DEPARTMENT OF COMPUTER ENGINEERING

UNIVERSITY OF PERADENIYA

CO421/CO425: PROJECT REPORT

IDENTIFYING KEYWORDS IN LEGAL ARTICLES USING ML TECHNIQUES

Team Members

E/15/077-Dilhani K.P.W.A.K.K.

E/15/211-Maduwanthi S.A.I.

E/15/279-Premathilaka L.S.W.S.

Course Coordinator

Dr.Upul Jayasinghe

02, November 2020

Supervisors: Dr. Janaka Alawathugoda (University of Peradeniya)

Dr.Damayanthi Herath(University of Peradeniya)

Mr.Thidas Maduwalthenna(Lawyer)

Acknowledgements

At first we would like to acknowledge both of our supervisors Dr.Janaka Alawathugoda and Mr. Thidas Herath Maduwalthanne for providing us a solid background for this project. So far they have been with us to guide, encourage, motivate and support to complete these tasks. In every meeting, they checked our progress and gave us feedback to improve our work. Not only that, they provided many references which were related to our project and tried to make it easy for us.

Next we would like to acknowledge the Department of Computer Engineering, Faculty of Engineering, University of Peradeniya. Specially to Department Head Dr. Kamalanath Samarakoon for providing us with necessary opportunities to succeed in our project and all the lectures, who support this project to make it true. Instructors of the computer engineering department also gave us the support. So we would like to give our special thanks to them too.

Furthermore, we would like to give our special thanks to our parents for the help and support that has been given us to succeed in every step. Next we would like to thank all our friends for thoughtful and valuable discussions that helped to the success of the project.

Finally, We would like to give our gratitude to each and every one who helped us in every possible way to succeed the project.

Abstract

This paper presents a survey of methods and approaches for keyword extraction tasks. The paper gives an extensive review of existing research, in addition to the systematization of methods. Related work on keyword extraction is involved for supervised and unsupervised methods.

Nowadays there are plenty of legal documents available in electronic format. Therefore, legal scholars and practitioners are in need of systems able to search and quantify semantic details of these documents. Legitimate information and common laws are available in raw form and hard to understand since they are not in organized form. All legitimate information is nowadays computerized since the legal information gets generated frequently in a large volume due to the increase of law courts. The purpose of this research is to explore an efficient way to implement an algorithm to identify keywords by predicting the relevance of legal documents from a huge file system which is difficult to do manually.

The system to analyze this legal data can serve effectively for lawyers and law students, which can address a lawyer's role and can even become powerful to release such a role in future. Designers of such systems are facing a key challenge that the majority of these documents are in natural language streams which are lacking formal structure or other explicit semantic information. In this research, we describe an unsupervised learning approach for automatically identifying important details in legal documents.

The machine learning and deep learning algorithms based analysis systems apply these methods mainly for document classification. Legal document classification, translation, summarization, data obtainment are part of the goals obtained from this research. In this study, we review the different methods of deep learning used in legal tasks such as Legal data search, Legal document analytics, and Legal perspective interface. Through this review, we instituted that machine learning models are giving advanced performance.

Table of Contents

Chapter 1
Introduction

1.1 What is a Keyword?

Keywords are people, places, words, or ideas that are understood as important in the given context. In this research, the context is a legal document, and thus the keyword is expected to reflect the real semantic essence of the legal document. Then, the keyness is a one of the quality measures of which shows the importance of the given text

We do not restrict the keyword to be one word because it is better to extract multi-word keywords, which is called key phrases. By combining words together we can make phrases , and they usually gain a new meaning that cannot be inferred from the single words. Therefore, if we assumed only one-word keywords, we would never find the keyness hidden in the phrases.

Two factors drive the process of identifying keywords. First, the more frequently a word occurs in the document, the more likely it is a keyword. And second, the more frequently a word occurs generally in a speech, the less likely it is a keyword of any document. The second factor reflects that words we use very frequently in the speech, such as prepositions, conjunctions or the most common nouns, are not considered as keywords, even though their occurrence is frequent.

1.2 What is keyword extraction?

Keyword extraction is one of the text analysis methods, extract the most important words in the document and express the idea of it. It helps to get a summarization of the content of the document. The definition of keyword is a significant word,or a word used to find information when researching. They can express approximately the overall idea of the document. Keywords are also called as 'Search Queries', since they are the words or phrases that people use when they are searching. Keywords are important since they provide the connection between what people search for and what system they have.

When there are thousands of documents, keyword extraction helps to find the best matching document for our purpose. Keywords may, for example, be considered as a dense summary for a document, lead to improved information extraction, or be the entrance to a document collection. However, relatively few documents have keywords assigned, and therefore finding methods to automate the assignment is desirable.

1.3 Why keyword extraction for legal articles?

1.3.1 Problem Statement

When lawyers, attorneys or paralegals obtain a legal case to work on, one of the first steps is the searching case related documents. They need to search, clean, and organize the important facts in legal documents in order to study the legal concepts and find pieces of evidence in a legal case. With increasing amounts of information in the electronic form, lawyers put more emphasis on an efficient process of searching documents using keywords. It is common that a legal case consists of a single page to thousands of related documents, which a lawyer needs to scan through to understand a case and find useful information, potentially a shred of evidence.

