Project Report on

Text Summarization on Indian Legal Judgements
On Supreme Court Verdicts

Submitted in partial fulfilment of
the requirements for the course

CS60092: Information Retrieval

Submitted by

Anadi Vashist	(19BM6JP05)
Srijan Gupta	(19BM6JP19)
Kowshik Moyya	(19BM6JP50)
Suman Basak	(19BM6JP62)

Under Guidance of

Paheli Bhattacharya (TA)

Submitted to

Prof. Saptarshi Ghosh & Prof. Pawan Goyal

Acknowledgement

Our journey of learning in this project has added a lot to our knowledge. We are glad to take this opportunity to express gratitude to everyone who supported us throughout the course. This project required huge amount of work and dedication. Still, implementation would not have been possible if we did not have support of many individuals.

First of all we would like to thank **Prof. Saptarshi Ghosh** and **Prof. Pawan Goyal** for providing us the project topic to work on. We are really grateful to their class room teachings for this course which provide us good foundation to start our work on this project.

We express deep and sincere gratitude to **Ms. Paheli Bhattacharya**, who took time to hear, guide and always available over a phone call or email, whenever we ran into any trouble or question about the project. She consistently steered us in the right direction throughout the project journey.

Last but not least we express deepest gratitude to **our team mates** who supported each other during difficult times of the project.

Table of Contents

Content Description **Page No.**

List of Figures

List of Tables

1. Introduction

India has a common law system that prioritizes the doctrine of legal precedent over statutory law, and where legal documents are often written in an unstructured way. These legal judgements are often quite long, and typically taking long man hours to go through it.

A solution to this problem is to develop summary for these judgements. The gist of judgement documents should encode important experience and viewpoints of the Supreme Court, and provides instrumental and educational information for judges, lawyers, practitioners, and students. But, summarization of such judgements requires specialized law professionals that are not only expensive, but very less in number.

To tackle this problem, what we propose is a Text Summarization mechanism. Text summarization is an extraction of important text from the original document. The objective of any automatic text summarization system, especially in legal domain, is to produce a summary which is close to human-generated summaries. Based on our observation of the existing gist statements, we can treat the generation of the gist as a sentence classification problem.

1.1. Problem Statement

With this project, we intend to develop a mechanism for summarization of legal documents as binary classification problem where fitness of the solution is derived using **Extractive Summarization Technique**, based on the statistical features of each sentence such as length of the sentence, type of entity, degree of similarity, term frequency–inverse sentence frequency and keywords to generate summary of the document.

Let R denote the statements of the full text section, our goal is to build a service, V, that is able to select some statements S, from R, that can serve as the summary statements, G.

$$\mathbf{V(R) \rightarrow S}$$

We treat this task of sentence selection as a classification problem. In addition to considering some relevant features based on knowledge and observations of linguistic and legal perspectives, we also employ different types of word vectors. We realized the service V with classifiers that were designed based the concepts of logistic regression, gradient boosting, neural networks, and some other methods.

1.2. Reference Paper

Chao-Lin Liu and Kuan-Chun Chen. 2018. Extracting the gist of Chinese judgments of the Supreme Court.
In Proceedings of 2019 International Conference on Artificial Intelligence in Law (ICAIL'19). ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3322640.3326715

2. Data Description

2.1. Data Source

The legal judgement and its summaries were extracted from Legal Information Institute of India (LIIofIndia). LIIofIndia is a not-for-profit publisher that provides free access to individual end-users of the content it provides, to enable them to read, print and copy materials for their personal use, and any other uses permitted by copyright law. LIIofIndia has built a collection of primary and secondary legal materials (including cases, legislation, treaties, journal articles and law reform documents) by agreements with the various sources of the documents and rights-holders in the documents, and by other means such as scanning documents where documents are out of copyright or with the permission of copyright holders.

The data, in interest of our problem statement, is available at this link between the years 1950 to 1995.

