Ekta Gandotra

d

Quick Submit

Quick Submit

Jaypee University of Information Technology

Document Details

Submission ID

trn:oid:::1:3241005184

Submission Date

May 6, 2025, 10:56 AM GMT+5:30

Download Date

May 6, 2025, 11:09 AM GMT+5:30

File Name

Abstract_to_Refernces.pdf

File Size

1.0 MB

23 Pages

3,974 Words

23,311 Characters

20% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Small Matches (less than 8 words)

Match Groups

31 Not Cited or Quoted 13%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

15 Missing Citation 7%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

16% Internet sources
11% Publications
9% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **31** Not Cited or Quoted 13%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **15** Missing Citation 7%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

16% Internet sources
11% Publications
9% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	
huggingface.co		4%
2	Student papers	
Jaypee University of Information Technology		2%
3	Internet	
github.com		2%
4	Student papers	
University of Bath		1%
5	Student papers	
Liverpool John Moores University		1%
6	Internet	
acl-arc.comp.nus.edu.sg		<1%
7	Internet	
lrec-conf.org		<1%
8	Internet	
arxiv.org		<1%
9	Internet	
www.coursehero.com		<1%
10	Publication	
Saloni Sharma, Surabhi Srivastava, Pradeepika Verma, Anshul Verma, Sachchida ...		<1%

11	Publication	
Nikita Chaudhari, Deepali Vora, Payal Kadam, Vaishali Khairnar, Shruti Patil, Keta...		<1%
12	Internet	
jist.ir		<1%
13	Internet	
aclanthology.org		<1%
14	Internet	
ijor.co.uk		<1%
15	Internet	
ceur-ws.org		<1%
16	Internet	
medium.com		<1%
17	Internet	
preview.aclanthology.org		<1%
18	Internet	
scholar.archive.org		<1%
19	Publication	
Aashish Ghimire, Raj Shrestha, John Edwards. "Too Legal; Didn't Read (TLDR): Sum...		<1%
20	Internet	
digital.library.unt.edu		<1%
21	Internet	
ir.juit.ac.in:8080		<1%
22	Publication	
"PRICAI 2024: Trends in Artificial Intelligence", Springer Science and Business Me...		<1%
23	Student papers	
Amrita Vishwa Vidyapeetham		<1%
24	Publication	
Esaú Villatoro-Tello. "Multi-document Summarization Based on Locally Relevant S...		<1%

25 Publication

Library Hi Tech, Volume 32, Issue 2 (2014-09-16) <1%

26 Internet

pta-dspace-dmz.csir.co.za <1%

27 Publication

"Natural Language Processing and Information Systems", Springer Science and B... <1%

28 Internet

card2brain.ch <1%

29 Internet

journalcrd.org <1%

30 Internet

peerj.com <1%

31 Internet

www.fluidtopics.com <1%

32 Internet

www.hindawi.com <1%

ABSTRACT

Legal documents are often very lengthy, time-consuming, complex and difficult to understand manually, even the legal professionals find it difficult and prone to human error. By using the technology, we are able to fill that gap and make the process much faster and easier. We have applied the Natural Language Processing(NLP) techniques to extract, interpret, and summarize legal content efficiently. Our approach involves extracting text from legal documents and performing preprocessing which includes tasks like tokenization, lemmatization, NER, word embedding and many more, with the help of libraries of python which include nltk, numpy, spacy, pandas etc.

For the summarization part, we combine both extractive and abstractive models. Textrank is used for extractive summarization to identify and extract the most significant sentences from the document and Legal Pegasus is used for abstractive summarization, which produces human-readable paraphrased summaries by comprehending the context and meaning of the content. The dual strategy combines factual completeness with readability. To measure the performance of the summaries produced, we utilize the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metric, which reports the precision, recall, and F1-score of our system with respect to reference summaries.

The tool is a valuable resource for legal experts, law firms, and scholars. This solution greatly cuts down on the time and mental effort of manual legal document reading while improving comprehension and productivity.

Chapter 01: INTRODUCTION

1.1 Introduction

In the rapidly evolving field of law technology there is a need to dill the bridge between technology and law to make a hybrid model which helps in improving the approach toward legal Documents which are very complex, verbose and time consuming to process. Manually reading these documents and finding their summary is inefficient and more likely to cause human errors. This project proposes a hybrid intelligent system that combines rule-based techniques and state-of-the-art Natural Language Processing (NLP) to automate the summarization of legal documents. This tool will help to analyze legal documents like contracts, litigation, property, government and personal documents efficiently and avoid human errors. This model will help in increasing the efficiency and providing an improved approach.

1.2 Objective

The key aim is to create and deploy a system that can read legal documents by extracting Text, identify crucial legal entities, and provide proper summaries with sophisticated Natural Language Processing (NLP) methodologies. The system has the purpose of helping legal experts minimize the workload, avoid mistakes, and accelerate the speed and precision of understanding legal documents and making decisions.

- Converting unstructured legal texts into structured summaries.
- Supporting both **extractive** summarization (highlighting important sentences) and **abstractive** summarization (rewriting the content in a concise and readable form).

1.3 Motivation

The motivation stream from the challenges faced by legal professionals and any help seeker in reviewing length documents and under the context of the documents. Automating this will help to reduce error, save time and enhance the efficiency of research.

