trained models.</td>
<td>31</td>
</tr>
<tr>
<td>5.3</td>
<td>Model results. For TF-IDF and TextRank models we take the sum of the token embeddings.</td>
<td>32</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Motivation

In any nation, groups, individuals, organisations, and governments are bound to have disagreements. A neutral authority must resolve all such disagreements in conformity with the rule of law. The rule of law means that the same rules bind all people. The judiciary’s primary function is to uphold the rule of law and guarantee that the law reigns supreme. It protects individual rights, handles conflicts according to the law, and guarantees that democracy does not give way to dictatorship by individuals or groups [NCERT, 2006]. The legal system plays a critical role in the functioning of a nation. It is paramount that cases are handled judiciously and efficiently. However, in recent times, the legal system in many populous countries (e.g., India) has been inundated with many legal pending cases. There is an imminent need for automated systems to process legal documents and help augment the legal procedures [Kapoor et al., 2022]. Legal Artificial Intelligence (LegalAI) is concerned with using artificial intelligence technologies to assist with legal tasks such as case summarisation, legal judgement prediction, similar case identification [Zhong et al., 2020a].

1.2 Legal Systems

The two significant legal systems in the world are the civil law and the common law systems. Civil law is a legal system originating in mainland Europe and adopted across the world. Civil law has its core principles codified into a referable system, which serves as the primary source of law. In contrast common law system is based on judicial precedence. Judicial precedence is a prior decision by the court used as a source to decide future judgements [Wikipedia, 2022a]. India maintains a hybrid legal system with a mixture of civil law, common law and other customary and religious law [Wikipedia, 2022d].
1.3 Indian Legal System

Judicial processes in large populated countries like India are slow and inefficient and a huge backlog of pending cases slows down the entire system. Most countries adopt a tiered court-system to improve productivity and streamline the judicial process. The United Kingdom has County Courts, High Courts and Supreme Court as its three levels. The United States of America also has District Courts, Courts of Appeals and Supreme Court in its hierarchical court structure. India follows a similar three-tiered system with the District courts, High courts and Supreme Court.

The Supreme Court, often known as the Apex Court, is India’s highest and last appellate court (court to appeal against conviction). The Chief Justice of India is its highest authority. High Courts are the highest judicial organisations in each state, and they are supervised and managed by Chief Justices of States. Below the High Courts are District Courts, also known as subordinate courts [Wikipedia, 2022c].

1.4 District Courts in India

The district courts of India are judicial bodies for each district or for one or more districts combined, taking into consideration the number of cases and population distribution in each district. In India, they are in charge of justice at a district level [Wikipedia, 2022b].

Discussions to expedite judiciary tends to focus on the cases in the High Court and the Supreme Court as they are more high profile cases. However, the majority of cases are handled in the lower courts. As of September 20, 2021, 40,035,606 total cases were pending in District Courts of India as opposed to 5,609,146 pending cases in High Courts of India. This highlights the importance of an efficient case disposal system at lower courts level [National Judicial Data Grid, 2021].

1.5 Legal Corpus

Processing of legal documents is challenging and is quite different from conventional text processing tasks. For example, legal documents are typically quite long (tens of pages), highly unstructured and noisy (spelling and grammar mistakes since these are typed), use domain-specific language and jargon; consequently, pre-trained language models do not perform well on these [Malik et al., 2021b]. Thus, to develop legal text processing systems and address the challenges associated with the legal domain,
there is a need for creating specialized legal domain corpora. In recent times, there have been efforts to develop such corpora. For example, [Chalkidis et al., 2019] have developed an English corpus of European Court of Justice documents, while [Malik et al., 2021b] have developed an English corpus of Indian Supreme Court documents. [Xiao et al., 2018] have developed Chinese Legal Document corpus. However, to the best of our knowledge, there does not exist any legal document corpus for the Hindi language (a language belonging to the Indo-European family and predominantly spoken in India). Hindi uses Devanagari script [Wikipedia contributors, 2021] for the writing system. Hindi is spoken by approximately 567 million people in the world [WorldData, 2021]. Most of the lower (district) courts in northern India use Hindi as the official language. However, most of the legal NLP systems that currently exist in India have been developed on English, and these do not work on Hindi legal documents [Malik et al., 2021b]. To address this problem, in this work, we release a large corpus of Hindi legal documents (HINDI LEGAL DOCUMENTS CORPUS or HLDC) that can be used for developing NLP systems that could augment the legal practitioners by automating some of the legal processes.

1.6 Legal AI

Legal Artificial Intelligence (LegalAI) is concerned with the use of artificial intelligence technologies, particularly natural language processing, to legal tasks. LegalAI is an important addition in the legal domain because it can help lawyers and judges save time and effort by reducing the amount of work they have to complete. Many legal duties need the experience of legal professionals as well as a full comprehension of numerous legal papers. Even for legal specialists, retrieving and comprehending legal documentation takes a long time. As a result, a certified LegalAI system will minimise the effort spent on these time-consuming tasks thus benefiting the legal system. Furthermore, LegalAI can serve as an economical legal help for people who are unfamiliar with the complex legal structure [Zhong et al., 2020a].

1.7 eCourts Project and Data

The eCourts Project was conceptualised on the basis of the ‘National Policy and Action Plan for Implementation of Information and Communication Technology (ICT) in the Indian Judiciary – 2005’ submitted by eCommittee, Supreme Court of India with a vision to transform the Indian Judiciary by ICT enablement of Courts [eCourt India, 2022].
eCourts project also was successful in creating an ecosystem to file, track and analyse case progress entirely online. The vision of the project enabled Indian courts to work during the pandemic without backlogs breaking the system. The case documents made public by the e-court were extensively used in the thesis.

1.8 Legal Judgement Prediction

Legal judgement prediction is the task of automatically predicting the outcome of a court case, given a text describing the case’s facts [Chalkidis et al., 2019] [Xu et al., 2020]. Improvements in LJP helps legal practitioners and citizens, while reducing legal costs and improving access to justice. It can also assist judges to make faster and more consistent decisions.

1.8.1 Bail Prediction

Bail Prediction is a specific case of Legal Judgement Prediction that predicts if bail is granted or not. Bail essentially means the judicial interim release of a person suspected of a crime held in custody, on entering into a recognizance, with or without sureties, that the suspect would appear to answer the charges at a later date [Law Commission India, 2017].

1.9 Thesis Contribution

In the thesis we provide a comprehensive overview of the LegalAI landscape in the India Context. We identify a prominent gap of limited work in local Indic languages and aim to bridge it.

In a nutshell, we make the following main contributions in this thesis:

- We create a Hindi Legal Documents Corpus (HLDC) of 912,568 documents. These legal case documents are cleaned and structured to make them usable for downstream NLP/IR applications. Moreover, this is a growing corpus as we continue to add more legal documents to HLDC. We release the corpus and model implementation code with this thesis: https://github.com/Exploration-Lab/HLDC.

- As a use-case for applicability of the corpus for developing legal systems, we propose the task of Bail Prediction - wherein given the facts of the case we predict if the bail would be granted or denied. Its the first work to deal with legal judgement prediction in Hindi.
For the task of bail prediction, we experiment with a variety of deep learning models. We propose a multi-task learning model based on the transformer architecture. The proposed model uses extractive summarization as an auxiliary task and bail prediction as the main task.