The goal of this research is to study, design and evaluate a mechanism that is able to highlight legal documents with valuable information and filter out the irrelevant. Such a tool could save the time spent on reading the documents and thus, increase the lawyers' efficiency. In this research, the client is required to have some brief

summary of each document when he searches it using their system. As then, he is able to easily find the suitable documents which he can use to handle the case. So our contribution is the implementation of the algorithm(tool) mentioned above and integration with the Lawciter system. Lawciter is a collaborative system for legal documents management that provides comprehensive control over legal cases including full-text search, sorting of relevant information and visualization of events in time.

The document is structured in the following way. The next section discusses the methods of keyword extraction that we should refer to do the research.The fourth section discusses the previous work on keyword extraction in the legal domain, similar tools, and understanding of the concept of relevance. Then the fifth chapter explains how we decide to use machine learning techniques to make an efficient searching algorithm and the system functionality that legal documents are stored. The paper is concluded with summary and future work directions.

1.3.2 Background

Reading legal documents is a very difficult task and sometimes it needs some domain knowledge related to that document. And also it is hard to read the full legal document without missing the key important sentences and it is a very time consuming task. With an increasing number of legal documents it would be convenient to get the essential information from the document without having to go through the whole document. Hence manual extraction of keywords is slow, expensive and prone to mistakes.

Finding database and e-Resource that provide legal and legislative information is vital need for lawyers of Sri Lanka.There are current implementations but those systems does not come up with efficient solution.There also manual work is costly.We need to reduce man work from the beginning of the portal.To implement user friendly and a system which learn itself to categorize documents the keyword extraction is essential.
Therefore, many algorithms and systems for automatic keyword extraction have been proposed in the recent past. Those experiments are the basic background for this paper.

Chapter 2

Literature Review

2.1 Introduction

Early automatically keywords extraction approaches were focused on evaluating corpus-oriented statistics of extracting individual words. In 1972 Jones and in 1975 Salton described positive results of selecting for an index vocabulary the statistically discriminating words across a corpus. Later keyword extraction research applies these metrics to select different words as keywords for individual documents. As an example, in 1998, Andrade and Valencia used the comparison of word frequency distributions within a text against distributions from a reference collection. But corpus-oriented methods also typically operate only on single words.

Document-oriented methods of keyword extraction have combined natural language processing approaches to identify part-of-speech (POS) tags. They are combined with supervised learning, machine-learning algorithms, or statistical methods. Considering the existing methods that are used in automatic keyword extraction, can be divided into four categories: simple statistics, linguistics, machine learning and other approaches.

2.1.1 Simple Statistic Approaches

Statistical methods are simple and training data is not needed.To identify the keywords in a document statistics information of the words can be used.The statistical keyword extraction methods are based on a sentence-term matrix.In this matrix, each row corresponds to a sentence of a document and each term corresponds to a column.A term can be a individual word or a sequence/set of words.

2.1.2 Linguistics Approaches

These approaches use the linguistics feature of the words mainly, sentences and documents. The linguistics approach includes lexical analysis, syntactic analysis, discourse analysis and so on.

2.1.3 Machine Learning Approaches

Keyword extraction can be seen as both supervised learning and unsupervised learning.Machine learning approach can be used to train a model by using previously extracted keywords and applying the model to extract keywords from new documents. Machine learning technique is mainly categorized into two sections as supervised learning methods and unsupervised learning methods. This approach includes Naïve Bayes, SVM, Bagging, etc. Some keyword extraction tools, e.g. KEA , GenEx have been developed. Following section will be described more on machine learning approaches.

2.2 Machine learning Approaches

In this research we followed machine learning techniques to develop the searching algorithm. Machine learning technique is mainly categorized into two sections as supervised learning methods and unsupervised learning methods.

2.2.1 Supervised Learning Methods

Supervised learning is a learning that is supervised by a set of examples with class assignments. The goal of this approach is to find a representation of the problem in some feature space that is used to build up profiles of the classes. The supervised approach problem is treated as a classification task. Well-known classification models for the supervised learning method are naïve Bayes classification, classification by the C4.5 decision tree induction and neural networks.

In this model, first documents should be trained which are already labeled with keyphrases assigned by humans because in this model keyphrases are picked out

from previously unseen documents. Peter Turney developed key-phrase extraction as a supervised learning problem. In step with Turney, all phrases in a document are potential key-phrases, but only phrases that match with human assigned ones are considered as correct keyphrases. Another notable keyphrase extraction system is KEA with the meaning of Keyphrase Extraction Algorithm. It builds a classifier supporting the Bayes‟ theorem using training documents, and it uses the classifier so as to extract keyphrases from new documents. Another system that relies on linguistic features is the Learning Algorithm for Key-phrase Extraction(LAKE) .It make use of linguistic knowledge for candidate identification. Hulth introduced linguistic knowledge, i.e. part-of-speech (pos) tags, in determining the candidate sets and uses 56 potential pos-patterns in identifying candidate phrases within the text. In Zhang and Lu, a fuzzy classification method for websites is proposed.

2.2.2 Unsupervised Learning Methods

It eliminates the necessity for training data. It selects a general set of candidate phrases from the given document, and uses some ranking strategy to pick the foremost important candidates as key-phrases for the document. Barker and Cornacchia extract noun phrases from a document and rank them by using simple heuristics that support their length, frequency. In, Bracewell et al. extract noun phrases from a document, then cluster the terms which share the identical noun term. The clusters are ranked on the idea of term and phrase frequencies. Finally, topmost n ranked clusters are chosen as keyphrases for the document. Another unsupervised method that utilizes document cluster information to extract keyphrases from one document is presented in.