- **Time and Efficiency Constraints**: Manual review consumes considerable time that can

otherwise be used for analytical work.

- **Error Reduction**: Human fatigue or bias can lead to missed details or misinterpretation, particularly in high-stakes legal matters.
- **Democratization of Legal Understanding**: Empowering users who lack legal expertise to understand the content of legal documents.

1.4 Language Used

Completely collab based project that use:

- Programming Language: Python
- Model Used: spaCy,TextRank,Pegasus,TF-IDF,N-grams,Scikit-learn, NLTK.
- Libraries/Frameworks: transformers, torch, spacy, summa, pandas, numpy, scikit-learn, nltk, rouge-score, datasets (all these are some good pretrained model in python)

1.5 Technical Requirements (Hardware)

The technical requirements for running our model are relatively modest and include:

- A standard desktop or laptop computer with an internet connection.
- Modern web browsers from where they can access https://colab.research.google.com
- GPU(Recommended): For significantly faster processing google collab is suggested to have GPU access.
- CPU(Alternative): If not able to access GPU, CPU will be automatically accessed but some features like pegasus may not work.

1.6 Deliverables/Outcomes

Upon completion, the project will deliver the following outcomes:

- A full functional model for summarization which will help in better understanding of legal documents.
- Results of each stage will provide a detailed description of how the model works and provide a better understanding.
- Quantitative evaluation using metrics like ROUGE and BLEU to validate the effectiveness of the summaries.

Chapter 02: Feasibility Study, Requirements Analysis and Design

2.1 Feasibility Study

2.1.1 Literature review

To develop a robust hybrid model for legal document summarization, a comprehensive review of existing approaches is essential. This section contains the comparison between different approaches used in prior research and commercial tools. We have prepared hybrid model using this approach which includes the Textrank(Extractive) and Legal pegasus (Abstractive)summaries approach

Table 1.1: Literature review

S	Title	Method Used	Benefits	Limitation
1	Indian Legal Text Summarization: A Text Normalisation-based Approach. [1]	Normalizes legal texts by replacing abbreviations, then uses BART (extractive) & PEGASUS (abstractive) for summarization	Summarization with domain-independent models.	PEGASUS performs poorly when was compared to legal advisor.
2	A High-Precision Two-Stage Legal Judgment Summarization [2]	Two-stage model: Extractive (BERT + LSTM for key sentences; BiLSTM for keywords) + Abstractive (UNILM with attention mechanism)	Handles lengthy texts by extracting key sentences; integrates legal keywords via attention; achieves 0.19–0.41 ROUGE	High computational complexity due to pre-trained models.
3	ROUGE: A Package for Automatic Evaluation of Summaries [3]	ROUGE-L, ROUGE-N, ROUGE-W, ROUGE-S.	Effective for summarization evaluation process.	Struggles with paraphrasing, less effective for abstractive summarization

4	Legal Document Summarization and analysis[4]	Extractive Summarization, Sentence Scoring using TF*IDF matrix, Multinomial Naive Bayes model.	Uses a hybrid summarization model using extractive summarization for key sentence identification and BERT based text classification for improved sentence selection.	TF*IDF lacks understanding of legal documents and BERT may be computationally heavy
5	A Cyclical Approach to Legal Document Analysis: Leveraging AI for Strategic Policy Evaluation[5]	Cyclical AI-Based Processing Framework that combines NLP, ML, and Rule-Based Systems to analyze legal documents.	The cyclic approach iteratively improves the analysis through each pass	High Computational Cost, Error Propagation
6	Case Law Analysis using Deep NLP and Knowledge Graphs[6]	Core technique used in this paper is Deep NLP-Based Legal Text Processing Combined with Knowledge Graphs (KGs).	Mapping case law documents, in particular the opinion and the holding, to KRs allows.	High Complexity in Legal Language, Knowledge Graph Limitations
7	LEGALSEVA - AI-POWERED LEGAL DOCUMENTATION ASSISTANT[7]	- Custom Trained GPT (NLP-based) - Optical Character Recognition (OCR) integration - User interaction through bot for document creation	- Enhances accessibility and reduces complexity of legal jargon - Time-saving and cost-effective for small businesses and individuals	- Privacy and data security concerns - Potential scalability issues for expanding to other legal domains

8	Enhancing Legal Document Analysis and Judgment Prediction with Machine Learning and Deep Learning Techniques[8]	Texas Wolf Optimization (TWO) + BiLSTM (Deep Learning)	Achieves 97% accuracy and 97.29% F1-score using TWO-BiLSTM; outperforms traditional and other advanced models	Performance may vary across different legal domains; requires annotated datasets and computational resources
9	An Overview of Information Extraction Techniques for Legal Document Analysis and Processing[9]	Comparative analysis of NLP-based, deep learning-based, and knowledge base population (KBP) approaches for legal text processing.	KBP approach effectively handles complex legal language. Enhances legal text retrieval through NLP-based and deep learning models.	Legal documents have varied structures, high computational cost for deep learning-based methods.
10	Summarization of Indian Legal Judgement Documents via Ensembling of Contextual Embedding based MLP Models[10]	Machine learning, NLP, rhetorical role labeling, summary-worthiness detection.	Improved legal document summarization, structured case analysis, efficient retrieval.	Limited dataset size, potential bias in labeling, domain-specific applicability
11	Legal Document Summarization Using Nlp and Ml Techniques[11]	NLP, ML, TextRank (based on PageRank), TF-IDF, Word2Vec, BERT, Extractive & Abstractive Summarization	Saves time, retains key legal info, aids lawyers & researchers	Complex legal language, bias, accuracy issues, high computations
12	Summarization of Lengthy Legal Documents via Abstractive Dataset Building: An	Extract-then-Assign: Key sentence extraction + Abstractive summarization using NLP &	Faster legal document processing, improved summarization	Requires high-quality training data, may lose key legal details, computationally expensive, potential