1.10 Thesis Organisation

The thesis is organised into 7 chapters. Chapter 2 covers the related work done in legal NLP sphere focusing on previously released legal corpus, legal summarisation and judgement prediction. Chapter 3 introduces the creation and properties of HLDC (Hindi Legal Document Corpus). In Chapter 4, we showcase the utility of the HLDC corpus through bail prediction task. We create a bail prediction corpus and segment the case documents to identify the facts of the case. In Chapter 5, we try out a multitude of Machine Learning and Deep Learning models to tackle the bail prediction problem. We also propose our Multi Task Learning model that outperforms the baselines. In Chapter 6, we discuss the ethical considerations of working with sensitive legal data. We also discuss the importance of mitigating bias in models, especially in a high stake task such as bail prediction. We finally conclude the thesis and discuss future direction in Chapter 7.
Chapter 2

Related Work

In this chapter we aim to provide a comprehensive understanding of prior literature in this field. In recent years there has been active interest in the application of NLP techniques to the legal domain [Zhong et al., 2020a]. A number of tasks and models have been proposed, inter alia, Legal Judgment Prediction [Chalkidis et al., 2019], Legal Summarization [Bhattacharya et al., 2019; Tran et al., 2019], Prior Case Retrieval [Jackson et al., 2003; Shao et al., 2020], Legal Question Answering [Kim and Goebel, 2017], Catchphrase Extraction [Galgani et al., 2012], Semantic Segmentation [Kalamkar et al., 2022; Malik et al., 2021a].

We focus on three main aspects:

- Legal Corpus
- Legal Judgement Prediction
- Legal Summarisation

2.1 Legal Corpus

Majority of corpora for Legal-NLP tasks have been in English; recently, there have been efforts to address other languages as well, for example, [Xiao et al., 2018], have created a large-scale Chinese criminal judgment prediction dataset with over 2.68 million legal documents. Work on Legal-NLP in languages other than English is still in its incipient stages. The Thesis contributes towards these efforts by releasing corpus in Hindi. Majority of the work in the legal domain has focused on the higher court [Malik et al., 2021b; Strickson and De La Iglesia, 2020; Zhong et al., 2020b]; however, the lower courts handle the maximum number of cases. We try to address this gap by releasing a large corpus of district court level legal documents.
Some of the recent work has explored other Legal-NLP tasks in languages other than English. [Chalkidis et al., 2021] released a multilingual dataset of 65K European Union (E.U.) laws for topic classification of legal documents. The data was translated into the 23 official E.U. languages and annotated with labels from the multilingual thesaurus, EUROVOC. [Luz de Araujo et al., 2018] have released 70 documents in Portuguese for Legal Named Entity Recognition. The dataset contains specific tags for law and legal cases entities in addition to the normal tags for named entities like person, locations, organisation and time-entities. COLIEE (Competition on Legal Information Extraction/Entailment) tasks [Kano et al., 2019, 2017] have published legal data in Japanese, along with their English translation. The competition has two sub-tasks, a legal information retrieval task and an entailment identification task between law articles and queries. Multiple datasets in Chinese have been released for different tasks, namely Reading Comprehension [Duan et al., 2019], Similar Case Matching [Xiao et al., 2019], Question Answering [Zhong et al., 2020b]. [Duan et al., 2019] proposed Chinese judicial reading comprehension (CJRC) dataset with about 10K documents and almost 50K questions with answers. [Zhong et al., 2020b] presented JEC-QA, a legal question answering dataset collected from the National Judicial Examination of China with about 26K multiple-choice questions. They augment the dataset with a database containing the legal knowledge required to answer the questions and also assign meta information to each of the questions for in-depth analysis. [Xiao et al., 2019] proposed CAIL2019-SCM, a dataset containing 8,964 triplets of the case document, with the objective to identify which two cases are more similar in the triplets. Similar case matching has a crucial application as it helps to identify comparable historical cases. A historical case with similar facts often serves as a legal precedent and influences the judgement. Such historical information can be used to make the legal judgement prediction models more robust.

[Kleinberg et al., 2017] proposed bail decision prediction as a good proxy to gauge if machine learning can improve human decision making. A large number of bail documents along with the binary decision (granted or denied) makes it an ideal task for automation. In this Thesis, we look into the bail prediction task (in Hindi language) on the HLDC corpus.
2.2 Legal Judgement Prediction

Legal Judgement Prediction (LJP) involves predicting the final decision from the facts and arguments of the case. [Chalkidis et al., 2019] released 11,478 cases from the European Court of Human Rights (ECHR). It contains facts, articles violated (if any), and the importance scores. [Malik et al., 2021b] provided 34,816 case documents from the Supreme Court of India for the prediction task. [Strickson and De La Iglesia, 2020] published 4,959 documents from the U.K.’s Supreme court (the highest court of appeal).

2.3 Legal Summarisation

Automatic text summarisation aims to condense documents in a precise manner preserving the important pieces of information. Existing legal systems use legal professionals or paralegals to prepare case summaries, which is a costly procedure. Hence legal summarisation is a critical use-case in the LegalAI ecosystem.

Text summarisation is broadly of two types: i) Extractive Summarisation, in which relevant lines from the source document are extracted and incorporated in the summary and (ii) Abstractive Summarisation, in which the model builds a summary using appropriate vocabulary and language structure. [Bhattacharya et al., 2021]

The summarisation of documents in not only useful for legal practitioners but can also be used as an intermediary step in a solution pipeline. Summarisation can help us identify relevant sentences which can provide some explainability to our deep learning models. Additionally, it reduces the length of documents, subsequently reducing the number of tokens. Many transformer based models have hard constraints on the maximum number of tokens, thus a summarisation step helps to overcome this challenge.
Chapter 3

HLDC: Hindi Legal Document Corpus

In this chapter we go through the creation and processing of the HLDC corpus. Hindi Legal Documents Corpus (HLDC) is a corpus of 912,568 Indian legal case documents in the Hindi language. The corpus is created by downloading data from the e-Courts website (a publicly available government service \(^2\)). All the legal documents we consider are in the public domain. We download case documents pertaining to the district courts located in the Indian northern state of Uttar Pradesh (U.P.). We focus mainly on the state of U.P. as it is the most populous state of India, resulting in the filing of a large number of cases in district courts. U.P. has 71 districts and about 161 district courts. U.P. is a predominantly Hindi speaking state, and consequently, the official language used in district courts is Hindi. We crawled case documents from all districts of U.P. corresponding to cases filed over two years, from May 01, 2019 to May 01, 2021. Figure 3.2 shows the map of U.P. and district wise variation in the number of cases. As can be seen in the plot, the western side of the state has more cases; this is possibly due to the high population and more urbanization in the western part. Table 3.1 shows %-wise division of different case types in HLDC. As evident from the table, majority of documents pertain to bail applications. HLDC corpus has a total of 3,797,817 unique words, and on average, each document has 764 words.

3.1 HLDC Creation Pipeline

We outline the entire pipeline used to create the corpus in Figure 3.1. The documents on the website are originally typed in Hindi (in Devanagari script) and then scanned to PDF format and uploaded.

---

1 The work is a full paper accepted at Findings of ACL 2022.

2 https://districts.ecourts.gov.in/
**Document Collection:** The first step in HLDC creation is the downloading of legal case documents (PDF) from the e-Courts website. The data for each court complex is hidden behind a captcha. So, we used selenium (a browser emulation library) to mock browser actions and decode captcha after image pre-processing to remove noise and use Tesseract, an OCR engine. As the captcha is vulnerable to time outs, we collected data in small chunks of time. The data was collected for a period of two years spanning from May 2019 to May 2021 for each district. We collated a total of 1,221,950 legal Hindi documents.

**OCR conversion:** To extract Hindi text from these, we perform OCR (Optical Character Recognition) via the Tesseract tool. Tesseract worked well for our use case as the majority of case documents were well-typed, and it outperformed other OCR libraries.

