International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 05 | May 2021 www.irjet.net p-ISSN: 2395-0072

# Extraction of Legal Documents for Assistance to Lawyers

**Shreyas Shetty[1] , Aditya Mhatre[2] , Ashok Singh[3] , Prof. Manasi Kulkarni[4]**

[1]*Student, Information Technology Engineering, Pillai College of Engineering, Maharashtra, India*
[2]*Student, Information Technology Engineering, Pillai College of Engineering, Maharashtra, India*
[3]*Student, Information Technology Engineering, Pillai College of Engineering, Maharashtra, India*
[4]*Faculty, Information Technology Engineering, Pillai College of Engineering, Maharashtra, India*

---

**Abstract -** *The functioning of court cases results in the engenderment of many documents, most of them in the form of digital copy or text documents. Many of them are licitly relevant and involute, which makes their understanding difficult. They are indicted in natural language for lawyers, which is hard for computer processing and hence hard for further analysis. Documents represent an abundance of data in unstructured form, So, We describe an information extraction and retrieval system, which extracts data and retrieves relevant information of prior cases from a database predicated on a query passed by the user. Hence with the motive of engendering such a system wherein, the input would be given as a keyword or number of keywords to obtain the desired result in the form of a relationship between the query and all the case documents. Our system employs a cumulation of information retrieval, Information Extraction, Natural Language Processing techniques like Term Frequency, Inverse Document Frequency, and Cosine Similarity which will rank case documents according to the query. The goal is to facilitate the work of professionals in terms of processing large magnitudes of documents. Incremented productivity should be propitious for both natural and legal professionals when working with textual and licit issues.*

***Keywords -*** *TF-IDF, Data Preprocessing, Cosine Similarity, Vectorization, Confusion Matrix, Document Retrieval, Summarization*

## 1. INTRODUCTION

Modeling, developing, and implementing systems capable of providing expeditious and efficacious content-predicated access to astronomically vast amplitudes of information is the goal of Information Retrieval (IR). An information retrieval system (IR system) attempts to determine the relevance of information objects such as text documents, photographs, and video to a user's information needs. A demand, which corresponds to a bag of words, is used to express such a need for information. Users are only interested in the knowledge products that are relevant to their needs.The information items should be represented and organised in such a way that the user can easily access the information that piques his interest. An IR system's main objective is to retrieve all information items relevant to a user query while removing as many non-germane items as possible. In addition, the information items retrieved should be classified from most relevant to least relevant.

The method of extracting unique (pre-specified) information from textual sources is known as information extraction. When your email extracts only the data from the message for you to add to your Calendar, this is one of the most basic examples. Judicial acts, medical records, social media interactions and streams, online news, government documents, business papers, and other free-flowing textual sources can all be used to obtain structured information.

Extraction of Legal Documents is a method for extracting related forms or domain cases so that users can more easily review the case. If we do it manually, studying each case and looking for related domain cases takes a long time. The proposed system will make the work simpler and faster, allowing the lawyer to devote more time to the case.

## 2. LITERATURE SURVEY

### 2.1. Extracted Summary Based Recommendation System for Indian Legal Documents

In the year 2020, at the 11th International Conference on Computing, Communication, and Networking Technologies (ICCCNT), Aashka Trivedi, Anya Trivedi, Sourabh Varshney, Vidhey Joshipura, Rupa Mehta, and Jenish Dhanani[1] proposed a novel framework to classify certain paragraphs of the case document that relate to its summary and use them to retrieve other homogeneous documents. This method used a dataset with specifically named summary paragraphs of Indian Supreme Court documents from before the 1970s to train a Support Vector Classifier to achieve this aim.They addressed a model that produced promising results with a high level of precision. When opposed to considering the document holistically, using only the extracted description to retrieve similar documents demonstrates better efficiency in terms of time and space involution.

We have utilized this paper as the base of our system, in this paper, they have a summary of the incipient document as input to retrieve homogeneous documents but we are utilizing keywords from the incipient document as a query

to retrieve relative documents from the corpus additionally in lieu of SVM we are utilizing cosine homogeneous attribute to visually perceive and retrieve cases from the corpus which are most proximate to the query.