2.2.3 Graph-Based Methods

The systematic review of methods was gathered which resulted during a comprehensive review of existing approaches. Work associated with keyword extraction is involved for supervised and unsupervised methods, with a special importance on graph-based methods. Various graph-based methods are analyzed and compared. Following are some related works through the years.

In 1998 an algorithm for automatic indexing by co-occurrence graphs constructed from metaphors was presented by Oshawa. It is called KeyGraph. it is based on the segmenting of a graph and representing the co-occurrence between terms in an article.

In 2013 Boudin compared various centrality measures for graph-based keyphrase extraction. Experiments on standard data sets show that Easy degree centrality achieves results which corresponding to the TextRank algorithm obtains the most effective results on short documents.

In 2009 community detection techniques for key terms extraction on Wikipedia's texts were used by Grineva . The results showed that the terms associated with the most topics of the document tend to create a community, thematically cohesive groups of terms.

A stochastic graph based method for computing the relative importance of textual units on the matter of text summarization was presented by Erkan and Radev. LexRank calculates sentence importance and supports the concept of eigenvector centrality in an exceedingly graph representation of sentences.

2.2.4 Other Methods

In 2004 Mihalcea introduced the textRank algorithm for the text summarization used in sentence extraction. It motivated a new branch of research supporting based extraction and ranking algorithms because it performed on supervised and unsupervised summarization methods.
In 2010 Tsatsaronis introduced a network based ranking algorithm called SemanticRank.Semantic relatedness between linguistic units and keywords is the basic of linguistic relation. The keyword extraction from the Inspec abstracts' results reported a good performance of SemanticRank over state-of-the-art counterparts - weighted and unweighted variations of PageRank and HITS.

In 2014 Abilhoa and de Castro proposed a keyword extraction method representing tweets as graphs. Also they applied centrality measures for locating the relevant keywords.Then they could develop Twitter Keyword Graph.In preprocessing step they used tokenization, stemming and stopwords removal method. Keywords are extracted from the graph by cascade applying graph centrality measures closeness and eccentricity. The performance of the algorithm is tested on one text from the literature and compared with the TF-IDF approach and KEA algorithm. Finally, the algorithm is tested on five sets of tweets of accelerating size. The computational time to run the algorithms proved to be a strong proposal to extract keywords from texts, especially from short texts like micro blogs.

Mihalcea in 2004 presented an extension to earlier work, where the TextRank algorithm is applied for the text summarization task powered by sentence extraction. On this task TextRank performed on a par with the supervised and

unsupervised summarization methods, which motivated the new branch of research supporting the graph-based extracting and ranking algorithms.

Tsatsaronis in 2010 presented SemanticRank, a network based ranking algorithm for keyword and sentence extraction from text. linguistic relation relies on the calculated knowledge-based measure of semantic relatedness between linguistic units, keywords or sentences. The keyword extraction from the Inspec abstracts' results reported a positive performance of SemanticRank over state-of-the-art counterparts - weighted and unweighted variations of PageRank and HITS.

Abilhoa and de Castro in 2014 proposed a keyword extraction method representing tweets as graphs and applying centrality measures for locating the relevant keywords. They developed a method named Twitter Keyword Graph where within the preprocessing step they use tokenization, stemming and stopwords removal method. Keywords are extracted from the graph by cascade applying graph centrality measures closeness and eccentricity. The performance of the algorithm is tested on one text from the literature and compared with the TF-IDF approach and KEA algorithm. Finally, the algorithm is tested on five sets of tweets of accelerating size. The computational time to run the algorithms proved to be a sturdy proposal to extract keywords from texts, especially from short texts like micro blogs.

Chapter 3

Methodology & Implementation

3.1.Methodology

The software that is going to develop should be able to identify certain key data contained in the judgements in the website and link the content with each other, based on the identified data.

Website will contain judgements, statutes and various other content. We need an intelligent system which can;

- Identify the names of the previous judgements from the body of a judgement and link those names to the previous judgements uploaded to the website;
- Identify the names of the statutes, section numbers from the body of a judgement and link those names and section numbers to the relevant statutes/section numbers uploaded to the website.
- Identify the key legal concepts discussed in the body of the judgement.

In addition,the system should extract the following data from the judgements.

- Court
- The date and the year of the judgement
- Judges who are involved in the judgement and their roles
- Lawyers who are involved in the judgement and their roles

3.1.1Identify the previous judgements referred to in the judgement and the surrounding sentence

Usually, most judgement will refer to various other previous judgements. These previous judgements can be found at various places of the document.
In a judgement, the previous judgements could be found inside the judgement, within paragraphs.

- In some cases, the previous judgments which are found inside the judgement could be given as a list, at the beginning or end of the current judgement.

ex:

be availed of as admittedly the body corporate A. D. R. Company Ltd., was non-existent.