	Extract-then-Assign Approach[12]	Transformer-based models (BERT, GPT, T5)	accuracy, structured dataset for training	legal misinterpretations

2.1.2 Problem Definition

Legal professionals face numerous challenges in reviewing, understanding, extracting legal info related to legal contracts, arguments that there is a need for a system which can intelligently and effectively interpret legal texts, identify key information and provide concise summaries in a timely and reliable manner.

2.1.3 Problem Analysis

The primary issue with manual legal document handling is:

- **High Complexity**: High load due to complex and verbose language.
- **Information Overload**: Increased chances of missing important information.
- **Time Consumption**: Time consumed while analysing long legal documents.
- **Human Error**: Fatigue or bias may lead to misinterpretation or oversight of crucial legal points.

2.1.4 Solution

We are proposing a Legal Document summarization that intelligently implements a hybrid model of both extractive and abstractive summarization techniques to produce concise summaries, ensuring accuracy, of complex documents in the legal space.

Model Pipeline:

The model pipeline consists of the following steps:

- **Text Extraction & Preprocessing-** Extraction of legal text from multiple formats followed by cleaning, normalization, and fragmentation.
- **Keyword and Sentence Analysis-** Utilization of TF-IDF and n-gram analysis to discover meaningful patterns and important sentences.
- **Summarization Engine-** Two paths for processing for:
    1. **Extractive Summarization:**

Models compared: TextRank, LexRank

Best performer: TextRank (on ROUGE score)

2. **Abstractive Summarization:**

   Models compared: LegalPegasus, LegalBART

   Best performer: LegalPegasus (on ROUGE, coherence couched in domain-relevance)

3. **Hybrid Summary**

   Hybrid Summary was generated using the - LegalPegasus (abstractive) + TextRank (extractive). Each of the summarizers complement and improve the summarization results.

2.2 Requirements

2.2.1 Functional Requirements

- Upload legal documents in .pdf or .docx formats or .xlsx format.
- Extract text from documents for further use.
- Clean and preprocess extracted data for NLP tasks.
- Segment text into sentences and tokens.
- Generate summary with help of both extractive and abstractive summaries.
- Display evaluation score for showing the efficiency.
- Export summarized text and metadata in structured formats

2.2.2 Non-Functional Requirements

- Usability: Simple and intuitive interface for non-technical users
- Performance: Should generate summaries in under 10 seconds for average-sized documents
- Scalability: Must be able to handle bulk documents in the future
- Maintainability: Modular code structure for easy updates