### 2.2. Automatic Catchphrase Identification from Legal Court Case Documents

In 2017, Arpan Mandal, Kripabandhu Ghosh, Arindam Pal, and Saptarshi Ghosh[2] published a model consisting of an unsupervised scheme for catchphrase identification from documents in CIKM. In comparison to non-legal domains, they implemented a new scoring function to determine the value of a word in the legal domain. There are a variety of methods/functions for scoring phrases, such as BM25, KL-divergence dependent score, Mysore, and they've also suggested a technique for estimating the value of a term defined in the legal domain relative to a non-local domain. This method generates a weighted list of extracted patterns from a legal text, with the patterns being ranked according to different scores. The extracted phrases are then ranked using the supervised KEA method. Then it determines if these methods correctly retrieve Manupatra(Indian Law Database) catchphrases.

### 2.3. Automatic Extraction of Catchphrases from Software License Agreement

Fareeha Zahoor and Imran Sarwar Bajwa[3] proposed an approach to extract catchphrases from software licence agreements at the Sixth International Conference on Intelligent Human-Machine Systems and Cybernetics in 2014. Extraction of catchphrases is done in four phases. First perform lexical analysis, which is a process of converting a string of characters in a series of symbols, it additionally contains tasks such as sentence splitting - divide the text into sentences that are discrete from each other by brackets, Tokenization, POS tagging - according to the nature of every word Part Of Verbalization Tagger tag the words, Thematic Segmentation - According to rules, dividing the text into structural blocks is done by thematic segmentation. Secondly, syntactic processing, which contains tasks such as Parse Tree Engenderer and the engenderment of dependencies. Thirdly perform filtering and at the cessation, we make a corpus of license terms that cull from software license acquiescence, and then the cull of catchphrases is done from utilizing this corpus.

### 2.4. Information extraction from case law and retrieval of prior cases

In the year 2003, Peter Jackson, Khalid Al-Kofahi, Alex Tyrrell, and Arun Vachher[4] identify a framework called "History Assistant" that non-nugatory combines knowledge extraction, machine learning, and information retrieval. It extracts judicial language from electronically imported court opinions, as anteriorly reported in and utilizes this information to retrieve cognate cases from a citator database. Its role in a production environment would be to suggest links between 'old' cases already in the citator and incipient cases under editorial review. The architecture has two principal components, a set of natural language modules and a prior case retrieval module, which perform the extraction and retrieval tasks, respectively. They present recall and precision data on both of these tasks, provide an overview of the plenarily implemented system, and discuss several theoretical and practical quandaries encountered. The pristine contribution of this work lies in the cumulation of techniques needed to approach industrial-vigor performance.

### 2.5. Automatic Extraction of Entities and Relation from Legal Documents

In 2018, Judith Jeyafreeda Andrew and Xavier Tannier[5] presented a framework that automatically distinguishes specified entities and the relationships between various entities within a dataset of certain types of licit documents that contain details about people investing in property at the Association for Computational Linguistics. This avails journalists to identify some utilizable information - information like the denomination of the person investing and the company invested in. They proposed a hybrid method to automatically detect variants of relationships after identifying the entities within the corpus. It follows a cumulation of statistical and rule-predicated techniques to achieve the goal. Firstly, To identify and relegate the entities within each of the text documents, and Secondly, To identify the relationships between the entities. To achieve the objectives, It presents a hybrid system that explores an amalgamation of two techniques for Designated Entity apperception (a statistical approach utilizing Conditional Arbitrary Fields (CRF) and rule-predicated techniques) and engenders a graph with all entities and their relationships, in the perspective of an investigative journalism use.

### 2.6. Legal Claim Identification: Information Extraction with Hierarchically Labeled Data

In 2010, Mihai Surdeanu, Ramesh Nallapati, and Christopher Manning[6] at Stanford University proposed a novel Information Extraction conundrum in which only parts of documents are relevant, and linguistic annotations are only available for these segments. The data is hierarchical: the top layer marks the pertinent text segments and the bottom layer annotated domain-concrete entity mentions, but only in the segments marked as germane in the top layer. They investigate this quandary in the licit domain, where we extract the text corresponding to litigation claims and entity mentions such as patents and

laws in each claim. Because entity mentions are not labeled outside claims in training data, a top-down approach that extracts claims first and entity mentions next seem the most natural. However, they show that other models are superior. Utilizing a simple semi-supervised approach they implement a bottom-up Conditional Random Field model; they will implement a joint hierarchical CRF utilizing a cumulation of pseudo-likelihood and Gibbs sampling. They show that both these models significantly outperform the top-down approach.