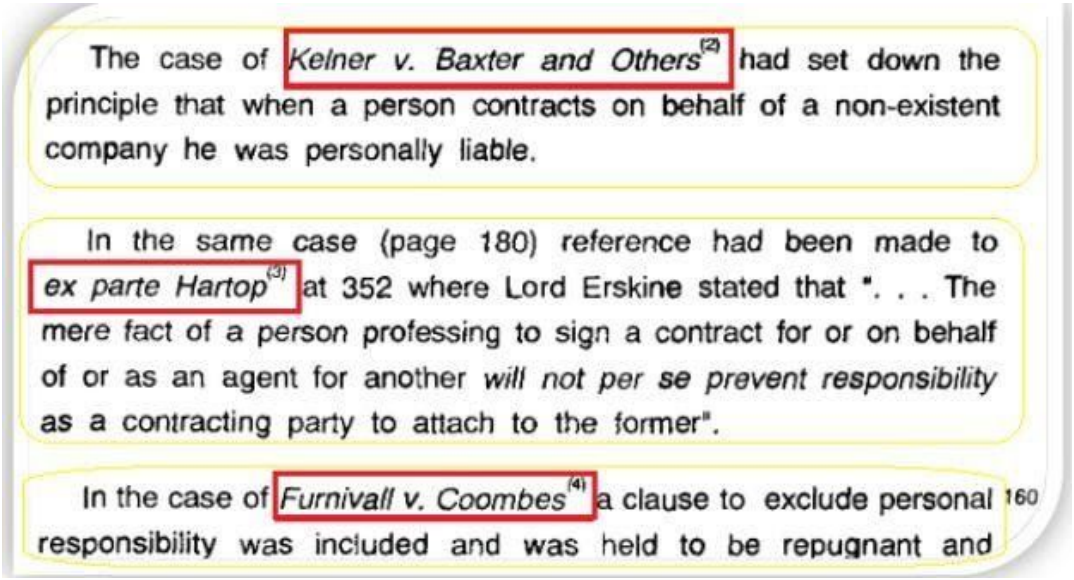
APPEAL from the judgment of the District Court of Colombo.

Cases referred to :

1. *Newborne v. Sensolid* – (GB) 1954 QB 45, 1953 1 ALL ER 708.
2. *Kelner v. Baxter and Others* – 1866 LR 2 Court of Common Pleas.
3. *Ex parte Hartop* –12 Ves 349 at 352.
4. *Furnivall v. Coombes* – 6 Scott NR 522.

Lalanath de Silva for defendant-appellants.

- In some cases previous judgements are not given as a separate list,they are mentioned inside the body of the judgement.

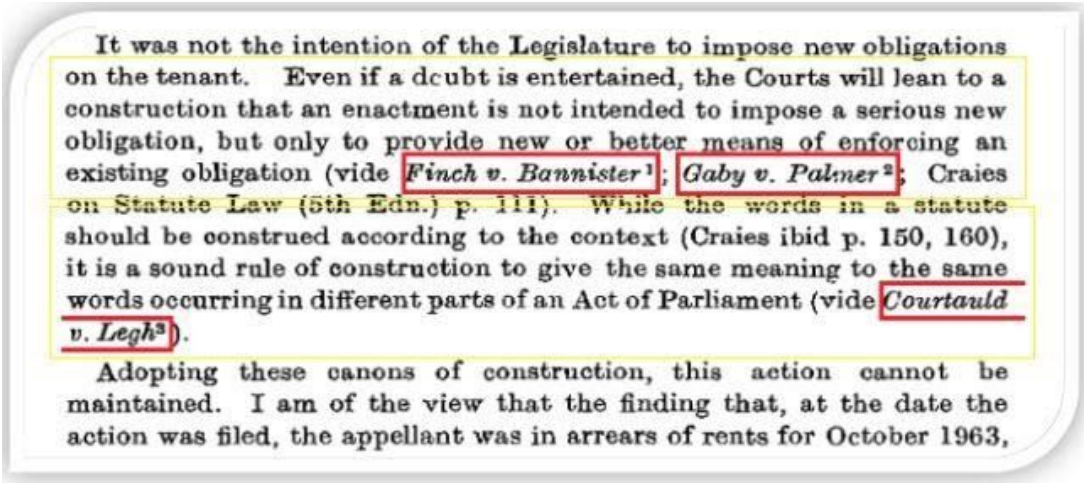


The case of *Kelner v. Baxter and Others*[2] had set down the principle that when a person contracts on behalf of a non-existent company he was personally liable.

In the same case (page 180) reference had been made to *ex parte Hartop*[3] at 352 where Lord Erskine stated that ". . . The mere fact of a person professing to sign a contract for or on behalf of or as an agent for another *will not per se prevent responsibility* as a contracting party to attach to the former".

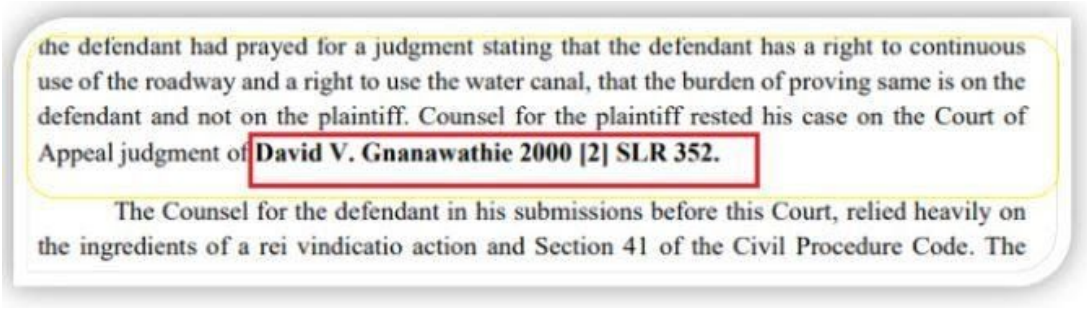
In the case of *Furnivall v. Coombes*[4] a clause to exclude personal responsibility was included and was held to be repugnant and