2.2. Data Retrieval

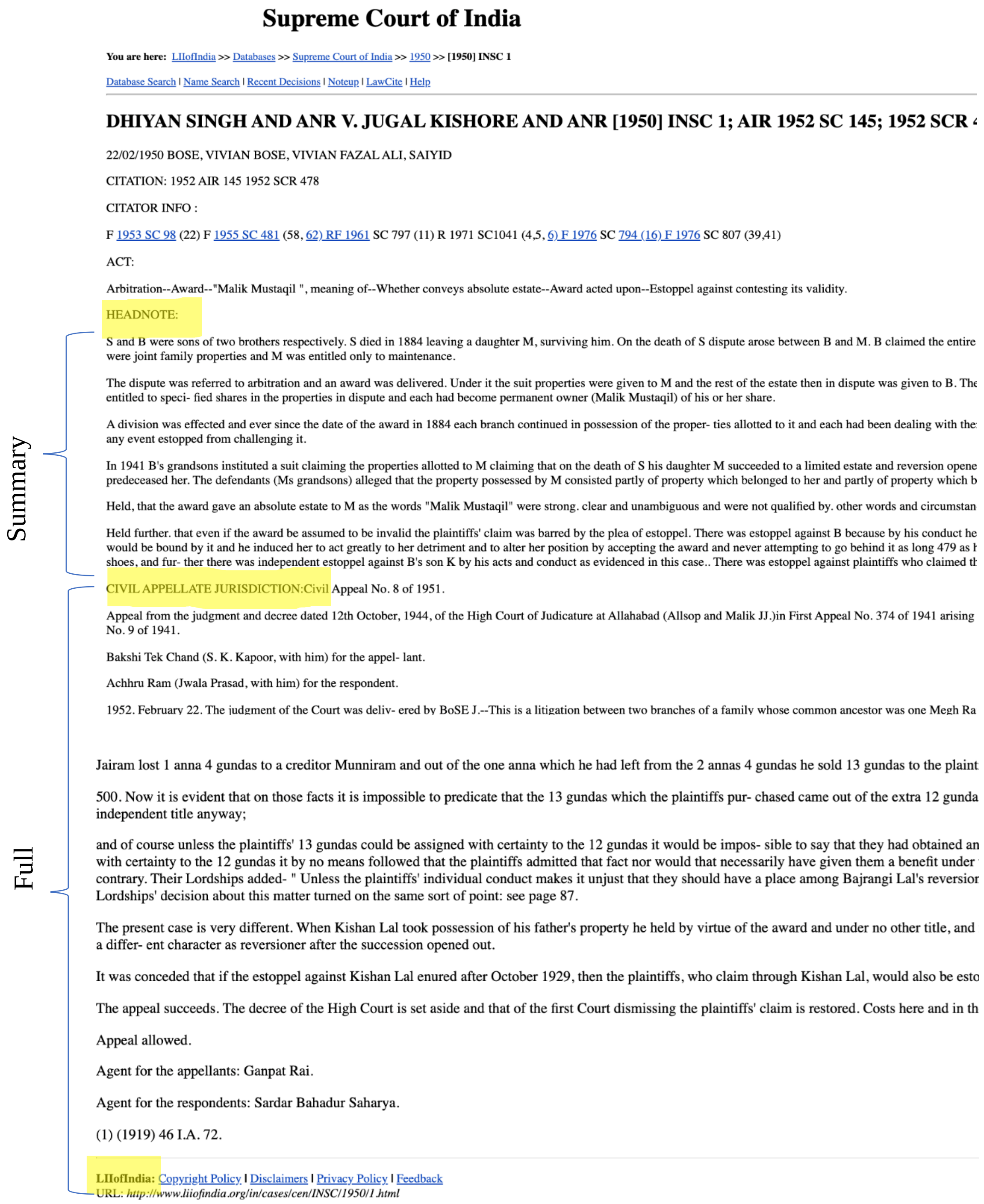
First step in this project pipeline is to access and retrieve the data necessary for our analysis and for building machine learning models as **no ready-made dataset is available**.

Data Extraction Procedure: Data from the source website was extracted by **web scraping** using **Beautiful Soup**. The complete judgement web pages from 1950 to 1995 were continuously iterated and saved in the local machine. Then, out of collection of html docs required data is extracted using Beautiful Soup.

For this project, html documents with the standard judgement format is considered, in which summary section represented as "**Head note**" and judgement or full text section represented as "**Civil Appellate Jurisdiction**". The above format was considered for most of html pages that are downloaded. Whereas, some html pages have spelling mistakes and follows another format which is not consistent with this format. Such documents were excluded.

Head note (Summary) were extracted by filtering the html page using Regular Expression i.e., extract text between 'Head note:' (case insensitive) and 'Civil Appellate Jurisdiction' (case insensitive). Civil Appellate Jurisdiction or Full text were extracted by following same strategy i.e., between 'Civil Appellate Jurisdiction' (case insensitive) and 'LIIofIndia' (case insensitive).

Please refer to the figure below for reference of the procedure.



Supreme Court of India

You are here: LIIofIndia >> Databases >> Supreme Court of India >> 1950 >> [1950] INSC 1

Database Search | Name Search | Recent Decisions | Noteup | LawCite | Help

DHIYAN SINGH AND ANR V. JUGAL KISHORE AND ANR [1950] INSC 1; AIR 1952 SC 145; 1952 SCR

22/02/1950 BOSE, VIVIAN BOSE, VIVIAN FAZAL ALI, SAIYID

CITATION: 1952 AIR 145 1952 SCR 478

CITATOR INFO :

F 1953 SC 98 (22) F 1955 SC 481 (58, 62) RF 1961 SC 797 (11) R 1971 SC1041 (4,5, 6) F 1976 SC 794 (16) F 1976 SC 807 (39,41)

ACT:

Arbitration--Award--"Malik Mustaqil ", meaning of--Whether conveys absolute estate--Award acted upon--Estoppel against contesting its validity.

HEADNOTE:

S and B were sons of two brothers respectively. S died in 1884 leaving a daughter M, surviving him. On the death of S dispute arose between B and M. B claimed the entire
were joint family properties and M was entitled only to maintenance.

The dispute was referred to arbitration and an award was delivered. Under it the suit properties were given to M and the rest of the estate then in dispute was given to B. The
entitled to speci- fied shares in the properties in dispute and each had become permanent owner (Malik Mustaqil) of his or her share.

A division was effected and ever since the date of the award in 1884 each branch continued in possession of the proper- ties allotted to it and each had been dealing with the
any event estopped from challenging it.