**Document Cleaning:** Obtained text documents were further cleaned to remove noisy documents, e.g. small documents (< 32 bytes) or too long (> 8096 bytes) documents, duplicates, and English documents. The exhaustive criteria to remove documents is listed below:

Out of the total 1,221,950 case documents scraped from eCourts website, 309,382 documents were removed in the cleaning and filtering process leaving over 900k documents in the final HLDC corpus. Following rules were used to remove documents.

- Removed very short documents (file size ≤ 32 bytes). A number of documents were stub files without any coverage of the case, this filtering criteria helped us to remove such documents.
- Removed duplicate documents, to prevent over-counting. The removal of document was done both on the basis of case id of the document and an exact match on the content of the file.
- Removed too large documents (>8096 bytes). Few case documents exceeded a file size of 8Mb and slowed the processing pipeline. Hence we removed this very small minority of excessively large documents.
- Removed document where majority text was in English language. We used a simple method comparing the range of ASCII values of the English alphabet to identify English words.

This resulted in a total of 912,568 documents in our Hindi Legal Document Corpus - HLDC.

---

3 https://github.com/tesseract-ocr
4 https://github.com/JaidedAI/EasyOCR
Figure 3.1: End-to-end HLDC (Hindi Legal Document Corpus) Creation Pipeline from extraction of raw documents from eCourts to final cleaned case documents.

3.2 NER Removal

We anonymized the corpus with respect to names and locations. We used a gazetteer along with regex-based rules for NER to anonymize the data. List of first names, last names, middle names, locations, titles like पंडित (Pandit: title of Priest), सर (Sir: Sir), month names and day names were normalized to <नाम> (Naam: <name>). The gazetteer also had some common ambiguous words (these words can be names or sometimes verbs) like प्रार्थना (Prathna: Can refer to prayer, the action of request or name), गाया (Gaya: can refer to location name or verb), किया (Kiya: can refer to infinitive ‘to do’ or name), लिया (Liya: can refer to infinitive ‘to take’ or name). These were removed. Further, we ran RNN-based Hindi NER model on a subset of documents to find additional entities and these were subsequently used to augment our gazetteer. Phone numbers were detected using regex patterns and replaced with a <फोन-नंबर> (<phone-number>) tag, numbers written in both English and Hindi were considered.

3.3 Legal Document Segmentation

Legal documents, particularly in lower courts, are highly unstructured and lack standardization with respect to format and sometimes even the terms used. We converted the unstructured documents to semi-
structured documents. We segmented each document into a header and a body. The header contains the meta-information related to the case, for example, case number, court identifier, and applicable sections of the law. The body contains the facts of the case, arguments, judge’s summary, case decision and other information related to the final decision. The documents were segmented using regex and rule based approaches.

### 3.4 Case Type Identification

HLDC documents were processed to obtain different case types (e.g., Bail applications, Criminal Cases). The case type was identified via the meta-data that comes with each document. However for each case, the uniformity for case type across districts and even within district is little. For eg: Bail applications was categorised as ‘Bail Appl’ or ‘Bail Application’ in Kanpur Nagar district. Similarly, there were variations of ‘Civil Revision’ as ‘Civil Reivision’ and ‘Civill Revision’ in multiple districts. Most of these issues occur due to typographical errors. We hypothesised that whenever multiple such names are occurring in a single district, they might have been subject to correction at some point of time. To verify, we did temporal analysis of case types, which confirmed our hypothesis. Figure 3.3 showcases how Bail Application were referred as both ‘Bail Appl.’ and ‘Bail Application’ in Kanpur-nagar. We order the cases by filing date and observe that there was a transition from ‘Bail Appl’ to ‘Bail...
Figure 3.3: Temporal variation of case type for Bail cases in district Kanpurnagar. We see that in 2020 Kanpurnagar switched from ‘Bail Appl’ to ‘Bail Application’ to represent bail cases.

<table>
<thead>
<tr>
<th>Case Type</th>
<th>% in HLDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bail Applications</td>
<td>31.71</td>
</tr>
<tr>
<td>Criminal Cases</td>
<td>20.41</td>
</tr>
<tr>
<td>Original Suits</td>
<td>6.54</td>
</tr>
<tr>
<td>Warrant or Summons in Criminal Cases</td>
<td>5.24</td>
</tr>
<tr>
<td>Complaint Cases</td>
<td>4.37</td>
</tr>
<tr>
<td>Civil Misc</td>
<td>3.4</td>
</tr>
<tr>
<td>Final Report</td>
<td>3.32</td>
</tr>
<tr>
<td>Civil Cases</td>
<td>3.23</td>
</tr>
<tr>
<td>Others (Matrimonial Cases, Session Trial, Motor Vehicle Act, etc.)</td>
<td>21.75</td>
</tr>
</tbody>
</table>

Table 3.1: Case types in HLDC. Out of around 300 different case types, we only show the prominent ones. Majority of the case documents correspond to bail applications.

Application’ in middle of year 2020. We resolved these standardization issues via manual inspection and edit distance metrics, resulting in a final list of 300 unique case types.

3.5 Lexical Analysis

Although Hindi is the official language, U.P. being a large and populous state, has different dialects of Hindi spoken across the state. We found evidence of this even in official legal documents. For example, the word साकिन (Sakin: motionless) appears 11,614 times in the dataset, 63.8% occurrences of the word come from 6 districts of East U.P. (Ballia, Azamgarh, Maharajganj, Deoria, Siddharthnagar and Kushinagar). This particular variant of motionless being used most often only in East U.P. Similarly,
the word गाउशिया (Gaushiya: cow and related animals) is mostly used in North-Western UP (Ramapur, Pilibhit, Jyotiba Phule Nagar (Amroha), Bijnor, Budaun, Bareilly, Moradabad). Three districts - Muzaffarnagar, Kanshiramnagar and Pratapgarh district constitute 81.5% occurrences of the word दंड (Dand: punishment). These districts are, however, spread across UP. An important thing to note is that words corresponding to specific districts/areas are colloquial and not part of the standard Hindi lexicon. This makes it difficult for prediction model to generalize across districts (§5.5).

3.6 Challenges and Limitations

Working with legal data from the eCourts website comes with numerous challenges. The two main challenges was the poor data quality and standardisation. Some of these challenges have been pointed out in the past [Narasappa et al., 2016] and are still prevalent.

- **Data Quality** - The quality of the current eCourts data is a key problem. District Court data is plagued with mistakes, rendering portions of it worthless in its current state. To analyse this data, it must be extensively reviewed and cleaned up — a process hampered by the large volume of data and the number of errors it includes.

- **Standardisation** - Every District court operates independently of each other. The courts operating in silos means there is no consistent structure to case documents. Section 3.4 highlighted this challenge and showed the problem of normalising case types. The lack of standardisation is further aggravated when we consider districts from different states.
Chapter 4

Bail Prediction

In this chapter we introduce the bail prediction task and the creation of the corpus to achieve it. The most prevalent case-type amongst the cases in the corpus was Bail cases. Additionally bail cases are very time-sensitive in nature as the hearing and decision needs to be completed in a short period. Bail cases on average are resolved within 7-15 days as compared to most other cases which can takes years.

Hence, considering the utility of quick bail decisions and as a use-case for applicability of the HLDC corpus for developing legal systems, we propose the task of Bail Prediction. Given a legal document with facts of the case, the task of bail prediction requires an automated system to predict if the accused should be granted bail or not. The motivation behind the task is not to replace a human judge but rather augment them in the judicial process. Given the volume of cases, if a system could present an initial analysis of the case, it would expedite the process [Zhong et al., 2020a]. As told to us by legal experts and practitioners, given the economies of scale, even a small improvement in efficiency would result in a large impact [Kapoor et al., 2022].