2.3 E-R Diagram / Data-Flow Diagram (DFD)

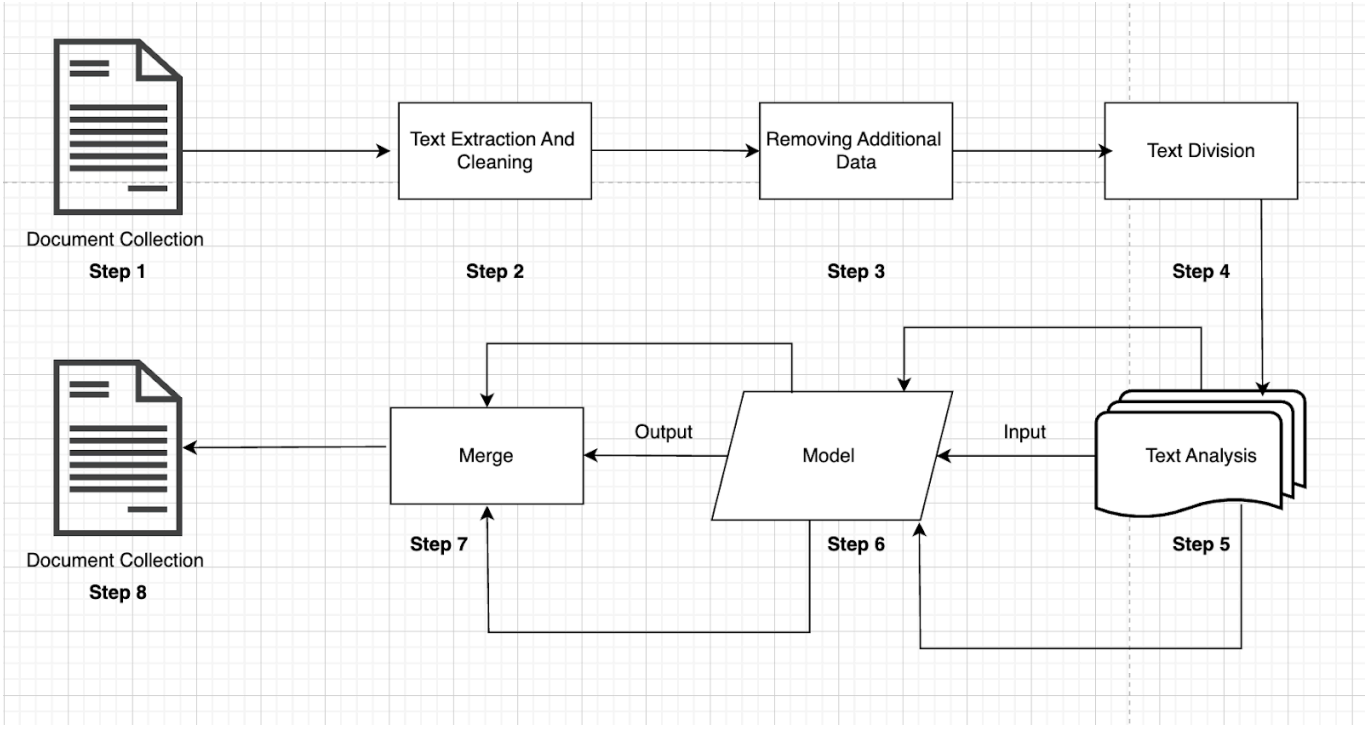


Figure 2.1: WorkFlow Diagram

Figure 2.1 illustrates our preprocessing and processing pipeline for summarizing legal documents:

- Step 1: Gather the legal documents.
- Steps 2-4: Extract the text, clean the text, and normalize the text, removing noise.
- Step 5: Splitting text into the chunks that we would like to feed to the summarization model.
- Step 6. Feed that chunked text into the summarization model.
- Step 7. Combine those outputs from the model into an integrated summary.
- Step 8: Add the summaries back to the collection of documents.

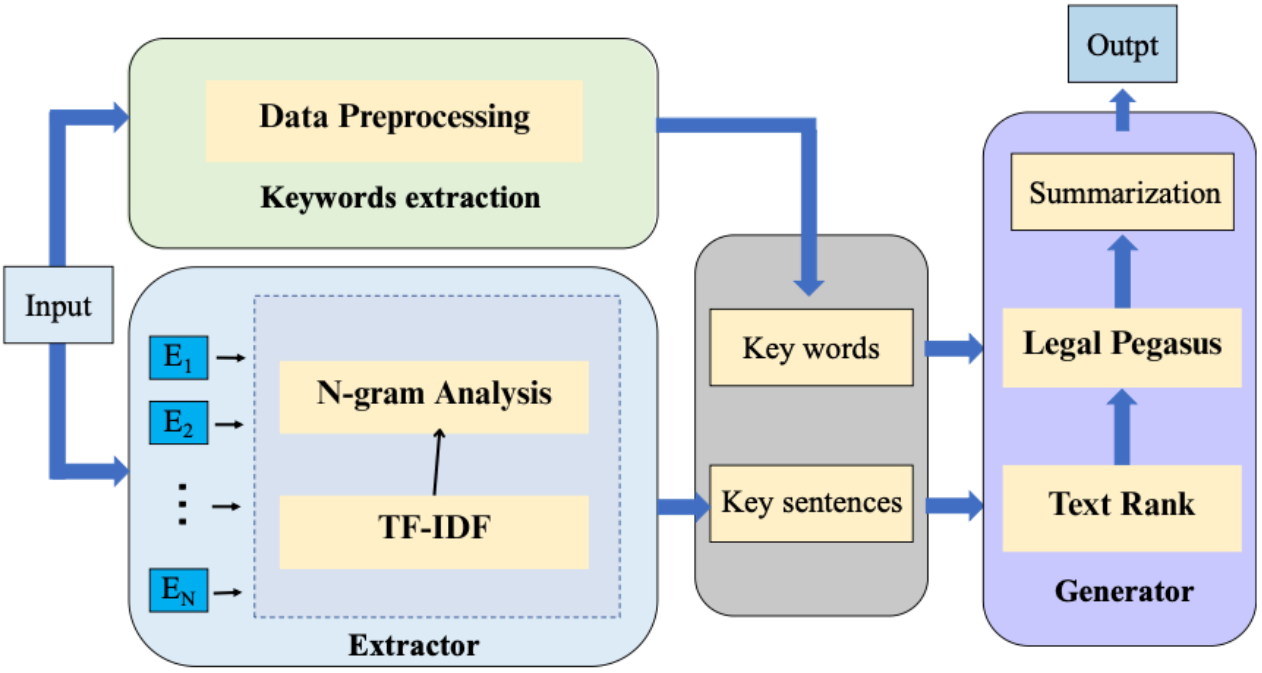


Figure 2.2 (Data Flow)

Figure 2.2 depicts the legal document that is input is initially subjected to data cleaning and preprocessing so that keywords can be extracted. After that, the document will be subjected to the TF-IDF along with N-gram analysis to extract the key sentences. The extractive summarization will plug into the TextRank algorithm and the abstractive summarization will plug into the Legal Pegasus model. Once the final summary is prepared, it will be generated to prepare for output.

Chapter 03: IMPLEMENTATION

3.1 Date Set Used in the Minor Project

a. **BillSum -** https://huggingface.co/datasets/FiscalNote/billsum:

This dataset consists of three parts: US training bills, US test bills and California test bills. The US bills were collected from Govinfo service provided by the US government Publishing Office(GPO) under CC0-1.0 license. The California bills from the 2015-16 session are available from the legislature's website

b. **Legal Case Document Summarization:Extractive and Abstractive Methods and their Evaluation -** https://zenodo.org/records/7152317

This repository contains the following 3 datasets for legal document summarization :

- IN-Abs : Indian Supreme Court case documents & their `abstractive' summaries, obtained from http://www.liiofindia.org/in/cases/cen/INSC/

- IN-Ext : Indian Supreme Court case documents & their `extractive' summaries, written by two law experts (A1, A2).