In 1941 B's grandsons instituted a suit claiming the properties allotted to M claiming that on the death of S his daughter M succeeded to a limited estate and reversion opene
predeceased her. The defendants (Ms grandsons) alleged that the property possessed by M consisted partly of property which belonged to her and partly of property which b

Held, that the award gave an absolute estate to M as the words "Malik Mustaqil" were strong. clear and unambiguous and were not qualified by. other words and circumstan

Held further. that even if the award be assumed to be invalid the plaintiffs' claim was barred by the plea of estoppel. There was estoppel against B because by his conduct he
would be bound by it and he induced her to act greatly to her detriment and to alter her position by accepting the award and never attempting to go behind it as long 479 as h
shoes, and fur- ther there was independent estoppel against B's son K by his acts and conduct as evidenced in this case.. There was estoppel against plaintiffs who claimed th

CIVIL APPELLATE JURISDICTION:Civil Appeal No. 8 of 1951.

Appeal from the judgment and decree dated 12th October, 1944, of the High Court of Judicature at Allahabad (Allsop and Malik JJ.)in First Appeal No. 374 of 1941 arising
No. 9 of 1941.

Bakshi Tek Chand (S. K. Kapoor, with him) for the appel- lant.

Achhru Ram (Jwala Prasad, with him) for the respondent.

1952. February 22. The judgment of the Court was deliv- ered by BoSE J.--This is a litigation between two branches of a family whose common ancestor was one Megh Ra

Jairam lost 1 anna 4 gundas to a creditor Munniram and out of the one anna which he had left from the 2 annas 4 gundas he sold 13 gundas to the plaint

500. Now it is evident that on those facts it is impossible to predicate that the 13 gundas which the plaintiffs pur- chased came out of the extra 12 gunda
independent title anyway;

and of course unless the plaintiffs' 13 gundas could be assigned with certainty to the 12 gundas it would be impos- sible to say that they had obtained an
with certainty to the 12 gundas it by no means followed that the plaintiffs admitted that fact nor would that necessarily have given them a benefit under
contrary. Their Lordships added- " Unless the plaintiffs' individual conduct makes it unjust that they should have a place among Bajrangi Lal's reversior
Lordships' decision about this matter turned on the same sort of point: see page 87.

The present case is very different. When Kishan Lal took possession of his father's property he held by virtue of the award and under no other title, and
a differ- ent character as reversioner after the succession opened out.

It was conceded that if the estoppel against Kishan Lal enured after October 1929, then the plaintiffs, who claim through Kishan Lal, would also be esto

The appeal succeeds. The decree of the High Court is set aside and that of the first Court dismissing the plaintiffs' claim is restored. Costs here and in th

Appeal allowed.

Agent for the appellants: Ganpat Rai.

Agent for the respondents: Sardar Bahadur Saharya.

(1) (1919) 46 I.A. 72.

LIIofIndia: Copyright Policy | Disclaimers | Privacy Policy | Feedback
URL: http://www.liiofindia.org/in/cases/cen/INSC/1950/1.html

Fig 1. Representation of an html page to highlight extraction mechanism

2.3. Challenges

a. During web scraping of source website, the server repeatedly responded with status 410 (resource request is permanently deleted), even though requested page can be seen in the website. Significant time was spent on finding work around for this issue. To solve this, the user-agent was specified as a field in request header while making a request to server.

```
headers = {'User-Agent': 'PGDBA_Students'}
result = requests.get(url ,headers = headers)
```

b. The html pages do not have proper DOM structure, due to which extracting the required data using Beautiful Soup required a lot of effort. To tackle this problem, regular expressions were used. Required data were extracted through multiple cycles of testing satisfying in other web pages.

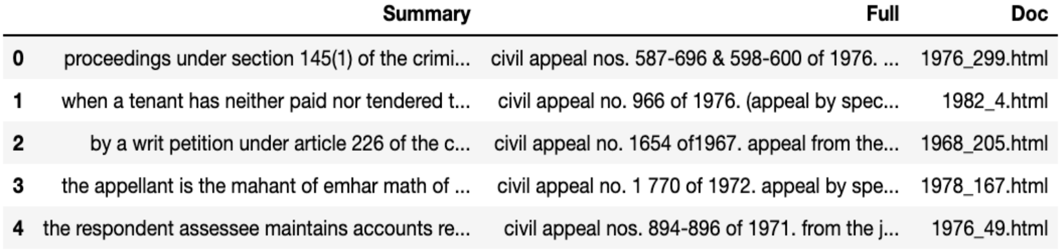
2.4. Data Insights

A few insights on the extracted corpus are as follows:

1. A total of 4946 legal judgements and their corresponding summaries were obtained.
2. Total Size of extracted data amounted to 140 MB in size
3. The developed data's structure is of format [Summary, Full, Doc], where

   Summary : Head Note of Judgement Data
   Full : Judgement Data or Civil Appellate Jurisdiction
   Doc : (year)_(document_number).html

A sampled view of the yielded data is as follows:



	Summary	Full	Doc
0	proceedings under section 145(1) of the crimi...	civil appeal nos. 587-696 & 598-600 of 1976. ...	1976_299.html
1	when a tenant has neither paid nor tendered t...	civil appeal no. 966 of 1976. (appeal by spec...	1982_4.html
2	by a writ petition under article 226 of the c...	civil appeal no. 1654 of1967. appeal from the...	1968_205.html
3	the appellant is the mahant of emhar math of ...	civil appeal no. 1 770 of 1972. appeal by spe...	1978_167.html
4	the respondent assessee maintains accounts re...	civil appeal nos. 894-896 of 1971. from the j...	1976_49.html

Fig 2. Snapshot of our structured dataframe

3. Data Preprocessing and Normalization

Once the data was extracted as a neat dataset, data preprocessing needed to be done. This stage involves text cleaning, and normalizing text to bring text components like words to some standard format. This enabled **standardization across document corpus**, which helps in building meaningful features and helps in **reducing noise** due to irrelevant symbols etc.