4.1 Bail Document Corpus

Bail is the provisional release of a suspect in any criminal offence on payment of a bail bond and/or additional restrictions. Bail cases form a large majority of cases in the lower courts, as seen in Table 3.1. Additionally, they are very time-sensitive as they require quick decisions. For HLDC, the ratio of bail documents to total cases in each district is shown in Figure 4.1. As a use-case for the corpus, we further investigated the subset of the corpus having only the bail application documents (henceforth, we call it Bail Corpus). Figure 4.4 shows a sample bail documents with different structured segments.
Figure 4.1: Ratio of number of bail applications to total number of applications in U.P. Gautam Budh Nagar (0.60), Kanpur Nagar (0.57) and Bagpat (0.52) have the highest ratio. The ratio is less than 0.5 for rest of the districts. The ratio for Allahabad is 0.14 which is one of the lowest.

4.2 Bail Document Segmentation

Out of 912,568 documents in HLDC, 340,280 were bail documents, which were further processed to obtain the Bail Document corpus. Bail documents were structured into different sections. We extracted these sections from the bail documents. Details are mentioned below. For the bail documents, besides the header and body, we further segmented the body part into more sub-sections (Figure 4.2). Body is further segmented into Facts and Arguments, Judge’s summary and Case Result. Facts contain the facts of the case and the defendant and prosecutor’s arguments. Most of the bail documents have a concluding paragraph where the judge summarizes their viewpoints of the case, and this constitutes the judge’s summary sub-section. The case result sub-section contains the final decision given by the judge.

Figure 4.2: Bail Corpus Creation Pipeline to extract relevant documents and information for the Bail Prediction task.
4.2.1 Header

Header refers to the meta data related to the case, for example, धारा (IPC (Indian Penal Code) sections), जाना (police station), case number, date of hearing, accused name, etc. Header is present at the top of the document. Header mostly ended with धारा (IPC) or जाना (police station) details. Hence, in order to cut the document to get header, we first find the indices of धारा (IPC) and जाना (police station), and from these indices we find the finishing word of the header. We then segment the document at the finishing word. We also include the first line of upcoming paragraph in header as it also didn’t contain case arguments but contained data like if this is the first bail application or not.

4.2.2 Case Result

Case Result refers to the end of the document where judge writes their decision. Judge either accepts the bail application or rejects it. If the judge had accepted the bail document then this section mostly also contains bail amount and bail terms for accused. We observed that result section mostly began along the following line, मामले के समस्त तथ्यों को देखकर (looking at all facts of the case), the keyword तथ्यों (facts) was very common around the start of the result section. Hence, we iterated over the indices of keyword तथ्यों (facts) in reverse order and checked if the division at that index is correct. To check if the division is correct we look for bail result in lower half of the division, if the bail result is present, we classify that division as correct else we move to next index of तथ्यों (facts).

4.2.3 Judge’s summary

Most of the bail documents have a concluding paragraph where the judge summarizes their viewpoints of the case. To extract this, we first constructed certain regex which often precedes judge’s summary, defendant’s and prosecutor’s arguments (described in Table 4.1). Since the document might have intermingling of different arguments and opinions, we opted for sentence level annotation of these labels using the regex pattern. The sentences not matching any criteria are given a tag of None. Next we try to replace the None by extending the tags of the sentences to paragraph level as long as no other tag is encountered. As the judge’s opinion mostly occurs at the end, we start iterating from end and start marking the None as judge’s opinion. If a label which is neither None nor judge’s opinion is encountered, the document is discarded as we cannot extract the judge’s opinion from the document using the process defined. If the judge’s opinion label is found in reverse iteration, then we claim that judge’s opinion can be extracted. Finally, all sentences labelled as judge’s opinion either during reverse
iteration or during paragraph level extension are extracted as judge’s summary and rest of the sentences form facts and opinions for further modelling. Using the above process, following are some cases where the judge’s opinion cannot be extracted:

1. Certain characters were mis-identified in the OCR pipeline and hence do not match the regex.

2. The segmentation of document into header, body and result caused a significant portion of the body and thus judge’s opinion to move to result section.

3. The document was written from judge’s perspective and hence judge’s summary also contains the prosecutor’s and defendant’s arguments.

4. The regex didn’t have 100% coverage.

<table>
<thead>
<tr>
<th>Field</th>
<th>Hindi phrases</th>
<th>English Translations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Judge’s Summary</td>
<td>उभय शत्रु की बहस मूनने, पत्राचार्य के अवलोकन, केस डायरी में उप-लव साक्ष्य के अनुसार, मामले के तथ्यों व परिस्थितियों में पूरी तरह से स्पष्ट है, प्रथम सूचना रिपोर्ट, पुलिस प्रश्न ...परीक्षण किया</td>
<td>Hearing the arguments of the parties, perusal of the records, as per the evidence available in the case diary, fully clear from the facts and circumstances of the case, First Information Report, Police Forms ...perused</td>
</tr>
<tr>
<td>Prosecutor</td>
<td>जमानत का विरोध करते हुये अभियोजन की ओर से तक हिरा गया है, जमानत प्राधिकार के विरुद्ध आपति</td>
<td>Opposing the bail, it has been argued on behalf of the prosecution, the objection against the bail application</td>
</tr>
<tr>
<td>Defendant</td>
<td>अभियुक्त के बित्रान अभियोजन का तक है, में झटका एक रजिस्तान फसाया गया</td>
<td>The learned counsel for the accused has argued, has been falsely and enmity implicated in this case</td>
</tr>
</tbody>
</table>

Table 4.1: Phrases used to construct regular expression for extracting judge’s opinion. The list is just an indicative of the various phrases and variants used; the entire list can be found in code.

4.2.4 Facts and Arguments

This section comprised of facts related to case, arguments from defendant and prosecutor. Mostly, this corresponds to the portion of the body after removing judge’s summary.
4.2.5 Extracting Bail Decision from Result

To extract the bail decision we searched for keywords in result section. Keywords like कार्जिंग (dismissed) and निरस्त्त (invalidated) identified rejection of bail application and words like स्वीकार (accepted) identified acceptance of bail application. Table 4.2 lists all the tokens used for extraction.

4.2.6 Extracting Bail Amount from Result

In case of granted bail decision, the judge specifies bail amount. We saw that the bail amount mostly comprises of personal bond money and surety money. There can be multiple personal bonds and sureties. The bail amount we extracted refers to the sum of all the personal bond money. Bail amount was present in two forms in result section, numerical and Hindi-text. Numerical bail amount was extracted by regex matching and text bail amount was extracted by creating a mapping for it. Table 4.3 shows few examples of bail mapping.
<table>
<thead>
<tr>
<th>Field</th>
<th>Tokens</th>
</tr>
</thead>
</table>
| Bail granted tokens | स्वीकार किया जाता (is being accepted)  
स्वीकार करते हुए (by accepting)  
स्वीकार किये जाते (accepted)  
रिहा किए जाने का आदेश दिया जाता (ordered to be released)  
स्वीकार किये जाने योग्य है (deserves to be accepted)  
प्रमाण आधार प्रतीत होता है (sufficient evidence found)  
प्रमाण आधार पाता हूँ (I see sufficient evidence)  
आधार प्रमाण है (sufficient evidence there)  
प्रमाण आधार दर्शित होता (we see sufficient evidence)  
रिहा किये जाने का आदेश दिया जाता (ordered to be released)  
रिहा किया जाये (should be released)  
रिहा किया जाए (should be released)  
मुक्त किया जाता (should be released)  
रिहा कर दिये जाये (should be released)  
रिहा किया जाता है (is being released)  
रिहा किया जाये (should be released)  
रिहा कर दिया जाये (is being released) |
| Bail denied tokens | निष्पत्त किया जाता (is canceled)  
निष्पत्त किये जाते (are canceled)  
निष्पत्त किए जाते (are canceled)  
खण्डित किया जाता (is broken)  
खण्डित किये जाते (are broken)  
प्रमाण आधार नहीं है (insufficient evidence)  
प्रमाण आधार प्रतीत नहीं होता (insufficient evidence)  
खारिज किया जाता (is rejected)  
अस्वीकार (denied) |

Table 4.2: Mapping of Bail decision tokens to the binary label of bail granted or denied.

