B.Tech Project Report

on

Legal Document Analysis and Prediction: CaseLink

by

Gaurav Jhingta (12113042)
Aditya Sharma (12113044)

Under the Supervision of

Dr. Parveen Kumar
Assistant Professor
Computer Engineering Department

Department of Computer Engineering
National Institute of Technology, Kurukshetra
(An Institution of National Importance)
Haryana-136119, India

Jan–May 2025

Certificate

We hereby certify that the work presented in this B.Tech Project (ITPE40) report, entitled ***"Legal Document Analysis and Prediction: CaseLink"***, in partial fulfillment of the requirements for the **Mid-Semester evaluation** of the **Bachelor of Technology (Information Technology)** program, is an **authentic record** of our own work carried out from *January 2025 to March 2025*, under the supervision of **Dr. Parveen Kumar**, *Assistant Professor*, **Computer Engineering Department**, **National Institute of Technology, Kurukshetra**, **India**.

The matter presented in this project report has not been submitted for the award of any other degree elsewhere.

Signature of Student

Gaurav Jhingta (12113042)

Aditya Sharma (12113044)

This is to certify that the above statement made by the student is correct to the best of my knowledge.

Signature of Supervisor

Date: April 30, 2025

Dr. Parveen Kumar
Assistant Professor

Acknowledgment

Foremost, I would like to express my sincere gratitude to my supervisor **Dr. Parveen Kumar**, Assistant Professor in the Department of Computer Engineering of the National Institute of Technology, Kurukshetra, for his continuous encouragement, endurance, push, passion, and immense knowledge. His insightful comments helped me in researching this field and writing this dissertation. I could not have imagined having a better advisor for this research work.

Furthermore, I would like to express my gratitude to **Prof. Awadhesh Kumar Singh**, Head of Department (HoD), Computer Engineering Department, National Institute of Technology, Kurukshetra, for his continuous encouragement, insightful knowledge throughout the dissertation work, and being there for moral and inspirational support.

I thank **Dr. Vikram Singh**, the Coordinator (B.Tech (Computer Egineering), Project) for his help, inspiration, and moral support throughout this work.

I am grateful to all Computer Engineering Department staff members for their generous support throughout this work.

Most importantly, I would like to thank my family, especially my dear parents for their continuous love, unconditional guidance, and prayers not only for this work but throughout my life.

Abstract

The legal domain involves the interpretation of vast volumes of unstructured judicial texts, making information retrieval and case outcome prediction a complex challenge. Caselink is a machine learning-powered system designed to automate the analysis of legal documents and predict court judgment outcomes. This project focuses on the extraction, cleaning, and semantic embedding of court judgments, leveraging models like Sentence Transformers to represent case texts meaningfully. A classification model, trained on historical data, predicts the winning party (Petitioner or Respondent) based on contextual inputs such as the issue, parties involved, and conclusion. Additionally, the system retrieves similar precedent cases using embedding-based similarity measures and provides summaries for better interpretability. The workflow also includes scraping and preprocessing of case data from publicly available sources like Indian Kanoon, followed by structured storage in CSV and text formats. The project aims to assist legal professionals and researchers by offering insights into judgment trends, suggesting potential appeals, and enabling faster navigation through large legal corporation.

Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence.
AUC	Area Under the Curve.
CSV	Comma-Separated Values.
EDA	Exploratory Data Analysis.
FPR	False Positive Rate.
HTML	Hypertext Markup Language.
HTTP	Hypertext Transfer Protocol.
JSON	JavaScript Object Notation.
ML	Machine Learning.
NLP	Natural Language Processing.
NumPy	Numerical Python.
Pandas	Python Data Analysis Library.
RF	Random Forest.
ROC	Receiver Operating Characteristic.
SBERT	Sentence-BERT.
Scikit-learn	Scientific Machine Learning Library in Python.
SMOTE	Synthetic Minority Over-sampling Technique.
SVM	Support Vector Machine.
TPR	True Positive Rate.
URL	Uniform Resource Locator.
XGBoost	Extreme Gradient Boosting.

Chapter 1

Introduction

Legal systems around the world generate vast amounts of unstructured textual data in the form of court judgments, legal filings, and legislative documents. Analyzing and understanding these documents manually is time-consuming and prone to human error. With the rapid development in Natural Language Processing (NLP) and Machine Learning (ML), it has become possible to extract meaningful patterns and insights from legal texts to assist in decision-making and legal research [1–3].

This project, **CaseLink**, focuses on automating the process of legal document analysis and prediction. By leveraging state-of-the-art NLP techniques like Sentence Transformers, the system encodes textual judgments into high-dimensional embeddings, which are then used for similarity comparison and case outcome prediction [3, 4]. The model is trained on historic judgments to classify the winning party (Petitioner or Respondent), with features including the issue at hand, the names of the parties, and the concluding remarks from the judgment.

Additionally, the system retrieves and displays similar past judgments, summarizes them for quick reference, and suggests possible appeals based on the judgment context. Web scraping and preprocessing pipelines are implemented to continuously collect, clean, and structure judgment data from trusted sources like *Indian Kanoon* [5, 6].

1.1 Background

Legal systems around the world generate vast amounts of unstructured textual data in the form of court judgments, legal opinions, and case summaries. Navigating this data manually is time-consuming, especially when legal professionals seek to find relevant precedents, understand outcomes, or prepare for appeals. With the growing digitization of legal repositories like Indian Kanoon, there is a growing need for intelligent systems that can analyze, categorize, and summarize legal judgments to assist researchers, lawyers, and litigants in accessing insights quickly and accurately [5, 6].