Following are different preprocessing techniques followed.

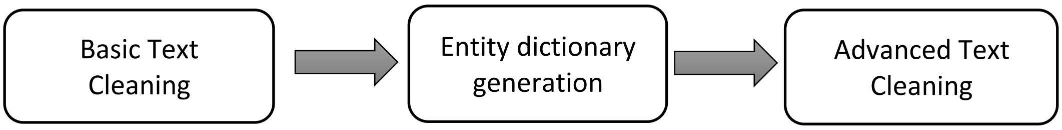


Fig 3. General Preprocessing Scheme

3.1. Basic Text cleaning

Before Implementing Named Entity Recognition (NER) some of the basic cleaning techniques are implemented such as

1. Reducing Extra Spaces**:** In between the words to single space. This was done as spaces are being recognized as tokens during tokenization. For instance,

   “Indian Extradition Act” → “Indian Extradition Act”

2. Cleaning Short Hands And Expanding Contractions: By nature, short hands and contractions pose a problem for NLP and Text analytics and there should be a definite process for dealing these. So, a mapping for contractions and short hands and corresponding expansion was used for this type of cleaning. For instance:

   ‘Respondent’s family members ‘re coming out’→ ‘Respondent family member are coming out

3. Replacing Special Characters: Some of the special characters are extensively used in formal or informal writings like “%”, “&” and “etc.” which needed to be mapped to the corresponding English word for preserving the meaning of the texts. For instance,

   ’%’ → ‘percent’, “&” → ‘and’

3.2. Implementation of Named Entity Recognition (NER)

Motivation Behind Implementing NER: The document corpus had so many unique proper nouns used under same context like Name of persons, Month, Nationalities etc. If there were no mechanism to club the different words used in same context under single entity, each would contribute to be a unique term in vocabulary, thus increasing vocab size and not capturing the essence of the words. So, NER was employed so that different proper nouns were tagged under the entity it belongs to.

Entity Dictionary Generation: For the task of recognizing to which entity a proper noun belongs to, spaCy a free open-source library for Natural Language Processing in Python was used. Complete document corpus was traversed to generate a “Entity Dictionary” which maps different proper nouns to corresponding entity it belongs to like person, number, date, or act etc.

The dictionary generated was saved as it is one-time activity on corpus and saves computation time. For instance,

```
Entity dictionary = {"october":"date",
                     "canadian":"NORP",
                     "Jawahar":"Person"}}
```

The above dictionary was used to replace the different proper nouns with entity it belongs to.

Challenges faced during NER Implementation

1. Data Challenge: Proper nouns written in short hand in summary i.e., for some documents or judgement data the name of the person in full text were represented as a short form in summary. For instance:

   Brijlal (full text) → B (summary)
   SpaCy 's NER was unable to recognize B as a" person".

2. SpaCy NER Inefficiency: The algorithm was unable to recognize and tag Indian names of people, states, organization
   For instance: Ram, Jawahar, Rajasthan, Tata Iron and Steel Company

3.3. Advanced Text Cleaning

After NER was implemented the proper nouns are replaced with entity it belongs to, now, some remaining text cleaning techniques are used to remove as much noise and retain as much underlying meaning as possible.

1. Removing Stop Words and Punctuations**:** Stop words are words that have little significance and are usually removed from text when processing it so as to retain the words having maximum significance or context. Punctuations were also removed as it has little significance. For instance,
   'Indian Extradition Act was passed which provided' → 'Indian Extradition Act passed provided'

2. Skip Continuous Entity String: During text analysis some patterns were observed like '24th October ,1994' replaced as 'date date date' after NER implementation. But practically the complete date '24th October ,1994'belongs to a single entity 'date' instead of 'date date date'. So, this type of cases was handled by writing custom code. For Instance,
   24th October ,1994 → date date date → date (single)

3. Replacing Proper Nouns With Entity Tags: The dictionary generated in previous step is used to replace the proper nouns with entity it belongs to. For Instance,
   'Indian Extradition Act passed provided' → 'Indian law passed provided'

4. Lemmatization: the words and handled pronouns (--PRON--). In order to normalize the text components like words to basic root form, lemmatization using spaCy is employed. Different variants of basic root word are lemmatized to bring all to a single root word called lemma. For Instance,
   'Indian law passed provided' → 'Indian law pass provide'

3.4. Final Result

Below is an example of text sentence in document corpus before text preprocessing and after preprocessing passing through Basic Cleaning, Implementing NER, Advanced Cleaning.