## 3. PROPOSED SYSTEM

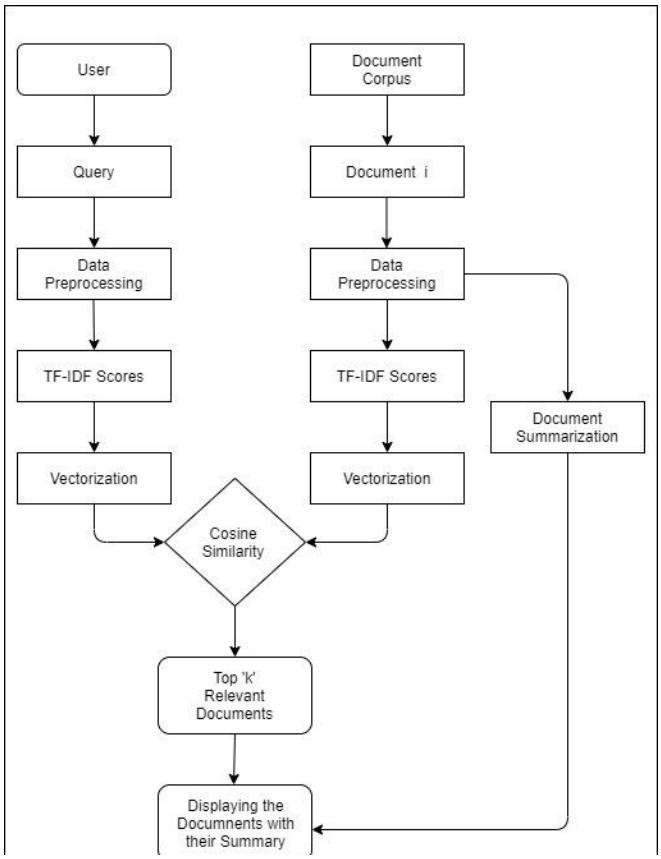


Figure 3.1. Proposed System

**Document Corpus**: It is a Collection of Unstructured Legal Case Documents taken from "Indiankanoon.org" which is in pdf format. It is the Knowledge Base on which our System will work.

**Data Preprocessing:** Data preprocessing is a significant step in building an NLP model, and the outcomes are dependent on how extensively the data has been preprocessed. When dealing with any kind of text model, preprocessing is one of the most important steps. During this phase, we must acknowledge the distribution of our data, the approach required, and the extent to which we should clean.

This phase has no hard and fast rules and is entirely dependent on the problem statement. The following are the basic preprocessing measures we used in our proposed system:

(i)Converting all the text into Lowercase: This is important because it is easy and reliable if all the words are in one form.

(ii)Removing Stopwords: stopwords are ordinary words that have no importance while text processing. So, all the words are removed before the operation.

(iii)Removing punctuation: Punctuation are redundant characters that are in our text corpus.So, it is better to remove them.

(iv)Converting a number to words: Since, an IR model will interpret "20" and "Twenty" as different tokens, we are going to convert numbers to words to improve our model.

(v)deleting single characters:Single are not convenient while text processing because most of them are not important for the document.

(vi)Tokenization:" Tokens are the major components of Natural Language". It is a way of dividing a sentence into smaller sections called tokens.
For instance, examine the sentence: "Physics is Awesome".
The most usual way of creating tokens is based on space. Assuming space as a delimiter, the tokenization of the sentence results in 3 tokens is [Physics,is,Awesome]. As each token is a word, it becomes an example of Word tokenization.

(vii)Stemming: Stemming is the process of manufacturing lingual forms of a root/base word. A stemming algorithm diminishes the words "consultant", "consulting", "consultative" to the root word, "consult" and "connection", "connecting", "connects" and diminishes to the stem "connect". Stemming is a main part of the pipelining process in Natural language processing. The input to the stemmer is tokenized words.