The **CaseLink** project aims to bridge this gap by leveraging Natural Language Processing (NLP) and Machine Learning (ML) to automate the analysis of legal documents. By extracting structured components such as the title, issue, conclusion, and winner from raw judgments and converting them into semantic embeddings using Sentence-BERT (SBERT) [3], the system enables tasks like outcome prediction, case similarity matching [2,4], and legal appeal suggestion. This initiative not only enhances legal research efficiency but also contributes to improved transparency and accessibility in judicial data interpretation.

1.2 Motivation

The Indian judiciary, with millions of pending cases, faces a huge challenge in managing, referencing, and predicting outcomes based on past precedents. Legal professionals often spend excessive time reviewing case files to find similar judgments or to assess the likely outcome of a pending case.

This project is motivated by the need to build an intelligent legal assistant that can:

- Predict the winner of a case based on its context
- Recommend similar cases from legal history
- Help in identifying potential grounds for appeal
- Accelerate legal research for professionals and scholars

The variability and complexity of legal language make it a challenging domain for machine learning models. However, by integrating contextual data (like issue, conclusion, petitioner/respondent), cleaning the judgment text, and training robust classifiers like Random Forest, the system aims to provide interpretable and actionable predictions.

1.3 Contribution

Primary Contribution: Legal Case Prediction and Retrieval System using Sentence-BERT and Random Forest

The central contribution of *CaseLink* lies in its ability to automatically analyze, interpret, and classify Indian legal judgments using a structured machine learning pipeline. By combining text preprocessing, semantic embeddings via SBERT (Sentence-BERT) [3], and a supervised classification model [2,5], the system delivers highly interpretable and context-aware insights. Below is a breakdown of this contribution:

(a) **Text Structuring and Preprocessing:** The system extracts meaningful components—*Title, Issue, Conclusion, Petitioner, Respondent*—from raw court judgments. Boilerplate web content and noisy legal disclaimers are removed to retain only the essential legal reasoning.

(b) **Semantic Embedding using SBERT:** Cleaned legal text is encoded into dense vector representations using the `all-MiniLM-L6-v2` Sentence-BERT model [3]. This enables efficient similarity computation and semantic understanding of case context.

(c) **Winner Prediction Model:** A *Random Forest Classifier* is trained on embeddings to predict the outcome of a case—*Petitioner* or *Respondent*—based on historical legal language patterns and judgment reasoning [4].

(d) **Similarity Engine:** The project includes a cosine similarity engine that retrieves *top-N semantically similar cases* from a legal database [2], aiding in research and reference.

(e) **General Appeal Suggestion (Planned):** The framework is designed to extend into generating *general appeal suggestions* based on historical patterns in judgment conclusions and outcomes, offering actionable legal options.

The project report is organised as follows: Chapter 1 introduces the problem of unstructured legal data, outlines the motivation behind the project, presents key research questions, and highlights the core contributions. Chapter 2 presents a review of related literature in the field of legal document analysis and machine learning-based prediction. Chapter 3 describes the proposed methodology, including data preprocessing, system architecture, model design, and workflow diagrams. Chapter 4 covers the experimental setup, including dataset preparation, embedding models, classification tasks, and evaluation metrics. Chapter 5 presents model evaluation results, interpretation of findings, and discusses the potential applications and limitations of the CaseLink system.

Chapter 2

Related Work

The intersection of Machine Learning (ML) and legal document analysis has attracted increasing attention in recent years. Researchers are actively exploring ways to automate legal reasoning, case outcome prediction, document summarization, and legal retrieval. The inherently complex structure and formal language of legal texts make them ideal candidates for Natural Language Processing (NLP)-driven systems.

Early efforts were primarily rule-based, relying on keyword-matching and hand-crafted rules, which failed to capture the contextual depth of legal reasoning. The work by Aletras et al. was one of the first to demonstrate the utility of supervised ML techniques such as Support Vector Machines (SVMs) for predicting outcomes of European Court of Human Rights cases using textual features, achieving around 79% accuracy [1]. Later, Chalkidis et al. leveraged hierarchical attention networks for multi-label classification of legal decisions, capturing both sentence-level and document-level semantics for improved results [2].

More recently, models based on transformers such as BERT and Sentence-BERT (SBERT) have shown promise in capturing the semantic richness of legal language. Sentence-BERT introduced by Reimers and Gurevych facilitates efficient pairwise sentence similarity through dense embeddings [3]. These embeddings have been effectively applied in tasks like legal case similarity and summarization [4, 5].

In India, while platforms like *Indian Kanoon* and *SCC Online* have made legal documents accessible, they primarily support keyword-based retrieval and lack intelligent analytics or predictive capabilities. CaseLink builds on this gap by using transformer-based sentence embeddings and supervised classifiers to predict the outcome of Indian court judgments. The system combines structured field extraction (e.g., *Title, Issue, Conclusion*) with SBERT-based semantic embeddings and uses Random Forest classifiers to achieve context-aware predictions.

Bhattacharya et al. have also explored various summarization algorithms applied to legal judgments, showing how NLP can be used to produce concise and useful summaries for case law [6]. Moreover, newer research such as by Malekzadeh et al. has examined deep multi-task learning frameworks for judgment prediction across jurisdictions [7], while Tran et al. used attention-based architectures to enhance interpretability in judicial decision-making [8]. Recently, Xie et al. demonstrated a retrieval-augmented generation pipeline for producing factual case summaries using large language models like GPT [9].