<table>
<thead>
<tr>
<th>Text Amount</th>
<th>In Value Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-5 हजार</td>
<td>10000</td>
</tr>
<tr>
<td>बीस-बीस हजार</td>
<td>40000</td>
</tr>
<tr>
<td>पच्चीस-पच्चीस हजार</td>
<td>50000</td>
</tr>
<tr>
<td>तीस-तीस हजार</td>
<td>60000</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Table 4.3: Mapping of the word tokens to the corresponding total bail amount.
4.3 Sample Segmented Document

<table>
<thead>
<tr>
<th>Field</th>
<th>Example</th>
<th>Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header:</td>
<td>This chunk of the document contains meta information related to the case like court hearing date, IPC sections attached, police station of complain, etc.</td>
<td>- Court Special Judge (SC, / ST Act) Allahabad. Bail Petition Number - 4438/2020 C. N. R. No. UPAD01008173-2020 NER Unknown Uttar Pradesh State Lawsuit Offense Number - 773/2020 Article 376, 377, 504 &amp; 506 Indian Penal Code &amp; Article 3(2)V SC /ST Act Station - Sorav, Prayagraj. 04.03.2021 This bail petition has been given of behalf of petitioner/acused NER son NER, resident of Dhamapur Abdalpur, Station - Sorav, District - Prayagraj Offense number 773/2020 Article 376, 377, 504 &amp; 506 Indian Penal Code &amp; Article 3(2) V SC /ST Act in Station - Sorav, Prayagraj which is endorsed by the affidavit of the pairokar father NER of the accused.</td>
</tr>
</tbody>
</table>

Continued on next page
<table>
<thead>
<tr>
<th>Field</th>
<th>Example</th>
<th>Translation</th>
</tr>
</thead>
</table>
| Facts and Arguments: This chunk of the document contains case facts related to the case and arguments from defendant and prosecutor. | राय्य में अभियोजन तथाय इस प्रकार है कि वादित्र का सम्पर्क <name> <name> पुत्र <name> नाखु <name> के साथ 66 वर्ष पूर्व हुआ। <name> <name> वादित्र के घर दूर पहुंचने आया था। ... वादित्र लोक लाजके <name> शात रही। वादित्र मुख्य <name> की तहसील <name> मुख्य पेपरकर्ता किया गया। एमिमेकुल की ओर से यह तरह <name> गया कि प्राथमिक का कभी कोई भीआपत्तिक दावा नहीं है। ... सामान्य आरोप है जो दृष्ट ब फर्मनी है। ... रान <name> <name> लोक का करने के लिए पीड़ितदाता दुरा एवं फर्मनी मुख्य <name> दर्ज कराया गया है। तहसील में वादित्र के हस्ताक्षर दत्तात्रेय का उल्लेख नहीं है। प्राथमिक निर्देश है और उसे दृष्टा पेपरकर्ता गया है। प्राथमिक दिनक 07.10.2020 से देर में निर्देश है। प्राथमिक जमानत देने को तैयार है। अतः प्राथमिकों जमानत <name> तहसील किया जाय | अभियोजन पव की ओर से विक्षेप लोक अभियोजन ने जमानत का विरोध करते हुए कहा है कि ... अभियोजन बाराकालित अध्ययन गंभीर <name> का है। अतः अभियोजन का जमाना अव्यवधान निर्रानकिया जाया। | In short, prosecution facts are such that, the litigant (female) came in contact with NER son of NER 6 years ago. NER used to come to litigant’s (female) house for delivering milk. Paraphrase ...Litigant (female) kept quiet due to public shame. Complaint was registered on basis of case record of litigant (female). It was argued on behalf of the accused that the petitioner has no previous crime record. Paraphrase ... it is a complete accusation which is untrue and fictitious. Paraphrase... With the intention of getting money, in order to blackmail, a false and fictitious complaint was filed by the victim (female). There is no mention of the signature of the litigant (female) and date on record. Petitioner is innocent and is being falsely framed. Petitioner is lodged in jail since 07.10.2020. Petitioner is ready to give bail. Hence petitioner be released on bail. On behalf of the prosecution party, refusing the bail application, special public prosecutor has said that... paraphrase ... the crime committed by the accused is of serious nature. So the bail petition of the accused be overruled.
Table 4.4 – continued from previous page

<table>
<thead>
<tr>
<th>Field</th>
<th>Example</th>
<th>Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Judge’s Opinion:</td>
<td>नैं अभियुक्त की ओर से उल्लिखित विधिक आदेश के अनुसार सम्पत्ति विवेचन लेकर अभियुक्त की ओर से समस्त प्राप्त अवसरों के अनुसार की गई थी। जांचकर्ता / अभियुक्त ने पूछे उसके बाद उसके साथ बललाभ एवं पारंपरिक कुर्य व्यवस्था तथा जानकारी देते हुए &lt;नाम&gt; के मार्ग की ध्वस्तकिरण। यह अपराध एक नातेला और समाज के &lt;नाम&gt; है, जो कमी वर्षों का है।</td>
<td>I heard the learned advocate representing the accused and the special public prosecutor representing the State and examined all documents. The victim (female) was raped and she was subjected to unnatural acts under the pretext of marriage and was subjected to caste-slurred abuse and threatened to be killed by the petitioner/accused. This crime is against a woman and society which is of serious nature.</td>
</tr>
<tr>
<td>Result:</td>
<td>अत: नामले के समान तथ्य, परिस्थितियों एवं अवसर की समीक्षा को देखते हुए प्राप्त अभियुक्त का अनुसार &lt;नाम&gt; छोड़ जाने का आवश्यकता प्रतिवेदन और अभियुक्त का जन्मादि का निर्णय निर्णय जाने &lt;नाम&gt; है। आदेश अभियुक्त राखिए &lt;नाम&gt; पूरा &lt;नाम&gt; नाम &lt;नाम&gt; नाम जाने का जन्मादि प्राप्ति पर निर्णय किया जाता है। (रामकेश) शिलेश यांत्याकेश (एसबीएफ/एसबीएफ एट) इलाहाबाद</td>
<td>JCO Code- UP592</td>
</tr>
</tbody>
</table>

Table 4.4: A sample segmented document with header, facts and arguments, judge’s opinion and result section.
4.4 Segmentation Pipeline Verification

We verified each step of the corpus creation pipeline to ensure the quality of the data. We initially started with 363,003 bail documents across all the 71 districts of U.P., and after removing documents having segmentation errors, we have a Bail corpus with 176,849 bail documents. The bail corpus has a total of 2,342,073 unique tokens, and on average, each document has 614 tokens.