**Term Frequency:** The word frequency (TF) indicates how often a term appears in a text. This metric determines the frequency of a word in a text. This is highly dependent on the document's duration and the word's generality; for example, a common word like "was" may appear several times in a document. However, consider two documents, one with 500 words and the other with 50,000 words.. There's a good chance that common words like "was" would appear more often in the 50,000-word text. However, we can't say that the longer document is more important than the shorter one. We perform normalization

on the frequency value precisely for this purpose. We divide the document's frequency by the total number of terms.
Each text and word has its own TF. hence we can formulate TF as follows.

tf(t,d) = count of t in d/ number of words in d

**Inverse Document Frequency:** The information quality of term t is computed using the inverse of document frequency (IDF). When we measure IDF, we'll see that it's very low for common terms like stop words (because stop words like "is" appear in almost every text, and N/df gives that word a very low value). Finally, we have a comparable weightage, which is just what we want.

idf(t) = N/df

There are a few other issues with the IDF; for example, if the corpus is big enough, say 20,000, the IDF value explodes. So, to counteract the effect, we use the IDF log.
If a word that isn't in the vocab appears during the query, the df will be 0. We smooth the value by integrating 1 to the denominator since we can't divide by 0.
idf(t) = log(N/(df + 1))

**TF-IDF:** TF-IDF is a weighting scheme that assigns each term in a document a weight predicated on its term frequency (tf) and inverse document frequency (idf). The terms with higher weight scores are considered to be more paramount.
Conclusively, by taking a multiplicative value of TF and IDF, we get the TF-IDF score, which is given by

tf-idf(t, d) = tf(t, d) * log(N/(df + 1))

**Vectorization**: Vectorization is a methodology in NLP to map words or phrases from the lexicon to the vector corresponding to the real numbers that are used to find pre-words and homogeneous features of the words / semantics. The process of converting words into numbers is called Vectorization. Here, each word in the Corpus document is mapped to its corresponding tf-idf to compute the identical attribute between the user-submitted query and each document in the set.

**Cosine Similarity:** Cosine similarity can be perceived visually as a method of standardizing document length during comparison. In the case of information retrieval, the similarity of the cosine for two documents will range from 0 to 1, because the term frequencies (using tf - idf weights) cannot be negative. What it does is it will tag all documents as vectors for tf-idf codes and draw them from the center. What will happen is that the length of the query will be few times short but it may be close to the document, in these cases, cosine similarity is best for finding a fit.

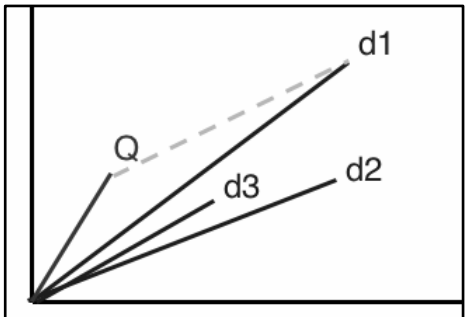

Figure 3.2. Cosine Similarity

Notice the drawing above, vectors d1, d2 and d3 are documents and vector Q is the query, as we can see, although the Manhattan distance (dash line) is very high for document d1, the query is still close to document d1. In these types of cases, the similarity of cosines would be better because it takes into account the angle between these two vectors.

**4. RESULT ANALYSIS**
We have tested our System on documents taken from the "IndianKanoon" website which contains the judgments of cases from sundry courts all over India. Since we tested our system for 10 different queries, our document corpus contains around 180 case documents from which 80 documents are relegated as "Relevant" and other 100 documents as "Irrelevant" for every query. While retrieving, we retrieved the top 60 documents which are most pertinent to the query.
The 10 Queries are:

| | |
|---|---|
| **Query 1** | murder cases in Punjab |
| **Query 2** | domestic violence act cases in Bombay high court |
| **Query 3** | cheating and forgery cases in the supreme court of India and Mumbai high court |
| **Query 4** | cases with bench as Y.K. Sabharwal and Arun Kumar |
| **Query 5** | Divorce case on cruelty and desertion ground |
| **Query 6** | rape and murder with the death penalty |
| **Query 7** | life imprisonment as punishment |
| **Query 8** | bench as G.S. Singhvi in Punjab court |
| **Query 9** | cases of dowry in Allahabad and Punjab courts |

| | |
|---|---|
| **Query 10** | cases of kidnapping and abduction |

*Table 4.1 Query Data*

The System is measured using metrics such as Precision(P), Recall(R), Accuracy(A), Fall Out(FO), and F-Score(F).