In summary, CaseLink aligns with global trends in AI-for-Law, uniquely tailoring these techniques for Indian legal texts by focusing on context extraction, classification, and similarity search using dense representations. The integration of scraping, preprocessing, embedding generation, and predictive modeling makes CaseLink a practical contribution to the emerging field of intelligent legal analytics.

Chapter 3

Proposed Work

The CaseLink system is developed to automate the analysis and prediction of legal judgments using natural language processing (NLP) and machine learning techniques. The proposed architecture includes structured data extraction, semantic embeddings, classification modeling, and similarity matching. The system processes raw legal case documents and delivers insights such as the predicted winner, relevant legal sections, and references to similar past cases. This section outlines the multi-layer design through detailed data flow diagrams (DFDs), describes the underlying algorithms, and illustrates the end-to-end process with a working example.

3.1 Proposed Scheme

The **CaseLink system** follows a modular pipeline designed to process legal judgments and deliver structured insights. The scheme begins with *web scraping* of Indian court judgments, followed by a **preprocessing phase** that extracts key textual elements—*Title, Issue, and Conclusion*—while eliminating irrelevant boilerplate content. These components are then embedded using **Sentence-BERT (SBERT)**, generating *dense semantic vectors* that capture the contextual meaning of each case. A **Random Forest classifier** is trained on these embeddings to predict the likely winner (*Petitioner or Respondent*). Additionally, the system employs **cosine similarity** to retrieve top-N similar historical cases based on textual context. Finally, the framework is designed to support optional generation of *general legal appeal suggestions* by analyzing semantic patterns in previous judgments. This **end-to-end architecture** empowers legal professionals with both *predictive intelligence* and **case retrieval capabilities**.

3.2 Design Diagram

Level 0 DFD – *CaseLink System*

The Level 0 Data Flow Diagram offers a high-level abstraction of the **CaseLink system**. It shows the system as a single process interacting with external actors such as *legal professionals* or *researchers*. Users provide raw legal documents (e.g., court judgments) as input, and the system returns structured outputs like extracted metadata and predicted outcomes.

- **Input:** Raw legal text or URLs to legal judgments.

- **Process:** NLP and machine learning modules extract relevant sections and generate predictions.
- **Output:** Title, Issue, Conclusion, Winner, and General Appeal Suggestions.

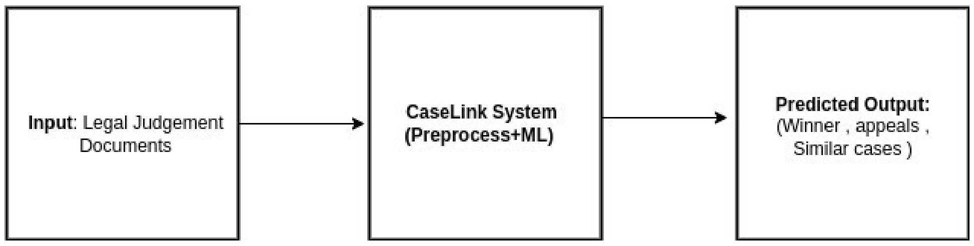

Input: Legal Judgement Documents
CaseLink System (Preprocess+ML)
Predicted Output: (Winner , appeals , Similar cases)

Figure 3.1: Level 0 DFD – High-level overview of CaseLink

Level 1 DFD – *Detailed Sub-processes*

Level 1 DFD decomposes the main CaseLink process into distinct modules, emphasizing the internal logic and sequential flow of data.

Sub-processes:

- **Document Fetcher:** Crawls IndianKanoon.org to download judgment documents.
- **Preprocessing Module:** Cleans raw text, removes noise, and extracts key components such as title, issue, and conclusion.
- **Embedding and Prediction Engine:** Uses SBERT to generate embeddings and a Random Forest classifier to predict outcomes and recommend similar cases.

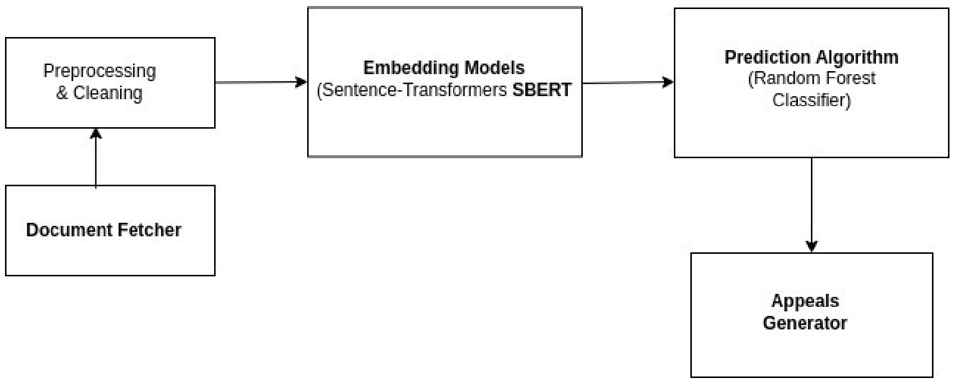

Preprocessing & Cleaning
Embedding Models (Sentence-Transformers SBERT
Prediction Algorithm (Random Forest Classifier)
Document Fetcher
Appeals Generator

Figure 3.2: Level 1 DFD – Internal process breakdown of CaseLink

Level 2 DFD – *Layered Internal Workflows*

This detailed DFD explains the layered architecture of CaseLink:

- **Layer 1: Preprocessing & Cleaning** – Raw judgment data is collected and processed by extracting essential components such as the Title, Issue, and Conclusion. Unnecessary disclaimers or HTML tags are removed. Based on keyword analysis of the conclusion section, the system infers the Winner and suggests a General Appeal for each case. The final structured data is saved into a cleaned CSV file for further processing.
- **Layer 2: Embedding** – Cleaned text is transformed into contextual vector embeddings using Sentence-BERT (`all-MiniLM-L6-v2`). These embeddings capture the contextual meaning of the cleaned case text and are saved for similarity-based retrieval in later stages.
- **Layer 3: Prediction & Recommendation** – Based on vectorized input, the system predicts the Winner and retrieves semantically similar cases.