- UK-Abs : United Kingdom (U.K.) Supreme Court case documents & their `abstractive' summaries, obtained from https://www.supremecourt.uk/decided-cases/

3.2 Features

3.2.1 Number of Attributes, fields, and description of the data set

- **Attributes:** The data consists of two columns, Text and Summary
- **Description:** Text, contains the text of the legal document, Summary contains the summary of this text used for evaluation
- **Shape:**

  Training Data: (26672, 2)

  Test Data: (3469, 2)

3.3 Design of Problem Statement

The project aims to generate concise and meaningful summaries of legal documents using machine learning-based summarisation techniques and natural processing language. In this project we are creating a document that can summarise documents by using both extractive

and abstractive summarisation methods.Extractive summarisation picks out important sentences from the original text. Abstractive summarisation rewrites the summary and is more human like a natural way.

We use tools like TextRank and LexRank for extractive summaries, and models like Legal Peagasus and Bart for abstractive summaries. Out of all four methods, we choose the best two by comparing their scores using ROUGE evaluation metrics. For Methodology diagrams refer to fig 1.3.

1. Data Collection And Input Handling

The first step in our methodology is to collect legal documents that the user wants to summarise. We will use libraries like pdfplumber to extract the raw text from the PDF file. Pdfplumber is especially helpful because it prevents the structure and layout of the text which is useful for understanding paragraph and section

2. Data Preprocessing

Once we extract text from the document the next important step is to clean and prepare the text so that it can be understood by summarisation models. This is known as preprocessing and it ensures that we remove any unnecessary noise from the text and focus on the key information

- Tokenization

  We break down the text into sentences and words using libraries like spacy. This helps us to analyze the text at a more granular level.

  Sentence tokenization : splits the text into sentences

  Word tokenization: splits the sentence into words

- Noise Removal

  We Remove:

  stop words(like "is", "the", "and")

  Punctuation Marks(like commas, dots)

  Special characters and digits that are not meaningful

- Lemmatization

  Each word is converted into base from a dictionary form using tool spacy.

  For example words like "running" "ran", "runs"-> become "run"

- Sentence Segmentation

  We divide the entire text into logically meaningful segments this helps in understanding where one context ends and other context begins(e.g., a new clause or a section in the legal document)

3. Feature engineering

Once the text is preprocessed, we convert it into numerical representations that machine learning models can understand and learn from. This step is essential for both extractive and abstractive summarisation models.

- **1. TF-IDF(Term Frequency-Inverse Document Frequency)**

TF-IDF is a statistical method used to evaluate how important a word is to document in a collection or corpus. it increases proportionality to the number of times a word appears in the document but is offset by the frequency of word in the entire corpus.

It helps to identify keywords or important terms in a sentence or a document.

TF-IDF(t,d) = TF(t,d) *IDF(t)

Where,

TF-IDF(t,d) = frequency of term t in document d

IDF(t) =log(N/df(t))

- **2. N Gram Analysis**

N-gram: A contiguous sequence of n words (or tokens) from a text.

- ☐ Unigram (n=1): "court", "ruled", "today"
- ☐ Bigram (n=2): "court ruled", "ruled today
- ☐ Trigram (n=3): "the court ruled", "court ruled today"

4. Choosing Summarization Approach

After the text is clean and represented numerically now it is time to generate the summary of the document. We have used to main Strategies for this:

- **Extractive Summarization**

It works by selecting and compiling the most relevant sentences directly which is present on the original document without altering the wording. Common techniques used are TextRank and LexRank.

TextRank is a graph based ranking model that assigns scores to the sentences based on similarity to other sentences in the text.

LexRank is another graph based approach that uses cosine similarity to identify the important sentences.

- **Abstractive Summarization**

It attempts to understand the context and the meaning of the text to generate new cosine sentences that may not appear verbatim in the original document. Abstractive Summarization approach realise on the advance Natural Language Processing model such as Transformers

In this system Legal Pegasus, Transformer based model trained on legal text is used to generate structure and readable summary

5. Training and Fine Tuning

Once the summarization approach is finalized particularly for extractive summarization a model is trained to identify the importance of each of the sentences in the legal document.

Random forest classifier is a powerful assembler learning algorithm model which is used here which combines multiple decision trees to improve performance and reduce overfitting. This classifier is trained on the labelled legal data where each sentence is marked on whether it should be included in the summary or not.

6. Model Evaluation

The final step is to evaluate the effectiveness and the performance of the summarization model. It ensures that the summary generated is not only concise but also accurate. Evaluation is typically done using these metrics:

A. ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

It values the overlap between the generated summary and reference summary.

Variants of ROUGE :

ROUGE 1: Measures overlap of individual words

ROUGE 2: Measures overlap of two word sequences

ROUGE-L : Evaluate the longest common subsequence

The higher the ROUGE score indicates better preservation of original contents key information.