Before: In 1903 the Indian Extradition Act was passed which provided
After: date Indian law pass provide

3.5. Challenges Faced in Preprocessing

1. Lack Of Computational Power
   - Preprocessing on complete 5000 docs lead to run time crashing due to limited availability of RAM in local machines as well as Google Collab.
   - Work Around: Dividing the complete data frame into multiple batches of size 30 and saving the preprocessed data frame iteratively for each batch into drive. This avoided loss of work even if Collab crashes and was able to resume on the remaining batches that need to be pre processed

2. Preprocessing Time Exceeding Time Limit Of Google Collaboratory
   - Preprocessing including Basic cleaning, NER implementation, Advanced Cleaning required a lot of time sometimes crossing time limit of Google Collab 12 hrs.
   - Work Around: Same strategy of dividing the data frame into batches i.e., each team member preprocessing on one fourth of the data, each using 2 Google Collab accounts, and 4 jobs per each Collab account, Total 4 * 2 * 4 = 32 preprocessing jobs parallelly.
   - This strategy drastically reduced the preprocessing time.

4. Mechanism

4.1. Normalization and PCA (for TF.iDF Sparse Matrix)

Since, classification models cannot interpret sparse matrix data, and dense matrix conversion of some of the achieved tfidf required **huge memory capacities** (174.67 GB), so features were generated from the tfidf using **Latent Schematic Analysis.**

Latent semantic analysis (LSA) is a technique in natural language processing for analysing relationships between a set of documents and the terms they contain by producing a set of concepts related to the documents and terms. It was achieved using **dimensionality reduction**.

These approaches do not centre the data before computing the dimensionality reduction operation. Hence, data was cantered using **normalize** function of SciKit Learn's preprocessing package along the column.

The features were then generated using **SVD transformation** on the existing data to yield 500 features. (Optimal number of features were selected based on memory capabilities for processing and trying to retain maximum number of feature).

4.2. Cosine Similarity Generation (for all embedding approaches)

We have three different sentence embedding arrays post the developed embedding process, namely full text sentences embeddings, summary sentences embeddings, and gold standard sentences embeddings.

Assumption: Here, we assume that a given full text sentence is similar to only a single summary sentence. This assumption is important, because the summary which we're using is not an extractive summary, but an abstractive summary.

So, for a given document, we compute the cosine similarity between all possible pair of full text sentence to a summary sentence, using the developed schematic embeddings of the pair of sentences.

Mechanism: The data of each embedding array was first normalized to make each sentence vector a unit vector in magnitude. Normalization was achieved using SciKit Learn's normalize function under 'l2' constraint. For each document, the following operation was done to develop all possible pair of summary-full text sentence pair's cosine similarity.

$$C_i = {S_i}^T F_i$$

Where,

S_i: Matrix containing normalized summary sentence embeddings for document i
F_i: Matrix containing normalized full text sentence embeddings for document i
C_i : Matrix with each element (<j, k>) representing cosine similarity if jth summary sentence and kth full text sentences for document i

Under the above assumption, we find the maximum cosine similarity of a given full text sentence to any summary sentence of the same document, and develop this for all the full text sentences.

A visualization of the mechanism is as follows:

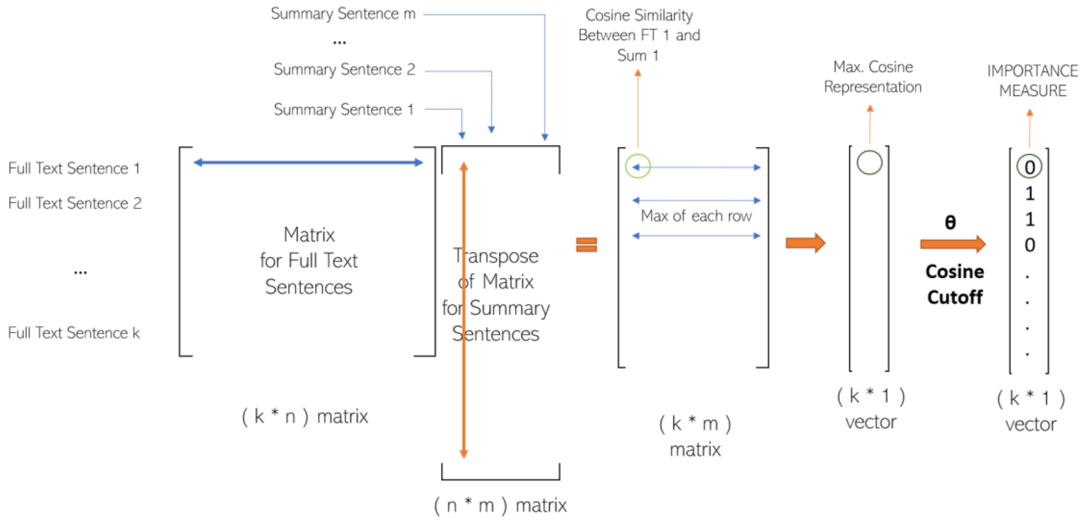


Fig 4. Step-by-step visual representation of generating importance of sentences using their embeddings

5. Embedding Generation

5.1. TF-IDF with Name and Entity Recognition

The tfidf embedding of the sentences were generated using TF.iDF vectorizer function available in feature extraction package of SciKit Learn's utility tools. The operation yielded us a numerical matrix with 69,229 features. The result generated was available in a sparse matrix format. These features were reduced to 500 features using PCA.