We used a validation set (0.1% of data) to evaluate our regex based approaches, the results are in Table 4.5. Note that metrics used for evaluation are quite strict and hence the results are much lower for Judge’s summary part. The segmentation and Judge’s opinion were strictly evaluated and even a single sentence in the wrong segment reduces the accuracy. We also see that the main binary label of outcome detection (bail granted or denied) had an almost perfect accuracy of 99.4%. Nevertheless, future works can improve the pipeline further by adopting a hybrid approach and training ML models in addition to traditional regex patterns.

<table>
<thead>
<tr>
<th>Process</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header, Body and Case Result Segmentation</td>
<td>89.7%</td>
</tr>
<tr>
<td>Judge’s Opinion and Facts extraction</td>
<td>85.7%</td>
</tr>
<tr>
<td>Bail Decision Extraction</td>
<td>99.4%</td>
</tr>
</tbody>
</table>

Table 4.5: Evaluation results of bail document division and bail decision extraction pipeline.
Chapter 5

Models for Bail Prediction

In this chapter, after the creation of the bail corpus, we can now tackle the bail prediction task. We first formally define the prediction task in terms of the extracted segments from the legal document before building models to solve the task of bail prediction. Fine-tuning large language models pre-trained on massive general domain corpora has significantly improved state-of-the-art results for various domain-specific natural language processing (NLP) tasks [Gu et al., 2021]. However, every domain-specific corpus has its own set of differentiating characteristics: vocabulary, which consists of specialized tokens unique to the domain in hand, structure, and size of the documents, and lastly, the ambiguity in the interpretation of words or phrases based on context. HLDC is an example of such a highly specialized dataset, which not only consists of legal domain documents but with many other unique characteristics such as the language (Hindi), long document length, and the inherent diverse writing styles followed across different districts in UP. These niche properties make it challenging to apply off-the-shelf pre-trained transformer models directly. Hence, in this chapter we setup multiple baseline models before going into detail of our proposed end-to-end trainable multi-task model.

5.1 Bail Prediction Task Formulation

To demonstrate a possible applicability for HLDC, we propose the Bail Prediction Task, where given the facts of the case, the goal is to predict whether the bail would be granted or denied. Formally, consider a corpus of bail documents $\mathcal{D} = b_1, b_2, \ldots, b_i$, where each bail document is segmented as $b_i = (h_i, f_i, j_i, y_i)$. Here, $h_i$, $f_i$, $j_i$ and $y_i$ represent the header, facts, judge’s summary and bail decision of the document respectively. Additionally, the facts of every document contain $k$ sentences, more formally, $f_i = (s_{i1}^k, s_{i2}^k, \ldots, s_{ik}^k)$, where $s_{ik}^k$ represents the $k^{th}$ sentence of the $i^{th}$ bail document. We formulate the bail prediction task as a binary classification problem. We are interested in modelling
\( p(y_i|f_i) \), which is the probability of the outcome \( y_i \) given the facts of a case \( f_i \). Here, \( y_i \in \{0, 1\} \), i.e., 0 if bail is denied or 1 if bail is granted.

5.2 Embedding Based Models

We experimented with classical embedding based model Doc2Vec [Le and Mikolov, 2014] and transformer-based contextualized embeddings model IndicBERT [Kakwani et al., 2020]. Doc2Vec embeddings, in our case, are trained on the train set of our corpus. The embeddings go as input to SVM and XGBoost classifiers. IndicBERT is a transformer language model trained on 12 major Indian languages. However, IndicBERT, akin to other transformer LMs, has a limitation on the input’s length (number of tokens). Inspired by [Malik et al., 2021b; Chalkidis et al., 2019], we experimented with fine-tuning IndicBERT in two settings: the first 512 tokens and the last 512 tokens of the document. The fine-tuned transformer with a classification head is used for bail prediction. Further details for the models are discussed in the following sections.

5.2.1 Doc2Vec

Numerical representation of written text is a critical component of any NLP application. Word2Vec was a pioneering approach to create numerical representations for words [Mikolov et al., 2013]. Doc2Vec built upon it to represent document as vectors [Le and Mikolov, 2014].

We used gensim\(^1\) library to train the embedding model. The training was done on the done on the training split only to avoid leakage of information from the test split. We test the model of the test set. The training was done for 100 epochs on the training split to generate the vocab and embeddings. The hyperparameters used were distributed-memory algorithm with the concatenation of context vectors rather than sum/average. Minimum count for words in vocabulary was set to 5. The vector size was set to 400 and alpha was set to 0.015. For training common hindi stopwords were removed. Using these embeddings for the train set we train two classifiers SVM and XGBoost from sklearn library. We train these classifiers for the bail prediction task with the Doc2Vec embeddings as feature vectors. The doc2vec vectors for the test set was inferred using the infer_vector() of the gensim library which predicts a vector for an unseen document and we predict the label by passing it through the classifiers.

\(^1\)https://github.com/RaRe-Technologies/gensim
5.2.2 IndicBERT

Due to the multilingual nature of our dataset, many existing models like BERT, RoBERTa and XL-Net trained on monolingual corpora fail to generalise to our task. Instead, we utilise the IndicBERT [Kakwani et al., 2020] model which is trained on large scale corpora encompassing 12 major Indian languages. However, transformer models are not readily applicable on long documents due to the bottleneck on their self-attention mechanism which limits the number of input tokens to the model. To account for this, we experimented with 2 strategies of finetuning IndicBERT:

- **First 512 Tokens**: We take only the first 512 tokens from the facts of the document and truncate the rest.

- **Last 512 Tokens**: We take only the last 512 tokens from the facts of the document and truncate the rest.

5.3 Summarization Based Models

The models described in the previous section either do not take into account the context of all the facts (static embeddings) or lose important input signals due to truncation which can be detrimental to the bail prediction task. In order to alleviate this issue, we generate extractive summaries of the long bail documents which helps us in two ways:

1. It shortens the long bail document such that it can fit into the input token length constraint

2. It can be used to qualitatively evaluate the most salient sentences from the bail document which contribute the most towards determining the outcome of the case.

In this section, we describe the two step models used for the bail prediction task. The first step generates the extractive summaries of the document while the second step trains a model to predict the outcome, similar to the models described in the previous section. We experimented with both unsupervised and supervised extractive summarization models for the first step, while we stick with IndicBERT as part of the classification model for the second step which we then fine-tune in order to enable fair comparison with the other baselines.

In particular, an extractive summary of a document goes as input to a fine-tuned transformer-based classifier (IndicBERT). Besides reducing the length of the document, extractive summarization helps to
evaluate the salient sentences in a legal document and is a step towards developing explainable models. The unsupervised and supervised approaches are elaborated below.

- **Unsupervised:**
  - **TF-IDF+IndicBERT:** We rely on the statistical measures of term frequency and inverse document frequency [Ramos, 2003] to recognize the most salient words and sentences.
  - **TextRank+IndicBERT:** We use the graph-based method TextRank [Mihalcea and Tarau, 2004] to extract the most important sentences from the documents.

- **Supervised:** **Salience+IndicBERT** In order to train an extractive summarizer on the legal documents, we take inspiration from [Bajaj et al., 2021], to train a simple saliency classifier based on sentences ranked through the cosine similarity between sentences from the document and the summary. We take a subset from the training data to use in building the saliency classifier. For each document \( b_i \) from this salience training data, we consider \( j_i \) as its summary and rank each sentence \( s^k_i \) in \( f_i \) based on its cosine similarity with \( j_i \). More formally, we assign a salience score to every fact as follows:

\[
\text{salience}(s^k_i) = \cos_{sim}(j_i, s^k_i)
\]

where \( \cos_{sim} \) is the cosine similarity function. We use the multilingual sentence encoder [Reimers and Gurevych, 2020] to generate embeddings for \( j_i \) and \( s^k_i \). Once we have the scores for each fact in the document, we sort them in descending order of their salience and take out the top half sentences to label them most salient. To build negative samples for the saliency classifier, we also take the bottom half sentences and label them as non-salient. Based on this binary saliency data, we train a Transformer Encoder [Vaswani et al., 2017] layer along with a single fully-connected saliency classification head to predict the salience of every sentence \( s^k_i \) in fact \( f_i \).