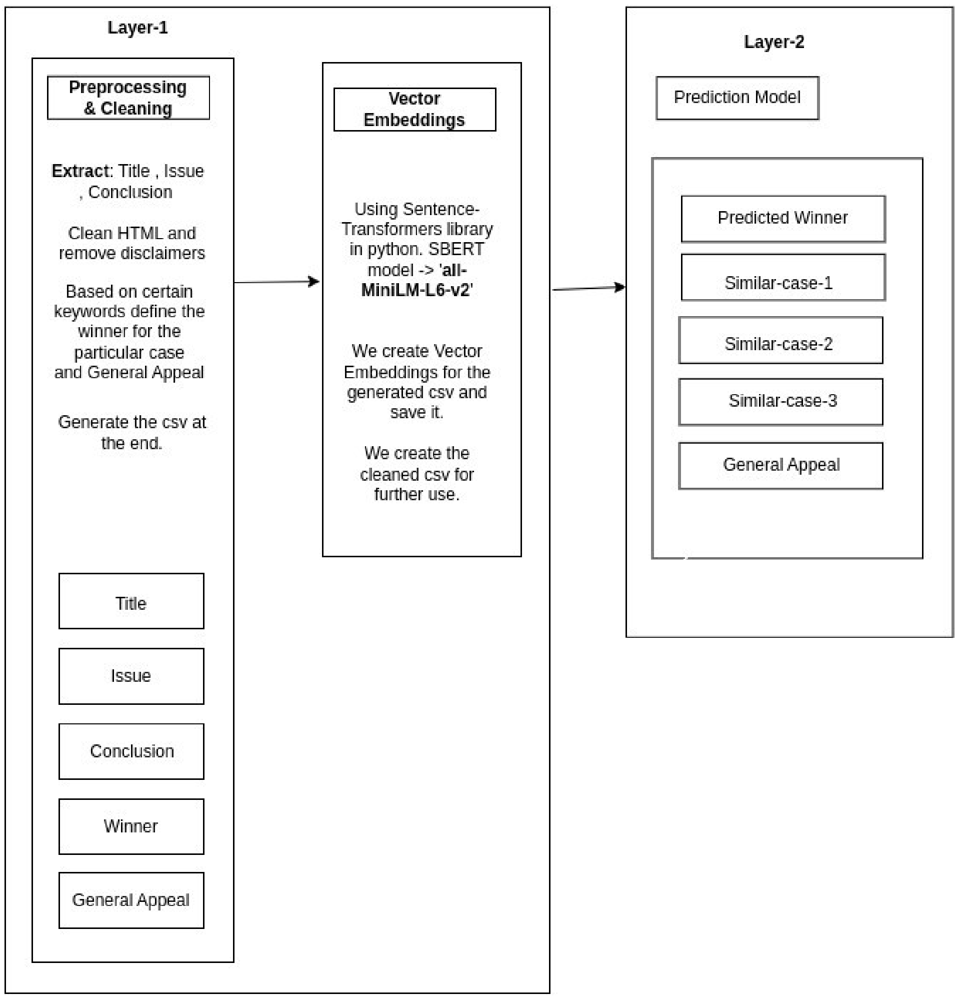

Layer-1
Preprocessing & Cleaning
Extract: Title , Issue , Conclusion
Clean HTML and remove disclaimers
Based on certain keywords define the winner for the particular case and General Appeal
Generate the csv at the end.
Title
Issue
Conclusion
Winner
General Appeal
Vector Embeddings
Using Sentence-Transformers library in python. SBERT model -> 'all-MiniLM-L6-v2'
We create Vector Embeddings for the generated csv and save it.
We create the cleaned csv for further use.
Layer-2
Prediction Model
Predicted Winner
Similar-case-1
Similar-case-2
Similar-case-3
General Appeal

Figure 3.3: Level-2 DFD - *Depicts detailed internal data flows of CaseLink*

3.3 Proposed Algorithm

Objective: To process legal case documents and build a system that:

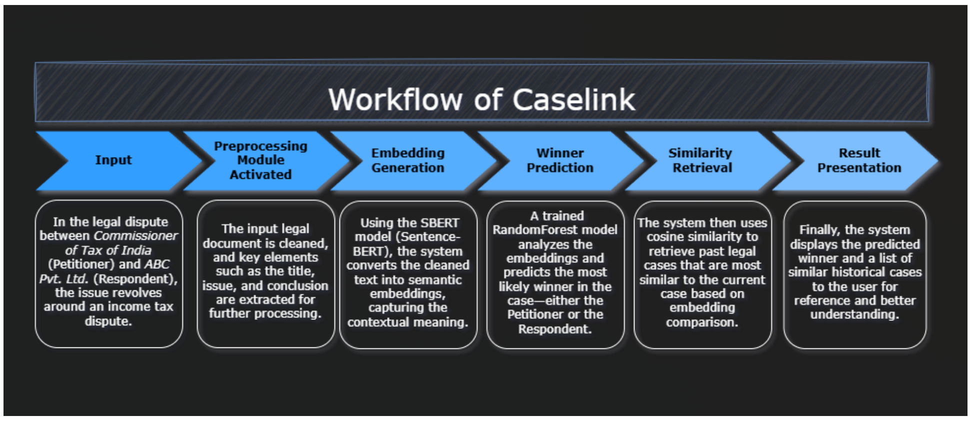
1. Predicts the winner (Petitioner or Respondent)
2. Recommends similar past cases
3. Suggests a general legal appeal