Mechanism Effectiveness Measurement: to test the effectiveness of the dimensionality reduction process on our data, we sampled 50 documents and developed the importance vector feature column once using tfidf operation and another time using tfidf followed by dimensionality reduction operation. The cosine similarity between the two vectors achieved was 0.87 that gave us sufficient confidence in our mechanism and also ensured that not much information loss was happening with dimensionality reduction operation.

A brief summary of the developed corpus is as follows:

Number of Judgement Documents	4,946
Number of Sentence Segments	667,104
Total number of sentences in the gist	142,745
Proportion of Gist Segments	21.398%
Number of Gold Standard Sentences	7,103
Dimension of Sentence Embeddings	200

Table 1. Summary of the developed corpus

The distribution of the cosine similarity, thus achieved, is as follows:

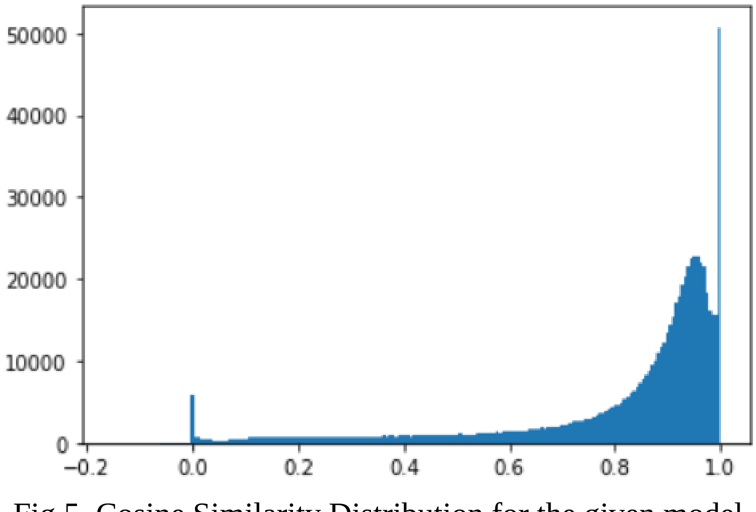


Fig 5. Cosine Similarity Distribution for the given model

Based on the above distribution, we intended to classify roughly 25% of the documents as important. So, a threshold of **0.95** was kept to classify a full text sentence as important or not. Importance of sentences with **cosine similarity more than or equal to 0.95 was marked as 1**, and the remaining sentences were marked 0. With this, we had prepared a dataset which had **29.68% important sentences** and 70.31% not so important sentences.

Note: the importance of sentence here is in context to that sentence being part of the summary sentence.

5.2. TF-IDF Generation using Legal Dictionary:

As a second approach we have created TF-IDF features using legal dictionary. We have taken this approach in order to incorporate only standard legal terms that are available in form of dictionary. We have used legal dictionary that is available at https://dictionary.law.com/. In the dictionary we have total of 2328 legal terms. The legal terms are in form of unigram, bi-gram or tri-gram. Examples: All Terms are in legal context, For instance, “your honor”, “goodwill”, “power of attorney” etc.

A brief summary of the developed corpus is as follows:

Number of Judgement Documents	4,946
Number of Sentence Segments	613,890
Total number of sentences in the gist	133,358
Proportion of Gist Segments	21.723%
Number of Gold Standard Sentences	8,916
Dimension of Sentence Embeddings	200
Explained Variance Ratio (post PCA)	81.70%

Table 2. Summary of the developed corpus

We have used TF-IDF vectorizer using text feature extraction of scikit-learn library. Here we have specifically defined our legal dictionary as the feature space. So, total of 2328 features are generated using this method.

As working with this huge data is still computationally expensive, PCA was applied on out developed embeddings to reduced the dimension of each entry from 2328 to 200, at the same time, retaining explained variance ratio of 81.7% post PCA.

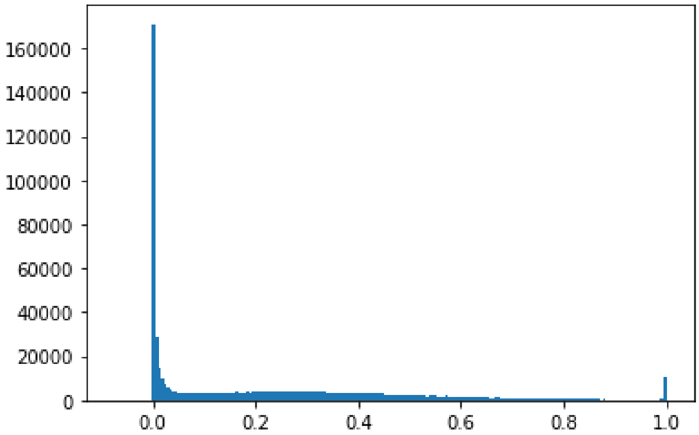


Fig 6. Cosine Similarity Distribution for the given model

Using cosine threshold value of **0.5**, we obtain the response variable for the binary classification problem was obtained. Flagging full text sentences with a cosine similarity greater than the threshold as important. For the given threshold **14.61%** of full text sentences were flagged as important.

Advantages: By taking the legal terms into account we have reduced the no of features drastically. Our computations become faster in subsequent steps like model training. We were able to eliminate unnecessary non-legal terms from feature space, which had none to less importance in model building.