| | P | R | A | FO | F |
|---|---|---|---|---|---|
| **Query 1** | 0.83 | 0.63 | 0.78 | 0.10 | 0.71 |
| **Query 2** | 0.88 | 0.66 | 0.81 | 0.07 | 0.75 |
| **Query 3** | 0.81 | 0.61 | 0.76 | 0.11 | 0.70 |
| **Query 4** | 0.85 | 0.69 | 0.79 | 0.09 | 0.78 |
| **Query 5** | 0.88 | 0.66 | 0.81 | 0.07 | 0.75 |
| **Query 6** | 0.87 | 0.65 | 0.8 | 0.08 | 0.74 |
| **Query 7** | 0.9 | 0.68 | 0.82 | 0.06 | 0.77 |
| **Query 8** | 0.78 | 0.59 | 0.74 | 0.13 | 0.67 |
| **Query 9** | 0.93 | 0.7 | 0.84 | 0.04 | 0.80 |
| **Query 10** | 0.87 | 0.65 | 0.8 | 0.08 | 0.74 |
| **Average** | **0.86** | **0.65** | **0.79** | **0.08** | **0.74** |

*Table 4.2 Evaluation metrics.*

**Precision:** Precision is the fraction of the documents retrieved that pertain to the utilizer's information need.
Precision(P) = TP/ (TP+FP)

**Recall:** Recall is the fraction of the documents that pertain to the query that is prosperously retrieved.
Recall(R) = TP/(TP+FN)

**Accuracy:** It is the ratio of several correct presages to the total number of input documents.
Accuracy(A) = (TN+TP)/ (TP+FP+TN+FN)

**Fall Out:** The proportion of non-germane documents that are retrieved, out of all non-germane documents available.
Fall Out(FO) = FP/(FP+TN)

**F-Score:** The weighted harmonic mean of precision and recall, the traditional F-measure or balanced F-score is:
F -Score(F) = (2 * Precision * Recall) / (Precision + Recall)

**Confusion matrix:** A Confusion matrix is a table that is often used to describe the performance of a Document Retrieval Model predicated on a set of test data for which the germane documents are kenned.

| | Relevant | Irrelevant | Total |
|---|---|---|---|
| **Retrieved** | a | b | a+b |
| **Not Retrieved** | c | d | c+d |
| **Total** | a+c | b+d | a+b+c+d |

Table 4.3 Confusion Matrix

Let us Evaluate the Confusion Matrix for Query 1 i.e. “murder cases in Punjab”. The corpus has 180 case documents from which 80 are relevant and 100 are Irrelevant to the Query. We will retrieve the top 60 documents from the corpus and calculate the Confusion matrix.

| | Relevant | Irrelevant | Total |
|---|---|---|---|
| **Retrieved** | 50 | 10 | 60 |
| **Not Retrieved** | 30 | 90 | 120 |
| **Total** | 80 | 100 | 180 |

Table 4.4 Confusion Matrix for a Query

Here,
True Positives (TP) = 50
True Negatives (TN) = 90
False Positives (FP) = 10
False Negatives (FN) = 30
Therefore,
Precision = TP/ (TP+FP) = 50/60 = 0.83
Recall = TP/ (TP+FN) = 50/80 = 0.63
Accuracy =(TN+TP)/ (TP+FP+TN+FN) = 140/180 = 0.78
Fall Out = FP/(FP+TN) = 10/100 = 0.1
F -Score(F) = (2 * Precision * Recall) / (Precision + Recall)
= (2 * 0.83 *0.63) / (0.83+0.63) = 0.71

The Confusion Matrix for all the Queries is shown below:

| | TP | FP | TN | FN |
|---|---|---|---|---|
| **Query 1** | 50 | 10 | 30 | 90 |
| **Query 2** | 53 | 7 | 27 | 93 |
| **Query 3** | 49 | 11 | 21 | 89 |
| **Query 4** | 51 | 9 | 29 | 91 |
| **Query 5** | 53 | 7 | 27 | 93 |
| **Query 6** | 52 | 8 | 28 | 92 |
| **Query 7** | 54 | 6 | 26 | 94 |
| **Query 8** | 47 | 13 | 33 | 87 |
| **Query 9** | 56 | 4 | 24 | 96 |
| **Query 10** | 52 | 8 | 28 | 92 |