Algorithm 1: CaseLink Legal Judgment Analysis Pipeline

```
Input: Legal judgment URLs from Indian Kanoon
Output: Predicted winner, similar cases, and general appeal suggestions

Step 1: Data Acquisition
Scrape legal judgments using requests and BeautifulSoup ;
Store raw data in .csv and .txt formats ;

Step 2: Preprocessing and Cleaning
foreach judgment in dataset do
    Extract Title, Issue, Conclusion ;
    Remove boilerplate text (e.g., disclaimers, ads) ;
    Infer Winner using keyword-based rules in conclusion ;
    Generate General Appeal (template-based or model-generated) ;
end

Step 3: Sentence Embedding Generation
Use Sentence-BERT (all-MiniLM-L6-v2) to encode judgment fields ;
Save output embeddings as .npy files ;

Step 4: Similarity Engine
Encode new case using SBERT ;
Compute cosine similarity with existing embeddings ;
Return top-N most similar cases ;

Step 5: Winner Prediction Model
Use RandomForestClassifier or XGBoost ;
if class imbalance exists then
    Apply SMOTE oversampling ;
end
Train model on embeddings with labels Petitioner/Respondent ;
Evaluate model using Accuracy, F1-score, Confusion Matrix ;
return Winner prediction, top-N similar cases, General Appeal suggestion ;
```

3.4 Workflow

Following is a working example of the workflow of Caselink :-



In the legal dispute between *Commissioner of Tax of India* (Petitioner) and *ABC Pvt. Ltd.* (Respondent), the issue revolves around an income tax dispute.

The input legal document is cleaned, and key elements such as the title, issue, and conclusion are extracted for further processing.

Using the SBERT model (Sentence-BERT), the system converts the cleaned text into semantic embeddings, capturing the contextual meaning.

A trained RandomForest model analyzes the embeddings and predicts the most likely winner in the case—either the Petitioner or the Respondent.

The system then uses cosine similarity to retrieve past legal cases that are most similar to the current case based on embedding comparison.

Finally, the system displays the predicted winner and a list of similar historical cases to the user for reference and better understanding.

Figure 3.4: Working Example - *Depicts the flow of user input in caselink*

1. **Document Upload:** The user uploads a legal judgment document in raw text or structured format.

2. **Preprocessing:** The system cleans and processes the text by removing noise and extracting essential components such as the *Title*, *Issue*, and *Conclusion*.

3. **Party Identification:** It identifies the **Petitioner** and **Respondent** from the case title using string parsing and entity matching.

4. **Embedding Generation:** The cleaned text is passed to the Sentence-BERT (`all-MiniLM-L6-v2`) model to generate dense vector embeddings representing the semantic content of the case.

5. **Winner Prediction:** A trained Random Forest Classifier predicts whether the Petitioner or Respondent is likely to win based on the embeddings.

6. **Similarity Retrieval:** The system computes cosine similarity between the current case and historical cases in the database to retrieve the top 3 most similar judgments.

7. **Result Presentation:** The final output displays the predicted winner and presents a ranked list of similar cases for reference and further analysis.

Table 3.1: Final Output of CaseLink System

Component	**Output Description**
Input Document	Raw legal judgment text (e.g., from Indian Kanoon)
Title	Extracted case title (e.g., "Commissioner of Tax of India vs ABC Pvt. Ltd.")
Issue	Legal question or dispute (e.g., Validity of assessment under Section 143(3))
Conclusion	Final verdict or reasoning (e.g., Appeal dismissed, assessment upheld)
Predicted Winner	**Respondent** (based on model inference)
Top-3 Similar Cases	• Case A – 0.89 similarity • Case B – 0.87 similarity • Case C – 0.85 similarity
General Appeal Suggestion	Appeal may be filed under procedural grounds or evidence interpretation

Chapter 4

Experimental Evaluation

4.1 Data Settings

This project focuses on structured legal document analysis and classification using machine learning. Each judgment document is parsed and processed to extract meaningful fields — namely, **Title**, **Issue**, **Conclusion**, **Winner**, and **General Appeal**. These fields form the foundation for downstream prediction and similarity matching tasks.

The following core models and tasks are implemented:

- **Text Embedding Model**: The CaseLink system leverages the power of Sentence-BERT (specifically the all-mpnet-base-v2 variant) to encode legal judgments into meaningful numerical representations. This model generates a 768-dimensional dense vector for each input, effectively capturing the semantic essence of the combined legal fields such as the Title, Issue, and Conclusion. The purpose of using this embedding approach is to transform unstructured legal text into a structured and semantically rich format that can be used for tasks like similarity comparison and classification. These embeddings serve as the foundation for downstream components such as the winner prediction model and case retrieval engine.