### 5.4 Multi-Task Learning (MTL) Model

As observed during experiments, summarization-based models show improvement in results (§5.5). Inspired by this, we propose a multi-task framework (Figure 5.1), where bail prediction is the main task, and sentence salience classification is the auxiliary task. The intuition is that predicting the important sentences via the auxiliary task would force the model to perform better predictions and vice-versa.
Figure 5.1: Overview of our multi-task learning approach, with bail prediction as the primary task and salient sentence identification as the auxiliary task.

Input to the model are sentences corresponding to the facts of a case: $s_1^i, s_2^i, \ldots, s_k^i$. A multilingual sentence encoder [Reimers and Gurevych, 2020] is used to generate contextualized representation of each sentence: $\{h_1^i, h_2^i, \ldots, h_k^i\}$. In addition, we append the sentence representations with a special randomly initialized CLS embedding [Devlin et al., 2019] that gets updated during model training. The CLS and sentence embeddings are fed into standard single layer transformer (shared transformer).

### 5.4.1 Bail Prediction Task

We concatenate a special trainable classification token $c \in \mathbb{R}^m$, at the beginning of the contextual fact embeddings $h_k^i$, thus making the input to the Transformer Encoder Layer $\{c, h_1^i, h_2^i, \ldots, h_k^i\}$. $c$ is randomly initialized before training starts and its parameters are trained during backpropagation to maximise the bail prediction accuracy. Similar to [Devlin et al., 2019]’s methodology, we regard the final hidden state $\hat{c}$ of this token as the aggregate sequence representation for the input sequence. We employ a single fully-connected layer followed by cross-entropy loss as the Bail-prediction classifier head, which takes as input the ($\hat{c}$) and calculates the loss ($L_{\text{bail}}$) for the main task - bail prediction.

### 5.4.2 Salience Classification Task

We use the salience prediction head (a Multi-Layer Perceptron) on top of sentence representations at the output of the shared transformer. For training the auxiliary task, we use sentence salience scores obtained via cosine similarity (these come from supervised summarization based model). For each
Table 5.1: Number of documents in the train, validation and test sets across the district wise and all district data split.

<table>
<thead>
<tr>
<th></th>
<th>Granted</th>
<th>Dismissed</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>All Districts</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Train</td>
<td>77010 (62%)</td>
<td>46732 (38%)</td>
<td>123742</td>
</tr>
<tr>
<td>Test</td>
<td>21977 (62%)</td>
<td>13423 (38%)</td>
<td>35400</td>
</tr>
<tr>
<td>Validation</td>
<td>11067 (63%)</td>
<td>6640 (37%)</td>
<td>17707</td>
</tr>
<tr>
<td><strong>District Wise</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Train (44 districts)</td>
<td>77220 (62%)</td>
<td>47121 (38%)</td>
<td>124341</td>
</tr>
<tr>
<td>Validation (10 districts)</td>
<td>9563 (60%)</td>
<td>6366 (40%)</td>
<td>15929</td>
</tr>
<tr>
<td>Test (17 districts)</td>
<td>23271 (64%)</td>
<td>13308 (36%)</td>
<td>36579</td>
</tr>
</tbody>
</table>

sentence, we use binary-cross entropy loss ($L_{\text{salience}}$) to predict the salience. Based on our empirical investigations, both the losses are equally weighted, and total loss is given by $L = L_{\text{bail}} + L_{\text{salience}}$.

5.5 Experiments and Results

5.5.1 Dataset Splits

We evaluate the models in two settings: all-district performance and district-wise performance. For the first setting, the model is trained and tested on the documents coming from all districts. The train, validation and test split is 70:10:20. The district-wise setting is to test the generalization capabilities of the model. In this setting, the documents from 44 districts (randomly chosen) are used for training. Testing is done on a different set of 17 districts not present in train set. The validation set has another set of 10 districts. This split corresponds to a 70:10:20 ratio. Table 5.1 provides the number of documents across splits. The corpus is unbalanced for the prediction class with about 60:40 ratio for positive to negative class (Table 5.1). All models are evaluated using standard accuracy and F1-score metric.

Implementation Details: All models are trained using GeForce RTX 2080Ti GPUs. Models are tuned for hyper-parameters using the validation set. We used Optuna \(^2\) for hyperparameter optimisation. Optuna allows us to easily define search spaces, select optimisation algorithms and scale with easy parallelization. We run parameter tuning on 10\% of the data to identify the best parameters before retraining the model with the best parameters on the entire dataset. The best parameters are listed in Table 5.2.

\(^2\)https://github.com/optuna/optuna
Table 5.2: List of Hyper-Parameters used for the final trained models.

<table>
<thead>
<tr>
<th>Model</th>
<th>Hyper-Parameters (L=Learning Rate), (E=Epochs), (D=Embedding Dimension(Default 200)), (W= Weight Decay), (E=Epochs(Default 15))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doc2Vec + SVM</td>
<td>District-wise Split: E=100</td>
</tr>
<tr>
<td>Doc2Vec + XGBoost</td>
<td>E=100, D=300</td>
</tr>
<tr>
<td>IndicBert - (First 512 Tokens)</td>
<td>L=3.69 × 10^{-6}, W=2.6 × 10^{-2}</td>
</tr>
<tr>
<td>IndicBert - (Last 512 Tokens)</td>
<td>L=5.60 × 10^{-5}, W=1.0 × 10^{-2}</td>
</tr>
<tr>
<td>TF-IDF + IndicBert</td>
<td>L=1.11 × 10^{-5}, W=1.9 × 10^{-2}</td>
</tr>
<tr>
<td>TextRank + IndicBert</td>
<td>L=3.17 × 10^{-6}, W=3.1 × 10^{-2}</td>
</tr>
<tr>
<td>Salience Pred. + IndicBert</td>
<td>L=1 × 10^{-5}, W=3.2 × 10^{-2}</td>
</tr>
<tr>
<td>Multi-Task</td>
<td>E=30, L=5 × 10^{-5}</td>
</tr>
</tbody>
</table>