5.3. Word-to-Vector Embeddings (word2vec)

Word2Vec package from the **Gensim** library was used to obtain embeddings for words in the sentences. These word embeddings were **weighted averaged**, based on occurrence and usage, to yield sentence embeddings. The embeddings were developed using **5 content window size**, for words with **minimum 2 occurrences** in the corpus, and under **skip-gram condition**.

A brief summary of the developed corpus is as follows:

Number of Judgement Documents	4,946
Number of Sentence Segments	613,890
Total number of sentences in the gist	133,358
Proportion of Gist Segments	21.723%
Number of Gold Standard Sentences	8,916
Dimension of Sentence Embeddings	500

Table 3. Summary of the developed corpus

Each sentence embedding has dimension of 500. The dimension was chosen based on ability to capture maximum context possible with **available resources** and **computational power**.

Cosine similarity between each summary text sentence embedding and a corresponding full text sentence embedding which is maximum similarity to it has the following frequency distribution. The observed distribution for the full text sentences are as follows:

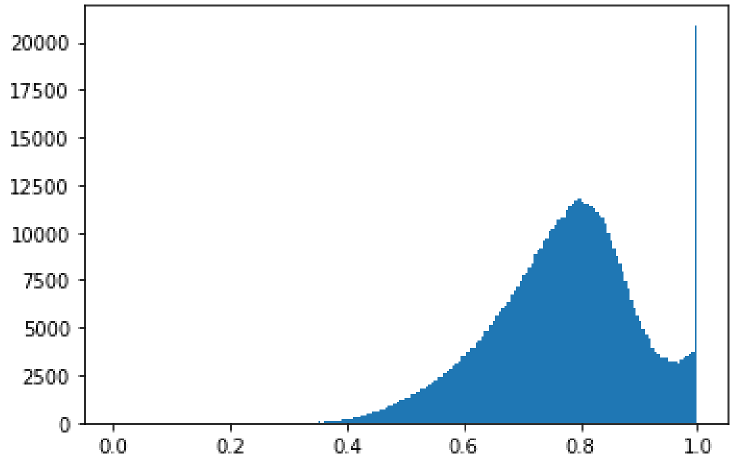


Fig 7. Cosine Similarity Distribution for the given model

Using cosine threshold value of **0.85**, we obtain the response variable for the binary classification problem was obtained. Flagging full text sentences with a cosine similarity greater than the threshold as important. For the given threshold **26.18%** of full text sentences were flagged as important.

5.4. BERT Embeddings:

Sentence transformers were used for obtaining semantically meaningful sentence embedding of RoBERTa. The Pre-Trained model used for obtaining the embeddings was "**roberta-base-nli-mean-tokens.**"

A brief summary of the developed corpus is as follows:

Number of Judgement Documents	1,000
Number of Sentence Segments	130,759
Total number of sentences in the gist	27,476
Proportion of Gist Segments	21.02%
Number of Gold Standard Sentences	9,092
Dimension of Sentence Embeddings	1024

Table 4. Summary of the developed corpus

Each sentence embedding is a vector with 1,024 numerical feature. Cosine similarity between each summary text sentence embedding and a corresponding full text sentence embedding which is maximum similarity to it has the following frequency distribution. The observed distribution for the full text sentences is as follows:

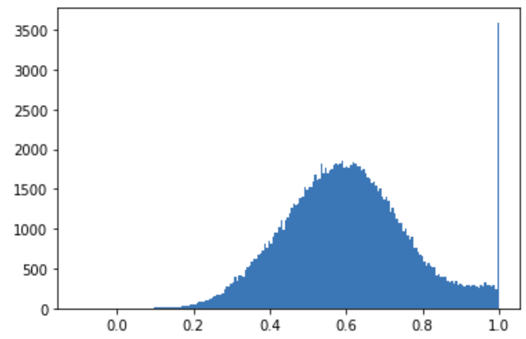


Fig 8. Cosine Similarity Distribution for the given model

The using threshold value of **0.73** we obtain the response variable for the binary classification problem was obtained. Flagging full text sentences with a cosine similarity greater than the threshold as important. For the given threshold **26.6%** of full text sentences were flagged as important.

6. Model Fitting

The **embeddings generated** for the sentences were used as the **input features**, and these embedded features were used to predict whether a sentence is **important** to be incorporated as part of the summary. Here, each row represents an embedding of a sentence and their corresponding importance classification. To incorporate statistical structures in data for classification of sentences, **Machine Learning models** (Logistic Regression, Random Forest Classifier, Extreme Gradient Boosting (XGB) Classifier, and simple Neural Networks) were used.

The whole prepared datasets were divided into two sets, **training** and **validation** dataset. Training dataset comprised of 70% of the data, and validation dataset the other 30%. The models were developed over training datasets, and the **generalization ability** and **applicability** of the developed models were tested using validation dataset, unseen to the model.

6.1. Evaluation Metric

Accuracy is not a good measure in our case. We have chosen **F1 score** as our metric. F1 score is harmonic mean of the recall and the precision.

$$F1\ score = \frac{2 * (\mathrm{precision} * \mathrm{recall})}{\mathrm{precision} + \mathrm{recall}}$$

It is also called the F Score or the F Measure. In another way, the F1 score conveys the balance between the precision and the recall. It **heavily penalises** a lower value of recall and/or precision. In order to have a significant F1 score, we **need to have both the recall and the precision** to be higher.