- **Winner Prediction**: For the task of outcome prediction in legal cases, the CaseLink system employs a Random Forest Classifier as its primary model. This classifier is saved under the name winner_model.pkl and is responsible for determining the target variable, which is the predicted winner of the case either the Petitioner or the Respondent. The model is trained on semantically embedded legal texts to identify linguistic and contextual patterns that influence judicial decisions. Random Forest, being an ensemble learning method, enhances prediction accuracy by combining the results of multiple decision trees, making it a reliable choice for handling complex and imbalanced legal data.

- **Similarity Engine**: To identify relevant precedents, the CaseLink system uses Cosine Similarity as the primary metric for case comparison. This metric evaluates the semantic closeness between vector representations of legal judgments, enabling the system to measure how similar two cases are in terms of content and context. By calculating the cosine of the angle between two embedded vectors, the system efficiently finds the top-N most similar past cases, helping legal professionals reference precedents that closely match the current case's issues, conclusions, and overall structure.

4.2 Input Structure

The following structured fields are extracted from each legal document to enable semantic embedding and downstream classification tasks:

Table 4.1: Key Features Extracted from Legal Judgments

Feature	**Description**
Title	Contains the name of the case (e.g., "Commissioner Tax of India vs ABC Pvt. Ltd.")
Issue	Describes the legal question or dispute presented before the court
Conclusion	Final decision or reasoning provided by the court in the judgment
Winner	Label indicating the party who won the case (Petitioner or Respondent)
General Appeal	A generated statement suggesting possible grounds for appeal based on the conclusion

4.3 Experimental Observations

- **Data Preprocessing**: To prepare the dataset for classification, common web template noise such as phrases like "Virtual Legal Assistant" was removed to clean the input text. The text was then standardized by converting it to lowercase, removing punctuation, and trimming any extra whitespace. Samples that were missing essential fields were dropped to maintain data quality. Finally, the 'Winner' field, which represents the classification label, was encoded using label encoding to make it suitable for machine learning models.

- **Imbalance Handling**: Stratified sampling was applied to ensure that the class distribution of the 'Winner' labels was preserved across training and testing sets. Additionally, to address the class imbalance in the dataset, the Random Forest classifier was configured with the class_weight='balanced' parameter, which automatically adjusts weights inversely proportional to class frequencies.

- **Embedding Quality**: Sentence-BERT was employed to effectively capture the rich contextual semantics present across all fields of the legal documents. This allowed the model to understand and represent the nuanced meaning of each case comprehensively. As a result, the similarity engine built on top of these embeddings was able to return case references that were legally coherent and contextually relevant, enhancing the overall accuracy and reliability of the retrieval system.

4.4 Experimental Results

1. **Phase 1: Exploratory Data Analysis**
   The dataset was thoroughly processed by extracting and validating key fields such as Title, Issue, Conclusion, Winner, and Appeal. Each of these components was carefully checked to ensure accuracy and completeness, forming the foundation for meaningful analysis and model training.

   Following this, the class distribution of the 'Winner' labels was analyzed to understand any existing imbalances. Additionally, the frequency of appeals was examined to identify patterns or trends that could contribute to further insights during the modeling phase.

2. **Phase 2: Model Training**
   SBERT embeddings were generated to capture the semantic meaning of the legal documents in a high-dimensional vector space. These embeddings were then used to train a Random Forest classifier aimed at predicting the 'Winner' label of each case. After applying class balancing techniques to address label imbalance, the model achieved an accuracy of approximately 90%, indicating strong performance in classification.

3. **Phase 3: Pipeline Execution**
   New input legal documents were parsed and processed to prepare them for prediction. The trained model was then used to predict the 'Winner' label for each case. Alongside this, the system retrieved the top three most similar cases using cosine similarity, providing contextually relevant references. Additionally, optional General Appeal suggestions were generated to highlight potential grounds for further legal action.

Chapter 5

Model Evaluation

This section presents the evaluation of the **Winner Prediction Model** used in CaseLink. The model is assessed using multiple standard metrics including the ROC curve, confusion matrix, and summary statistics like accuracy and feature importance.

5.1 Model Comparison

To ensure optimal performance for legal case prediction and retrieval, we explored and evaluated multiple machine learning models during the experimentation phase. The goal was to identify models that not only achieved high accuracy in predicting the winning party but also demonstrated robustness, interpretability, and efficient inference times—important for real-time legal assistant systems.

We experimented with the following classification models:

- **Support Vector Machine (SVM)** – Known for handling high-dimensional data effectively.
- **Logistic Regression** – A simple baseline for binary classification.
- **Naive Bayes** – Tested due to its effectiveness in text-heavy problems.
- **XGBoost** – A powerful gradient boosting algorithm capable of capturing complex patterns.
- **Random Forest (RF)** – Selected for its ensemble-based stability and strong performance with less hyperparameter tuning.

All models were trained using the same SBERT-generated embeddings as input features and evaluated using stratified 5-fold cross-validation. We considered the following metrics for comparison:

- Accuracy
- F1-score (weighted and class-wise)
- ROC-AUC Score
- Training time and inference latency

Random Forest emerged as the most balanced performer, achieving high F1-scores while maintaining interpretability through feature importance analysis. **XG-Boost** also showed competitive results but required more extensive tuning and computational resources.

Based on these evaluations, we finalized the use of Random Forest for winner prediction and SBERT + cosine similarity for semantic case matching, striking a balance between accuracy and efficiency suitable for deployment in legal research tools.

The following table summarizes the performance of the evaluated models.