160



It was not the intention of the Legislature to impose new obligations on the tenant. Even if a doubt is entertained, the Courts will lean to a construction that an enactment is not intended to impose a serious new obligation, but only to provide new or better means of enforcing an existing obligation (vide *Finch v. Bannister*[1]; *Gaby v. Palmer*[2]; Craies on Statute Law (5th Edn.) p. 111). While the words in a statute should be construed according to the context (Craies ibid p. 150, 160), it is a sound rule of construction to give the same meaning to the same words occurring in different parts of an Act of Parliament (vide *Courtauld v. Legh*[3]).

Adopting these canons of construction, this action cannot be maintained. I am of the view that the finding that, at the date the action was filed, the appellant was in arrears of rents for October 1963,



the defendant had prayed for a judgment stating that the defendant has a right to continuous use of the roadway and a right to use the water canal, that the burden of proving same is on the defendant and not on the plaintiff. Counsel for the plaintiff rested his case on the Court of Appeal judgment of **David V. Gnanawathie 2000 [2] SLR 352.**

The Counsel for the defendant in his submissions before this Court, relied heavily on the ingredients of a rei vindicatio action and Section 41 of the Civil Procedure Code. The

Previous judgements which are referred to in the body of the current judgement will generally be surrounded by a specific set of words and will be in a specific format. For example, in the above extracts the following words have been used to introduce previous judgements.

The case of Klener **v.** Baxter and others
vide Finch **v.** Bannister
; Gaby **v.** Palmer
Vide Courtauld **v.** Legh
held in Peter **vs**. James that
the judgement of Peter **v.** James
in Peter **vs**. James,

We will be able to develop a comprehensive list of words that precede and follow the names of judgements. Then, the system, by looking at these pre-identified preceding and following words, will identify the name of the judgement and the sentence in which it is included.

3.1.2.Link previous and current judgements uploaded to the website

The system should be able create links between the names of the previous judgements which are identified in the body of the current judgement and the relevant previous judgements, which have been already uploaded to the website.

- The judgements are uploaded to the system in the chronological order.

- When we upload the 1st available judgement, it will not have any previous judgements, because it is the 1st judgement.

However, when we upload the 2nd judgement, if it refers to the 1st judgement, the system will have the name of the 1st judgement (since it has already been uploaded).

- Once the names of the previous judgements are identified, we want the system to go through our database of judgements and identify those relevant previous judgements and create links between those names with those judgements.

3.1.3.Tf-Idf method for keyword extraction

Term frequency(Tf)–Inverse document frequency(Idf), is a formula that measures how important a word is to a text document.Term frequency means the number of times a word appears in a text and Inverse document frequency means how rare a word is in the entire text.After Multiply these two quantities it provides the TF-IDF score of a specific word in a document. When the score is high,word is more relevant to the document.

TD-IDF algorithms in machine learning are used for several applications.Most of the time search engines use various types of TF-IDF algorithms when searching queries..

This Tf-Idf algorithm metric helps to identify the most relevant words in a document which have the higher tf-idf scores in keyword extraction.

In some cases, the words that appear frequently in a collection of documents are not necessarily the most relevant. And also a word that appears in a single text but doesn't appear in the remaining text may be very important to understand the content of that text.

In a text,words like and, if, the, this or what, will probably have the higher frequenciesAnd also there will be content related words which have higher frequencies but not provide much details about the document.But in this TF-IDF algorithm,we are able to weigh the importance of each term and extract the keywords that helps to get the overall idea of the document.

If we write this in formal mathematical terms,tf-idf for a word t in the document d from document collection D is calculated as,

$$tfidf(t,d,D) = tf(t,d).idf(t,D)$$

Where,
tf(t,d)=log(1+freq(t,d) and idf(t,D)=log(N/count(d⊂D:t⊂d)

When machine learning is used to work with natural language processing(NLP) the major problem is its algorithms usually work with numbers.Therefore we need to transform text into numbers.This is also known as text vectorization.It is one of the fundamental steps for analyzing data in machine learning.
After transforming text into numbers that machine learning algorithms can understand ,tf-idf scores can be fed to machine learning algorithms like Support Vector Machine(SVM) and Naive Bayes.
In this method ,a word vector represents a text as a list of numbers,with one for each possible word of the corpus.Text vectorizing means that taking the text and creating one of these vectors and the vector numbers represents the content of the text.TF-IDF enables us to gives us a way to associate each word in a document with a number that shows how relevant each word is in that document. Then, documents with synonyms will have similar vectors, which is what we are looking for in a machine learning algorithm.