5.5.2 Results

Results are shown in Table 5.3. As can be observed, in general, the performance of models is lower in the case of district-wise settings. This is possibly due to the lexical variation across districts, which makes it difficult for the model to generalize. Moreover, this lexical variation corresponds to the usage of words corresponding to dialects of Hindi. Another thing to note from the results is that, in general, summarization based models perform better than Doc2Vec and transformer-based models, highlighting the importance of the summarization step in the bail prediction task. The proposed end-to-end multi-task model outperforms all the baselines in the district-wise setting with 78.53% accuracy. The auxiliary task of sentence salience classification helps learn robust features during training and adds a regularization effect on the main task of bail prediction, leading to improved performance than the two-step baselines. However, in the case of an all-district split, the MTL model fails to beat simpler baselines like TF-IDF+IndicBERT. We hypothesize that this is due to the fact that the sentence salience training data may not be entirely correct since it is based on the cosine similarity heuristic, which may induce some noise for the auxiliary task. Additionally, there is lexical diversity present across documents from different districts. Since documents of all districts are combined in this setting, this may introduce diverse sentences, which are harder to encode for the salience classifier, while TF-IDF is able to look at the distribution of words across all documents and districts to extract salient sentences.
<table>
<thead>
<tr>
<th>Model Name</th>
<th>District-wise</th>
<th>All Districts</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Acc.</td>
<td>F1</td>
</tr>
<tr>
<td>Doc2Vec + SVM</td>
<td>0.72</td>
<td>0.69</td>
</tr>
<tr>
<td>Doc2Vec + XGBoost</td>
<td>0.68</td>
<td>0.59</td>
</tr>
<tr>
<td>IndicBert-First 512</td>
<td>0.65</td>
<td>0.62</td>
</tr>
<tr>
<td>IndicBert-Last 512</td>
<td>0.62</td>
<td>0.60</td>
</tr>
<tr>
<td>TF-IDF+IndicBert</td>
<td>0.76</td>
<td>0.74</td>
</tr>
<tr>
<td>TextRank+IndicBert</td>
<td>0.76</td>
<td>0.74</td>
</tr>
<tr>
<td>Salience Pred.+IndicBert</td>
<td>0.76</td>
<td>0.74</td>
</tr>
<tr>
<td>Multi-Task</td>
<td><strong>0.78</strong></td>
<td><strong>0.77</strong></td>
</tr>
</tbody>
</table>

Table 5.3: Model results. For TF-IDF and TextRank models we take the sum of the token embeddings.

### 5.5.3 Error Analysis

We did further analysis of the model outputs to understand failure points and figure out improvements to the bail prediction system. After examining the misclassified examples, we observed the following. First, the lack of standardization can manifest in unique ways. In one of the documents, we observed that all the facts and arguments seemed to point to the decision of bail granted. Our model also gauged this correctly and predicted bail granted. However, the actual result of the document showed that even though initially bail was granted because the accused failed to show up on multiple occasions, the judge overturned the decision and the final verdict was bail denied. In some instances, we also observed that even if the facts of the cases are similar the judgements can differ. We observed two cases about the illegal possession of drugs that differed only a bit in the quantity seized but had different decisions. The model is trained only on the documents and has no access to legal knowledge, hence is not able to capture such legal nuances. We also performed quantitative analysis on the model output to better understand the performance. Our model outputs a probabilistic score in the range \( \{0, 1\} \). A score closer to 0 indicates our model is confident that bail would be denied, while a score closer to 1 means bail granted. In Figure 5.2 we plot the ROC curve to showcase the capability of the model at different classification thresholds. ROC plots True Positive and False Positive rates at different thresholds. The area under the ROC curve (AUC) is a measure of aggregated classification performance. Our proposed model has an AUC score of 0.85, indicating a high-classification accuracy for a challenging problem.
Figure 5.2: ROC curve for the proposed model. The total AUC (Area under curve) is 0.85.

We also plot (Figure 5.3) the density functions corresponding to True Positive (Bail correctly granted), True Negative (Bail correctly dismissed), False Positive (Bail incorrectly granted) and False Negatives (Bail incorrectly dismissed). We observe the correct bail granted predictions are shifted towards 1, and the correct bail denied predictions are shifted towards 0. Additionally, the incorrect samples are concentrated near the middle ($\approx 0.5$), which shows that our model was able to identify these as borderline cases.

### 5.6 Challenges and Limitations

There are certain unique challenges in building models for bail prediction on the HLDC data. We expand on the three major challenges in this section

- **Domain Specific Knowledge** - Court documents contain very specific lexicon. The technical legal jargon used in case documents means that pre-trained models built on other corpus like wikipedia or tweets fail.

- **Low Resource Language** - Our case documents were present in Hindi. Despite being the third most spoken language in the world [Ethnologue, 2022], Hindi remains a low resource language in the context of NLP with limited high quality training data and pre-trained models. This means
Figure 5.3: Kernel Density Estimate (KDE) plots of our proposed bail prediction model. The majority of errors (incorrectly dismissed / granted) are borderline cases with model output score around 0.5.

even fundamental NLP tasks like NER (Named Entity Recognition) are challenging due to unavailability of training data, richer morphology and lack of capitalisation in Hindi [Athavale et al., 2016].

• Long Documents - Court cases can be extremely long in nature spanning tens of pages. One of the constraints for transformer based models is the input length constraints ( \( \leq 512 \) tokens). Thus handling large legal documents is challenging for existing architectures and we tried out a number of creative ways to overcome this constraint.
Chapter 6

Ethical Considerations

We create HLDC to promote research and automation in the legal domain dealing with under-researched and low-resource languages like Hindi. The documents that are part of HLDC are in the public domain and hence accessible to all. Given the volume of pending cases in the lower courts, our efforts are aimed towards improving the legal system, which in turn would be beneficial for millions of people. Our work is in line with some of the previous works on legal NLP, e.g., legal corpora creation and legal judgement prediction (section 2). Nevertheless, we are aware that if not handled correctly, legal AI systems developed on legal corpora can negatively impact an individual and society at large. Consequently, we took all possible steps to remove any personal information and biases in the corpus. We anonymized the corpus (Chapter 3) with respect to names, gender information, titles, locations, times, judge’s name, petitioners and appellant’s name. As observed in previous work Malik et al. [2021b], anonymization of a judge’s name is important as there is a correlation between a case outcome and a judge name. Along with the HLDC, we also introduce the task of Bail Prediction. Bail applications constitute the bulk of the cases (3), augmentation by an AI system can help in this case. The bail prediction task aims not to promote the development of systems that replace humans but rather the development of systems that augment humans. The bail prediction task provides only the facts of the case to predict the final decision and avoids any biases that may affect the final decision. Moreover, the Bail corpus and corresponding bail prediction systems can promote the development of explainable systems [Malik et al., 2021b], we leave research on such explainable systems for future work. The legal domain is a relatively new area in NLP research, and more research and investigations are required in this area, especially concerning biases and societal impacts; for this to happen, there is a need for corpora, and in this Thesis, we make initial steps towards these goals.
Chapter 7

Conclusion and Future Works

In this work, we introduced a large corpus of legal documents for the under-resourced language Hindi: Hindi Legal Documents Corpus (HLDC). We semi-structure the documents to make them amenable for further use in downstream applications. We then introduce the task of Bail Prediction. We experimented with several models and proposed a multi-task learning based model that predicts salient sentences as an auxiliary task and bail prediction as the main task. Results show scope for improvement that we plan to explore in future. The thesis lays the ground for future work in the legal NLP sphere for Indian languages. Now, we explore the possible future directions to build upon this thesis.

1. Multiple Languages

We plan to expand HLDC by covering other Indian Hindi speaking states. Furthermore, as a future direction, we plan to collect legal documents in other Indian languages. India has 22 official languages, but for the majority of languages, there are no legal corpora. The use of such corpora is not restricted only to the legal domain. It can also act as a source of unsupervised data in low resource languages to train transformer models.

2. Infusing Domain Specific Knowledge  Another interesting future direction that we would like to explore is the development of deep models infused with legal knowledge so that model is able to capture legal nuances. For Indian legal documents, we can incorporate information from the sections and acts in the IPC (Indian Penal Code).

3. Other LegalNLP tasks  Apart from judgement prediction, a number of other tasks like legal summarisation and prior case retrieval can help expedite judicial process.
Related Publications

Bibliography


Yoshinobu Kano, Mi-Young Kim, Randy Goebel, and Ken Satoh. 2017. Overview of coliee 2017. In COLIEE@ICAIL.


Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.