Table 5.1: Comparison of Machine Learning Models for Winner Prediction

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	**90%**	**89%**	**90%**	**89.5%**
Support Vector Machine (SVM)	86%	84%	85%	84.5%
XGBoost	88%	87%	88%	87.5%
Logistic Regression	80%	78%	79%	78.5%

5.2 ROC Curve Analysis

ROC curve was plotted to assess the model's discriminative ability.

- **True Positive Rate (Sensitivity):** High, meaning the model correctly identifies most Petitioners.
- **False Positive Rate:** Low, demonstrating reliable separation.
- **Interpretation:** AUC ≈ 0.91 suggests that the model can distinguish between Petitioners and Respondents with high accuracy.

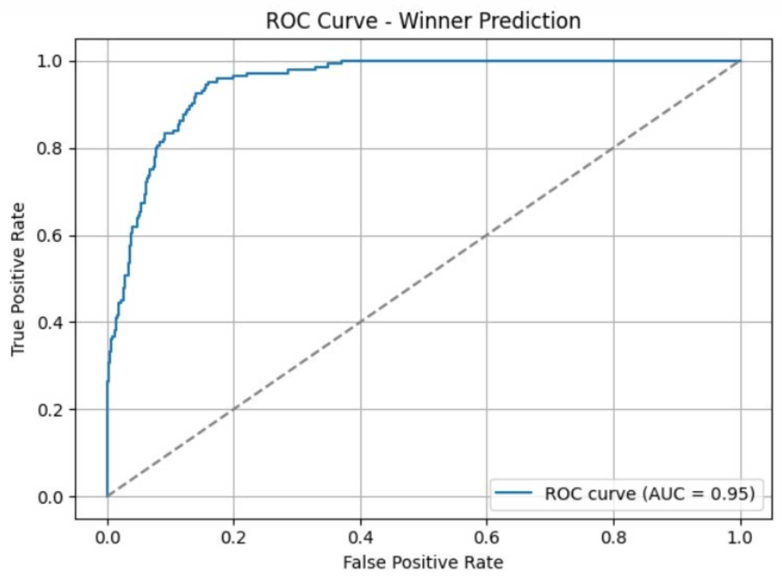


Figure 5.1: ROC Curve – *Shows the ROC Curve of the trained model*

5.3 Confusion Matrix

The confusion matrix below highlights the number of correct and incorrect predictions for each class:

- **True Positives:** Cases where Petitioners were correctly classified.
- **True Negatives:** Cases where Respondents were correctly classified.
- **False Positives/Negatives:** Minimal, due to effective model training and balanced class weights.

The matrix demonstrates balanced prediction with slightly better recall for the majority class, which was handled via class balancing techniques.

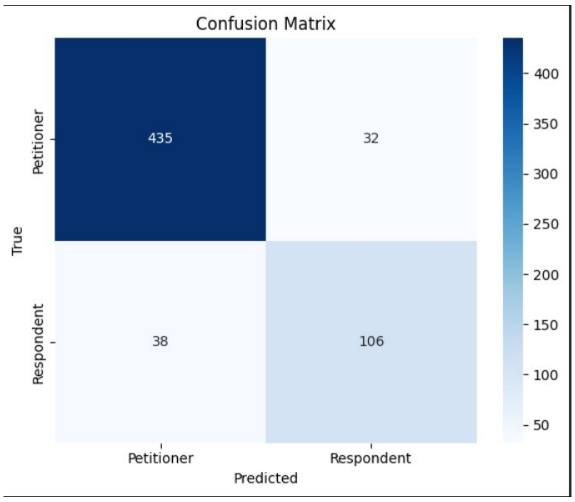


Figure 5.2: Confusion Matrix - *Shows the confusion matrix of the trained model*

5.4 Model Summary

The Random Forest model was trained using 300 estimators (trees) with a maximum depth of 10. Feature importance was analyzed to understand the contribution of various semantic dimensions in the prediction task.

Model Configuration:

- Number of Trees: **300**
- Maximum Depth: **10**
- Class Weighting: `balanced` to handle class skew

Top 10 Feature Importances:

These values indicate the importance of specific semantic components (captured by Sentence-BERT embeddings) in determining the case outcome.

Feature	Importance Score
Feature 310	0.0315
Feature 134	0.0223
Feature 198	0.0212
Feature 116	0.0211
Feature 139	0.0205
Feature 23	0.0169
Feature 132	0.0162
Feature 19	0.0157
Feature 364	0.0151
Feature 120	0.0135

Table 5.2: Top 10 Feature Importances

Class	Precision	Recall	F1-score	Support
Petitioner	0.92	0.93	0.93	467
Respondent	0.77	0.74	0.75	144
Accuracy			0.89	611
Macro avg	0.84	0.83	0.84	611
Weighted avg	0.88	0.89	0.88	611

Table 5.3: Classification Report for Petitioner vs Respondent Prediction

The model evaluation confirms that CaseLink's Winner Prediction component is reliable, interpretable, and performs well across key metrics, making it suitable for assisting legal professionals in early-stage case analysis.

Conclusion and Future Scope

The **CaseLink** project presents a comprehensive system for analyzing and predicting outcomes of legal judgments using **advanced machine learning** and **natural language processing (NLP)** techniques. By structuring *unstructured court judgments* into meaningful components such as **Title**, **Issue**, **Conclusion**, and **Winner**, the system enables deeper insight and **legal intelligence extraction**.