3.4 Algorithm / Pseudo code of the Project Problem

1. Data Preparation:

- Load a test dataset containing text with corresponding summaries.
- Install and import the following operating libraries: transformers, sentencepiece, rouge-score, pandas, openpyxl, nltk, networkx.
- Download: punkt (for tokenizing), stop words from nltk.

2. TextRank Sentence Ranking:

- Pre-process the sentences by converting to lowercase, tokenize, and remove stop words and punctuation
- Build a similarity matrix of each sentence based on common words.
- Construct a graph from a similarity matrix.
- Calculate PageRank scores in the graph to establish significance of sentences from the corpus.
- Choose the number of top-ranked sentences (i.e. top 10 sentences) to create a summary.

3. Legal-Pegasus Summarization:

- Load the pre-trained v1.0 Legal-Pegasus model and tokenizer.
- Tokenize the top-ranked sentences.
- Run the Legal-Pegasus model to create a summary.
- Decode the summary.

4. Hybrid Summarization:

- Integrating TextRank and Legal-Pegasus
- Rank the Sentence using TextRank, select the top-ranked sentences and use Legal-Pegasus to summarize the top-ranked sentences.

5. ROUGE Evaluation:

- Evaluate the resulting Legal-Pegasus summaries against its reference summaries for ROUGE scores (ROUGE-1, ROUGE-2, ROUGE-L).
- Average all ROUGE scores for given metrics.
- Print average values of the ROUGE scores.

3.5 Flow graph of the Minor Project Problem

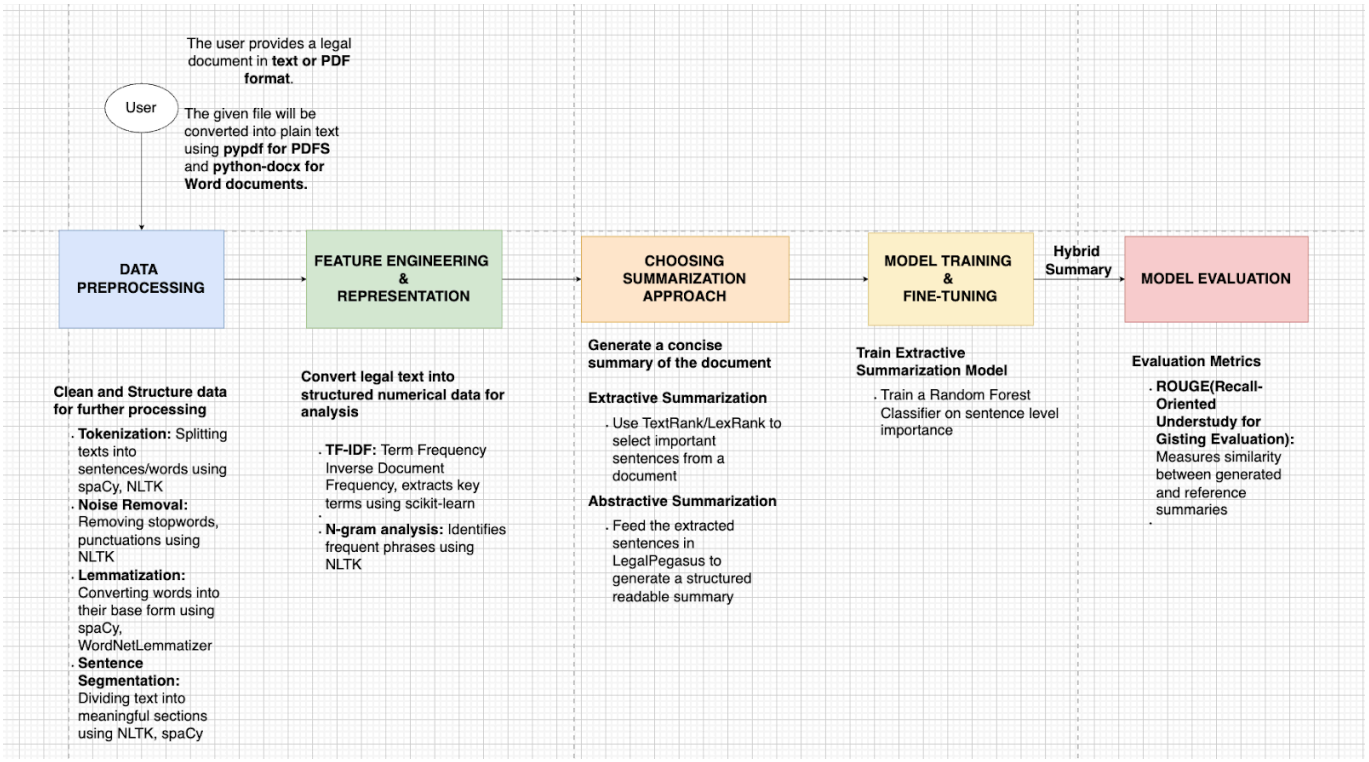


Figure 3.1: Methodology Diagram

3.6 Screenshots of the various stages of the Project

```
!pip install transformers sentencepiece rouge-score pandas openpyxl
import pandas as pd
import numpy as np
import nltk
import networkx as nx
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from rouge_score import rouge_scorer
import string
import torch
from transformers import PegasusTokenizer, PegasusForConditionalGeneration

nltk.download('punkt_tab')
nltk.download('stopwords')
```