3.2 Implementation

3.2.1.Keyword Extraction from Tf-Idf method

3.2.1.1Extract content of the PDF and convert it into text format.

```
import slate3k as slate
import PyPDF2

def pdf_to_text(docPath):
   pdfFileObject = open(docPath, 'rb')
   pdfReader = PyPDF2.PdfFileReader(pdfFileObject)
   count = pdfReader.numPages

   with open(docPath, 'rb') as f:
      extracted_text=slate.PDF(f)

   file = open("data.txt","w")

   for i in range(0,count):
      file.writelines(extracted_text[i]) #write extracted text into a text file
   file.close()
```

3.2.1.2Data preprocessing

Here we Preprocess the input text in several steps.They are,
-tokenization and Lemmatization
-Removing stop words
-Creating and removing custom stop words.
-Generating required Vocabulary from input
-Preprocessing the input

```
import spacy
nlp = spacy.load("en_core_web_sm")
file = open("data.txt", "rt")
data = file.read()
def textProcessing(doc):
    Nouns = []
    Noun_set = []
    trimmed_noun_set = []
    removing_duplicates = []
    arr = []
    vocab = []
    vocab_dict = {}

    doc = nlp(doc.upper())#Convert to upper case

    for possible_nouns in doc:
        if possible_nouns.pos_ in ["NOUN","PROPN"] : #PROPN ->Proper Nouns
            Nouns.append([possible_nouns , [child for child in possible_nouns.children]])
#append all nouns and pronouns into nouns[] array


    for i,j in Nouns:
        for k in j:
            Noun_set.append([k,i])


    for i , j in Noun_set:
        if i.pos_ in ['PROPN','NOUN','ADJ']:
            trimmed_noun_set.append([i ,j])


    for word in trimmed_noun_set:
        if word not in removing_duplicates:
            removing_duplicates.append(word)


    for i in removing_duplicates:
        strs = ''
        for j in i:
            strs += str(j)+" "
        arr.append(strs.strip())
```

```
    for word in Noun_set:
        string = ''
        for j in word:
            string+= str(j)+ " "
        vocab.append(string.strip())


    for word in vocab:
        vocab_dict[word]= 0

    for word in arr:
        vocab_dict[word]+= 1

    return vocab_dict,arr
```

3.2.1.3.Functions for Tf-Idf

```
def computeTF(wordDict,bow):
        tfDict = {}
        bowCount = len(bow)
        for word, count in wordDict.items():
                tfDict[word] = count/float(bowCount)
        return tfDict


def computeIDF(doclist):
        import math
        count = 0
        idfDict = {}
        for element in doclist:
                for j in element:
                        count+=1
        N = count

idfDict = dict.fromkeys(doclist[0].keys(),0)

for doc in doclist:
        for word,val in doc.items():
                if val>0:
                        idfDict[word]+= 1

for word,val in idfDict.items():
        if val == 0:
                idfDict[word] = 0.0
        else:
                idfDict[word] = math.log(N / float(val))

return idfDict

def computeTfidf(tf,idf):
```

```
        tfidf = {}
        sorted_list = []
        for word , val in tf.items():
                tfidf[word] = val * idf[word]

        ranking_list  = sorted(tfidf.items(),reverse=True, key = lambda kv:(kv[1], kv[0]))[:5]
        for i, _ in ranking_list:
        sorted_list.append(i)

        return sorted_list
```

3.2.2.Find Previous Judgements,Case name and Court name,

```
def find_prevJudg(data):
        sentences_list=[]
        previous_judgment_list=[]
        sentences_list=data.split(".")

        previous_judgments = ["THE CASE OF","THE JUDGMENT OF","VIDE","VS","HELD IN"]
        count=0
        for sentence in sentences_list:
                tokens = nltk.word_tokenize(sentence)
                for i in range(4):
                        for token in tokens:
                                if token==previous_judgments[i]:
                                        if(i==3):
                                                count=count+1
                                                if(count>1):
                                                  previous_judgment_list.append(sentence)

                                                else:

previous_judgment_list.append(sentence)


        if len(previous_judgment_list)>0:
                print("The Previous Judgements:")
                print( previous_judgment_list)
        else:
                print("No previous judgments")


def find_court:
        sentences_list=[]
        sentences_list=data.split(".")
```

```
court_search = ["PRIMARY COURT","DISTRICT COURT","MAGISTRATE
COURT","SUPREME COURT","COURT OF APPEAL","LABOUR COURTS","JUDICIAL
SERVICE COMMISION","C.R"]
count=0;
for sentence in sentences_list:
        for i in range(8):
                if sentence.count(court_search[i])>0:
                        count=count+1
                        if count==1:
                                print("Court and Reference Number:"+sentence)
```

Chapter 4

Results and Evaluation

4.1 Results

Sample1: ALAWATUGODA RATEMAHATMEYA v. KIRIWANTE

ALAWATUGODA RATEMAHATMEYA *v.* KIRIWANTE.

P. C., Nuwara Eliya, 8,926.

Forest Ordinance, No. 10 of 1885, chapter IV.—Prosecution under rules of 3rd February, 1887—Proof in such cases—Validity of judgment—Criminal Procedure Code, s. 372.

In a prosecution for clearing (for chena cultivation) a land at the disposal of the Crown without a permit, in breach of a rule framed under chapter IV. of the Ordinance No. 10 of 1885, it is necessary to prove that the land is not one within a reserved or village forest ; that it is at the disposal of the Crown ; that it is a chena ; that its extent and boundaries are so and so ; and that the accused cleared it.

A judgment of a criminal court should specify the offence with which the accused is charged, in terms of section 372 of the Criminal Procedure Code.