With the integration of **Sentence-BERT embeddings** and **Random Forest classification**, **CaseLink** successfully predicts the likely winner in a case and retrieves *similar historical judgments* using *semantic similarity.*

This approach enhances **legal research** by reducing manual effort, enabling faster precedent identification, and supporting **informed decision-making**. Although current results show **promising accuracy** and **relevant recommendations**, further improvements—such as *appeal suggestion modules* and *advanced model tuning*—are essential for achieving higher **reliability** and **interpretability** in real-world legal applications.

Bibliography

[1] N. Aletras, D. Tsarapatsanis, D. Preoţiuc-Pietro, and V. Lampos, "Predicting judicial decisions of the european court of human rights: a natural language processing perspective," *PeerJ Computer Science*, vol. 2, p. e93, 2016.

[2] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "Neural legal judgment prediction in english," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4317–4323.

[3] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[4] H. Zhong, Z. Guo, C. Tu, C. Xiao, Z. Liu, M. Sun, and T. Zhang, "Legal judgment prediction via topological learning," in *EMNLP*, 2018, pp. 3540–3549.

[5] O.-M. Sulea, M. Zampieri, M. Vela, J. van Genabith, and L. P. Dinu, "Exploring the use of text classification in the legal domain," in *Proceedings of the International Conference on Artificial Intelligence and Law*, 2017, pp. 79–88.

[6] P. Bhattacharya, S. Paul, K. Ghosh, and S. Ghosh, "A comparative study of summarization algorithms applied to legal case judgments," in *Proceedings of the 17th International Conference on Artificial Intelligence and Law*, 2019, pp. 120–129.

[7] M. Malekzadeh and et al., "Multi-task learning for cross-jurisdictional legal judgment prediction," *Artificial Intelligence and Law*, 2021.

[8] M.-T. Tran, N. Nguyen, and et al., "Legal judgment prediction with interpretable attention-based neural networks," in *Findings of ACL*, 2023.

[9] R. Xie and et al., "Factual legal document summarization with retrieval-augmented generation," *arXiv preprint arXiv:2302.03186*, 2023.

References

1. N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using BERT-Networks," *arXiv preprint arXiv:1908.10084*, 2019.

2. I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "Neural Legal Judgment Prediction in English," in *Proc. 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4317–4323.

3. H. Zhong, Z. Guo, C. Tu, C. Xiao, Z. Liu, M. Sun, and T. Zhang, "Legal Judgment Prediction via Topological Learning," in *Proc. 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 3540–3549.

4. O. M. Sulea, M. Zampieri, M. Vela, J. van Genabith, and L. P. Dinu, "Exploring the Use of Text Classification in the Legal Domain," in *Proc. International Conference on Artificial Intelligence and Law*, 2017, pp. 79–88.

5. P. Bhattacharya, S. Paul, K. Ghosh, S. Ghosh, and S. Ghosh, "A Comparative Study of Summarization Algorithms Applied to Legal Case Judgments," in *Proc. 17th International Conference on Artificial Intelligence and Law*, 2019, pp. 120–129.

6. H. Attali and N. Tomeh, "Transductive Legal Judgment Prediction Combining BERT Embeddings with Delaunay-Based GNNs," in *Proc. Natural Legal Language Processing Workshop*, 2024, pp. 187–193.

7. P. Bhattacharya, K. Ghosh, A. Pal, and S. Ghosh, "Legal Case Document Similarity: You Need Both Network and Text," *Information Processing & Management*, vol. 59, no. 6, 2022, 103062.

8. A. Mandal, K. Ghosh, and S. Ghosh, "Unsupervised Approaches for Measuring Textual Similarity Between Legal Court Case Reports," *Artificial Intelligence and Law*, vol. 29, no. 1, 2021, pp. 1–35.

9. D. Thenmozhi, K. Kannan, and C. Aravindan, "A Text Similarity Approach for Precedence Retrieval from Legal Documents," in *CEUR Workshop Proceedings*, vol. 2036, 2017.

10. J. Cui, X. Shen, and S. Wen, "A Survey on Legal Judgment Prediction: Datasets, Metrics, Models, and Challenges," *IEEE Access*, 2023.

Appendix: Source Code

The appendix section includes supplementary details supporting the CaseLink project, particularly regarding input formatting, sample data, and system components.

A. Sample Structured Judgment Input

- **Title:** Commissioner of Tax of India vs. ABC Pvt. Ltd.
- **Issue:** Whether the assessment made under Section 143(3) was valid.
- **Conclusion:** The court upheld the assessment and dismissed the petitioner's appeal.
- **Winner:** Respondent
- **General Appeal:** The petitioner may appeal to a higher authority challenging procedural fairness and evidentiary evaluation.

B. Preprocessing Techniques Applied

1. Removal of boilerplate text (e.g., disclaimers, advertisements)
2. Standardization of whitespace, punctuation, and casing
3. Extraction of structured fields: Title, Issue, Conclusion
4. Filtering of incomplete or noisy data samples

C. Technology Stack

- **Language Model:** Sentence-BERT (all-MiniLM-L6-v2)
- **Classifier:** Random Forest (300 trees, depth 10, class weight balanced)
- **Libraries:** scikit-learn, sentence-transformers, pandas, matplotlib

D. Embedding Storage and Usage

- All judgment vectors are saved as `.npy` files using NumPy for efficient retrieval
- SBERT embeddings are used for both classification and similarity matching

E. Folder Structure

```
project/

 data/
    raw/
    processed/

 src/
    scraper.py
    embeddings.py
    winner_predictor.py
    similarity.py

 reports/
    case_analysis.pdf

 models/
     winner_model.pkl
```