Table 4.5 Confusion Matrix Data for All Queries

Below are the graphs for Precision, Recall, Fall Out, Accuracy, and F-Score for all the Queries from which Query 9("cases of dowry in Allahabad and Punjab courts") highest precision, Recall, Accuracy and F-Score, and Lowest Fall Out and Query 8("bench as G.S. Singhvi in Punjab court") has Lowest precision, Recall, Accuracy and F-Score, and Highest Fall Out

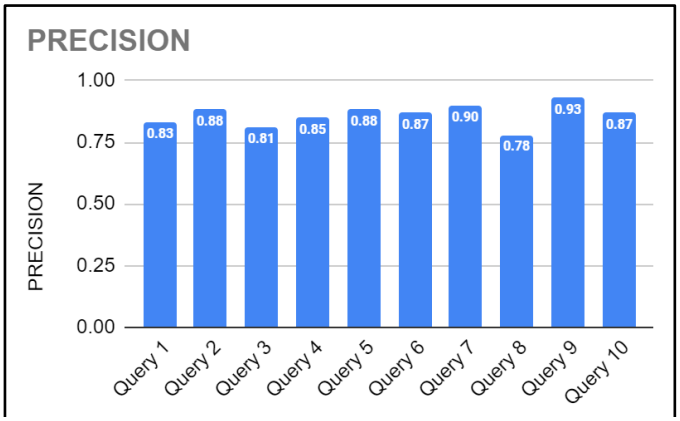


Figure 4.1 Graph plot of Precision vs different Queries.

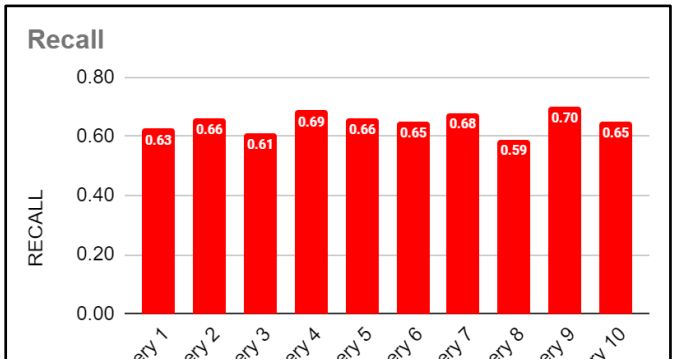


Figure 4.2 Graph plot of Recall vs different Queries.

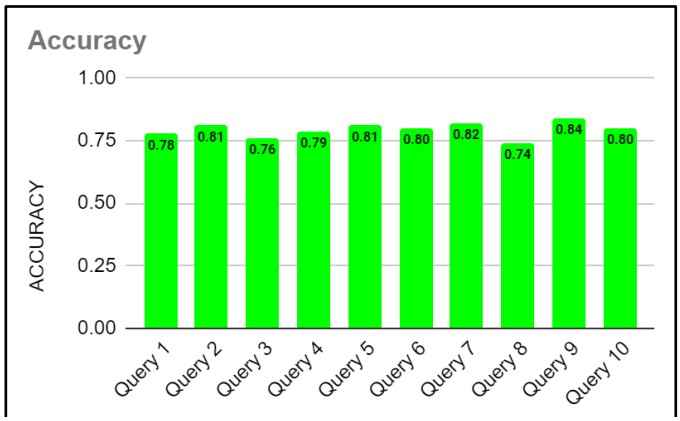


Figure 4.3 Graph plot of Accuracy vs different Queries.

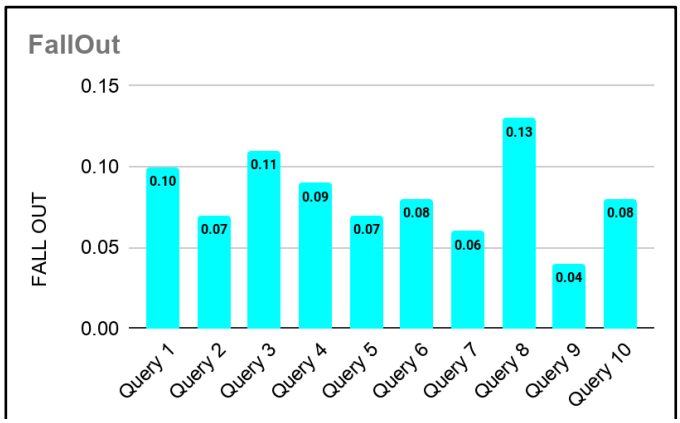


Figure 4.4 Graph plot of Fallout vs different Queries.