THE charge against the accused in this case was that he cleared for chena cultivation a land known as Komarikagalgawahena (situated at Thenpila in Walapane), at the disposal of the Crown, without a permit from the Government Agent or Assistant Government Agent within whose jurisdiction the land was situated, in breach of clause 1 of the rules dated 3rd February, 1887, framed under chapter IV. of Ordinance No. 10 of 1885, and the Police Magistrate, after evidence heard, delivered judgment as follows : " This is Crown land under the Ordinance 12 of 1840. " Defendant is convicted and fined two rupees and fifty cents."

On appeal (taken with leave of the Court below), *Wendt* appeared for accused appellant.

The Supreme Court quashed the conviction and remitted the case for further evidence.

19th March, 1895. WITHERS, J.—

The accused has been sentenced to pay a fine of Rs. 2·50, but the judgment which precedes the sentence is defective for this, if for no other, reason, that it does not specify the offence in the mode required by section 372 of the Criminal Procedure Code.

VOL. I. L

When we submit this sample1 pdf document to the model,it gives the following results.

```
E:\Academic\7th Semester\CO421425-Final year project>main.py
C:\Users\Wathsari\AppData\Local\Programs\Python\Python38\lib\site-packages\spacy\util.py:275:
 is incompatible with the current spaCy version (2.3.2). This may lead to unexpected results
train your custom model with the current spaCy version. For more details and available update
  warnings.warn(warn_msg)
Case Name:( 73 ) ALAWATUGODA RATEMAHATMEYA VS KIRIWANTE
Court and Reference Number: PRIMARY COURT, NUWARA ELIYA, 8,928
No previous judgments

keywords:
['CHENA CULTIVATION', 'PROCEDURE CODE', 'CRIMINAL CODE', 'RESERVED FOREST', 'ORDINANCE .']
```

Case Name:(73) ALAWATUGODA RATEMAHATMEYA VS KIRIWANTE

Court and Reference Number: PRIMARY COURT, NUWARA ELIYA, 8,928

No previous judgments

Keywords:

['CHENA CULTIVATION', 'PROCEDURE CODE', 'CRIMINAL CODE', 'RESERVED FOREST', 'ORDINANCE .']

sample2:CAVE v. KRELTSZHEIM

1895.
July 30. August 2 and 9.

CAVE *v.* KRELTSZHEIM.

P. C., Colombo, 38,081.

Evidence—Forgery—Comparison of handwriting in disputed document with handwriting in genuine document.

In a case of forgery, a Judge sitting without a jury should not arrive at a decision on the comparison of handwritings without some proof that the handwriting of the disputed document is the handwriting of the accused.

BONSER, C.J.—I am not satisfied that it was intended to allow a jury to find a man guilty of forgery, when no qualified witness could be found willing to state his belief that the alleged forgery was in the handwriting of the accused.

WITHERS, J. (with much hesitation).—It is permissible for a jury or Judge to bring in a verdict of guilty on the comparison of a disputed handwriting with a well-proved handwriting, unsupported by other evidence as to the disputed handwriting.

But a decision of Judge or jury resting solely on a comparison of other than ancient documents is dangerous.

THE complaint against the accused was that he committed criminal breach of trust of certain goods entrusted to him as Value-Payable Parcel Register Clerk of Messrs. Cave & Co. of Colombo. At the inquiry it appeared that some of the goods mentioned were parted with by Cave & Co. on the strength of two letters (marked C and D), which purported to come from one Fernando from Kandy. The first letter ordered the goods, and

```
E:\Academic\7th Semester\CO421425-Final year project>main.py
C:\Users\Wathsari\AppData\Local\Programs\Python\Python38\lib\site-packages\spacy\uti
 is incompatible with the current spaCy version (2.3.2). This may lead to unexpected
train your custom model with the current spaCy version. For more details and availab
  warnings.warn(warn_msg)
Case Name:( 146 ) CAVE VS KRELTSZHEIM
The Previous Judgements:
, IN REGINA VS HARVEY, IS STATED, IN ARCHIBALD'S PLEADING AND EVIDENCE IN CRIMINAL C
] ACCORDING TO SOLITA VS YARROW (1 MOODY AND ROBINSON, 133) A JURY MAY JUDGE OF A DI
RPOSES, AND ADMITTED TO BE OF THE HANDWRITING OF THE PARTY
 AGAIN, ACCORDING TO GRIFFITH VS WILLIAMS (7 CROMPTON AND JERVIX,P
 IN SUPPORT OF HIS CONTENTION, APPELLANT'S COUNSEL CITED THE CASE OIREGINA VS WILLIA
 THE POLICEMAN IS CERTAINLY NOT A SKILLED WITNESS, AND, ACCORDING TO REGINA VS WILLI
 HE CITES IN SUPPORT OF HIS OPINION THE CASE OF COBBETT VS KILMINSTER (4, FOSTER AND
VOL
 THE LAW LAID DOWN IN ALLPORT VS MEEK (4, CARRINGTON AND PAYNE, P
 133 (SOLITA VS YARROW), REFERRED TO IN THE COURSE OF ARGUMENT, IS THUS SUMMED UP IN
 OTHER DOCUMENTS IN " EVIDENCE FOR OTHER PURPOSES AND ADMITTED TO BE THE HANDWRITING
 IN GRIFFITH VS WILLIAMS (1, GROMPTON AND JERVIS, P
 MUDD VS SUCKERMORE (5, ADOLPHUS AND ELLIS, P
 PERRY VS NEWTON (5, ADOLPHUS AND ELLIS, P
keywords:
['JURY JUDGE', 'D C', 'POLICE MAGISTRATE', 'PENAL CODE', 'OTHER DOCUMENTS']
```

4.2 Evaluation

4.2.1.Results evaluation

In this phase we have shown that,

- Identify the names of the previous judgements from the body of a judgements
- Extract the keywords from a particular document.
- Identify the case name
- Identify the Court and reference numbers.