6.2. Classification Probability Threshold:

The decision for converting a predicted probability or scoring into a class label is governed by a parameter referred to as the “decision threshold,” “discrimination threshold,” or simply the “threshold.” The default value for the threshold is 0.5 for normalized predicted probabilities. The threshold was customized to yield maximum F1 score on validation results. A typical distribution of validation F1 score and the validation accuracy with varying threshold is as follows:

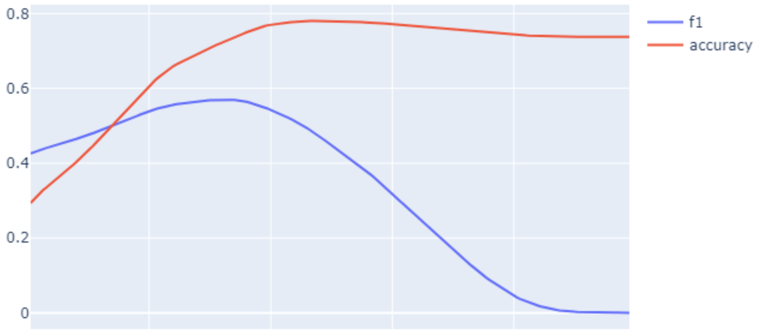


Fig 9. Plot of validation F1 score and accuracies with varying thresholds

Here, x-axis represents the threshold value of probabilities, and the y-axis represents the values. The above curve was developed for Deep Neural Network Model for Word2Vec developed embeddings.

6.3. Models Developed

1. **Logistic Regression:** Logistic regression statistical model here was used to model a binary class variable, using the embedded features of a sentence. The parameter space of the model was explored with different **regularization techniques** (L1 and L2), along varying **C** values to yield the maximum F1-score on validation dataset. The **classification probability threshold** was further tuned to yield maximum value of evaluation metric.

2. **Random Forest Classifier:** Random Forest Classifier is used to classify a sentence is part of summary or not using different types of embedding features of the sentence. The parameter space of the model was explored by tuning different parameters like '**no of estimators', 'min samples leaf'** and **min leaf nodes'** over a range of values to yield maximum F1-score on validation dataset. The classification probability threshold was further tuned to yield maximum value of evaluation metric.

3. **SVM:** Support vector machine (SVM) was used to classify binary outcome using different embedded features of sentences. SVM model was optimized by parameter space exploration. Different values **Regularization parameter**, C and **kernel type** ('linear', 'polynomial', or 'radial basis function') are explored to yield the maximum F1-score on validation dataset. The **classification probability threshold** was further tuned to yield maximum value of evaluation metric.

4. **XGB Classifier:** Gradient boosting algorithm which provides parallel tree boosting, **XGBOOST**, was used to train the classification model with the objective function '**binary: logistic**' to maximize the evaluation metric '**ROC AUC** '. The hyper-parameter tuning was done using Bayesian optimization to build a probabilistic model of the objective function and use it to select the most promising hyperparameters to evaluate in the true objective function. We used **5-fold cross validation** while tuning the hyper-parameters. Finally, F1 scores were evaluated for the trained model on the test data.

5. **Neural Network:** Deep neural networks were developed using Tensorflow 2.0. The **number of layers** and **hidden stages** were decided to maximize the F1 score on validation dataset. The loss function used here was **binary_crossentropy** using **Adam** optimizer, with varying **decay** and **learning rates** to maximize our evaluation metric score. The **classification probability threshold** was further tuned to yield maximum value of evaluation metric.

6.4. Challenges in Model fitting:

- **Resource Constraint:** We had limited memory access, with very low RAM capacities. This hindered us to perform more accurate embeddings by getting more feature representations.
- **Computational Power:** hindered us from exploring other advanced Neural Network architectures like LSTM Models, Bi-Directional RNN Models, and Attention Models due to lack of faster processing units like GPUs.

7. Results & Discussion

The F1 scores corresponding to different models trained on the sentence embeddings is given below. The scores are calculated by fitting the trained model on validation dataset.

	NER TF.iDF	**Vocab TF.iDF**	**Word2Vec**	**RoBERTa**
Logistic Regression	0.5636	0.3916	0.5370	0.4608
Random Forest Classifier	0.5823	0.1551	0.5338	0.4381
Deep Neural Network	0.6544	0.4168	0.5704	0.4707
XGB Classifier	0.6530	0.423	0.5341	0.4529
SVM	-	0.235	-	-

Table 5. Summarizing model results for different kinds of dataset produced

It is evident from the table that the classifier based on the **Deep Neural Network Models** outperformed the classifiers trained using Logistic Regression, Random Forest, XGBOOST and Support Vector Machine for each of the validation datasets created using the four different methodologies for generating the sentence embeddings.

Based on the F1 scores of validation dataset we see the embeddings obtained by performing PCA on normalized tf-idf using Name Entity Recognition have higher scores for each of the 5 types of models trained.

The Rouge scores for the 50 gold standard documents will help better analyze the relative comparison between the four embedding methodologies for our project.

8. Future Scope

1. We have achieved good accuracy and F1-score using machine learning method for the purpose of extractive summarization. We can extend our study to abstractive summarization of Indian legal documents.
2. We can use advanced models like LSTM to classify sentences whether it is an important one or not.
3. We can further improve our results by implementing ensemble and stacking to the classifiers that we have already implemented.