Figure 3.2: Environment Setup and Library Installation

```
test_df = pd.read_excel("test_data.xlsx",)
texts = test_df["Text"].tolist()
references = test_df["Summary"].tolist()
```

Figure 3.3: Data Loading

```
stop_words = set(stopwords.words('english'))

def preprocess_sentence(sentence):
    words = word_tokenize(sentence.lower())
    return [w for w in words if w not in stop_words and w not in string.punctuation]
```

Figure 3.4: Text Preprocessing

```python
def textrank_rank_sentences(text):
    sentences = sent_tokenize(text)
    if len(sentences) <= 1:
        return sentences
    preprocessed = [preprocess_sentence(s) for s in sentences]

    similarity_matrix = np.zeros((len(sentences), len(sentences)))
    for i in range(len(sentences)):
        for j in range(len(sentences)):
            if i == j:
                continue
            words_i = set(preprocessed[i])
            words_j = set(preprocessed[j])
            if not words_i or not words_j:
                continue
            common_words = words_i.intersection(words_j)
            if common_words:
                similarity_matrix[i][j] = len(common_words) / (np.log(len(words_i)+1) + np.log(len(words_j)+1))

    graph = nx.from_numpy_array(similarity_matrix)
    scores = nx.pagerank(graph)
    ranked_sentences = sorted(((scores[i], s) for i, s in enumerate(sentences)), reverse=True)
    return [s for _, s in ranked_sentences]
```

Fig 3.5: Hybrid Implementation - Text Rank

```python
model_name = "nsi319/legal-pegasus"
tokenizer = PegasusTokenizer.from_pretrained(model_name)
model = PegasusForConditionalGeneration.from_pretrained(model_name)
device = "cuda" if torch.cuda.is_available() else "cpu"
model = model.to(device)

def generate_pegasus_summary(text, max_length=128, min_length=30):
    inputs = tokenizer(text, return_tensors="pt", truncation=True, padding="longest", max_length=512).to(device)
    summary_ids = model.generate(inputs["input_ids"], max_length=max_length, min_length=min_length, length_penalty=1.0, num_beams=4, early_stopping=True)
    return tokenizer.decode(summary_ids[0], skip_special_tokens=True)
```

Fig 3.6: Hybrid Implementation - Legal Pegasus

```python
generated_summaries = []
for text in texts:
    ranked_sentences = textrank_rank_sentences(text)
    top_sentences = " ".join(ranked_sentences[:10])  # select top 10 sentences
    summary = generate_pegasus_summary(top_sentences)
    generated_summaries.append(summary)
```

Fig 3.7: Hybrid Implementation - generating summary

```python
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = {"rouge1": [], "rouge2": [], "rougeL": []}

for ref, gen in zip(references, generated_summaries):
    score = scorer.score(ref, gen)
    for key in rouge_scores:
        rouge_scores[key].append(score[key].fmeasure)

average_scores = {key: np.mean(vals) for key, vals in rouge_scores.items()}
print("ROUGE Scores:")
for key, val in average_scores.items():
    print(f"{key}: {val:.4f}")
```

Figure 3.8: Rouge Implementation

```
ROUGE Scores:
rouge1: 0.2795
rouge2: 0.1383
rougeL: 0.1677
```

Figure 3.9: Results

Chapter 04: RESULTS

4.1 Discussion on the Results Achieved

After building and training the model on the dataset, the following results were obtained:

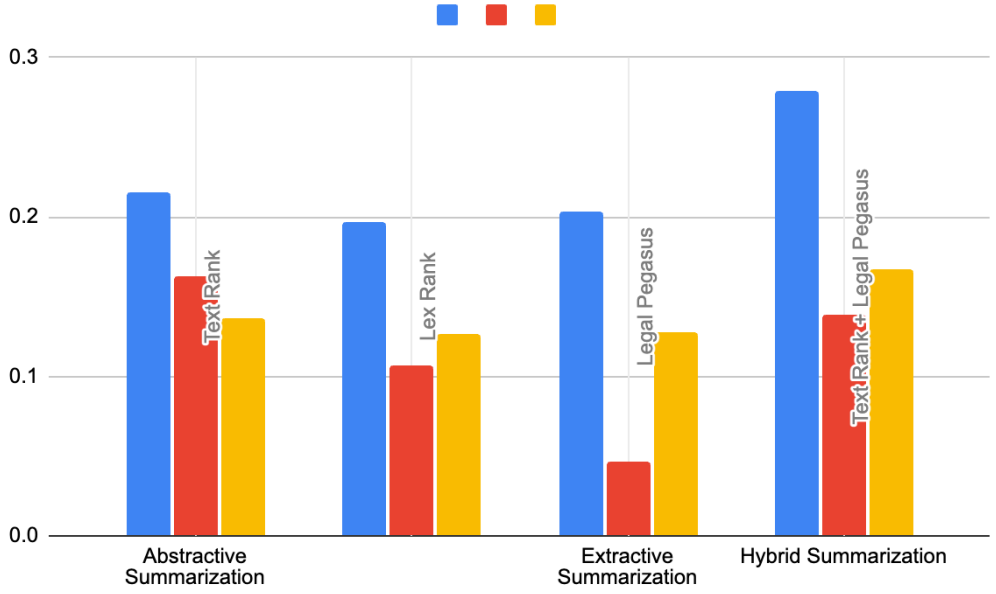
ROUGE score evaluation:

Technique Used	Model	R1	R2	RL
Abstractive Summarization	Text Rank	0.2159	0.1633	0.1366
	Lex Rank	0.1966	0.1071	0.127
Extractive Summarization	Legal Pegasus	0.2035	0.0468	0.128
Hybrid Summarization	Text Rank + Legal Pegasus	0.2795	0.1383	0.1677

Figure 4.1: Comparison analysis

Sample Summary:

Section 19 of the Evacuee Property Act, 1954 provides that a managing officer or managing corporation may cancel any allotment etc., under which any evacuee property acquired under the Act is held or occupied by a person whether such allotment or lease was granted before or after the commencement of the Act.<n>A managing officer or a managing corporation may sell any property in the compensation pool entrusted to him or to it, cancel an allotment or terminate a lease, or vary the terms of any such lease or allotment if the allottee or lessee, as the case may be, has sublet or parted with the possession of the whole.



4.2 Application of the Minor Project

a. **Law Firms and Legal Professionals**
   Aid law firms, lawyers, and legal professionals by rapidly generating a summary of lengthy case files, contracts, and judgments for quick review
b. **Corporate and In-house legal teams**
   Quickly extract and understand key clauses, obligations, and risks in vendor or employee contracts, and compliance documents
c. **Journalists and Analysts**
   Summarize court rulings, legislative changes, bills, and laws for media coverage and help in quickly producing news reports
d. **Education and training**
   Students can use this to digest large readings and focus on key legal doctrines

4.3 Limitation of the Minor Project

a. **Model knowledge cutoff and domain limitations**
   The model uses pretrained models like Pegasus that are trained on specific data and may not generalize well to legal documents from different countries, newer laws, and specialized domains
b. **No Legal Validation or Certification**
   AI-generated summaries cannot replace legal advice and notarized reviews
c. **Formatting and Parsing Challenges with PDFs**
   The model may not properly read legal documents with complex layouts, tables, or images, which may lead to inaccurate input for summarization

4.4 Future Work

a. Develop and deploy a functional website for a better user experience
b. Train the model using legal documents from different countries and different domains to overcome domain limitations

References

1. S. Ghosh, M. Dutta, and T. Das, "Indian Legal Text Summarization: A Text Normalisation-based Approach," presented at the 2022 IEEE 19th India Council International Conference (INDICON), 2022.
2. Y. Huang, L. Sun, C. Han, and J. Guo, "A High-Precision Two-Stage Legal Judgment Summarization," Mathematics, vol. 11, no. 6, p. 1320, Mar. 2023
3. C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in Text Summarization Branches Out, Barcelona, Spain, 2004, pp. 74–81.
4. R. Raj and Dr. Kavitha S N, "Legal Document Summarization and Analysis," International Journal of All Research Education and Scientific Methods, vol. 13, no. 1, pp. 2073-2081, 2025.
5. K. Lipianina-Honcharenko, N. Maika, S. Sachenko, L. Kopania, and M. Soia, "A Cyclical Approach to Legal Document Analysis: Leveraging Al for Strategic Policy Evaluation," in ICyberPhyS-2024: 1st International Workshop on Intelligent & CyberPhysical Systems, 2024, pp. 164-173
6. D. Cavar, J. Herring, and A. Meyer, "Case Law Analysis using Deep NLP and Knowledge Graphs," in Proceedings of the LREC 2018 "Workshop on Language Resources and Technologies for the Legal Knowledge Graph," Georg Rehm, Víctor Rodríguez-Doncel, Julián Moreno-Schneider, Eds. Miyazaki, Japan, 2018, pp. 1-9.
7. Pandey, R. R., Khandelwal, S., Srivastava, S., Triyar, Y., & Almas, M. (2024). LEGALSEVA - AI-Powered Legal Documentation Assistant. International Research Journal of Modernization in Engineering Technology and Science, 6(3), 6418-6421.
8. Enhancing Legal Document Analysis and Judgment Prediction with Machine Learning and Deep Learning Techniques" by Avadhut Shelar and Minal Moharir .
9. An Overview of Information Extraction Techniques for Legal Document Analysis and Processing" by Ashwini V. Zadgaonkar and Avinash J. Agrawal
10. Jagirdar, S. Gandage, B. Waghmare, and I. Kazi, "Enhancing Legal Document Summarization Through NLP Models: A Comparative Analysis of T5, Pegasus, and BART Approaches," Int. J. Creative Res. Thoughts (IJCRT), vol. 12, no. 3, Mar. 2024
11. R. C. Kore, P. Ray, P. Lade, and A. Nerurkar, "Legal Document Summarization Using NLP and ML," Int. J. Eng. Comput. Sci., vol. 9, May 2020.
12. D. Jain, M. D. Borah, and A. Biswas, "Summarization of Lengthy Legal Documents via Abstractive Dataset Building: An Extract-then-Assign Approach," in Proc. Forum for Information Retrieval Evaluation (FIRE), India, Dec. 13–17, 2021.
13. Extractive Summarization:Text Rank and Lex Rank

Extractive_Summarization.ipynb

14. Abstractive Summarization: Legal Pegasus - Abstractive Summarization.ipynb

15. Hybrid model: Hyrid model.ipynb

16. Reference Summary : reference_summary

17. Datasets:https://zenodo.org/records/7152317, https://huggingface.co/datasets/FiscalNote/billsum