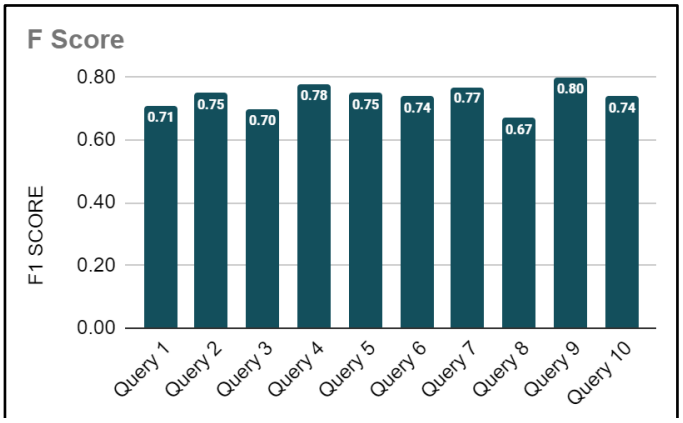


Figure 4.5 Graph plot of F-Score vs different Queries

## 5. CONCLUSIONS

The paper's innovative method of extracting words with high frequency from Indian licit case documents and then using these words text analysis to develop a model yielding good precision in recommending relevant licit documents from the entire document corpus to the one at hand is used to develop a model yielding good precision in recommending relevant licit documents from the entire document corpus to the one at hand. Models like these have a wide range of real-world applications for practitioners in the Indian legal system.

We used query as keywords such as court name, bench on the case, form of case, and so on, then vectorized the query using TF-IDF scores and compared the query vector to all document vectors using cosine similarity, which gives us a value between 0 and 1. (0 being most irrelevant and 1 most relevant document). From the corpus, the top k related documents are retrieved. During the assessment of our method, we achieved the highest accuracy of 84 percent and precision of 93 percent. Since we are passing keywords but not the document description, the system works quickly when retrieving documents. Certain terms that appear in all legal documents may add to the text analysis' difficulty, but it still yields sufficient results while retrieving.

## ACKNOWLEDGEMENT

Prof. Manasi Kulkarni, our Project Guide, has been extremely helpful in constructing and conceptualizing this project subject, as well as providing us with invaluable support. We are grateful to our college, Pillai College of Engineering, New Panvel , and our principal, Dr. Sandeep Joshi, for providing us with all of the resources we needed to complete our project. We'd also like to thank Dr. Satishkumar Varma, H.O.D. of Information Technology, and Prof. Gayatri Hegde, BE Project Coordinator, as well as other educators, for their invaluable assistance. Nonetheless, we would like to express our heartfelt gratitude to all of the people and workers who made it possible for us to complete this document.

## REFERENCES

[1] Aashka Trivedi, Anya Trivedi, Sourabh Varshney, Vidhey Joshipura, Rupa Mehta, Jenish Dhanani, "Extracted Summary Based Recommendation System for Indian Legal Documents", 11th International Conference on Computing, Communication, and Networking Technologies (ICCCNT), 2020.

[2] Arpan Mandal, Kripabandhu Ghosh, Arindam Pal, Saptarshi Ghosh, "Automatic Catchphrase Identification from Legal Court Case Documents", FIRE, 2017.

[3] Fareeha Zahoor, Imran Sarwar Bajwa, "Automatic Extraction of Catchphrases from Software License Agreement", 2014 Sixth International Conference on Intelligent Human-Machine Systems and Cybernetics.

[4] Peter Jackson, Khalid Al-Kofahi, Alex Tyrrell, Arun Vachher, "Information extraction from case law and retrieval of prior cases", Thomson Legal & Regulatory, 2003.

[5] Judith Jeyafreeda Andrew, Xavier Tannier, "Automatic Extraction of Entities and Relation from Legal Documents", Association for Computational Linguistics, 2018.

[6] Mihai Surdeanu, Ramesh Nallapati, Christopher Manning, "Legal Claim Identification: Information Extraction with Hierarchically Labeled Data", Stanford University, 2010.

[7] Madhulika Agrawal,Prasenjit Majumdar, Parth Mehta, Mandar Mitra, "Information access in the legal domain", FIRE, 2014.