When considering performance of the developed model, in keyword extraction it consists of accuracy,precision,recall and F1 score. Those are considered as perfect matches.In this phase we could achieve basic goals of our experiment. It is very important to extract the basic idea of the document by using keywords. We could identify names of the previous judgements from the body and we could identify case name,court and the reference numbers accurately.

But by using predefined keywords we can check perfect matches, and to check partial matches length and number of sequences that can be defined manually are the measures. To conclude, we hope to evaluate the keyword extraction by using more keyword extraction methods and by combining them.

4.2.2.Difficulties and challenges

There are several challenges we have to face in this scenario as follows,

1. The judgments in our website will be HTML files and not PDF files.But Since the system is still in the developing stage there are no sufficient case law reports in HTML format.Therefore,we have to use pdf files until HTML files will be available.When extracting the data from pdf there are several issues.When compare the pdf with extracted text there are some mismatching words .The reason for that is unclearness of the pdfs since they are scanned pdfs.

2. There is not exactly one format in case law.Mainly there are four types.They are,
   1. NLR format
   2. SLR format
   3. Unreported Supreme Court cases format
   4. Unreported Court of Appeal cases format

The way the content is arranged in each judgement will differ from one another.Therefore the system should understand all of the variations in all formats.

3. Sometimes, the names of the cases may have some variations. For example, in one case, a previous case may be referred to as "***Silva vs Perera and three others***" in another case the same case may be referred to as "***Silva vs Perera and 3 others***," or "***Silva vs Perera et all***" etc. Therefore, the system should be able to intelligently link the names of the previous judgements, appearing on the body of the judgement, with the correct previous judgements we have uploaded to the website.

4. There are a number of dates mentioned in these case law reports both past and future dates.So it is important to identify the date which is the exact date that case is conducted at the court.Also there are several formats in dates.As an example,23rd December 2020 can be mentioned as 2020-12-23 or 23-12-2020 and etc.Therefore System should identify all those formats.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This research searches answers on the questions of keyword identifying process of legal documents, especially, how are the legal documents different from others, what
makes a legal document relevant, what features of a legal document matters the most, if the structure is important in the relevance prediction, and what tools and estimators work best for relevance prediction in the legal domain.

During the data analysis, some ideas on improvement of the relevance prediction were proposed. Our plan is to implement a method searching documents using a single keyword or keyword phrase. There are still opportunities for further enhancements, which might rapidly speed up and ease the work for lawyers. Hopefully, the presented research will help lawyers and software developers in future decisions to achieve these enhancements.

Besides the research, a simultaneous outcome of the research was also the implementation of a system that applies the mentioned techniques and is integrated with an E-discovery system Lawciter. The system presented all documents of law cases. If the user testing confirms usability and usefulness of the extension, it will reach the final release.

5.2 Future Works

In the next phase we are going to do the followings

1. Identified previous judgements in the document, link to the previous judgements which have already uploaded to the website

In this stage we are able to identify and extract the sentences which contain previous judgements by looking at the given specific word phrases.But also we want to extract the exact case name.So in the next phase we are going to extract the exact case name and link them to the previous judgements uploaded to the website.

2. Identify the names of the statutes, section numbers from the body of a judgement and link those names and section numbers to the relevant statutes/section numbers uploaded to the website.

There will be different patterns of words in statutes which precede and follow, with the name of the statute. So system should identify those statute's names
and link to the relevant contents.So in the next phase we are going to develop the system as it can show those linked judgements on the statute pages and statutes on the judgement page, of our website.

3. Identify the key legal concepts discussed in the body of the judgement

We are going to achieve this by using legal dictionary whis is online available.Once these key concepts identified we have to show the key concepts discussed in the text of the judgement on the web page with the ability to click on those key concepts/key words to generate a list of judgements where those have been discussed

4.Develop an API with efficient machine learning model

To use this in the industry we should write an API and some sort of documentation describing how to use the API.Also we are going to develop a efficient machine learning model

Chapter 6

References

[1] Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents.

[2] Kore, R. C., Ray, P., Lade, P., & Nerurkar, A. (2020). Legal Document Summarization Using Nlp and Ml Techniques. International Journal of Engineering and Computer Science, 9(05), 25039–25046. www.ijecs.in

[3] Krapivin, M., Autayeu, A., Marchese, M., Blanzieri, E., & Segata, N. (n.d.). Improving Machine Learning Approaches for Keyphrases Extraction from Scientific Documents with Natural Language Knowledge. Retrieved July 27, 2020, from http://citeseer.ittc.ku.edu

[4] Lopez, P., & Romary, L. (2010). HUMB: Automatic Key Term Extraction from Scientific Articles in GROBID. Association for Computational Linguistics. http://wikipedia-miner.sourceforge.net

[5] Hulth, A. (n.d.). Improved Automatic Keyword Extraction Given More Linguistic Knowledge.